AX.25 Transport Layer Drivers for TCP/IP

bY
Mark Sproul, KB2ICI
Tim Hayes, N2KBG

sproul@eos.ap.org
thayes@noc.rutgers.edu

Abstract

Current TCP/IP over packet radio is largely implemented with KA9Q's NOS and its many variations on PCs and NetMac on Macintoshes. NOS was written before good general purpose networking software was available and was written as one monolithic do-everything program. Today's multitasking operating systems have built in networking support and there exists a large base of good server and client software for all platforms (Mats and Windows) that use this networking ability. Because of these things, a large, monolithic program is no longer needed and is actually a severe handicap. By implementing a transport layer AX.25 driver for use with a system's native TCP/IP protocol stack, the existing programs that utilize TCP/IP for Internet activities can be used for amateur packet radio activities. These include many excellent free and shareware programs for both Mats and Windows machines.

Background

NOS was written in 1985 [l] to run under DOS which does not allow multitasking at all. NOS was written as its own multi-threaded environment to implement all of the various functions needed in a network station including mail server, file server, router, user interface for these functions etc. Because DOS is not multitasking, NOS normally requires a dedicated machine to run and NOS is very complicated to set up. By today's graphical user interface standards and multitasking operating system environments, NOS is obsolete. In addition to being obsolete, there are so many different versions of NOS, each one to fix or address different issues, it is impossible to keep track of.

On the Macintosh platform, NetMac was a port of NOS to the Mac operating system [2]. NetMac suffers from many of the same problems as NOS and it has never been up to normal Macintosh standards for user friendliness.

Software Reauirements for a tdcal TCP/IP Dacket radio station

A user participating in an amateur radio TCP/IP network needs software to fulfill a basic set of functions to successfully participate. In the past, most of these functions have been accomplished with NOS. With the advances in the current TCP/IP networking software, the capabilities of NOS have fallen way behind those of most systems with access to the Internet and it will never catch up in its current implementation. If we could utilize existing software packages (see tables below) we would be able to immediately overcome these limitations. In addition, we would not have to reinvent the wheel each time a new network feature became available; software packages created for use on the Internet could be immediately put into service by amateurs over packet radio.

Server software

Function	Mac	Windows
Domain Name server	MIND	
IRC	Chat	
FTP server	FTPd, FTPShare	EMWAC FTPd
Gopher server	FTPd	
ListServer	Macjordomo, AutoShare ListSTAR	MajorDomo (NT)
Mail server	MailShare(AIMS), MailStop, MacPost	EMWAC POP3 EMWAC SMTPd
News Server	311	
WWW server	WebStar, MacHTTP, FTPd	EMWAC, NCSA-HTTP NetSite (NT)

Client software

Function	Mac	Windows
Archie		WSArchie, Prosper0
FTP	Fetch, Anarchic, XferIt	Chameleon, FTP
Gopher	TurboGopher	WSGopher
IRC	ircle	WinTalk, WSIRC
Mail	Eudora, PopMail, BlitzMail lee-mail, MacPost,	Eudora,

	Mail-Connect	
News	NewsWatcher, Nuntuis	WinVN,
	NewsHopper, NewsSucker	
	InterNews	
	Newsview, MacSlurp	
Terminal	NCSA Telnet	NCSA Telnet, EWAN
WAIS	MacWAIS	
www	MacWeb, Mosaic, Netscape	Mosaic, Netscape,

Imdementation

How do we utilize all of this existing software without re-writing it'? We have to write a transport layer for TCP/IP that implements AX.25 Once this is implemented, all of the current TCP/IP client software and server software for Mats and Windows is immediately usable on the packet radio TCP/IP - AX.25 network.

Status

Currently under development by the authors of this paper is an implementation for the Macintosh. It is called MacAX.25 and is a transport layer driver for use with MacTCP. Once finished it will be able to accomplish all of the goals outlined above. The current projection is that it will be finished by the end of 1995 or sooner.

There is also a project to implement an AX.25 TCP/IP driver for Windows, is in progress called EtherAX.

Linux already has an AX.25 driver for use with its TCP/IP stack.

