Adaptation of the KA9Q TCP/IP Package for Standalone Packet Switch Operation

Bdale Garbee, N3EUA
Don Lemley, N4PCR
Milt Heath

Several new hardware systems intended for, or adaptable to, standalone packet switch use have appeared on the market in 1990. *These include the* Grace *Pa&Ten, the Kantronics Data Engine, and the soon to be available AEA PS-186. One obvious use for these systems* is *implementation of a TCPIIP-based amateur packet network. This paper discusses some of the design issues uncovered in porting the KA9Q TCPIIP package to a standalone hardware environment, and will touch on some &tails of the implementations available now (or soon) for each of the mentioned hardware systems.*

Background

For several years, Phil Karn KA9Q has spear-headed an effort to develop software based on the TCP/IP internetworking protocol suite for use on amateur packet radio. The "NET" package has enjoyed increasing popularity in countries worldwide.

As the number of users has grown, it has become increasingly obvious that certain limitations in the base software need to be addressed in order to permit the development of networks. Paramount among these are the use of static routing in most NET installations, and the dependence on end-user nodes to serve a dual function as network routers or gateways.

In some areas, the separation of end user and network gateway (or router) functions has been accomplished by reliance on existing TNC-2 based NET/ROM installations to provide network routing and "reliable" network service. This approach has helped to point out deficiencies in the routing algorithms and implementation used in NET/ROM, and has resulted in very poor performance in the areas where NET/ROM switches are configured in a single-port configuration, against all recommendation. In other areas, existing hardware platforms such as IBM PC clones and Atari ST systems have been pressed into service running the existing, standard NET software. This can be a very functional configuration from the standpoints of hardware availability and cost, software availability, and familiarity to network administrators. It can also be a real hassle, for reasons ranging from the lack of useable network routing and remote system management capabilities, to the seemingly mundane but very annoying issues of temperature range and power availability at switch sites.

The advent of more powerful digital processing hardware targetted for the amateur packet radio environment clearly is an attempt to address some of the problems related to use of existing user hardware platforms. Unfortunately, the new hardware addresses less than half the problem. The KA9Q TCP/IP software must be ported to these platforms, and some of the original software's design limitations, which are quite reasonable in an end user's system but are unworkable for a standalone diskless mountaintop switch, must be addressed and resolved.

Understanding the network model envisioned by the authors may help explain our motivation in developing NOS-based packet switches. In Figure 1, we show the typical "cloud-model" of a network. Attached to this network, and providing access service to several local user bases, are several switches Each switch provides connectivity for zero or more AX.25 "terminal nodes" (as in TNC, or Terminal Node Controller), zero or more TCP/IP-based service providers (larger DOS or Unix machines), and zero or more TCP/IP-based user machines.

