TCP/IP: A Proposal For Amateur Packet Radio Levels 3 and 4

Phil Karn, KA9Q

Radio Amateur Satellite Corporation

ABSTRACT

This paper presents a case for basing Level 3 (the network layer) of Amateur Packet Radio on the "datagram" concept. It further proposes that the DARPA protocols IP (Internet Protocol) and TCP (Transmission Control Protocol) be adopted intact as the standard Level 3 (Network) and Level 4 (Transport) protocols for Amateur Packet Radio.

I will then provide an overview of TCP/IP, explain why it, as a datagram protocol, is more suitable for our needs than the virtual-circuit protocol CCITT X.75, and show how it would be used above the AX.25 Level 2 protocol already in use.

1. Datagrams and Virtual Circuits

A fundamental characteristic of ARPA (and several others, e.g., Xerox PUP [15]) protocols is the choice of the "datagram" as the fundamental unit of communication within the network. To understand what this means, a comparison of the datagram approach with its chief rival, the "virtual circuit," is needed.

1.1 What is a Datagram?

The word "datagram" is coined from the words "data" and "telegram." Like telegrams, datagrams are simple one-shot messages; each is self-contained in that it includes the full source and destination addresses, control information and user data. Each datagram is independently processed by the network. All information needed by a packet switch to route datagrams through the network is wholly contained within each datagram. No state need be maintained by a packet switch between datagrams. There are many analogies to this mode of operation besides telegrams: mailing a letter, sending electronic mail, or entering a message into the amateur radio National Traffic System.

The network makes a "best effort" attempt to deliver each datagram. If datagram delivery is impossible (e.g., due to network congestion, buffer overflow or an unknown or unreachable destination address), a packet switch may discard a datagram. Some datagram protocols (such as IP, to be described later) require that an effort be made to notify the sender of the problem.

Datagrams are never discarded lightly; however, there are usually varying degrees of "best effort" that can be expended before "giving up" on a datagram. Frequently, a greater effort at reliable delivery increases the "cost" (in some sense) of sending the datagram, or affects the user in some other way, e.g., by decreasing throughput or increasing delay. IP, as discussed later, gives the sender the ability, if desired, to specify the importance (i.e., the precedence) of a datagram and to influence any tradeoffs between delay, reliability and throughput that might exist in individual links and gateways within the network.

In any event, a datagram user must always be prepared to cope with the occasional loss, out-of-sequence delivery or duplication of datagrams caused by network congestion or switch or link failure. Since many applications require guaranteed service, a separate, higher level protocol using end-to-end acknowledgements and retransmission of lost datagrams is generally used "on top" of the unguaranteed datagram service.

1.2 And In The Other Corner... The Virtual Circuit

As the name implies, "virtual circuit" networks (hereafter abbreviated "VC networks") are oriented to provide the appearance of a direct connection between a pair of users. The network sets up a fixed path though the network to the destination for the duration of the user's connection. Because the path may be shared by several users (i.e., the physical facilities are not dedicated to a single user), the connection is "virtual."

A special "call setup" packet propagates through the network, and each switch adds a "virtual call" to an internal table so that the data packets that follow may be correctly routed to their destinations. The best analogy to a VC network is the telephone system, although the analogy isn't perfect because the telephone network usually dedicates fixed physical resources (a wire pair or a channel on an RF carrier) to each call.

Routing in most VC networks is static; once it is established at setup time, all packets follow the same route to the destination. As long as all links and switches traversed by the virtual call remain functional, the users' data will be properly delivered in sequence. However, should a switch crash or a link fail, all virtual circuits using the affected switch or link will be dropped and any data in transit will be lost.

When the user is done with a virtual circuit, it is cleared. This removes the information about the call from the memory of each packet switch along the call's path.

1.3 Discussion: Datagrams Vs. Virtual Circuits

Many applications, such as remote terminal access to a computer, require a reliable, flow-controlled "stream connection" between two end points, regardless of how this might be implemented in the bowels of the network. Therefore, the issue is NOT *whether* the user should be provided with a reliable end-to-end stream, but rather *how* it ought to be implemented. Should the concept of a "virtual circuit" be confined to the endpoints of a "connection" or should it permeate the design of the lower levels of the network?

The choice has many implications for reliability, flexibility, ease of implementation, efficiency, and adaptability to varying user-level service requirements. The decision is a tradeoff, and often the choice depends on those characteristics considered most important. Neither approach is always superior.

