JNOS

and

JNOS40

COMMANDS MANUAL

(Document ID: JN_CMD02)

(C)1993 Johan. K. Reinalda, WG7J
and
Douglas E. Thompson, WG0B

for
JNOS release 1.10
February 28, 1994
&
JNOS40 release 1.00
February 28, 1994

(based in part on the NOS Reference Manual,
by Phil Karn, KA9Q
and
Gerard van der Grinten, PA0GRI)

TABLE OF CONTENTS (cont)

This document combines the JNOS and JNOS40 command sets into a single document. This manual is current as of the versions indicated on the title page. Prior versions of commands or commands which have been deleted are not included. Also, this document addresses only the commands that are available in the WG7J distribution compile (uses distconf.h) Information about commands for other modules which are not included in the "standard" WG7J compile may be found in the "history" or "readme" files.
There are now two manuals for JNOS and JNOS40:
The COMMANDS Manual (this document)
JNOS40 Configuration Guide

The user interface for JNOS is very similar to most of the well-known BBS programs. The user interface for JNOS40 is very similar to the G8BPQ, TheNet and NetRom based nodes. This is to provide an easy transition for users. The user interface in the JNOS program is commonly called the 'mailbox'. Since JNOS40 doesn't provide any mail services, the terms 'node shell' or simply 'node' seem more appropriate. These terms will be used interchangeably throughout this document when describing JNOS40 commands and will appear in descriptions of JNOS commands which are common with JNOS40. The JNOS40 commands use the same syntax as JNOS as much as possible to allow ease of movement for sysops working on both types of systems. This Commands Manual contains the commands and their descriptions for using and operating a JNOS or JNOS40 tcp/ip and ax.25 packet switch, BBS, and network node. Also contained in this manual is information about the FORWARD.BBS, REWRITE, ALIAS, and USERS files used in a JNOS installation. There is substantial information of use to anyone operating packet station using tcp/ip in the "JNOS40 Configuration Guide" as well as in the History and README.NOW files. You are encouraged to obtain those files for reference purposes.
Questions, remarks and suggestions about JNOS and JNOS40 are welcome and should be sent to:

Johan Reinalda, WG7J/PA3DIS
420 NW 9th
Corvallis, OR 97330
U.S.A.

email: johan@ece.orst.edu (or wg7j@wg7j.ampr.org.)
(or the sloooower WG7J@WG7J.OR.USA.NA via packet)

or for the documentation only:

Doug Thompson, WG0B
PO Box 21108
Wichita, KS 67208-7108

email: wg0b@delphi.com (or thompsond@awsil5.boeing.com)
or packet wg0b@k0hyd.#scks.ks.usa.na

Corrections (and comments) to the documentation must include the following information:
1) Document ID (See the Title Page)

2) Page Number

3) Text as it exists
This does not have to be the complete text. But it must be enough to ensure unambiguous identification of the area under discussion.

4) Text as it is proposed to be or an explanation of the problem which I will convert into appropriate text.

DO NOT send a copy of the whole document with revisions scattered throughout. I have neither the time nor the inclination to wade through that much text.

Send the corrections to WG0B at one of the addresses on the preceding page. If it comes to the PO Box, please ensure you send it on floppy disk, IBM format.

The documents have been prepared using Microsoft Word Version 5.0. Submittals using MS Word 4.0 or 5.0 format, plain ASCII text, or Rich Text Format (RTF) (supported by WordPerfect) are all easily handled

Here are some of the abbreviations and terminology used throughout this manual.

HOSTNAME is the tcp/ip name of a computer or packet system.

INTERNET is a worldwide high speed computer network. It has thousands of computers at schools, companies and amateur packet radio systems connected to it.

MTU, or Maximum Transmission Unit, is the maximum data size in one packet. Most often the data referred to by MTU is the transported data, i.e. data frame in a network connection. With tcp/ip, the size of the tcp/ip frame inside the ax.25 packet is the MTU; with net/rom, the size of the data inside the netrom packet is the MTU.

NRS, or Net/Rom Serial protocol, is what TNCs with Net/Rom or TheNet eproms talk on the serial port.

NODE, NODESHELL, MAILBOX are terms used interchangeably for the user interface when connected to the system.

PACLEN, or packet length, is most often used to refer to data size in a link packet. The data in an ax.25 packet can be up to paclen bytes.

PORT or INTERFACE means the physical connection to a radio or other system (i.e. radio port or serial interface). These two terms are used interchangeably.

RFCs, or Requests For Comment, are standard papers used on Internet to discuss and propose new networking protocols and other related topics.

RSPF, or Radio Shortest Path First, is a tcp/ip routing protocol especially targetted at radio environments.

RTT, or Round Trip Time, indicates the time needed for data to be sent and acknowledged.

SLIP, or Serial Line IP, is a way to send IP frames over a serial port without using ax.25 or ethernet to carry the data. You can use SLIP to connect to PCs or Unix systems also running SLIP, and interchange tcp/ip data.

There are several command line options which can be exercised when starting JNOS. These commands are used to set environment variables, select configuration and autoexec files, and other functions.
Options should be separated by tabs or spaces. If there is an option argument, there should NOT be any whitespace between the option and the argument. The only option not preceded by '-' is the alternate startup file

```
     -b          : Use direct video for the screen output.
     -c#         : Set the number of COLUMNS on the screen to #.
     -drootdir   : Set the root directory for the configuration file
                 :  path.  This is overwritten by the file specified
                 :  in the -f option configuration file.
     -e          : Pause after each error line in autoexec.nos
     -ffile.cfg  : Set JNOS configuration file and path names as
                 :  indicated in the file 'file.cfg'
     -l          : Do NOT delete .lck files on startup
     -mn         : Set the default screen swap mode.
                 :   n = 0 Use EMS  (If compiled in and available.)
                 :           Default is EMS Available
                 :   n = 2 Use memory.  (Default if NO EMS
     available.)
                 :   n = 3 Use a temporary disk file
     -n          : No trace session
     -r#         : Set the number of ROWS on the screen to #
     -s#         : Set the number of sockets available (Default is
     40)
     -t          : trace the autoexec.nos file. You will be asked
                 : before each if you want to execute it. 'y'
     accepts,
                 : anything else skips the line.
     -v          : Verbose.  Print each line from autoexec.nos before
                 :   parsing.
     -u#         : Set the number of status lines. Valid values 0-3
     -wf+b       : Set foreground/background colors for system status
                 :     default: white on magenta
     -xf+b       : Set foreground/background colors for session
     status
                 :     default: white on blue
     -yf+b       : Set foreground/background colors for 'main' window
                 :     default: the system colors when jnos starts.
                 :     (most often lightgray on black) :
     -zf+b       : Set foreground/background colors for 'split'
     window
                 :     default: white on green
                 : 0=black
                 : 1=blue
                 : 2=green
```

```
             : 3=cyan
             : 4=red
             : 5=magenta
             : 6=brown
             : 7=white/gray

autoexec.new: Name of the startup file. Default is
'autoexec.nos'
             :  or as set with -fnos.cfg
```

JNOS now has a (up to) 3-line status display which shows:

On the first line:

time, heap and core free memory, number of connections to the different servers. Then a list of active sessions, where sessions with data waiting are blinking.

The 2nd line shows:

the users connected to the bbs. A status symbol in front means one of the following:

```
  none - user is idle
     * - user is a bbs
     @ - user is in sysop mode
     ! - user has gatewayed out
     # - user is reading or sending mail
     = - user is transfering data (up/download)
     ^ - user is in convers or sysop-chat mode
     ? - use is in none of the above, but not idle...

  Eg: BBS: *w0rli johan #ka7ehk !n7ifj
```

(lines 1 and 2 are the 'system' window)

On the 3rd line is data depending on the current session.

Always displayed are the current session number and type.

If the sessions are network connections, displayed are remote connection name, tx-queue (bytes for tcp, packets for ax.25), and state of the connection. It then shows the retry timer, with current time left, and initial value.

In 'repeat', 'more' or 'look' sessions, the 3rd line show the command, filename or user/socket for the session.

The command session 3rd line shows the current directory.
(the 3rd line is the 'session' window)

The number of status lines displayed can be set with the '-z#' commands line option. '-z0' turns it off. Default is '-z3'
NOTE: if tracing is enabled, this will 'bleed' through the status window.

