RADIOSERVER - A PACKAGE FOR TNC ACCESS TO A LAN IN A UNIX SYSTEM

A. Giordano (I1TD), S. Zappatore (IW1PTR)

Department of Communications, Computer and Systems Science (DIST),
University of Genoa (IK 1 HLF)

Via Opera Pia, 11A - 16145 Genova, Italy

ABSTRACT

A software package is described that interfaces a Terminal Node Controller (TNC) to a computer (host) running the 4.2 BSD Unix Operating System, over a serial line. An OM-shell is opened for every incoming AX.25 connection to the TNC so that the remote user needs only a terminal to log in a HAM devoted environment. Some utilities are then offered by the OM-shell and among others the possibility to issue telnet-like commands to the hosts and gateways of the Department LAN.

WHY WE BUILT IT

The increasing appeal of the KA9Q TCP/IP package is mainly due to the capability of performing Computer Networking in place of the primitive Terminal Networking. The implementation of the major ARPA intemet protocols over IP in this package allows AX.25 users to connect to remote hosts over a wide number of computer networks by performing end to end automatic functions.

Also in our country there is a trend to encourage strongly the use of this protocol, and IP reserved channels and repeaters are already in use. Unfortunately, the channel load resulting from higher level protocols like this may be high, and for some applications over a low speed (i.e., 1200 baud) channel it may become overwhelming.
We refer for example to a popular application like a remote-terminal session that is established with the telnet command ın the ARPA intemet protocol suite.

Some of us, OM and at the same time researchers at DIST, like to have the possibility to log in hosts of the Department LAN over the radio channel, and the immediate solution was to connect a KA9Q gateway between the TNC and the Ethernet cable.
It was easy to note that the 1200 baud speed allowed by the UHF voice transceiver was only a dream... In fact, some problems arise from this easy solution. The main one is due to the difference in bandwidth between the two networks. The host drives a high speed (Ethernet) line and for every message it sends, it expects to receive an ack for a certain time. This time is short if compared with the time needed for the radio channel to route the message to the remote station (the radio channel at this speed is about ten thousand times slower than Ethernet).

For this reason, thc host sends several times the same messagc always expecting an ack and stopping the turnover of the half-duplex channel. Only when the sequence of repeated messages ends, the radio channel reverses its direction and the ack sequence is eventually received from the host. This fact was already noted in [1].
Just sometimes, when the Ethernet traffic is high, the delay between two successive retransmissions allows the turnover of the radio channel, and this expedites the exchange. Only for few implementations of TCP, once the connection has been established, the system on the Ethernet side learns its correct timeout.

The solution would be to change the timing of the various ARPA intemet protocols (ICMP, UDP, TCP...) by every host: in presence of packets routed to the radio gateway, the equivalent of the AX.25 FRACK time should be appropriately set. This may be cumbersome or impossible on a wide LAN. On the other way, what we needed was merely an error free connection, and so we decided the AX.25 protocol was enough for a comfortable remote-terminal session”

WHAT THE SOLUTION

The idea was then to write a package, running on a host (specifically a Microvax II) connected to the Department LAN, capable of managing the TNC through a RS232 serial port and offering the possibility to open something like a shell forthe terminal active on the other side of the radio channel.

We got our goal by implementing a Radioserver, supported by the Unix 4.2 BSD Operating System, capable of managing multiple connections between the TNC and remote stations. The user interface is a shell offering the possibility to issue some commands to the host.

We never forget we are firstly OM, and so an “OM-shell” was created to **satisfy** the local community more interested to common HAM applications rather than to the direct access to the Unix resources.

The package was provided with some administrative and statistic facilities like user registration, command history and activity tracing. Mail box, database of software utilities, BBS service and similar are then supported by the OM-shell. Also a game session is provided.

For the hardware support, an AEA TNC was chosen for its possibility to dialogue with the host in “host-mode”, a non verbose interface providing greater direct control over the TNC with a communication based on character blocks [2].
In this way, we could simplify the transfer and subsequent encoding and decoding of commands and informations. The host mode lets the TNC fully manage the AX.25 protocol (particularly retransmission for error recovery), so we did not need to implement it in our package.

The server is made up by concurrent processes communicating with “sockets”. The socket, that looks like a file shared between different processes, is the basic building block within Unix communications domains. Domain is an abstraction introduced to bundle common properties of communicating processes [3].
The socket is proper of the BSD Unix version and it offers an easy tool for real interprocess communication.

HOW THE PROGRAM LOOKS

For a concise description of the package we refer to figure 1. The following processes are always active: The Supervisor, the RS232_reader, the RS232_writer, the PTIP daemon. The others are activated upon an **incoming** call.