1.3.1 Ease of Implementation Datagram packet switches are considerably easier to implement than VC switches. The lack of special "call setup" and "call clearing" packets means that all packets are alike as far as the switch is concerned. All that it has to do is select an outgoing link for the packet (typically based on a routing table that is periodically updated from its neighbors) and send the packet on its way. If there is a serious problem with the packet, the switch is entitled to discard it; no intricate error-recovery procedures are needed. Since the "what to do when things go wrong" section is the largest, most difficult to write and least reliable section of almost any programming project, this results in an enormously easier coding job.

1.3.2 Dynamic Routing As already mentioned, virtual circuit networks establish fixed paths through a network of packet switches and links. If a given link fails or becomes overly congested, there is no easy way to re-route established virtual circuits via alternate paths.

Datagrams, with their self-contained nature, may be individually routed without regard to any end-to-end connections that might exist at a higher protocol level. This makes it possible to react on a per-packet basis to changing traffic conditions and network reconfigurations. Much of the work to date in non-amateur packet radio has been done in a mobile environment, and dynamic routing is essential here because of the constantly changing topology of the network.

While it is certainly possible to make routing decisions based on link loading at circuit setup time in a virtual circuit network, this is less responsive to rapidly changing network conditions than the ability to route on a per-packet basis.

1.3.3 Overhead This is *the* primary objection that is made against datagram protocols. Virtual circuit protocols require that complete addresses be sent only at circuit setup time. Once the table entries are made in each switch along the path of a virtual circuit, only the index into this table (referred to as a "virtual circuit number") need be part of each data packet for the switch to route it properly. Depending on the size of the data fields, the larger headers involved in datagram packets can involve considerable overhead. This is primarily true with interactive terminal traffic that often consists of single character packets; it is much less of a factor when data fields are larger.

On the face of it, virtual circuit protocols seem to win the overhead argument hands down. However, there are applications where the direct availability of a datagram service to the user (e.g., the ARPA User Datagram Protocol, UDP [11]) results in fewer packets and bits being exchanged to accomplish the same task.

Such applications typically have a "client-server" characteristic. For example, a database server might be set up to provide "directory information" (i.e., providing the network number corresponding to a given station's name). Most transactions with such a database server are short; the request and replies each fit easily into single datagrams. In a virtual circuit network, a virtual circuit must be first set up between the client and the server, the request made, the response received, and the virtual circuit torn down. This clearly results in more network traffic than if one-shot datagrams were used at the network level, avoiding the overhead of setting up a virtual circuit for such a short "connection."

To answer this objection, the X.25/X.75 protocols include an optional "fast select" feature that allows user data to be sent in the same packet with a call request. Fast select is not, however, a substitute for datagrams. A virtual circuit is still being established, although for a short time. A reply packet (typically a CLEAR INDICATE), with or without data, is still expected from the destination within a time limit imposed by the network, and the data fields contained in either packet are limited to 128 bytes; there is no fragmentation facility. This is considerably less general than a "true" datagram facility.

However, in the common situation where the application requires an end-to-end connection for a relatively long time, virtual circuit networks do require fewer bits to be transmitted than do datagram-based networks. In situations where the traffic consists primarily of single-character packets (e.g., interactive terminal access) and efficient use of slow and expensive transmission facilities is of supreme importance, the lower per-packet overhead of the virtual circuit approach can be the overriding factor. [13]

While amateur packet radio is currently severely constrained by obsolete Bell 202 modems and 1200 baud transmission, dedicated RF modems operating at much higher speeds (orders of magnitude) are now being introduced. [16] Since these modems will not cost much more than those currently used (assuming a dedicated radio), this will almost completely mitigate the overhead argument.

1.3.4 Reliability Because a datagram contains all information necessary to forward it onto its destination, no state has to be maintained in a datagram packet switch between packets. This makes datagram networks much more resistant to real-world occurrences such as power glitches, software failures and nosy visitors who push reset buttons.

Since the reliability requirements are less for a datagram switch (since it has no volatile table of virtual circuits to safeguard) such measures as battery backup can often be dispensed with.[1] The only information that is typically lost within datagram packet switches during failures are routing connectivity tables (assuming a distributed routing algorithm is used), but these can be quickly rebuilt from one's neighbors. Virtual circuit switches, on the other hand, must maintain the information provided to it at circuit setup time to route successfully each data packet of a virtual connection. In general, this information cannot be rebuilt from one's neighbors, and the end user must re-establish the virtual circuit and recover from any lost data.

To achieve maximum reliability against internal network problems, both datagram and virtual circuit networks require a higher-level end-to-end "transport" protocol. A transport protocol recovers from various errors that might occur in the network (lost, reordered or duplicated packets in a datagram network, or dropped virtual circuits in a virtual circuit network). The transport protocol used atop the ARPA datagram protocol, IP, when reliable stream communication is desired is called TCP (Transmission Control Protocol).

