Packet Radio and IP for the Unix[1] Operating System

Clifford Neuman, N1 DMM

Department of Computer Science
University of Washington
Seattle, WA 98195

ABSTRACT

Many services are currently available on the ARPA Internet that would be of interest to amateur packet radio users. The ARPA Internet connects universities and other organizations around the world that speak TCP/IP. One advantage of running TCP/IP on packet radio is the ability to access these services, and to interconnect with other systems that are part of the internet. This paper describes the implementation of AX.25 as a link layer protocol for the Unix operating system and the use of this system as an IP gateway between our local amateur packet radio network and our department's ethernet at the University of Washington, which in turn provides access to the entire Internet. The potential role of such a system for amateur packet radio is discussed, and a mechanism to allow users that don't have the resources to run TCP/IP themselves to access such services is described.

1. Introduction

In the past year, we have started to see wider acceptance and use of layer three protocols in amateur packet radio. So far, most of this activity has been by people who are interested in advancing the state of amateur packet radio. Many are people who deal with computer networks outside of amateur radio, and would like to see similar facilities available within packet radio. Others have worked on mechanisms to solve some of the problems that amateur packet radio produced, and their solutions have made a significant difference in the way people use packet radio in the parts of the country where their solutions are being tested.

In order to get more use of layer three protocols by the users instead of the developers, there are two requirements that should be met. First, they need incentive. There should be something that they can do using layer three protocols that they can't do using connection[2] mode. One incentive is the ability to access some of the services available on more established networks such as the ARPA Internet Among these services are nameservice, file transfer, access to various databases, a more flexible system for electronic mail, and the ability to log into hosts on connected networks. These services can be made available in two ways. Servers can exist directly on amateur packet radio hosts, or they can exist on other networks with a gateway set up between the two networks. By connecting our local packet radio subnet to the internet, it is possible to access files on, and log into, computers at other internet sites (or at least, those where we have accounts).

Secondly, we have to lower the cost of entry. Most packet users do not have IBM PCs, or computers of equivalent or greater power. Many are simply using terminals connected to their TNC. This is probably one of the reasons that NET/ROM is so popular. With a packet station and no special hardware, one is able to connect to a NET/ROM node, connect to another node through the network, and come out the other end. If we are to generate as much interest

[1] UNIX is a trademark of AT&T

[2] By "connection" mode we mean the existing connection mechanism provided with TNCs when higher level protocols are not being used.

in layer three protocols such as TCP/IP, then we must make it easy for connection mode users to connect to, and use, a system that speaks IP. We can then point out the advantages they would have if their own system spoke IP directly. Among these advantages are the abilities to exchange mail and transfer fi les while simultaneously connected to one or more other systems.

2. System Overview

We decided one way to approach the above problems would be to get a machine that is on our department's ethernet onto packet radio. We had a MicroVax-I[3] available for our use. The advantage of using such a machine is that it already supports many of the network services that are desirable in the packet radio community. Among these are electronic mail, remote login, fi le transfer, and name service. Although not presently running on the machine, there are other applications available too, such as NNTP[4] which could be used as a bulletin distribution mechanism.

The existence of such a system gives users who have brought up TCP/IP something to connect to. The next step is to give people who aren't yet running TCP/IP something with which they can connect. To do this, we want to allow users to connect to our system in connection mode, and for them to be able to login to our system by this mechanism. The only other service we want to support in connection mode is mail. We would like to be able to exchange mail with PBBSs, which don't speak TCP/IP.

3. Role in a packet network

A machine such as the one I described above serves several functions in a packet radio network. It functions as a server for various network services. It is useful as a "home" machine for those users who do not have computers of their own. It also can serve as a gateway between multiple packet subnets, and perhaps even non-amateur networks. I have already described its use as a server. In this section I describe its use in some of these other functions.

[3] MicroVax is a trademark of Digital Equipment Corporation

[4] Netnews Transfer Protocol

3.1. Home machines

For those users who don't have IP running, our machine can serve as a home machine. Users can connect to it by using connection mode. The system can support multiple connections of this type simultaneously. When a user is connected to our system, he can use the various services available to IP hosts. He also will be allocated a limited amount of disk space, and will be able to retrieve fi les in which he is interested. The mail interface the user will be able to use presents a better interface than the central BBOARD mechanism which is currently in use. The user will be able to store messages indefinitely as long as he doesn't exceed his quota.

3.2. Level 2 to Level 3 Gateway

Since the user can connect to the system using connection mode, and since the system also speaks TCP, the system serves as a Level 2 to Level 3 gateway. Users will not have to give up connectivity with the old in order to begin using the new.

3.3. IP Gateway

A machine such as the one described above is also a logical machine to use as an IP gateway, at least until such time as we have dedicated machines for such purposes. Gatewaying could be between multiple packet radio networks, and even between non-radio networks such as the ARPA Internet

There are services available on the ARPA Internet that are of interest to packet radio users. If there is a university in the area, it is likely that there may be an online database of upcoming events. There are also many mailing lists on the Internet that might be of interest to Amateurs.