This is NOT a bug, but is inherent to the way tracing works in JNOS. (It would take a major rewrite to fix.) The status window will be rebuilt about twice a second to overcome this.

This section describes the commands recognized. Syntax is:

command
command literal_parameter
command subcommand <parameter>
command [<optional_parameter>]
command a | b

Many commands take subcommands, parameters, or both, which may in turn be optional or required. In general, if a required subcommand or parameter is omitted, an error message will summarize the available subcommands or required parameters. (Giving a '?' in place of the subcommand will also generate the message. This method is useful when the command word alone is a valid command.) If a command takes an optional value parameter, issuing the command without the parameter generally displays the current value of the variable. (Exceptions to this rule are noted in the individual command descriptions.)
Two or more parameters separated by vertical bar(s) '**|**' denote a choice between the specified values. When one of the choices is default, that choice will be in UPPERCASE. Optional parameters are shown enclosed in **[brackets]**. Parameters enclosed in **<angle brackets>** should be replaced with actual values or strings. The generic notation for number values is **<nnnn>**, and for string values, it is **<string_id>**. For example, the notation **<hostname>** means the actual host or gateway callsign or id. Numerical defaults are explicitly stated as such, e.g., "Default = 7."
All commands and many subcommands may be abbreviated. You only need type enough of a command's name to distinguish it from others that begin with the same series of letters. Parameters, however, must be typed in full.
All commands are printed in **bold** (if you have the version of this document that supports fancy formatting), and most commands have an example following the textual description of the commands. Commands or descriptions unique to one program or the other are annotated with **{JNOS}** or **{JNOS40}**.
When JNOS40 is in the Data Engine, many variables are kept in battery backed ram. They are protected against corruption by a 16 bit CRC. If the CRC is valid, the value or state of these variables will be maintained across power outages or warm restarts. This is only true if the variables are not re-set in the startup configuration. These variables are marked with '(B)' at the beginning of their command description. For example, if you set the tcp retries value to 0, if will be still 0 after a power down or warm restart. If the conference server was stopped, it will not be restarted after a power down or warm restart. However, if your startup configuration contains a line 'start convers', the server will be started! To

override this, either change it from remote sysop mode, or use the 'add' command to add the 'stop convers' to the configuration. (see the 'add' command for more).

The following section contains the comprehensive set of commands for JNOS and JNOS40. Commands unique to one system are annotated with **{JNOS}** or **{JNOS40}**, as appropriate.

{JNOS}

Entering a carriage return (empty line) while in command mode puts you in converse mode with the current session. If there is no current session, Nos remains in command mode and reissues the 'net>' prompt.

{JNOS}

An alias for the **shell** command.

Commands starting with the hash mark (**#**) are ignored. The hash mark is mainly useful for comments in the autoexec.nos.

{JNOS40} The CFG.EXE program eliminates lines beginning with the '#' when building the image files for the eproms.

Same as the 'help' command.

{JNOS}

Abort a FTP get, put or dir operation in progress. If issued without an argument, the current session is aborted. (This command works only on FTP sessions.) When receiving a file, abort simply resets the data connection; the next incoming data packet will generate a TCP RST (reset) response to clear the remote server. When sending a file, abort sends a premature end-of-file.
Note that in both cases abort will leave a partial copy of the file on the destination machine, which must be removed manually if it is unwanted.

{JNOS40}

(B) The **add** command allows you to add commands to the configuration after the system is up and running. **<other_command>** is a valid command line which will be executed and then stored in battery-backed RAM. Next time the system restarts, all commands put in eprom with the cfg.exe program will be executed followed by any commands saved with the **'add'** command. **'add'** with no arguments will display the commands stored. Each line will show a line number that should be used if you want to delete the line with the **'del'** command.
To add another ip route after the system is started and have it be remembered after the next power outage or warm restart:

```
add route add 1.2.3.4 1
```

Display the Address Resolution Protocol table that maps IP addresses to their subnet (link) addresses on subnetworks capable of broadcasting. For each IP address entry the subnet type (e.g., AX.25), subnet address and time to expiration is shown. If the link address is currently unknown, the number of IP datagrams awaiting resolution is also shown.

arp add <hostid> ax25 | netrom <callsign> <iface> {JNOS40}
or

arp add <hostid> ether | ax25 | netrom | arcnet <ether_addr>| <callsign> <iface> {JNOS}

Add a permanent entry to the table. It will not time out as will an automatically created entry, but must be removed with the **'arp drop'** command.

```
          arp add 44.26.0.19 ax25 wg7j-2 port1
```

arp drop <hostid> ax25 | netrom <iface> {JNOS40}
or

arp drop <hostid> ether | ax25 | netrom | arcnet <iface>   {JNOS}

Delete a permanent entry from the arp table.

```
          arp delete 44.26.0.19 ax25 port1
```

arp eaves [<iface>] [on | OFF]

Display or set the **'arp eavesdrop'** function per interface. If set, all arp replies overheard on the interface will be logged in the arp table. This speeds up arp discovery, but might build a huge arp table taking up lots of memory. Default for each interface is off.

```
          # Set arp eavesdrop on interface port1
          arp eaves port1 on
```

arp flush

Drop all automatically-created entries in the ARP table; permanent entries are not affected.

arp maxq [n]

Display or set the maximum number of packets to be buffered waiting for an arp resolution to finish. {JNOS} Default = 5. {JNOS40} Default = 2.
arp maxq 5

arp poll [<iface>] [on | off]

Display or set the 'arp keepalive polling' per interface. If set, when an arp entry expires, a query will be sent for the address. This keeps the arp table fresh, but possibly retains many unneeded entries.
arp poll port1 on

arp publish <hostid> ax25|netrom <callsign> <iface>

This command is similar to the **'arp add'** command, but the system will also respond to any ARP request it sees on the network that seeks the specified address. (Use this feature with great care!)

arp publish 44.26.1.19 ax25 wg7j-2 port1

arp sort [ON | off] {JNOS}

Sorts the arp display

{JNOS}

Display statistics on attached asynchronous communications interfaces (8250 or 16550A), if any. The display for each port consists of three lines. The first line gives the port label and the configuration flags; these indicate whether the port is a 16550A chip, the trigger character if any, whether CTS flow control is enabled, whether RLSD (carrier detect) line control is enabled, and the speed in bits per second. (Receiving the trigger character causes the driver to signal upper layer software that data is ready; it is automatically set to the appropriate frame end character for SLIP, PPP and NRS lines.)
The second line of the status display shows receiver (RX) event counts: the total number of receive interrupts, received characters, receiver overruns (lost characters) and the receiver high water mark. The high water mark is the maximum number of characters ever read from the device during a single interrupt. This is useful for monitoring system interrupt latency margins as it shows how close the port hardware has come to overflowing due to the inability of the CPU to respond to a receiver interrupt in time. 8250 chips have no FIFO, so the high water mark cannot go higher than 2 before overruns occur. The 16550A chip, however, has a 16-byte receive FIFO which the software programs to interrupt the CPU when the FIFO is one-quarter full. The high water mark should typically be 4 or 5

when a 16550A is used; higher values indicate that the CPU has at least once been slow to respond to a receiver interrupt.
When the 16550A is used, a count of FIFO timeouts is also displayed on the RX status line. These are generated automatically by the 16550A when three character intervals go by with more than 0 but less than 4 characters in the FIFO. Since the characters that make up a SLIP or NRS frame are normally sent at full line speed, this count will usually be a lower bound on the number of frames received on the port, as only the last fragment of a frame generally results in a timeout (and then only when the frame is not a multiple of 4 bytes long.)
Finally, the software fifo overruns and high water mark are displayed. These indicate whether the <bufsize> parameter on the attach command needs to be adjusted (see the Attach Commands chapter).
The third line shows transmit (TX) statistics, including a total count of transmit interrupts, transmitted characters, the length of the transmit queue in bytes, the number of status interrupts, and the number of THRE timeouts. The status interrupt count will be zero unless CTS flow control or RLSD line control has been enabled. The THRE timeout is a stopgap measure to catch lost transmit interrupts, which seem to happen when there is a lot of activity (ideally, this will be zero).