A major advantage of an end-to-end protocol such as TCP is that it provides protection against data corruption (as well as loss) along the ENTIRE network path. Link level error detecting codes (such as the 16-bit CRC in AX.25) protect only against errors on transmission links. Without end-to-end protection, a user is still vulnerable to data corruption that can occur in a packet switch between the reception of a packet and its retransmission with a freshly regenerated CRC. The probability of this occurring in a single packet switch may be acceptably small, but in a large network composed primarily of inexpensive microcomputers without memory error detection, errors are inevitable.

Many virtual-circuit proponents claim that their networks provide "reliable" VC service with less complexity than datagram networks because they do not "need" an elaborate end-to-end transport protocol. However, many X.75 networks provide NO end-to-end transport protocol at all, and as a result the user is still vulnerable to failures within the network that can lose information or drop connections. In practice, this happens often enough to be annoying. With an end-to-end transport protocol on a VC network, the reliability can approach that of, say, a TCP/IP network, but now the total implementation complexity is greater because of the redundancy at multiple levels.

1.3.5 Grades of Service VC networks are implicitly based on the assumption that *all* applications require a reliable, flow controlled stream "connection." However, there are several real-time[2] applications that either do not require

1. Most power failures are very brief, and if the switch recovers within the abort timeout interval of the end-to-end protocol (or if an alternate route is available), the only effect may be a momentary "freeze" on data transfer, not a dropped connection.
2. "Real time" as used here means that the information being transmitted is useful only for a short time until a

this grade of service or cannot tolerate any overhead introduced by it.

The best example of an application in this category is packet voice. People conversing on a telephone channel are sensitive to long transmission delays, especially if they are irregular. In contrast to data transmission, however, human speech can tolerate a certain amount of lost or corrupted data because of its great redundancy, and "perfect" reliability may be sacrificed to reduce delay.

Other examples of real-time applications might include television ("digital SSTV") and satellite telemetry. In each case, there is little point in retransmitting lost "old" data because "new" information will arrive shortly to take its place. For example, real time satellite telemetry gains little from retransmission of lost frames; the user might as well wait for updated information (and then interpolate the missing values) instead of falling behind by trying to recover data that is already out of date. If a somewhat higher degree of reliability is needed (but the cost of "perfect" reception is too high) the satellite might simply repeat each frame of data several times to increase the chances of successful reception, or use other more complex forms of forward error correction (FEC).

In a datagram network, these kinds of application-specific tradeoffs are easy. For example, control bits in each datagram might select the use of hop-by-hop acknowledgements. In a VC network, however, all applications get hop-by-hop acknowledgments whether they need them or not. Other applications might place different levels of importance on different messages (e.g., emergency vs routine traffic) but because VC networks typically handle traffic on established virtual circuits on a first-come, first-served basis this is difficult to do.

While it may be a while before amateur packet radio networks have the capacity to handle packet voice at a practical level, it would be unwise and shortsighted to adopt a protocol that would effectively *preclude* it from our network. Much could be gained through the joining of resources that could occur if a common network could satisfy the needs of amateur data *and* voice users.

1.3.6 Broadcasting Virtual circuits are inherently point-to-point and usually full-duplex, and thus they do not lend themselves easily to the notion of a "broadcast" message. Sending the same information to N receivers requires that N virtual circuits be created, one to each receiver, and that N copies of the data be transmitted. This is clearly wasteful when the underlying media permits broadcasting (such as Ethernet [10] or radio), and datagrams are a much more natural solution.

Given that reliable delivery to every receiver in a broadcast environment is much more expensive than reliable delivery to a single destination, it is even more appropriate to provide an unguaranteed service. As with simple point-to-point connections, a reliability-improving mechanism appropriate for the specific application would then be implemented on top of basic broadcast datagrams.

Other situations where broadcast mechanisms are useful include the construction and exchange of routing information, and in distributed processing to manage a collection of systems providing a set of services.

As with dynamic routing, datagrams do not, in themselves, solve every problem involved in broadcasting, but they do not *preclude* it outright as do virtual circuit networks.

"deadline" is reached. Information arriving after the deadline is useless, even if it is received correctly.

2. What is TCP/IP?

The ARPA Transmission Control Protocol (TCP) [3] and the ARPA Internet Protocol (IP) [1] are part of a larger collection of protocols that enjoy widespread and rapidly growing usage within numerous commercial, research and military computer networks.

Before delving into the internals of these two protocols, it is necessary to understand the needs of their developers and environment where they were designed.