Connecting to non-amateur networks does bring up a number of issues, such as screening of messages in both directions. I discuss solutions to this problem later in this paper.

3.4. NET/ROM

NET/ROM fits in nicely with TCP/IP. IP can be run on top of NET/ROM. In such an arrangement, users on a LAN would speak IP on top of AX25. Multiple LANs could then be linked together using NET/ROM. An IP gateway would exist on each local area network and would appear as a NET/ROM node to other NET/ROM stations. This arrangement is similar to the way that local area networks are linked by the

ARPAnet. NET/ROM nodes correspond to the IMPs on net IO.

4. Related Work

There has been a lot of work recently in the TCP/IP arena. Work has been done on Phil Karn's IBM-PC code, and it has been ported to other machines such as the Amiga, the Mac, and others. Steve Ward and Mike Chepponis have been working on additional features in order to give users greater incentive to upgrade to TCP/IP.

Implementations of the TCP/IP code are needed for many more machines. Services such as the ones I have described also are needed for these machines. Not many people have access to a MicroVax as I did. It is a good machine to use in order to determine how network users react to such services. The more machines such services are available on, the more people will be able to set them up.

5. Implementation

The Ultrix[5] kernel already had all the code necessary for Internet Protocol. Because we did not modify the "upper" IP interface, layers riding on top of IP were able to use the packet radio medium without modification. Thus, TCP and UDP did not need to be modified and, similarly, applications running on top of those protocols worked without modification. The IP code in the kernel did not require modification either. All we had to do was to find a way to take the IP packets generated by the kernel, encapsulate them in AX.25 packets, and send them off, using SLIP, to the KISS interface of the TNC.

5.1. IP and AX.25 and the gateway

We chose to implement a pseudo-device driver for the packet radio interface. The driver supports the same calls as network device drivers do for other media such as ethernet Our driver is a pseudo driver because there is not really any hardware on the bus for our packet radio controller. Instead, our controller is plugged into a dz[6] port, and the kernel must communicate with it through that port

Teaching the kernel to recognize the new interface was easy. There is a structure called if-net that is associated with each interface. This structure contains pointers to the kernel procedures, which are used to initialize the interface, send a packet, change parameters, and a few other operations. The next trick was to figure out how we could receive packets. This was done by including a routine similar to the one that gets called in the ethernet driver when a packet arrives. The difference, though, is that our routine is called by the dz driver whenever a character is received on the line to which the TNC is connected.

As each character is read, we do some initial processing on the fly. In particular, we unescape frame end characters that are embedded in the packet. When the final frame end is read, we check the header of the message, note the callsigns, note the layer three protocol type, and if it is IP, we add the encapsulated IP packet to the queue of incoming IP packets to be dealt with by the existing upper layers.

In order to implement the routines described above, we started with a few routines from Phil Karn's code for the IBM PC. These routines encapsulated and decapsulated AX.25 packets. With a few modifications these routines were made to work in the Ultrix kernel.

The gateway functionality came for free. The way an IP gateway works is that when a packet is received, the system looks at its IP header to determine the destination address. If the destination address is not its own, it then decides which is the correct destination interface, and which system is the correct next hop. This is all done at the IP layer, and the same code that existed for gatewaying packets on ethernets works for AX.25 subnets too.

5.2. Address Resolution Protocol

The final task was to translate internet addresses into AX25 addresses. This is done using ARP, the address resolution protocol, in the same manner that IP addresses are translated into ethernet addresses. But, AX25 addresses look like amateur radio callsigns followed by a 4 bit system ID. To make matters worse, some entries may contain additional callsigns for digipeaters that are to repeat the packet Thus, what is needed is a different set of ARP routines for the packet radio interfaces. Phil Karn's IBM-PC code includes an ARP implementation that supports both AX25 and ethernet addresses. Because we did not want to modify the code for our system that is

[5] Ultrix is a trademark of Digital Equipment Corporation

[6] A controller for multiple RS-232 ports

used on the ethernet side, we decided not to take this code. ARP lookup occurs at layer two, and thus, gets called inside either the ethernet driver, or the AX.25 driver. The routing tables at the **IP** layer determine which driver is called. Since the ARP lookup occurs inside our code, we are able to call a separate routine that deals specifically with AX.25 addresses.

5.3. Connection mode

As already discussed, we would like to support connection mode on our gateway. Doing so would allow users who do not have the resources to run TCP/IP to be able to access **IP** network services. Further, users can give **IP** a try, and if they like it, then they might consider running it themselves. However, there is no reason, though, that connection mode should be supported in the kernel as is **IP.**

The way our implementation is set up, it is easy to allow user level process deal with connection mode. We can tell the kernel that if a packet comes in, and its protocol **ID** is not **IP,** that the packet should be placed on the input queue for the appropriate tty line. A user program can then read packets that the system isn't interested in from that line, and deal with the packets itself. By setting appropriate parameters for the kernel, additional filtering could be provided, though one would not want to do anything too complex in the kernel.