{JNOS}

The 'at' command is used to provide automatic starting of other JNOS commands at predetermined times.

at time <cmd>

time takes the form yymmddhhmm
hhmm
mm
now+hhmm

<cmd> is any legal JNOS command. Multiple word commands must be enclosed in double quotes (" "). Commands which invoke the DOS shell must include 'c/' as the first argument in order for the shell to be exited and NOS to be re-entered automatically.

at k <id_num> <id_num>

This form of the **'at'** command kills jobs <id_num>...

attach <1|2|3|axip|kiss|pkiss|netrom> [options] {JNOS40}

Attaching the Data Engine's serial port.

Attach an interface to the Data Engine. See also the section **ATTACHING INTERFACES.**
Syntax is:

'attach 1 <mode> <name> <buffer> <mtu> <speed> [c]'

where:

```
     mode      - is one of 'ax25', 'pkiss', 'slip' or 'nrs'
     name      - is the interface name, e.g. 'port1'
     buffer    - is the receive buffer size
     mtu       - is the maximum transmission unit
     speed     - any common speed from 300 - 38400 Bd.
     c         - forces use of hardware handshaking except
                 for mode='pkiss' (G8BPQ polled-kiss)
```

Note: 19200 and higher is not very reliable if you use DE9600 modems, due to a hardware design limitation.

See also the '**attach kiss**' command

attach 1 ax25 port1 512 256 9600

Attach the Data Engine's internal radio ports

The modem type in each port is automatically sensed when a port is attached. Modem types A,B and D have been tested. Modem type C is supported, but currently has not been tested. Syntax is:

'attach <2|3> <name> <mtu> <speed> [f][n]'   {JNOS40}

where:

2|3	- is one of the Data Engine internal ports
name	- is the interface name
mtu	- is the maximum transmission unit.
speed	- is the radio speed.
[f][n]	- are optional parameters. 'f' indicates full duplex and 'n' is the value written to the mode AUX pins. This is used in the Kantronics 1200Bd modem to choose the type of CD circuitry to be used.

```
attach 3 port2 256 1200 2
attach 2 port2 1024 9600 f
attach 2 port3 256 1200 f1
```

Note: full duplex is not yet supported on port B.

Attaching the ports on a PC

attach <hw_type> <io_addr> <vector> <mode> <label> <bufsize> <mtu> [<speed>]   {JNOS}

Configure and attach a hardware interface to the system. Detailed instructions for each driver are in the Attach Commands chapter. An easy way to obtain a summary of the parameters required for a given device is to issue a partial attach command (e.g., attach asy). This produces a message giving the complete command format.
<hw_type> is the kind of I/O device being attached to the system.
<io_addr> is the base address of the control registers for the device.
<vector> is the interrupt vector number. Both the address and the vector must be in hexadecimal. You may put "0x" in front of the numbers, but they will be interpreted in hexadecimal even without the prefix.
<mode> controls how IP datagrams are to be encapsulated in the device's link level protocol.
Choices are ax25 or slip.

slip Encapsulates IP datagrams directly in SLIP frames without a link header. This is for operating point-to-point lines and is compatible with 4.28BSD UNIX SLIP.

ax25 Similar to slip, except that an AX.25 header and a KISS TNC control header are added to the front of the datagram before SLIP encoding. Either UI (connectionless) or I (connection-oriented) AX.25 frames can be used.

<label> defines the name by which the interface will be known to various commands, such as "connect", "route", "trace", etc.

<bufsize> For ASYNCHRONOUS PORTS, specifies the size of the ring buffer in bytes to be statically allocated to the receiver; incoming bursts larger than <bufsize> may cause data to be lost.
For ETHERNET, specifies how many PACKETS may be queued in the receive queue at one time. Excess packets will be discarded as they are received. This is useful to prevent the system from running out of memory should another node suddently develop a case of diarrhea.

<mtu> is the Maximum Transmission Unit size in bytes. See the System Configuration Manual for a discussion of the effect of MTU on system performance.

attach asy <io_addr> <vector> <mode> <label> <bufsize> <mtu> [<speed>] {JNOS}
Configure and attach a standard PC asynchronous I/O port using the National 8250, 16450, or 16550 chip or equivalent to the system, where:
<io_addr> is the comm port address; e.g., com1 = 0x3f8
<vector> is the comm port IRQ value.

attach axip <iface> <mtu> <ipaddress> [<callsign>]

Create a RFC1226 compatible AX.25 frame encapsulator for transmission of AX.25 frames over the IP. This command is used to establish a point-to-point AX.25 'tunnel' between two systems.

<iface> will be the name of the new interface,
<mtu> is the maximum transmission unit for the interface,
<ipaddress> is the address of the system on the other side of the 'tunnel,
<callsign> is the optional AX.25 callsign this station is listening on for frames to digipeat. Note that if you want cross-tunnel digipeating to work, each attached axip interface should listen to a different callsign. These should also be different from other callsigns used on this station.

attach axip axip1 256 44.26.1.19 WG7J-15

attach pkiss <iface> <mtu> <ipaddress> [<callsign>]

Attach a serial port in G8BPQ polled kiss mode.

To attach other TNCs to be polled on the same port, attach them as additional kiss devices (see below). They will automatically be polled.

attach kiss <asy_iface_label> <port> <label> [mtu]

Attach a second kiss interface on the serial port. This command allows the use of multiport TNCs.

<asy_iface_label> is the name of the serial port interface.
<port> is the port number (1-15) to use, and probably should be 1. (the original asy port is automatically port 0 !)
<label> is the name for this second kiss port.
<mtu> is an optional mtu, if different from the mtu on the first kiss port.

```
# Attach a dualport tnc in kiss mode.
# Ports are labelled 'port1' and 'port2'

# First, attach the serial port on the Data Engine
```

attach 1 ax25 port1 512 256 9600 {JNOS40}

```
# or

# Attach a PC asynch port (com1 in this example)
```

attach asy 03f8 4 ax25 port1 512 256 9600 {JNOS}

```
# Attach the second port on the multiport tnc
```

attach kiss port1 port2

attach netrom {JNOS}
attach netrom [ipaddress] {JNOS40}

This makes available a pseudo interface to enable NET/ROM operations. This command is automatically executed when the netrom server is started with the 'start netrom' command. In JNOS40, the netrom server is started by default and you do not have to specifically attach the netrom interface in the autoexec file.

[ipaddress] is an optional ip address for the netrom interface. If not set, the system 'ip address' will be used.

attach packet {JNOS}
Driver for use with separate software "packet drivers" which conform to the FTP Software, Inc., Software Packet Driver specification.

attach scc {JNOS}
PE1CHL driver for generic 8530 cards

{JNOS}
Displays or sets the global "I am present" flag in NOS. This flag is used in the welcome header for incoming ttylink connections.

All AX.25 parameters are configurable per interface. Commands of the form 'ax25 <command>' set the default or global values. Use the 'ifconfig <iface> ax25 <command>' form to set or show the specific interface values.

To set the system default ax.25 parameters, you must do so BEFORE attaching interfaces. After attaching, you must use the 'ifconf <iface> ax25' command form.

ax25 alias <aliascall>
The alias command shows or sets the system's alias call. If netrom is enabled, this modifies the same call as the **'netrom alias'** command. The **'ax25 alias'** command is NOT needed in that case! If netrom is not used, this command allows an alias name to be set such that users can connect to it.

ax25 bbscall <bbs_call> {JNOS}
For all interfaces NOT set to <mycall> or <bbs_call>, change or set the id to 'bbscall'. 'ax25 mycall' will not override bbscall once bbscall has been set. See also 'iface <iface> bbscall <bbs_call>'.

#Example: (in the following order)

'ax25 mycall <mycall>'
'attach <(all interfaces)>'
'ax25 bbscall <bbscall>' sets all ifaces to bbscall

or

'iface <name> bbscall <bbscall>' sets only iface <name>

ax25 bc <iface>

The **bc** command forces an immediate broadcast on the given interface. The particular interface or port must have been enabled with the **ax25 bcport** command first. If this is so, the ID will be broadcast as set with the **ax25 bctext** commands.

```
ax25 bc port1
```

ax25 bcinterval [<seconds>]

(B) The bcinterval displays or sets the time in seconds between broadcasts. On display, both the interval and the countdown values are shown. Default = 600 (10 minutes).

ax25 bcport [<iface>] [on | OFF] (Deleted in 110x16)

Display or set the active interfaces for ax.25 broadcasting (i.e. beacons). You must add this statement for each port that the system should beacon on. Default is off.

```
          ax25 bcport port1 on
```

ax25 bctext ["broadcast text"]

Display or set the default text to be sent for broadcast messages sent out every **ax25 bcinterval** seconds.