The ARPA research community that designed TCP/IP is a *user* (as opposed to a *vendor*) of computer hardware and communications facilities, and this had a major impact on its design. [9] A basic requirement was the interconnection of many dissimilar types of computers, using the widest possible variety of link-level networking hardware and protocols. This was important for two reasons: first, much of the hardware already existed and couldn't be thrown away. Second, the user community wanted a "hardware independent" protocol to guard against becoming "locked in" to any one vendor's products. This is in contrast to a vendor's usual incentive to establish standards that favor the use of ones' own products over those of the competition.

It was found that the networks in use vary radically in their characteristics. Some support the concept of "connections"; others only provide unguaranteed delivery. All vary widely in reliability, transmission speeds and addressing formats. The datagram was the only feasible choice as the "common unit" of transmission that could be "encapsulated" on each of these heterogeneous networks.

Other ARPA requirements included robustness in the face of internal network failures and reconfigurations, provisions for precedence, class-of-service and security classification, and optional user specified routing. Because no existing protocols satisfied these requirements (including CCITT X.25/X.75), it was necessary to design a new set of protocols.

The widespread acceptance of TCP/IP outside the military community that originally sponsored its design shows its success in meeting the needs of a wide variety of users, not just those of the military. The latest available figures show that address assignments have been made to a total of over 3,000 distinct networks of varying sizes. About half of this total represent Defense and Government-sponsored research organizations that are interconnected to form the ARPA Internet, while the rest are independent private (mostly commercial) networks. While the exact total number of hosts that support the Internet protocols is unknown, the ARPA Internet host file currently contains 1,146 hosts, ranging from IBM PCs to large timesharing systems.

Planning activities within the International Standards Organization (ISO) now include the design of a protocol based on TCP/IP (TP4), although the CCITT seems to remain adamantly opposed to this type of protocol.

In the following sections, I will discuss the major features of the ARPA IP and TCP protocols. In general, the statements made earlier about datagram protocols apply to TCP/IP. In addition I will point out some differences between TCP/IP and X.25/X.75 that are specific to those protocols and not necessarily related to the datagram/virtual circuit selection.

2.1 The Internet Protocol (IP)

The Internet Protocol (IP) occupies level 3, the network layer, in the ARPA protocol "suite." As its name implies, IP is the "universal language" of the network; it is the "Esperanto" of a larger network built up through the interconnection of many smaller, heterogeneous networks.

IP is a datagram protocol that makes minimal assumptions about the links it uses. IP headers contain only that information necessary to provide network functions such as addressing, classes of service, precedence, etc. In particular, there are no end-to-end features such as guaranteed delivery, flow control, sequencing, or other services commonly found in virtual circuit protocols. As a result, IP is simple and easy to implement on a wide variety of networks, including many that cannot directly support virtual circuits. Intermediate relay points (hereafter called "gateways") only need implement IP in addition to whatever link level protocols are being used; any end-to-end functions remain the domain of higher level protocols in the user systems.

The maximum size of an IP datagram is 65,536 bytes. Since many (most?) networks cannot handle such large packets, IP provides a feature called *fragmentation*. This allows a gateway faced with a datagram that won't "fit" into a given link level protocol to split it into several smaller datagrams that will. Each "fragment" behaves like a separate datagram in its own right and will propagate independently through the network. Only when they arrive at their destination will they be "reassembled" into the original, larger datagram and passed to the next layer protocol. As we will see later, this is an important feature that eases the use of IP on top of AX.25 Level 2. [12]

Each IP datagram contains a "Time To Live" (TTL) field that is decremented as the datagram propagates through the network. If the TTL reaches zero before the datagram is delivered, it is destroyed. Of course, the TTL field is set to a large enough value so that the datagram is likely to reach its destination; however, if a transient routing loop occurs in the network the datagram will not circulate indefinitely. Even if the routing loop eventually disappears, the TTL field protects higher level protocols by establishing the maximum interval when they must guard against duplicates of a particular datagram. The TTL feature provides backup protection against buffer deadlock by ensuring that a datagram never remains in the network indefinitely. A colorful (although admittedly impractical) analogy might be the placing of a time bomb in each car entering New York City. Normally, the cars leave the city in plenty of time, and there are few (or no) explosions. If gridlock occurs, however, even in the absence of any other actions taken the problem is guaranteed to go away eventually by itself!

Several features of IP are not used frequently enough to justify their inclusion in every datagram. These "IP options" are listed in [2], but the most interesting ones include:

1. Source routing. Normally, gateways do their own routing, but two forms of user ("source") specified routing are available. One, "strict source routing" specifies the exact path that must be used by the datagram; if this path is invalid, the datagram is discarded and a failure report is sent to the user. The other form, "loose source routing" allows the user to only partially specify the route; the gateways are free to determine paths between each user-specified point.
2. The "record route" option asks the gateways to route the datagram automatically, but to record in the datagram the path used.
3. Related to the "record route" option is the "Internet Timestamp" option. This option requests that each gateway record the time when it processed the datagram.