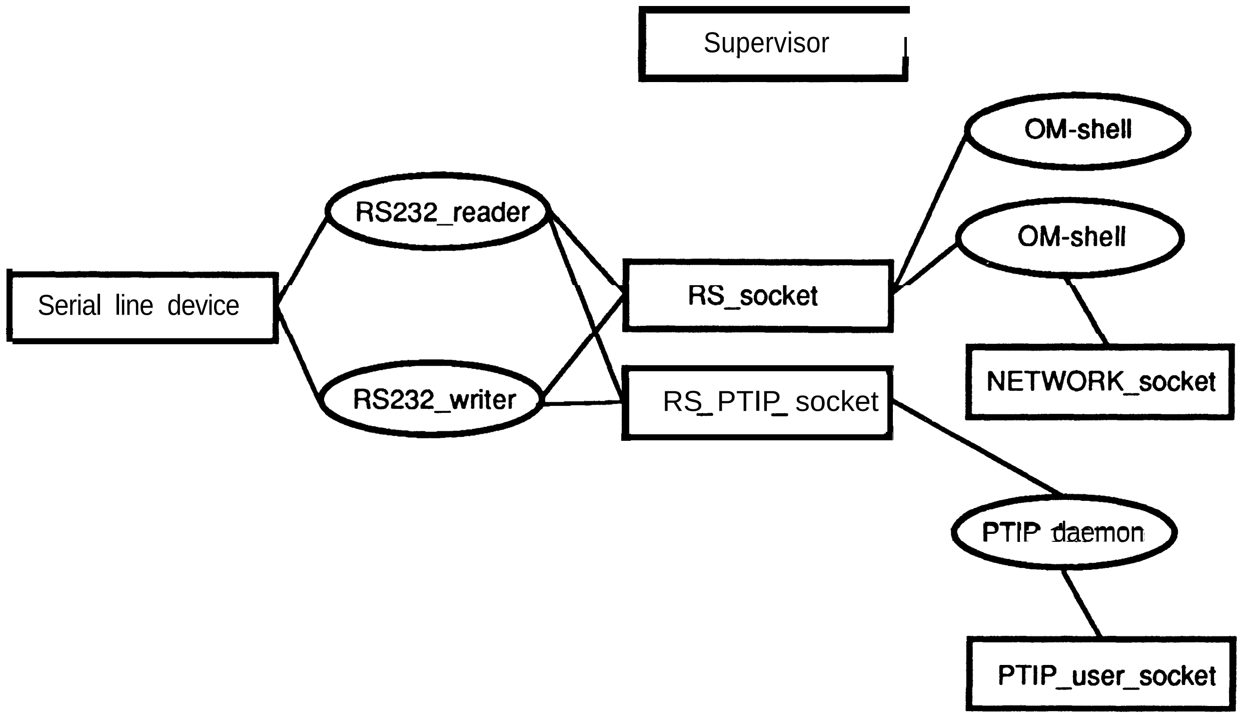

Fig. 1

The **RS232_reader** is interfaced to the input port device (RS232 serial interface) and it disassembles the received character blocks out of the "host mode" format. This process also ensures transparency by unstuffing of the DLE characters. The incoming packets are then placed in the RS_ socket or in the RS_PTIP socket according to the stream number. (The AEA TNC accepts a maximum of 10 users, i.e. 10 stream numbers).

The **RS232_writer** acts in the same way, but in the other direction. It reads from the RS socket or from the RS P_TIP s_ocket data structures and assembles the host format packets to the TNC.

The Supervisor is the parent originating all other processes; after initialization, it keeps a record of the active connections in a connection table, manages the packets containing control data, and kills the child processes related to disconnected links.

Besides the above mentioned services, the OM-shell may connect the RS_socket with a NETWORK-socket. In other words, the OM- shell may activate the gateway function between AX.25 and the LAN. Every connected user disposes of a OM-shell.

In this way, it is possible for the authorized users to connect to the desired host with a telnet-like command. After this connection, the user holds the Unix shell of the connected host.

A user program can access the Radioserver using the PTIP USER socket: a function library has been developed containing primitives that -are -useful to manage the data exchange between the user and the PTIP daemon process. Thus, the user program is allowed to manage the TNC in the issuing of outgoing connections (some of the virtual channels of the TNC are reserved for outgoing connections for the possibility of automatic mail forwarding).
With this library, several utilities have been implemented. Among these are mail forwarding with WORLI rules, and a terminal emulator that allows an authorized user logged on a LAN's host to access the TNC as if it were in the normal mode. It is so possible to connect AX.25 stations operating on the radio channel, to monitor the channel, and to temporarily change the TNC parameters, disposing of every TNC command.

The implementation of further utilities like file transfer commands or others was not justified because the superiority of those existing in the KA9Q software. Actually, even the implementation of this telnet-like utility might appear needless, but the package has been now "on the air" for two years and many amateurs are using it daily. More than 5000 connections at this time demonstrate some usefulness of this work.

CONCLUSIONS

An interface between a TNC and Unix environment has been described. The aim was not to substitute the use of existing network IP based protocols but to offer an easy tool to communicate with a computer running the Unix O.S. using a radio channel and AX.25. At this level, a set of selected utilities proper of the Unix environment are then easily accessible.

REFERENCES

[1] Clifford Neuman, N1DMM, "Packet Radio and IP for the UNIX Operating System" ARRL 6th Computer Networking Conference, Redondo Beach, California, August 29, 1987.

[2] Advanced Electronic Applications, INC., "Technical Reference Manual Model PK-232 Data Controller".

[3] S.J. Leffler, R.S. Fabry, and W.N. Joy, "A 4.2BSD Interprocess Communication Primer" Draft of August 3 1, 1984, Computer Systems Research Group Department of Electrical Engineering and Computer Science, University of California, Berkeley.