See also '**ifconfig <iface> beacon ["bctext"]**.

```
          ax25 bctext "This is the beacon text!"
```

ax25 blimit [<secs>]

(B) Display or set the default AX25 retransmission backoff limit. Normally each successive AX25 retransmission is delayed by twice the value of the previous interval; this is called binary exponential backoff. When the backoff reaches the blimit setting it is held at that value. Default = 30.

To prevent the possibility of "congestive collapse" on a loaded channel, blimit should be set at least as high as the number of stations sharing the channel. Note that this is applicable only on actual AX25 connections; UI frames will never be retransmitted by the AX25 layer.

```
#Set ax25 blimit to 15 seconds
```

```
          ax25 blimit 15
```

ax25 digipeat [<iface>] [ON | off]

Display or set digipeating per interface. If cross-band or AXIP digipeating is to be allowed, digipeating must be enabled on both interfaces involved. Default is on.

```
          # Display digipeat status of port1
          ax25 digipeat port1
```

ax25 ecall <callsign>   {JNOS40}

Set or display and "emergency" callsign to be used as described under the 'ax25 etext' command which follows.

ax25 etext [<your text>]

If you connect a monitor signal to the serial port RTS pin, when the RTS pin is pulled LOW (less than approximately 2 volts) this condition can be used to define an emergency state meaning that the node is operating on emergency or backup power. While in this state, every 5 minutes the node will transmit an ax.25 UI ("broadcast") packet containing the 'etext' message on all ax.25 interfaces if 'ax25 ecall' is set.

Users connecting to the node alias or via telnet will receive the 'etext' message as part of the logon. 'motd' and 'info' will also append the 'etext' message.

If the emergency condition ceases, broadcasting stops and etext is not displayed as part of other messages.

Example:

```
ax25 ecall pwrdwn
ax25 etext "System is running on backup power.  Please contact
sysop!"
```

Note: This feature was added after an area sysop had a node die after 24 hours of operation on emergency power. Prior to shutdown, there was no indication of the problem.

ax25 flush

Clears the AX.25 "heard" list (see **ax25 heard** and **ax25 hport**)

ax25 heard [<iface>]

Display the AX.25 "heard" list. For each interface that is configured to use AX.25 heard listing (see '**ax25 hport**'), a list of all ax25_source addresses heard on that interface is shown, along with a count of the number of packets heard from each station and the time since each station was last heard. The maximum length of the heard table can be set with the '**ax25 hsize**' command. If interface is given, only the heard list for that interface is displayed.

```
          ax25 heard port1
```

ax25 hport [<iface>] [ON | off]

Display or set the status of the ax.25 heard feature. If no interface is given, all interfaces with ax.25 heard enabled will be listed. If interface is given, the status of ax.25 heard for that interface is shown. Default is on.

```
#Display port1 status
          ax25 hport port1
```

ax25 hsize [<size>] {JNOS}

Set or display the size of the heard list table. Default is 0 which means no limit.

ax25 irrt [<milliseconds>]

(B) Display or set the initial value of smoothed round trip time to be used when a new AX25 connection is created. The actual round trip time will be learned by measurement once the connection has been established. Default is 5000ms.

```
#Set irtt to 10 seconds  (10000 milliseconds)
          ax25 irtt 10000
```

ax25 kick <axcb>

Force a retransmission on the specified AX.25 control block. The control block address can be found with the **ax25 status** command. This is useful to reactivate connections that have long time-out values.

ax25 maxframe [<count>]

(B) Establish the maximum number of frames that will be allowed to remain unacknowledged at one time on new AX.25 connections. This number cannot be greater than 7. Without **<count>** it will display the current setting. Note that the maximum outstanding frame count only works with virtual connections. UI frames are not affected. Also note that for optimal performance, a value of 1 should be used. Default is 1 frame.

ax25 maxwait [<msec>]
Sets a limit (in msec) to the retry timeout values. Default = 30000 (30 secs). A value of 0 disables maxwait.

ax25 mycall [<ax25call>]
Display or set the default local AX.25 address. The standard format is used, (e.g. WG7J or KA7EHK-5). This command must be given before any attach commands using AX.25 mode are given.

```
          ax25 mycall wg7j-3
```

ax25 paclen [<size>]
(B) This sets the default paclen used when attaching interfaces that will carry AX.25 connections. See also '**ifconfig <iface> paclen**'. Default is 256 bytes.
This parameter limits the size of I-fields on new AX.25 connections. If IP datagrams or fragments of datagrams larger than paclen are transmitted, they will be transparently fragmented at the AX.25 level, sent as a series of I frames, and reassembled back into a complete IP datagram or fragment at the other end of the link. IP datagrams will not be affected if this parameter is greater than or equal to the MTU of the associated interface.
If NET/ROM communication is configured, the NetRom MTU value should be Paclen - 20. !!! The Net/Rom header takes 20 bytes, and is part of the AX25 data. Default netrom mtu is 236.

Note1: the AX.25 Level 2 Version 2 definition specifies a maximum paclen of 256 bytes. Some systems are not equipped to handle larger packets (e.g. G8BPQ based systems), so be careful when using this parameter.

Note2: see also the discussion on paclen, mtu etc., elsewhere in the document.

ax25 pthresh [<size>]
(B) Display or set the poll threshold to be used for new AX.25 Version 2 connections. The poll threshold controls retransmission behavior as follows. If the oldest unacknowledged I-frame size is less than the poll threshold, it will be sent with the poll (P) bit set if a time-out occurs. If the oldest unacked I-frame size is equal

to or greater than the threshold, then a RR or RNR frame, as appropriate, with the poll bit set will be sent if a time-out occurs.

The idea behind the poll threshold is that the extra time needed to send a "small" I-frame instead of a supervisory frame when polling after a time-out is small, and since there's a good chance the I-frame will have to be sent anyway (i.e., if it were lost previously) then you might as well send it as the poll. But if the I-frame is large, send a supervisory (RR/RNR) poll instead to determine first if retransmitting the oldest unacknowledged I-frame is necessary; the time-out might have been caused by a lost acknowledgment. This is obviously a tradeoff, so experiment with the poll threshold setting. The default is 128 bytes, one half the default value of **<paclen>**

ax25 reset <axcb>

Delete the AX.25 connection control block at the specified address. This deletes a connection and everything associated with it. The control block address can be found with the '**ax25 status**' command.

ax25 retries [<count>]
(B) Limit the number of successive unsuccessful retransmission attempts on new AX.25 connections. If this limit is exceeded, link re-establishment is attempted. If the link can't be re-established in **<count>** times, then the connection is abandoned and all queued data is deleted. Default is 5.

ax25 route [<subcommand>]
Without optional subcommands, display the AX.25 routing table that specifies the digipeaters to be used in reaching a given station.

ax25 route add <target> <iface> [digis ...]
Add an entry to the AX.25 routing table. An automatic '**ax25 route add**' is executed if digipeaters are specified in an AX25 link from the node or a connection is received from a remote station via digipeaters. Such automatic routing table entries won't override locally created entries, however. **<target>** is the destination call to reach via digipeaters **<iface>** is the interface this is a route for. (I.e. allows different digi routes for different interfaces.) **[digis...]** is a list of one or more digipeaters, separated by spaces.

```
ax25 route add k7uyx-1 port1 wg7j wa7tas n7dva
```

ax25 route drop <target> <iface>

Drop an entry for **<target>** from the AX.25 routing table.

```
          ax25 route drop k7uyx-1 port1
```

ax25 route mode <target> <iface> [vc|dg|interface]

Sets the interface ip mode to one of **vc | datagram | interface** for **target**. This indicates how ip links to the destination call **<target>** should be established. If nothing is given for a certain destination or target, the interface default mode is used, which defaults to datagram. (See also the '**mode**' command).

vc is a virtual circuit (ax25 connected mode, meaning that ip frames are sent using ax.25 connections)

datagram is unconnected mode, (AX25 UI frames).