Most datagrams are sent without any of these options. However, they are extremely useful for special functions such as testing or collecting statistics about specific paths within the network. It should also be pointed out that X.25/X.75 provides none of these features. If they are considered necessary for amateur packet radio they would have to be added to those protocols.

A special "protocol" called ICMP (Internet Control Message Protocol) is considered an integral part of IP. ICMP is simply a standard way for an IP gateway to send a report back to the originator of a datagram when some unrecoverable error occurs. Gateways are required to generate ICMP messages whenever a datagram must be dropped for any reason, with the sole exception that ICMP messages are never generated about other ICMP messages (to avoid endless loops). ICMP messages report such error situations as invalid IP header formats, unreachable destinations, buffer congestion, time-to-live fields expiring, etc.

Other ICMP messages are of a more advisory nature. The "Redirect" ICMP message is used to notify a host's routing algorithm that an alternate gateway is a more optimal path to a given destination. There is also an ICMP "echo/echo reply" message pair that allows a host to monitor network performance by "pinging" echo requests off selected destinations, perhaps using specified routes.

2.2 The Transmission Control Protocol (TCP)

TCP is the standard ARPA Level 4 (Transport) protocol. TCP, and not IP, provides optional end-to-end "virtual circuit" service to applications that require it. Consistent with the concept of a datagram network, TCP resides only at the endpoints of the connection (typically in the computers containing the application programs) and not in the intermediate gateways. TCP takes the (potentially) unreliable service provided by IP and provides a reliable stream. Therefore it must resequence datagrams that have been delivered out of sequence by the network, detect and discard duplicate datagrams generated by the network, and request retransmissions when data is lost altogether.

While internal details of TCP are beyond the scope of this paper (see [3] for the formal specification of TCP), several of its key features can be mentioned here.

One is that each character of data has its own sequence number. This doesn't mean that TCP sends single-character packets; TCP "segments" (i.e., the datagrams that it sends by way of IP) can be of any length up to a limit negotiated by the TCP "maximum segment size" option. However, in contrast to X.25, receiver acknowledgments show exactly how many bytes of buffer space (the "window") are available.

In X.25 level 3, sequence numbers refer to "packets" that could be of any size from 1 to, say, 256 bytes. The receiver's "vocabulary" for flow control is limited to two phrases: "receiver ready (RR)" and "receiver not ready (RNR)." How MUCH the receiver is actually ready to accept when it says "RR" must be agreed on in advance and specified to the protocols. The maximum is 7 with standard sequence numbering, while a typical value is 2 packets. This means that the receiver cannot confidently indicate that it is ready to receive any data at all unless it has at least enough buffer space for two full-size (256 byte) packets. At the other end of the circuit, the sender may send no more than two (in our typical example) packets, even if each of these packets contains only a single character. This obviously limits efficient buffer utilization, and can severely limit throughput as well.

2.3 AX.25 and Digipeaters: A Poor Man's TCP/IP?

Those who have followed the development of AX.25 Level 2, particularly the digipeater feature, may have noticed a certain peculiar similarity to the way things are done under TCP/IP.

What we call "AX.25 Level 2" is, in fact, composed of two distinct "sub-layers." The upper of these two sub-protocols is the familiar connection-oriented, end-to-end

byte stream protocol essentially identical to X.25 LAPB. However, the essential changes that were made to X.25 LAPB to form what is now known as AX.25 Level 2 consisted entirely of inserting a *datagram* layer below LAPB! Every packet contains full source and destination addresses, the touchstone of a datagram protocol. In addition, should digipeaters be used, each packet contains a "strict source route," in IP terminology.

When AX.25 is used directly between two points (i.e., no digipeaters are used) it looks reasonably like an ordinary link level protocol. However, when digipeaters are used, the virtual-circuit level of AX.25 (the transmission of connect requests, sequence numbers, etc) is "promoted" to serve as an end-to-end *transport* protocol analogous to TCP. The digipeater is, in fact, nothing more than a *datagram* based packet switch, although it is very simple because it can't do routing.

Nor would automatic routing by digipeaters be desirable, since LAPB, which was intended solely as a *link* level protocol, does not make a very good *transport* protocol. For example, it is totally confused by packets that arrive out of order or duplicated, events that inevitably occur occasionally in datagram networks with automatic routing mechanisms, but not on the point-to-point links for which it was designed. There are also situations where information can be lost in LAPB requiring recovery by higher level protocols; this is unacceptable behavior by a transport protocol, the user's last defense against data corruption.