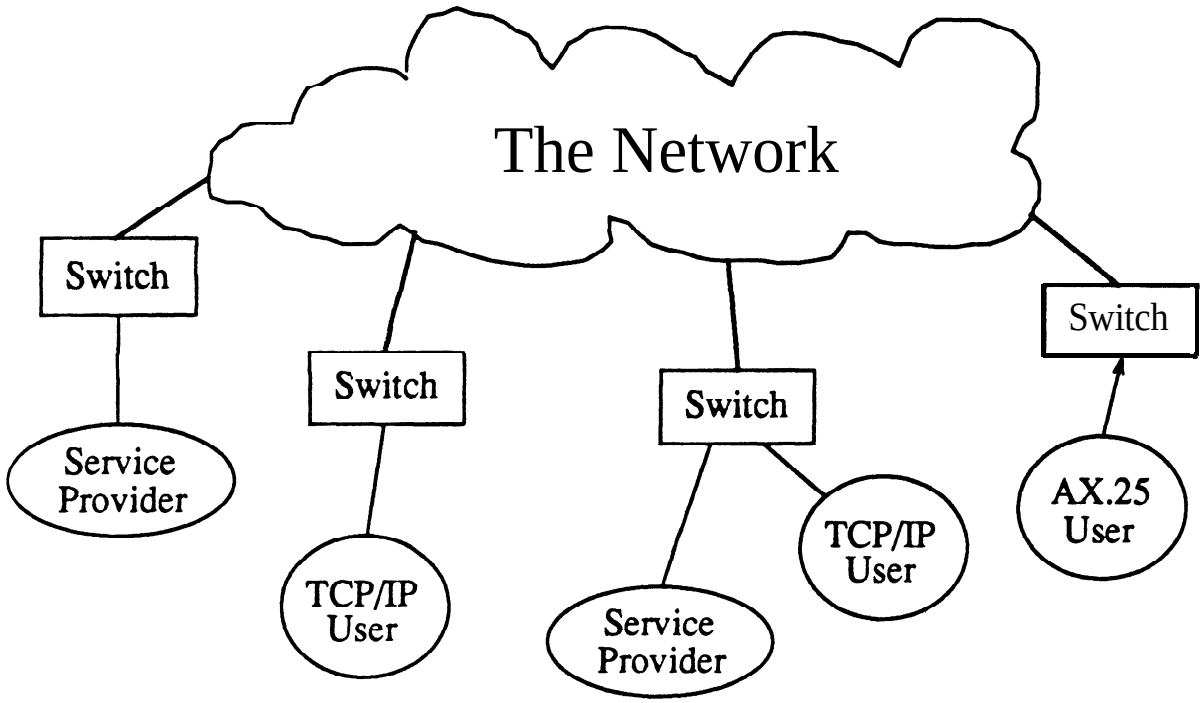

Figure 1: Our Model of an Amateur Packet Network

The most important points about this diagram are that individual users and their machines need know nothing at all about the **topology of "the network",** and that a switch can provide service to any number of AX.25 or TCP/IP users.

For the TCP/IP users, operation is exactly as might be expected. Sessions can be initiated to any number of servers, other systems can initiate sessions to local servers, etc. Adding a new user's host at maximum requires knowledge of the local switch address, and even that may not be required **if user nodes run** automated routing protocols too. Connections are established through the **network by knowing only the address or symbolic name of a remote system, and the service desired on that** system, which could be further simplified by the addition of a location broker on the network.

For an AX.25 user, the local switch appears to be the service provider. A user connects to the local switch with a normal AX.25 connection, **and is presented with a** menu of available services. Upon selecting **a service, a TCP-based connection is automatically** opened from the local switch, across the **network, to a remote service provider.** Services could include the ability to login to **a BBS or Unix machine,** where a full range of options exist. Or, services could be as directed as obtaining the current weather report from the National Weather Service, **looking up a callsign in an electronic callsign** server database, etc. The important point is that AX25 users are fully supported as **network citizens** through the terminal-serving capability of their local switch.

The implementation of KA9Q's NOS TCP/IP software on the platforms discussed in this paper has become commonly known as "NOSINABOX", and **will be referred to by that term for the remainder** of the paper.

Issues

The initial implementation of NOSINABOX has focused on the fundamental changes required in the KA9Q TCP/IP package to allow it to function in an environment without disk drives, operating systems, and direct human intervention. This has involved disabling services that make no sense in a standalone environment, rewriting facilities that are necessary regardless of environment but which depend on resources unavailable to us, and adding new capabilities that are uniquely necessary in a standalone, remote hardware environment.

The NOS software package includes protocol support

for a variety of functions that only make sense in the presence of users, file systems, or both. These include the FTP file transfer protocol client and server, the SMTP electronic mail client and server, and the Finger server among others. Phil provides a compile-time mechanism to eliminate all servers from the package, but additional effort was required to locate and disable code for nonsensical clients.

One of the features of NOS versions of the KA9Q software is the ability to query domain name servers such as BIND for hostname to IP address translation. Unfortunately, the stock name server client provided in NOS assumes the availability of a file system for caching the results of previous name queries. This code required a major rewrite, and included modifications made by PAOGRI and others to the stock NOS implementation.

File system references also appeared in the initial configuration section of the software. In a PC or workstation environment, NOS loads configuration information from a disk file that the user provides. Considerable effort was expended to provide fundamental configuration from ROM in all versions of NOSINABOX. Because we take advantage of special hardware features available on the different systems, this is discussed further in a later section.

The standard NOS software was written assuming that a keyboard and display local to the processor would be used as a "console" for the package, providing a configuration and debugging interface to the software. This is not true in most packet switch environments, where the switch lives on top of a mountain reachable only on snow-shoes much of the year, or high on a tower requiring professional assistance to access. Fortunately, as part of the NOS rewrite, Phil based the console interface to the software on the same "sockets" mechanism used to attach applications to networking services. It was therefore relatively easy to provide a mechanism for remotely attaching to the console over the network. Rudimentary but long-term inadequate security for this facility is provided in the initial release.

Implementation-Specific Issues

Since the author's efforts in developing standalone versions of NOS have been carried out independently due to differences in the hardware platforms involved, and geographic location, the actual feature sets provided by the implemented code are somewhat different. However, considerable effort has been expended to ensure that at least philosophically the versions of NOSINABOX are aligned in such a way as to provide compatability for future development of Dew features. The following paragraphs outline sane of the software features of each NOSINABOX version as of this writing.

Grace PackeTen

Because the Grace Communications PackeTen is based on the Motorola 68302 "Integrated Multi-Protocol Processor" instead of the Intel platform traditionally supported by KA9Q's NOS, much effort was expended porting the code to the new processor. This effort involved both making a standalone version of the code function, and optimizing the software to take advantage of powerful new 68302 features. The resulting version of NOSINABOX has been dubbed "NOS302".