interface is the default interface mode, as set with the '**mode**' command.

```
          ax25 route mode k7uyx-1 port1 vc
```

ax25 status [<axcb>]

Without an argument, display a one-line summary of each AX.25 control block. If the address of a particular control block is specified, the contents of that control block is shown in more detail. Note that the send queue units are frames, while the receive queue units are bytes.

ax25 t3 [<milliseconds>]

(B) Display or set the AX.25 idle "keep alive" timer. Value is in milliseconds. Default is 0, i.e. no 'keep-alive' polling.

ax25 t4 [<seconds>]

(B) Display or set the AX.25 Link "redundancy" timer. Value is in seconds. When no exchange has been had during this time the link is reset and closed. Default = 900 seconds (15 minutes).

ax25 timertype [LINEAR|exponential|original]

(B) Sets or displays the type of timer used for retransmission and recovery. **Linear** means that each retry will use a time-out that is RTT greater then the previous time-out. I.e. 4 sec, 8 sec, 12 sec, 16 seconds etc. **Exponential** means that each retry will use a time-out that is twice as large as the previous time-out. I.e. 4 seconds, 8 seconds, 16 seconds, 32 seconds etc. **Original** means that each retry will use a time out that is twice the RTT, i.e. 4 seconds, 8 seconds, 8 seconds, 8 seconds, etc.

Default is **linear.**

```
          ax25 timertype exponential
```

ax25 ttycall [ttycall]   JNOS}
Set or display the tty-link call for direct keyboard access. Remember to have both 'attended on' and 'mbox attend on' to be able to use this function.

ax25 version [n]

(B) Display or set the version of the AX.25 protocol to attempt to use on new connections. Version 1 is the version that does not use the poll/final bits. Default is version 2.

ax25 window [<size>]

(B) Set the number of bytes that can be pending on an AX.25 receive queue beyond which I frames will be answered with RNR (Receiver Not Ready) responses. This presently applies only to suspended interactive AX.25 sessions, since incoming I-frames containing network (IP, NET/ROM) packets are always processed immediately and are not placed on the receive queue. However, when an AX.25 connection carries both interactive and network packet traffic, an RNR generated because of backlogged interactive traffic will also stop network packet traffic from being sent. Default is 512 bytes.

This will show or configure the host and portnumber that will be used for the 'CAll' command. Node shell users give the 'CAll' command access a remote network callbook server. To see the current server, simply type callserver. To configure one use the full form. If a hostname is used instead of dotted decimal ip address, be sure that this name is either in the domain.txt file, or a valid domain name server has been configured already.

cbaud [<2400 | 4800 | 9600>] {JNOS40}
Set or display the serial port baud rate for the console connection. Valid values are 2400, 4800, or 9600 baud.

{JNOS40}
Coldboot is an IMMEDIATE command (i.e. no 'are you sure? warning...)

to restart the system. The system will 'cold-boot'. All variables in battery backed ram will be reset to the original values in EPROM. All links and connections to or through the node will be immediately lost.

comm <asy_iface> <"text_string">

Sends "text_string" via the specified asynch interface to the TNC. Normally, this command is used to place the TNC in KISS mode. Any number of 'comm' commands may be entered, one per line, with "text_string" in the syntax required by the TNC. Use the 'param' command after the TNC is in KISS mode.

connect <iface> <destination> [<digi1,digi2...digin>] [d] [e]]

Initiate an ax25 connection at interface <iface> to <destination>. Use the "ports" command when in the mailbox or nodeshell to discover the proper id of <iface>. <destination> may be either callsign-ssid or an alias. Note that there is only a 'space' between <destination> and <digi...>.

[d] disables the 'stay here' feature on node connections. If 'd' is not placed at the end of the connect command, the circuit will stay open to the last node connected to. 'stay here' default is ON.

[e] is the 'escape enabling' switch. In order for JNOS to abort a netrom connection in progress, the escape character must be sent to the node. Default escape character is CTRL-T. While in the mailbox or nodeshell, the user may disable escape or change the escape character or both.

Examples:

```
c rlimb e      -> connect to w0rli's bbs with escape
                     enabled. 'stay' is on
c 3 wg7j-2 d   -> connect to wg7j-2 on port 3, don't stay
                     connected afterward
c salem ed     -> connect to salem, with escape checking but
                     no 'stay'.  c salem de works the same.
```

These commands configure the network conference server. See the section '**SETTING UP THE CONFERENCE BRIDGE**' for details on the inner workings of the conference system.

convers drop <name>

Drop the remote convers link to **<name>**. See also 'convers link'.

```
          convers drop 44.26.1.19
```

convers filter {JNOS}

Set how the convers node will respond to connect requests.

convers filter mode [accept | refuse]

Sets or displays the filter mode. 'filter mode accept' allows links from only the hosts in the filter list. 'filter mode reject' allows links from all hosts except those in the list.

convers filter [ipaddress | hostname]

Builds the filter list used in conjunction with the 'convers filter mode' command.

convers host <name>

Displays or set the convers hostname as will be used when announcing the system to conference users or remote links. Maximum length is 10 chars, but if you want to stay compatible with NOS.EXE based convers servers use a maximum of 8 character for the convers host name (unless the system runs jnos-v1.04 or later).

If the **'hostname'** gets set and the **'convers host'** isn't set yet, it will be set to the first 10 chars of the **'hostname'**. After this, if any sub domains (i.e. periods) exist in the hostname, the convers hostname will be terminated at the right-most period. e.g. If **'converse host'** is not set, and **'hostname jnos.wg7j.ampr.org'** is given, then after this the converse hostname will be 'jnos.wg7j'.

```
convers host Corvallis
```

convers interface [<iface>] [on|OFF]

Displays or sets the active convers interfaces. This command needs to be given for each interface that which will allow connections to the conference call (see **'convers mycall'**); e.g., this command can be used to allow conference call access only on the user ports but not on the backbone/linking ports. This can also be useful to avoid confusion when different nodes have the same conference call. (Locally, we use the call 'QSO' for the conference server for different nodes, and ran into problems when a user tried to connect to it from a backbone node. All of a sudden two nodes were answering the connect !) Default is off.

```
convers interface port1
```

convers link <addr> [name]

Add a convers link to another (remote) conference server.

<addr> is the ip address or hostname of the remote server to link to.
[name] is the optional name that will show up in the links listing

shown with the '/links' command if the link has not yet been established. [name] can be a maximum of 10 characters.

After the link has been established, the name will be set to the name the remote system introduced itself with. The link command will automatically add an entry for this host into the 'refuse-list' (See **'convers refuse'** command.) This is to avoid dual links, and convers loops.

convers link 44.26.1.19 Testing

convers maxq [<bytes>]   {JNOS40}
convers [u|h]maxq [<bytes>]   {JNOS}

Display or set the upper limit for the number of bytes that can be queued up waiting for transmission on a connection to another server. If there is more data than this limit, the connection to the other server will be closed.

{JNOS} You are able to set individual limits for users and hosts with 'convers hmaxq' and 'convers umaxq'. If set to 0, there is no limit, otherwise connections will be reset if there is more than the []maxq value data outstanding on the connection. The connections will be RESET instead of gracefully closed.

Default is 2048 bytes.

convers maxq 1024

convers maxwait [<seconds>]

Display or set the upper limit for the time the system will wait to reestablish a disconnected convers link that originated at this system. Time is given in seconds.
convers maxwait 600

convers motd ["<yourmessage>"]

Set or show the message of the day for the convers server. This message is displayed when users connect to the server.

convers <mycall>

Display or set the 'conference call'. 'mycall' is a separate ax.25 callsign. If set, users can connect to it to get immediately connected to the conference bridge. However, each port or interface that this call should be allowed on should be enabled with the **'convers interface'** command. Conference call connections bypass the

regular node interface. This is independent from the settings of **'mbox convers'** or whether the network conference server has been started. See also **'convers t4'**.
convers mycall QSO

convers refuse <addr>   {JNOS40}
Refuse link-requests from **<addr>**. This is primarily intended to avoid loops. An entry on this list is automatically created when you add a link with the **'convers link'** command.

convers refuse 44.26.1.19

convers t4 [<seconds>]
(B) Display or the set the conference call connection T4 timer. t4 is the 'redundancy timer' for ax.25 connections to the conference server. This allows you to set a different inactivity time-out for ax25 node and conference connections. Default is 7200, i.e. 2 hours.
convers t4 900

{JNOS}
Deletes the specified file. <filename> may include a complete path. Functions the same as the DOS Delete command.
{JNOS40}
The delete command will delete the specified line number from the configuration lines stored in battery backed ram with the **'add'** command. To show the line numbers, execute the **'add'** command.
delete 4

The domain commands control and show the working of the name to Internet address mapping software. JNOS40 has both a network client and server. The server will answer queries from data in the domain cache, and from information stored in the rom DOMAIN.TXT file.

domain addserver <hostid> [<timeout>]
Add a domain name server to the list of name servers.