However, AX.25 *without* digipeaters is entirely suitable as a link level mechanism for relaying IP datagrams from one packet switch to another, and it can play an important synergetic role here. A major problem with our existing ad-hoc digipeater networks is the lack of hop-by-hop acknowledgements. If a packet in transit down a chain of digipeaters is lost for any reason, the transmission must be restarted back at the source. Even more wasteful is the loss of an acknowledgement in transit, as this requires the retransmission of the data being acknowledged as well as the retransmission of the acknowledgement. If the probability of successful transfer between adjacent digipeaters were high enough, end-to-end retransmission would be rare and would not be much of a performance problem. However, many problems, including the "hidden terminal problem,"[3] poor RF links and channel overloading, cause significant numbers of packets to be lost in real digipeater networks. If IP datagrams were to be sent in "raw" HDLC frames, a long multihop TCP/IP connection would suffer as much from this problem as a long multihop AX.25 connection. However, if an AX.25 link existed between each point of a long path, with the regular acknowledgment mechanism being used to increase the chances of a packet being successfully relayed onward, the efficiency and throughput of our network would increase dramatically. The end-to-end transport functions would instead be handled by TCP, a much more robust protocol specifically intended for this job. Retransmissions by TCP would be quite rare and would occur only when a link failed or became congested within the network.

It is in this way that AX.25, as a layer 2 protocol, and TCP/IP, as layer 4 and 3 protocols, respectively, can complement each other to produce an effective multihop network. The next section deals with the details of consummating such a marriage. Much of it is modeled on an existing standard for the transmission of IP datagrams over Public Data Networks using the X.25 interface. [7]

3. The hidden terminal problem in packet radio occurs when stations interfere with the reception of packets intended for other stations because they are unable to hear (and defer to) the transmission being interfered with.

3. Sending IP Datagrams on AX.25 Links

IP was designed to be easily "enveloped" in a wide variety of link level protocols, and the AX.25 link level is easily capable of supporting it. However, a standard must be established, and several items have to be addressed. A proposed standard is presented here; a summary is contained in Appendix A.

3.1 Protocol ID

The Layer 3 Protocol ID byte immediately follows the control field in AX.25 Level 2. Until version 2.0 of AX.25, only a single value had been defined: hex F0, meaning "no layer 3", i.e., send the packet directly to the terminal. For AX.25 Version 2.0, the Protocol ID byte hex CC has been defined to mean "Internet Protocol."

3.2 Service Mappings

Two types of frames (I and UI) in AX.25 may carry information. I-frames are sent using the full LAPB (connection-oriented) facility, while UI-frames allow access to the underlying datagram sublayer of AX.25 and do not provide guaranteed delivery. How should the "class of service" bits in the IP header control the selection of the frame type used to send the datagram?

Two "class-of-service" bits are relevant, "low delay" and "reliability." A reasonable mapping would seem to be the following: If the "low delay" bit is set (and the others are not), then send the datagram in a UI frame (i.e., do not use per-hop acknowledgements). On the other hand, if the "reliability" bit is set, then use I frames (i.e., use the per-hop acknowledgments of AX.25 level 2 to increase the chances of the datagram being successfully transferred to the next gateway). If neither (or both) of these bits are set, then it is up to the individual gateway whether to use I or UI frames. This choice may be based on local conditions, e.g., experience in the reliability of a given RF link.

It is not clear how the "throughput" bit should be interpreted, so for the time being it should be ignored. In the ARPA Internet, it is used by gateways that must choose between a low delay (but low throughput) terrestrial channel and a high throughput (but large delay) satellite channel in reaching a given destination.

3.3 Fragmentation

The AX.25 Level 2 document specifies that the maximum size of an I-field shall be 256 octets. This means that an IP datagram that is larger than 256 octets must be split into several using IP's fragmentation facility. Since hosts on the ARPANET often send 512 byte datagrams (to reduce header overhead) this facility must be incorporated if our network is to interconnect with non-AX.25 based sites.

3.4 Address Resolution

An IP address is a fixed-size 32 bit field, too small to contain an amateur callsign. Widening this field is out of the question since it would no longer be IP (remember that IP is the basic, universal protocol that is absolutely standard across a wide variety of systems). Nor would it be desirable to use amateur callsigns as IP addresses even if they did fit, but this is a topic relegated to my companion paper, "Addressing and Routing Issues In Amateur Packet Radio."

Therefore, there must be some way to map between the addresses used at the IP level and the addresses (call signs) used at the AX.25 link level. Fortunately, an almost identical problem has already been solved in the ARPA community, that of sending IP datagrams over the Ethernet local area network.