NOS302 is the NOSINABOX version most fully developed at this time. In addition to the feature set neccessary for minimal standalone operation, the NOS302 system provides:

1. Software Watchdog Function

   This feature implements a software sanity check within the NOS package. Since a remote packet switch is by definition not easily accessable, it is imperative that some form of dynamic software integrity test be performed during normal operation. The purpose of this test is to prevent a system which has suffered a software or firmware failure from "going off in the weeds" and becoming nonfunctional forever, requiring some poor soul to reach the site and manually restart the system. This "NOS independent" software "watches" the system software, and if for some reason NOS is not running through it's normal routine, the system is restarted with a hard reset.

2 EEprom Configuration

   Since amateur packet switches are usually inaccessable, they must be capable of surviving a power glitch or outage, rebooting to a known functional state appropriate for the site. Since the PackeTen switch hardware provides non-volatile memory in the form of EEPROM, menu driven software is provided for configuration of such options as IP address, AX25 address, link definitions, and other neccessary site specific information.

3 Attachable / Detachable Console Port

   In order to avoid wasting one of the serial links for use as a local console during operation as a standalone switch, a "Console" device was devel-

oped which can be attached and detached just like all other links. This allows a remote switch to use all links during normal operation, but if on-site service of a node is required, a simple jumper strap will configure one of the ports as a local console for debugging.

4. Scatter Gather Transmit

   This feature involves the use of an advanced 68302 capability to dynamically build a transmit frame "on tbe fly" from a list of buffer pointers. The major advantage of this capability is that the data does not have to first be copied from NOS's "mbuf" chains into a contiguous data frame area before transmission, thereby decreasing processor overhead, and increasing throughput on a link. It is Worth noting that KA9Q's approach to the problem of layered protocol frame construction, namely mbuf chains, fits perfectly with the design of the 68302 serial communications controllers. His design allowed the "scatter gather" feature to be implemented relatively easily, and efficiently.

5. Receive Mode Buffer Chaining

   Receive buffer chaining is another very important feature of the 68302 platform. The 302 serial channels are programmed with a list of available receive buffers. Then they begin to accept incoming frames on the link. Upon completion of a received frame, they simply mark the frame as ready for processing, and then proceed, utilizing the next available receive buffer. All of this functionality is performed with no main processor intervention, or interrupt service required. Since the serial channel's buffer chain list is up to eight (8) buffers deep, the main processor is only required to service a receive interrupt every 8 frames in order to prevent receive data overrun on a link.

6. Software Timer Services Added to NOS.

   It was considered important in the 68302 NOS implementation to be able to handle "multiple" high speed da&a links, without the use of wasteful polling loops for such things as keyup/keydown radio control. Therefore software programmable timer services were added to NOS to allow the programmer flexibility in writing device driver timing routines.

7. Automatic Power Conservation Mode

   One of the features of the MC68302 processor is the ability to cut it's power consumption from a typical 60 ma down as bw as 1 ma in it's lowest power mode. The current NOS302 release 1.2 implementation supports this type of operation, allowing consumption to be reduced by approximately half, to around 30 ma, during idle periods between received packets.

Kantronics DE and AEA PS-186

NOSINABOX for the Kantronics Data Engine and AEA PS-186 is under development at the time of this writing. Because the processor used (V40 and 80C186, respectively) and I/O hardware configuration (V40 DMA to 8530, and NEC DMA to 8530, respectively) are so similar, the two platforms are being treated as functionally equivalent, differing only in the number of ports (2 or 4) that are provided in addition to the dedicated console port.

Despite the inclusion of battery-backed CMOS RAM capability on both of these systems, the initial release of NOSINABOX will probably include a configuration utility intended to customize the ROM images to reflect configuration information for a given site. Configuration changes will of course be possible after boot from the system consoles.

The initial release also will not take advantage of the DMA capabilities of these systems. Because the DMA controllers used do not provide a "scatter gather" chaining mode identical to the 68302 version, additional effort will be required to interface the NOS mbuf handling to DMA buffers for each port. For this reason, initial performance of these two platforms in interrupt-driven operation will be far from spectacular.

Both the AEA and Kantronics platforms include hardware watchdog functionality, and NOSINABOX will use this functionality to force a hardware reset after software crashes.

Future Directions

Once fundamental hardware-specific issues are resolved, the authors hope to integrate the two currently divergent implementations of NOS for standalone hardware into a single NOSINABOX environment for multiple hardware platforms. It is highly likely that the Grace PackeTen system will always be "one step ahead", since it is a commercial product with TCP/IP protocol support provided by the manufacturer. The TCP/IP support for the AEA and Kantronics products is currently available due to entirely volunteer effort.