{JNOS} <timeout> is an optional timeout setting in seconds for this server. If <timeout> is not included in the command, the value defaults to 3 * (tcp_irtt).

Example: domain addserver wg7j.ece.orst.edu 30

Note: {JNOS40} When this command is first given in the startup 'file', the <hostid> can be either a name or an ip address. If a name is used, be sure that a valid address record exists in the rom DOMAIN.TXT file!!! Otherwize, the ip address should be used instead of the hostname. (If not, since no servers are setup yet, JNOS40 will not know how to resolve the name!)

domain addserver 128.193.48.1
domain addserver ucsd.edu

domain cache <subcommand>

Following commands work on the domain cache. These are resource records held in memory. (described in RFC1033/1034)

domain cache clean [<yes | NO>]

Displays or sets the discard of expired resource records. Expired records have their time-out value decremented to zero. Normally resource records get a default time-out value of 1800 seconds. After this time they are considered "old" and if referenced again the domain name resolver should be inquired again. When clean is off (the default), expired records will be retained; if no replacement can be obtained from another domain name server, these records will continue to be used. When clean is on, expired records will be removed from the file whenever any new record is added to the file.

```
          domain cache clean yes
```

domain cache dump

Immediately clears the domain cache

domain cache list

This command shows the current content of the in-memory cache of resource records.

domain cache size [<size>]

(B) Display or set the maximum size of the local in-memory domain cache. Default is 5.

```
          domain cache size 10
```

domain dns [on|off]

(B) Display or toggle the state of the Domain Name Server. If on, the system is active as a Domain Name Server. The system will then answer queries from other tcp/ip hosts regarding hostname to ip-address, and ip-address to hostname translations. For more information, see the section SETTING UP THE DOMAIN NAME SYSTEM.

domain dropserver <hostid>

Remove a domain name server from the list of name servers. You are warned when you delete the last name server.

domain dropserver ece.orst.edu

domain listservers

List the currently configured domain name servers, along with statistics on how many queries and replies have been exchanged with each one, response times, etc.

domain maxwait [<time-out>]

(B) This sets a time-out value (1 to 255 seconds) to a query or domain name server. This is not set for an already defined server but will be used for a newly defined name server. Also the value is used for domain name lookups (E.g. when a user does a telnet to a host with the nodeshell 'T host' command). Note that (PC based) name servers can have trouble finding records in a large database. Default is 60 seconds.

domain maxwait 10

domain retries [<retries>]

(B) The retry value (number) limits the number of queries sent out to remote domain name resolvers before giving up and telling you that host xyzzy.ampr.org does not exist. The total time lost with a query is (retries * time-out * number of domain servers defined); i.e., the delay between requesting a hostname to ip-address translation and getting the answer can become very long if you use many servers, and set the time-outs/retries high ! Default is 2.

domain retries 1

domain subnet [on | off] {JNOS}

This command works in conjunction with 'domain translate' to allow or disallow translation of any address ending in 0 or 255. On systems which have a lot of subnets, turning off subnet translation can result in a considerable speedup when displaying routes with 'domain translate on'.

domain suffix [<domain suffix> | none]
Display or specify the default domain name suffix to be appended to a host name when it contains no periods. For example, if the suffix is set to "ampr.org." and the user enters 'telnet ka9q', the domain resolver will attempt to find 'ka9q.ampr.org.' If the host name being sought contains one or more periods, however, the default suffix is NOT applied if the last part of the name is less than 5 characters and contains only letters; e.g., 'telnet foo.bar' would NOT be turned into 'foo.bar.ampr.org.' 'telnet foo.ka9q' will be turned into 'foo.ka9q.ampr.org.' Note that a trailing dot (.) is required for the suffix. If the suffix is the string 'none' (without trailing period), the current suffix is cleared and forgotten. Default is "ampr.org."

domain suffix ece.orst.edu.

domain trace [on| OFF]
(B) Display or set the flag controlling the tracing of domain server requests and responses. This only works when console is enabled. Default is off.

domain trace on

domain translate [on | OFF]
(B) Display or set the flag that controls the translation of ip addresses in dot notation into symbolic names. The translation process makes heavy use of reverse domain name lookups. Do not set this flag unless you have a good and fast connection to a domain name server.

domain translate on

domain ttl [ttl] {JNOS}
Select a default 'ttl' value to be applied to server responses than contain none.

domain update [on | off] {JNOS}

Controls whether or not domain.txt file is updated with server responses.

domain verbose [on | off]

(B) Display or set the flag controlling the return of a full name (on) or only the first name (dot delimiter) (off). This is for IP address to name translation only. If off, home.wg7j.ampr.org. will show as 'home.wg7j', whereas if on it will show as 'home.wg7j.ampr.org'

```
domain verbose off
```

The dump command shows memory in hex and ascii. Hex-address is a 32-bit value split into page address and page offset. A splitting colon is not used nor accepted. If decimal-range is not given , 128 bytes are displayed. 'dump .' displays memory starting at the end of a previous dump command. This command is primarily useful for debugging.

```
dump 12fe0008
```

{JNOS}

Set whether the system will send messages about system errors and permission infringements to user 'sysop'. Default is on.

{JNOS40} When in remote sysop mode (after connection to the node), returns the sysop to the node shell
{JNOS} Causes the JNOS program to terminate when at the NET> prompt. When shelled to DOS, causes a return to the NET> prompt. When terminating the program, an "Are you sure?" query is given. Enter "y(es) <cr>" to end the program. Any other response returns to JNOS.

Display a brief summary of top-level commands.

Display or set the local host's name. By convention this should be the same as the host's primary domain name. This string is used only in the greeting messages of the various network servers; note that it does NOT set the system's IP address.

```
hostname crv.kuyx.ampr.org.
```

These commands are used for the Internet Control Message Protocol service.

icmp echo [ON | off]
(B) Display or set the flag controlling the asynchronous display of ICMP Echo Reply packets. This flag must be on for pings to work. Default is on.

icmp quench [on | OFF] {JNOS}
With 'icmp quench off', when a packet is received and memory available < threshold, the packet will be dropped (i.e., no quench or anything.) The higher protocol layers will keep track of re-transmitting the dropped packets.

With 'icmp quench on', when packets are received and the high water mark for dynamically allocatable storage has been exceeded, JNOS submits an ICMP Source Quench to the originator. Usually, before the originator will have reacted to the source quench, JNOS's dynamically allocatable storage will have been exhausted. What happens after that is uncertain, but it is assumed to be unfavorable. Many tcp/ip implementations don't even respond to Source Quenches at all. See also 'memory threshold command.'

Default is OFF.

icmp status
Display statistics about the Internet Control Message Protocol (ICMP), including the number of ICMP messages of each type sent or received.

icmp timeexceed [<ON | off>]
Allows 'time exceeded' message to be sent when the ttl of an ip packet to be routed becomes zero. When turned OFF, no message is sent which allows the system to become invisible for 'traceroutes', etc.

icmp trace [on | off]
(B) Display or set the flag controlling the display of ICMP error messages. These informational messages are generated by Internet routers in response to routing, protocol or congestion problems. This only functions when in console mode. Default is off.

The iface command is used to assign a callsign or alias or to (re)set a flag on a specific port or interface after the interface is attached. Global assignments, i.e., setting a particular value on ALL interfaces which do not already have a value assigned, are made using <function> only.

If a valid subcommand is given, it will be executed (see below). When no subcommand is given, display a list of interfaces, with a short status for each. See the '**ifconfig <iface>**' command for a description of the display.