Ethernet link level addresses are 6 bytes long. Unique addresses are programmed by the manufacturer into a ROM on each Ethernet controller, and they cannot be easily changed by the user. After several unsatisfactory ad-hoc kludges, a solution proposed by David Plummer of MIT was widely adopted, and it has worked well. Plummer's Address Resolution Protocol, or "ARP" [6] (not to be confused with "ARPA") has been widely adopted and is general enough to work on other broadcast-type local area networks besides Ethernet (such as packet radio).

ARP works as follows. Whenever a station needs to determine the link layer address (e.g., the Ethernet 6-byte address or the AX.25 Level 2 call sign and sub-station ID) corresponding to a given 32 bit IP address, it broadcasts a special "ARP Request" packet on the channel. The station with the requested address responds with an "ARP Reply" packet containing the desired link level address. Naturally, to avoid having to invoke ARP for every datagram each IP station maintains a cache table of IP to link address correspondences. This table does not have to be large since it will contain only those stations that are "neighbors," i.e., stations to which packets can be directly sent using level 2. Packets addressed to more distant sites will be sent first to a gateway, and it is only the gateway whose link layer address is needed. Entries in the ARP table are aged, occasionally purged, and replaced with the reply to a fresh ARP request to allow for the possibility of network reconfiguration (i.e., stations changing their IP addresses.)

The beauty of ARP is that it works automacally and transparently. The IP layer need not be concerned with linklayer addresses, and there is no need to maintain manually a table of IP and link level addresses. In practice, it is even possible to swap Ethernet boards (and their addresses) between computers without any adverse consequences.

ARPA has already assigned an ARP "hardware type" value of 3 to AX.25 Level 2. ARP request and reply packets will carry AX.25 Level 3 Protocol Identifier CD (hex).

3.5 Addressing and Routing

This a major challenge facing any higher level Amateur Packet Radio protocols, virtual circuit or datagram. Since the purpose of this paper is to argue the case for TCP/IP, I have devoted a companion paper, "Addressing and Routing Issues in Amateur Packet Radio," to this topic as these issues apply equally to an IP or a X.75 network. I will simply mention here that ARPA "Class A" network number 44 has already been assigned to amateur packet radio through the foresight of Hank Magnuski, KA6M. IP addresses are always 32 bits wide; addresses within the ARPA assignment would contain 44 (decimal) in the first 8 bits of the address, and the assignment of the remaining 24 bits is left up to us. This provides the ability to address 16,777,216 individual stations.

4. Conclusions

It is difficult to summarize in just a few words what has been argued about at length by so many people, only a tiny fraction of whom are involved in amateur packet radio. Nevertheless, TCP/IP's proven flexibility and adaptability makes it an extremely attractive candidate for our needs. It already provides virtually every function we need in an amateur packet network and can be adopted exactly as-is, keeping open the possibility of direct interconnection with non-amateur TCP/IP networks.

TCP/IP is ideal for amateur radio, a heterogeneous environment where stations come and go, propagation paths change, satellites rise and set, and users experiment with new applications and transmission schemes unforeseen at present.

In contrast, X.25 and X.75 are much more limited protocols designed for a homogeneous common carrier environment with static network topologies, reliable nodes, point-to-point transmission lines, and a limited set of user services. Many ad-hoc changes would be required if they were to be used on amateur packet radio, creating new, unique and incompatible protocols. Interconnection with non-amateur networks would require protocol conversion gateways, a totally unnecessary complication.

Virtually all non-amateur packet radio systems use TCP/IP (and none whatsoever use X.75, to my knowledge). Furthermore, the fact that amateurs are already using a protocol much like TCP/IP (i.e., AX.25 with digipeaters) gives strong support to the convenience, flexibility and simplicity of this approach. If we adopt TCP/IP, we will be able to tap the enormous amount of experience that has been gained (and made public) with it over its 10+ year evolution. If instead we adapt X.75 and the higher layers of X.25, we will be forced to solve (or endure) many of its deficiencies in an ad-hoc, unique and time consuming way.

5. References

[1] Postel, J., ed., "Internet Protocol - DARPA Internet Program Protocol Specification," RFC 791, USC/Information Sciences Institute, September 1981.

[2] Postel, J., "Internet Control Message Protocol - DARPA Internet Program Protocol Specification," RFC 792, USC/Information Sciences Institute, September 1981.

[3] Postel, J., ed., "Transmission Control Protocol - DARPA Internet Program Protocol Specification," RFC 793, USC/Information Sciences Institute, September 1981.