The user level process that reads such packets would have to keep track of any connections and support connection mode itself. Such a program could maintain multiple connections, and direct input to and output from pseudo terminals. This would allow connection mode users to log into the system. Such a program could accept connections to multiple SSIDs, thus allowing one SSID to be used for the transfer of mail with local non-IP bulletin boards.

5.4. Other layer 3 protocols

In addition to supporting connection mode, support could be provided in a similar manner for other layer 3 protocols. I already mentioned how NET/ROM can be used to forward **IP** packets. One could conceivably support the rest of the NET/ROM interface in the same manner as connection mode is supported. Of course, NET/ROM users would not have the benefit of the additional services available using **IP.**

6. Unresolved issues

The ability to interconnect amateur packet radio networks and non-amateur networks introduces a few problems which have not been completely resolved as of this time. In this section, I present those problems, and for some of them, **I** suggest some possible solutions.

6.1. Timeouts

One problem that comes up is the difference in bandwidth for the two networks. Hosts on the ethernet side expect fast response, and if they don't get a response quickly, they time out and retry their transmission. We have found that when connected to a system on our department's ethernet from a machine on the packet side of the gateway, the system on the ethernet side initially retransmits packets several times before a response makes it back. This results in wasted bandwidth on the radio side as the packet is needlessly retransmitted, and this in turn delays other packets. Fortunately for some implementations of TCP, once the connection has been established, the system on the ethernet side learns the correct timeout, and things settle down.

6.2. Internet routing

Routing is another problem that arises if we want to allow connections to internet hosts beyond our department's ethernet In order for a response to come back, all the gateways between the source and the destination must know the route to the appropriate packet radio subnet. Since a class 'A' network is allocated for AMPRnet, and since most systems by default will maintain a single route for a class 'A' network, only one path exists for all of AMPRnet, whereas what is desired is different: gateways for different subnets. It is conceivable that something like this could be handled using ICMP[7] redirects, but at this time, no mechanism is in place.

6.3. Access Control

Another problem we face is access control. Since operation is on frequencies assigned to the amateur radio service, any communication must be initiated by licensed amateurs. One way we can solve this is to maintain a table of authorized addresses on the non-amateur side of the gateway. Associated with each of these addresses is

[7] Internet Control Message Protocol

a list of hosts on the amateur side of the gateway with which that host can communicate. Initially the table starts off empty. Whenever a packet is received on the amateur side destined for a non-amateur host, an entry is made in the table, enabling the non-amateur host to send packets in the other direction. After a certain period of time, these entries time out if packets have not been received from the amateur side of the gateway.

This scheme can be augmented with a few new ICMP messages. One message can force an entry to be removed from the table of authorized non-amateur systems. This allows the amateur radio operator that initiated the link to exercise his control operator function to cut off the link if he detects inappropriate use. Another message would allow one to add an authorized non-amateur host to the tables with an appropriate time to live. Both these message are allowed to come from either side of the gateway, but if they come from the non-amateur side, they must include a call sign and a password of for an authorized control operator for the gateway.

7. Status

The packet radio implementation of IP works. We have successfully connected from an **IBM** PC with a packet radio controller to a machine on our department's ethernet using telnet.[8] The connection was made using our MicroVax-I as a gateway. We also were able to telnet from the machine on the ethernet to the PC.

In the Seattle area, we are using a duplex repeater as the base for our local area network. Our network extends from Seattle, south to Tacoma, west to a station on the other side of Puget sound, and east to the Cascades.

We have not yet written the user program to support connection mode logins, but that is being considered. We also have not yet done anything towards using NET/ROM to interconnect our local area networks with others, but we would like to do that soon.

8. Conclusions

The Unix operating system provides a nice base upon which network services can be provided for the amateur packet radio community. At the same time, such a system can serve as a central node in the interconnection of local area networks running **IP,** and even those that don't run **IP.** By linking packet radio networks with more established networks, additional services become available. Such services are available in the Seattle area. These services are necessary if we are to interest people in running TCP/IP. Further, interconnection with non-IP packet radio users is necessary if we are to interest users who would like to try **IP,** but still want to maintain connectivity with those still using connection mode.

9. Acknowledgments

A number of people were helpful in getting our implementation running and in discussing some of the ideas presented in this paper. Among them are Bob Albrightson (N7AKR), Bob Donnell (KD7NM), Dennis Goodwin (KB7DZ), Mike Chepponis (K3MC), Steve Ward (W1GOH), and Ed Lazowska (KG7K). Thanks are also in order to Bob Hoffman (N3CVL) for typesetting this paper.

10. Bibliography

1. Fox, Terry L.: AX.25 Amateur Packet-Radio Link-Layer Protocol. Version 2.0. American Radio Relay league, October 1984.
2. Karn, Phil: "TCP/IP, A Proposal for Amateur Packet Radio Levels 3 and 4", Fourth *ARRL* Computer *Networking Conference,* San Francisco, 1985.
3. Leffler, S; Joy W.; Fabry R.; Karels M.: Networking Implementation Notes 4.3 BSD Edition. Computer Systems Research Group, University of California, Berkeley. June 1986.

[8] One of several remote login protocols.