A growing concern in the commercial and educational network markets that is or will soon be mirrored in the amateur packet environment is the issue of network monitoring and management. The SNMP Simple Network Management Protocol has recently come into wide use as the mechanism of choice for interrogating and controlling network entities. Several vendors are now producing slick, graphically-oriented user interfaces for network management that rely on the SNMP protocol for communication with diverse networking hardware systems from multiple manufacturers. The effort required to implement at least a subset of SNMP target functionality seems quite acceptable, and it is highly likely that we will add SNMP support in a future release of NOSINABOX.

Considerable room for experimentation exists in the area of dynamic route determination. Fred Goldstein, K1IO, has drafted a specification for a protocol called RSPF, or Radio Shortest Path First. A preliminary implementation for NOS was written some months ago by Anders Klemets, SMORGV, and we are currently investigating including the RSPF code in the next NOSXNABOX release. RSPF has several features that make it an attractive potential alternative to the routing provided by NET/ROM and the RIP protocol currently included in NOS. One of the keys to automatic, dynamic route determination on radio links is the issue of determining link quality, and this is an area that will no doubt be addressed at great length over time.

Currently, a very simplistic approach is used to "secure" the remote system administration facility. An alternative that intrigues the authors is the adoption of an IP-level option to cypher-checksum each frame sent on a secure connection. This approach involves encrypting each packet as it is assembled for transmission, calculating the checksum of the encrypted packet., and transmitting it along with the plaintext packet and normal data corruption checksum. By performing the same process on incoming packets, a receiver could know positively whether the sender knew a pre-negotiated key, thereby validating control packets as coming from an authorized control station. This is a clearly legal use of encryption for authentification purposes accepted by the FCC, with no restrictions since the actual data content of each packet is sent in cleartext. An implementation of this option, and a requirement for use of the option for remote sysop functions, would be a very secure way to control access to the configuration of a switch site.

Each of the hardware platforms discussed in this paper include some mechanism for the inclusion of custom, "expansion" hardware. One possible use of this facility would be to drive hardware interfaces for telemetry and control. It might be much easier to obtain permission to co-locate a switch with an existing voice repeater facility if added value could be offered in the farm of a secure remote control and monitoring station for repeater equipment, power equipment, or weather data collection hardware, as examples. This area of functionality will be investigated for future releases of NOSINABOX, either as support for specific future hardware add-ons to the existing platforms, or perhaps as a generic interface for driving custom hardware at the I/O address and data level.

The exact performance of NOSINABOX on all of these hardware platforms is not yet known. However, the PackeTen has as of this writing been placed in service running 56 kb/s fullduplex radio links as well as having been tested with 2 megabit/s full-duplex wire-line links. In one configuration it has simultaneously supported six (6) 56 kb/s links, two (2) links at 9600 and two (2) others at 1200 baud. As more NOSINABOX implementations become available and are placed into service, the additional performance information obtained will be used to further redesign and optimize the internal data handling algorithms, thus providing additional performance improvements.

We do not currently fully understand all of the relationships of hardware architecture, processor family, and protocol &sign on the real aggregate throughput of NOSINABOX packet switches. Gaining and documenting additional knowledge in this area will help potential customers make intelligent decisions about which switch to buy for a given environment, in addition to fueling continued performance enhancements.

Different approaches to the "terminal server" functionality provided by NOSINABOX have been proposed, including a suggestion by WB6RQN that AX.25 users be treated to an automatic, blind "rlogin" connection a local Unix server when a connection to the local switch is established. These ideas deserve further investigation, and it is likely that a future release of NOSINABOX will provide support for multiple approaches to solving the continuing problem of AX.25 support in a computer-to computer networking environment, with some mechanism provided to configure the mechanism desired for a given local area

Conclusion

This paper has discussed issues related to use of the KA9Q "NOS" TCP/IP software in a standalone, amateur packet radio packet switch environment. Current status and future directions of the authors' efforts in this area have been reported.

For continuing progress reports and other information about the effort to develop intelligent packet switch systems, the reader is directed to "Packet Status Register", the newsletter of TAPR, the Tucson *Amateur* Packet *Radio organization.* The authors may be reached by a variety of means, listed below.

We strongly encourage you to provide feedback on the ideas expressed in this paper, and encourage you to acquire and experiment with one or more of the packet switch platforms discussed, and report your experiences to us

Contact Information

Bdale Garbee, N3EUA
4390 Darr Circle
Colorado Springs, CO 80908
bdale@col.hp.com
CIS:76430,3323

Don Lemley
623 Palace Street
Aurora, IL 60506
donl%ldhmi.uucp@col.hp.com
CIS 73230,310

Milt Heath
75 south chestnut street
Aurora, IL 60506
milt%ldhmi.uucp@col.hp.com