ALL ax25 and MOST tcp parameters are now configurable per interface. The 'ax25 <cmd>' commands set the system default values and the 'ifconfig <iface> ax25 <command>' commands set or show the interface specific value(s). The 'tcp <cmd>' commands work in the same manner.

As a result of this change, 'ifconfig' NO LONGER takes multiple commands on one line. 'ifconfig ln0 netmask ffffff00 broadcast 255.255.255.255' is invalid. The command line must be separated into two commands as: 'ifconfig ln0 netmask ffffff00' and 'ifconfig ln0 broadcast 255.255.255.255'

ifconfig <iface> [<subcommands>]

When only **iface** is given, the interface status is displayed.

Interface status shows:

IP addr - the ip address assigned to this interface

MTU - the maximum transmission unit for this interface.

Link encap - the type of link protocol to send packets with over this interface (AX.25, NETROM etc.)

Paclen - if the interface is an AX.25 interface, this is the Paclen used for connections on this interface

flags - interface flags, the sum of all the options set with the various commands. See below.

netmask - the ip network mask. See elsewhere for a discussion.

broadcast - the ip broadcast address on this interface. Used when doing arp, etc.

sent ip - the number of ip packets sent on the interface

sent tot- the total number of packets sent (i.e. ip, ax.25, etc.)

sent idle - the elapsed time this interface hasn't transmitted any data.

recv ip - the number of ip packets received on the interface

recv tot- the total number of packets received (i.e. ip, ax.25, etc.)

recv idle- the elapsed time this interface hasn't received any data.

descr - a description of the interface

Interface flag values are the sums of the following options, and can be set or unset (i.e. toggled) with the following commands (See their individual descriptions for more)

command	value		description of flag
mode iface	DATAGRAM_MODE	0	/* Send datagrams in raw link frames */
	CONNECT_MODE	1	/* Send datagrams in connected mode */
netrom interface	IS_NR_IFACE	2	/* Activated for netrom use */
	NR_VERBOSE	4	/* broadcast routes verbose */
convers interface	IS_CONV_IFACE	8	/* Activated for conference call access */
ax25 bport	AX25_BEACON	16	/* Broadcast AX.25 beacons */
mbox hide	HIDE_PORT	64	/* Don't show port in mbox 'P' command */
ax25 digi	AX25_DIGI	128	/* Allow digipeating */
arp eaves	ARP_EAVESDROP	256	/* Listen to ARP replies */
arp poll	ARP_KEEPALIVE	512	/* Keep arp entries alive after time-out */

```
ax25 hport        LOG_AXHEARD       1024 /* Do ax.25 heard logging on
                                           this interface */
ip hport          LOG_IPHEARD       2048 /* Do IP heard logging on
this
                                           interface */
```

ifconfig <iface> ax25 [<subcommand>]
Sets the value for 'subcommand' per description in the ax25 commands. 'ifconfig <iface> ax25' by itself displays the following list of parameters and their values:

bctext
blimit
cdigi
irtt
maxframe
maxwait
paclen
pthresh
retry
timertype
t3
t4
version
window

ifconfig <iface> broadcast <addr>
Set the ip broadcast address of interface <iface> to **<addr>**.

ifconfig <iface> cdigi <call>
Set the 'crossband digipeater only' callsign. If this call is set, digipeating works independently from the 'ax25 digipeat' setting. Connections cannot be made to the cdigi call!

ifconfig <iface> description "descr"
This command sets the interface description to the string specified. If no **descr** is supplied (i.e. ""), the current description will be cleared. The description is displayed with the mailbox or nodeshell P command (if the interface wasn't hidden from that display). It is also shown in the **ifconfig** command.

ifconfig <iface> encapsulation <mode>
Sets the encapsulation for interface iface to slip or ax25. This should never be needed, since it is automatically executed when

interfaces are attached.

ifconfig <iface> forward <iface-2>

When a forward is defined, all output for interface **<iface>** is redirected to **<iface-2>**. To remove the forward, set **<iface-2>** to **<iface>**.

ifconfig <iface> ipaddress <addr>
Set the IP address to **<addr>** for this interface. Normally the ip address is assigned from the system **ip address** when the interface is first attached. However, it might be necessary to change it when a system acts as a ip-gateway.

ifconfig <iface> linkaddress <linkaddr>
Set the hardware dependent address for this interface. For AX.25 this is the callsign. If you want to allow cross band digipeating, give each port a different ax.25 call with this command.

ifconfig <iface> mtu <num>
Set the maximum transfer unit to **<num>** bytes.

ifconfig <iface> netmask <address>
Set the sub-net mask for this interface. The **<address>** takes the form of an IP address with 1's in the network and subnet parts of the address, and 0's in the host part of the address. Sample: ifconfig ec0 netmask 0xffffff00 for a class C network (24 bits). This is related to the '**broadcast**' subcommand. See also the '**route**' command.

ifconfig <iface> paclen <num>
Set the AX.25 paclen for this interface. This is useful if you want to use a value different from the default as set with the '**ax25 paclen**' command; e.g., if you have a port with an HF link, you might want to set it to 128. You can also set it to greater than 256 if you have a high speed port. (This command only works for interfaces that can carry AX.25 connections, i.e., it is not for SLIP interfaces, etc.)

NOTE1: The AX.25 V2 specification specifies a MAXIMUM of 256 for paclen. If you have a paclen > 256, you may run into problems when interfacing to other non-NOS systems (in particular G8BPQ-based systems.)

NOTE2: The value of paclen influences NETROM behavior if the interface is activated for netrom with the **'netrom interface'** command! If the paclen for this interface is smaller than any other (netrom active) paclen, the netrom mtu value will be set to this paclen - 20 ! This is to assure that you will not get fragmentation at the ax.25 level when trying to send large data packets over netrom connections. AX.25 V2.1 fragmentation is presently handled only by NOS and derived code as far as is known. Other systems, such as TheNet, BPQ, MSYS, etc., may not include proper handling of V2.1 fragmentation.

What the preceding means is, if you have a VHF port with paclen 256, and an Hf port with paclen 128, and BOTH are active with netrom, the netrom mtu will be 108 !

ifconfig <iface> tcp [<command>]

Sets or displays the 'tcp' command parameters for <iface>. 'ifconfig <iface> tcp' by itself displays the following list:

irtt
maxwait
mss
retries
syndata
timertype
window

OUTGOING tcp connections get the values for the interface on which the initial sync packet ('connect request') is routed out. INCOMING tcp connections get the values for the interface the initial request arrives on.

System default TCP parameters must be set PRIOR TO attaching interfaces. After attaching interfaces, use the 'ifconfig <iface> tcp' commands to set the interface.

Causes the mailindex program to be run and re-establish the indexes for the mailbox. 'index *' indexes ALL mailbox files.

{JNOS40}

This shows or sets the info message that will be shown when users

connect to the system alias, and by the user 'I' command.
info "Corvallis area packet network controller, Vineyard Mt. 1600Ft asl. Run by the OSU ARC"

These commands are used for the Internet Protocol service.

ip address [<addr>]
Display or set the default local IP address. This command must be given before an **'attach'** command if it is to be used as the default IP address for the interface.

ip heard
Display the ip-heard list. This shows the recently heard tcp/ip systems. See also the **'ip hport'** command.

ip hport [<iface>] [on | OFF]
Display or set the ip-heard facility. If no argument is given, show the interfaces on which ip-heard is currently active. If **<iface>** is given, shows the status of the ip-heard flag for the given interface. If <iface> <on|off> is given, it will set the flag on or off. Default is off.

If this flag is on, ip heard frames will be logged in a table. This table can be shown with the **'ip heard'** command or with the nodeshell **'IHeard'** command. Ip-heard logging on ax.25 interfaces logs all ip stations heard on the port, even if the system wasn't directly involved in the ip activity. For non-ax.25 interfaces, only ip frames that we were actively involved in (i.e. that we routed) are logged. (this difference is due to code internals)

ip hport port1 on

ip hsize [n]
Display or set the maximum size of the Ip heard table. 0 means no limit._ Default=8

ip rtimer [<seconds>]
Display or set the IP reassembly time-out. Default = 30 seconds

ip status
Display Internet Protocol (IP) statistics, such as total packet counts and error counters of various types.

ip ttl [<hops>]

(B) Display or set the default time-to-live value placed in each outgoing IP datagram. This limits the number of switch hops the datagram will be allowed to take. The idea is to bound the lifetime of the packet should it become caught in a routing loop. You should make the value slightly larger than the number of hops across the network you expect to transit packets. The default is set at compilation time to 255, the official recommended value for the Internet.