[4] Postel, J., "Internetwork Protocol Approaches," IEEE Transactions on Communications, April 1980, page 604.

[5] Postel, J., and Reynolds, J., eds. "Assigned Numbers," RFC 923, USC/Information Sciences Institute, September 1981.

[6] Plummer, D., "An Ethernet Address Resolution Protocol or Converting Network Protocol Addresses to 48-bit Ethernet Addresses for Transmission on Ethernet Hardware," RFC 826, MIT-LCS, November 1982.

[7] Korb, John T., "A Standard for the Transmission of IP Datagrams Over Public Data Networks," RFC 877, Purdue University, September 1983.

[8] Padlipsky, M. A., "A Critique of X.25," RFC 874, Mitre Corporation, September 1982.

[9] Padlipsky, M. A., "The Illusion of Vendor Support," RFC 873, Mitre Corporation, September 1982.

[10] "The Ethernet, A Local Area Network: Data Link Layer and Physical Layer Specification," X3T51/80-50, Xerox Corporation, Stamford, CT., October 1980.

[11] Postel, J., "User Datagram Protocol," RFC 768 USC/Information Sciences Institute, August 1980.

[12] "Amateur Radio AX.25 Link Layer Protocol Version 2.0," American Radio Relay League, publication pending.

[13] L. G. Roberts, "The Evolution of Packet Switching," Proc. IEEE, November 1978, page 1307-1313.

[14] Lampson et al., eds., "Distributed Systems - Architecture and Implementation," Springer-Verlag, 1981., chapter 6. Distri

[15] ibid, chapter 19.

[16] Steve Goode, K9NG, personal communications.

6. Appendix A

6.1 Datagram Encapsulation

1. Each datagram shall be sent in a separate AX.25 Level Two I or UI frame. The IP header immediately follows the AX.25 Level 3 Protocol ID byte, which shall be set to CC hex denoting INTERNET PROTOCOL. No other headers are to be used, and the AX.25 Level Two Frame Check Sequence (FCS) immediately follows the last byte of the datagram.

2. The maximum length of an IP datagram, including the IP header, headers for higher level protocol(s), and user data, shall not exceed 256 bytes. The IP fragmentation/reassembly facility shall be used on any datagram larger than this limit. The possibility of increasing the 256 byte fragment size limit is a subject for future study.

   Gateways may choose to fragment datagrams to sizes less than 256 bytes when necessary to increase the chances of successful transfer over links with high bit error rates; however, this should be done as a last resort.

3. The choice between the use of an I or a UI frame for the transmission of a datagram is made by examination of the "Class of Service" bits in the IP header. Datagrams with the "Reliability" bit set to one must be sent via I frames over regular AX.25 Level Two connections. Datagrams with the "Low Delay" bit set to one must be sent in Unnumbered Information (UI) frames. If neither or both bits are set, then each link node may make its own choice between I and UI frames based on local considerations.

   The interpretation of the "Throughput" bit is a subject for future study; in the meantime it should be ignored.

4. Buffer space permitting, each Level Two implementation should be able to accept and process UI frames containing IP datagrams whenever they are received, whether or not a regular Level Two connection exists with the sending station or any other station.

5. AX.25 Level Two connections may be established on demand when needed to transmit datagrams with the reliability bit set, or they may be continuously maintained; this is a local option to be agreed on by the stations concerned. If an AX.25 Level Two connection is to be taken down, each station should make every effort possible to ensure that any outstanding datagrams sent via I frames (i.e., datagrams with the "reliability" bit set) have been sent and acknowledged before going into the disconnected state.

6. There is no effort to maintain Level Two connections corresponding to any end-to-end virtual circuits that may exist at higher protocol levels.

6.2 Address Resolution Protocol

Whenever a station needs to determine the AX.25 Level Two address (i.e., the amateur radio callsign) of another station in its local area corresponding to a given Internet address, it shall use the ARPA Address Resolution Protocol (ARP) as described in RFC 826. The value of the "hardware type" field in an ARP packet has been assigned by ARPA to be 3, meaning "Amateur Radio AX.25".

Each ARP broadcast and reply packet is sent in a separate UI frame immediately after the AX.25 Level 3 Protocol ID byte, which is set to CD hex denoting ADDRESS RESOLUTION.

The contents of the AX.25 Level Two destination field to be used for all broadcast packets (including ARP) shall be "QST" with SSID 0.

---

1 It may be occasionally necessary to use an IP gateway that cannot support the full multiple-connection-oriented AX.25 facilities due to memory limitations. Such gateways should be used only when absolutely necessary, and when the quality of the associated RF links is high enough to provide reasonable reliability without acknowledgments.