MacAX.25 Design Specifications

SCOPE:

This document describes an AX.25 driver for IP stacks. All AX.25 procedures conform to the ARRL's AX.25 V2.0 specifications

DESIGN SPECS:

- Uses Generic C Code

For easy porting across platforms. In its final form this driver should be cross-platform compatible. Except for routines to pass data back and forth between the IP stack and this drive no code should be OS-specific.

- AX.25 Transport Method
  AX.2SV2 I-J1 frames will be used to send datagrams

+ IP encapsulation
  IP datagrams will be directly inserted into the data field of the AX.25 packet.

  The PID of the AX.25 packet will be set to “Oxcc”

- ARP encapsulation
  ARP datagrams will be directly inserted into the data field of the AX.25 packet.

  The PID of the AX.25 packet will be set to “Oxcd”

- ARP handling on the client system
  This driver should use the native IP stack of the client system for handling ARP information. This means it is necessary to encode AX.25 addresses in a manner that the system can store in its ARP tables.

  This encoding should be direct if possible- the AX.25 address field to be stored should have it’s C and R bits cleared (if set) and all 56 bits stored in the ARP table.

  In the event that the IP stack’s ARP table can not be used to store AX.25 addresses directly they should be encoded and stored as an ethernet MAC address. In this case the 56-bit AX.25 addresses should be encoded into the 48 bit MAC address as follows:

```
  rrrrrrr p cccccc cccccc cccccc cccccc cccccc cccccc ssss
. MSB                                                  LSB
  <---------------------- 48 bit- -----_--_------------->
```

  - The most-significant 7 bits are reserved and should be set to ones.
  - The “p” bit should be set if an AX.2SL2 path is in use for the address. In all other cases it should be cleared.

- The next 6 sets of 6 “c” bits each (36 bits total) contain the letters of the uppercased callsign translated to the offset of the character from an ASCII space. Empty positions should contain an translated ASCII space (ie: 0x00).
- The least significant 4 bits contain the SSID of the address which are not translated in an way.

Any published ARP entries must be published in the encapsulated form.

It is not permissible to publish ARP entries for addresses which have AX.2SL2 paths.

- IP should handle datagram delivery and routing, not this driver

  Simple and complex mechanisms for doing dynamic AX.2SL2 routing of AX.25 encapsulated IP datagrams have been developed. None of these should be implemented or used by this driver. All routing should be handled by IP.

  A mechanism should be implemented to allow user-defined point-to-point paths at the AX.25 level for links beyond the reach of the local transmitter. This may be especially useful when the “local gateway” is a hidden transmitter.

  - This is not routing- a path defines a single static link much as a telephone number defines a dial-up PPP link.

  - The AX.25 driver will be responsible for managing the paths.

  - Any AX.25 path mechanism should be capable of being used by both ARP and IP.

- Plain AX.25 L2 Support

  There will be no support for vanilla AX.25 connections

  A.s connections are not implemented the driver will not be able to accept AX.25 SABM requests. SABM requests from other stations should be handled in accordance with the AX.2SL2V2 spec. (All incoming SABM requests should be immediately replied to with a DM frame in which the Poll/Final bit is set)

  Incoming I frames will be dropped with no response.

Incoming S frames other than those of type SABM will be dropped.

Incoming I-71 frames with PID's other than those for IP and ARP will be dropped.

URLs:
Authors Home pages
http://pilot.njin.net/-msproul
http://www-ns.rutgers.edu/-thayes

AX25 specs
http://www-ns.rutgers.edu/-thayes/ax25
http://www-ns.rutgers.edu/-thayes/ax2S/MacAX25
http://www-ns.rutgers.edu/-thayes/ax2S/MaxAX25/MacAX25spec.html

Mac Software
http://rever.nmsu.edu/-elharo/faq/software.html
http://leuca.med.cornell.edu/Macjordomo

HI *The KA9Q Internet (TCPIIP) Package: A Progress Report,* Phil Karn, KA9Q ARRL 6th Computer Networking Conference, pp 90-94, Redondo Beach, California.

[21 *KA9Q Internet Protocol Package on the Apple iMacintosh,* Dewane Hendricks, WAgDZP, Doug Thorn, N60YU, ARRL 8th Computer Network Conference, Colorado Springs, Colorado, pp 83-91