{JNOS40}

(B) The keep command allows you to add commands to the configuration after the system is up and running. **<other command>** is a valid command line, which will be executed and then stored in battery backed ram. The next time the system restarts, all commands put in eprom with the cfg.exe program will be executed. Next any commands saved with the 'keep' command will be executed. **'Keep'** with no arguments will show the commands stored.   Each line will show a line number that should be used if you want to delete the line later with the **'delete'** command.

```
 #To add another ip route after the system is up, and
 #have it remembered after the next power outage,
```

'keep route add 1.2.3.4 port1'

{JNOS40}

(B) Toggle use of the status LEDs. Primarily intended to save power.

{JNOS}

This command allows peeking into a user session on the BBS or into any non-local socket. 'look [user | socket#]' brings up a window which follows the specified user or socket. If looking in on a bbs user, you can initiate a 'chat' session with that user.

```
  There are a few commands available:

  /? or /h - will show a sort help line
  /m <msg> - send a message to user. (only if 'looking' at user)
             User sees: '<sysop>: message text'

  /c       - initiate a 'chat' with user. The bbs will suspend
             while you talk with the user.
  /q or /b - if in 'chat' mode, finishes it and returns the user
             to the bbs.
             if in 'look' mode, finishes looking at user/socket.
```

Note: if not in 'chat' mode, all non-command strings will be ignored.

Looking at non-bbs sockets might be helpful debugging things or seeing what an ftp user or a smtp user is doing.

Note that often you won't see what you expect: e.g., ftp user sockets don't show the data transfered including directory listings because data transfer occurs on a separate socket.
(Not seeing it is NOT a bug, for once...8))

Without a subcommand, display the current users of the nodeshell. With a valid subcommand, it will execute the following commands.

mbox alias [<alias>] ["cmd"]

Set or show alias commands for the mailbox/nodeshell. To use an alias, the user must type the full alias which will then cause "cmd" to be executed. If the user types 'A', a list of sysop-defined aliases will be displayed. 'alias' is 6 characters maximum. "cmd" is 64 characters maximum. Note that "cmd" is in double quotes.

'mbox alias' displays the current aliases.
'mbox alias <alias>' displays only <alias> if it is defined.
'mbox alias <alias> <cmd>' adds, deletes, or redefines <alias>

Example: To add a new BBS command -

mbox alias bbs "c port bbscall"

To delete an alias-

mbox alias myalias ""

To redefine an alias-

mbox alias bbs "c newport newbbscall"

mbox attend [ON|off] {JNOS}

This displays or sets the attend flag. If set, users can initiate chat with the sysop via the O)perator mailbox command. (You need to have started the ttylink server for this !)

mbox bbsonly [iface] [on | OFF] {JNOS}

Specify whether iface is only accessible to stations which have the BBS flag set.

mbox convers [on|OFF]

Display or set the access to the conference bridge. The mailbox or nodeshell 'CONV' command is disallowed when off. Default is ON.

mbox fwdinfo [string]   {JNOS}

Displays or sets the string that is used in the BBS R: header when forwarding mail to other BBSes.

mbox haddress [string]   {JNOS}

Displays or sets your system's hierarchical bbs address. As of version 1.08, you must include the bbs call with the hierachical address.

Example:

mbox haddress wg7j.or.usa.na

mbox header [on|OFF]   {JNOS}

Explicitly_turns the R: line in the message header on or off. With mbox header off, regular users can forward from jnos to another system designated as a full-service bbs without leaving a trace to their system. This avoids having a downstream bbs improperly identify the originating station as a bbs.

The elements of the R: header are optional as of jnos107. If not set the line will be empty.

NOTE: 'mbox header off' also turns off 'mbox thirdparty.'

mbox hideport [<iface>] [on | OFF]

Display or set a port that should not be displayed when the 'P' command in the mailbox or nodeshell is given. Display if available to users with sysop permissions. This command is useful for ports that should be backbone linking only, etc. Note that currently users can still do ax.25 connections via that port, if they know the name. Default is OFF, the port is displayed.

mbox hideport port1 on

mbox kick {JNOS}

This is an immediate command. It forces the system to start a new bbs forwarding cycle.

mbox last {JNOS40}

Shows previous users of the nodeshell.

mbox mailfor [interval] and {JNOS}
mbox mailfor exclude [areaname areaname ...] {JNOS}

Interval is in seconds. At the end of each interval, the system reads all non-area mailboxes and checks them for unread mail. Next, a beacon is sent on all active interfaces. (see 'mbox mport' command). The beacon is addressed to the AX.25 address 'MAIL'. The data in this packet contains a list of the mailboxes with unread mail in the format 'Mail for: <user> ... '. Systems such as LAN-LINK can trigger mail-snatches from this beacon.

'mbox mailfor' will show the status of the timer and list mailboxes with unread mail.

In certain cases you might not want a private area to show up in the mail beacon; e.g., private areas to be forwarded to another bbs.
You can exclude areas from the beacon with the
     'mbox mailfor exclude' command.

'mbox mailfor exclude area1 area2 area3 ...'

will disable these area names from the beacon. You can give multiple exclude commands to build the list.

A simple 'mbox mailfor exclude' shows the list.

mbox maxmsg [n]   {JNOS}

This displays or set the maximum number of messages in a mailboxfile, for both private and public (i.e., area-) mail. If there are more than n messages, the user will only see the first n messages.

mbox motd [string] {JNOS}

This displays or sets the message of the day which is displayed when regular users log on to the system.

mbox mailstats {JNOS}

This command shows how many messages have been read and sent by users and how many messages have been received from and forwarded to BBSes.

mbox mport [<iface>] [on | off]   {JNOS}

Displays or sets the interfaces for 'MAIL' beacons. Set this flag for each port you want the mail beacon to be sent on.

mbox newmail [ON|off]   {JNOS}

When ON, users will be notified during login which areas have new mail since the last time that user logged out. This function uses a time stamp on a status file for each message area. The information shown can later be repeated with the 'AN' mailbox command.

mbox nobid [on | OFF]   {JNOS}

Set whether to accept or refuse bulletins with NO BID. Off means refuse if there is no BID. On means accept regardless of BID. Default is OFF (refuse)

mbox nrid [[ON|off] {JNOS}

When set, the first time a user logs onto the system, the system will add the netrom node id in NODE:CALL format to the prompt. Users can then change it with the mailbox XN command. The system will use the new prompt format as set by the user the next time the user logs in.

mbox past {JNOS}

Displays the users that have logged on since the system has been running.

mbox password <newpassword>   {JNOS}

This sets a new remote sysop password.

mbox prompt [ON | off]   {JNOS}

Turns the command prompt display of the current directory on or off. Default is ON.

mbox prompt [0|1|2|3|4|5|6|7] {JNOS40}

The node prompt is configurable by selecting a value in the range 0 to 7.

```
0 = no prompt
1 = netrom-id prompt     e.g.,  JNOS40:WG7J-1}
2 = command prompt       e.g.,  ??,A,B,C,E...>
3 = combined prompt      e.g.,  JNOS40:WG7J-1}??,A,B,C...>
```

mbox qth [location] {JNOS}

This displays or sets the location of your system, and uses it in the R: headers when doing bbs forwarding.

mbox register [ON | off]
Enable or disable user registration. If the user gives an e-mail address during registration, messages sent will include a 'Reply-To:' header with that e-mail address. Setting this command to OFF will prevent the 'please register' message from being sent to new users.

mbox secure [on |OFF]   {JNOS}
This displays or sets the mailbox secure flag. If set, only users coming on over netrom and ax25 connections can make gateway connects if they have the right priveleges. If not set, anyone with the right privs can do a telnet connect.

mbox sendquery [ON|off] {JNOS}
If on, users will be queried if they really want to send the message after they have typed the ^Z or /ex when sending a message. 'N' or 'n' will abort, anything else will send the message.

mbox smtptoo [on | OFF] {JNOS}

This displays or sets the smtp header flag. If set, the system will include most smtp headers when forwarding the message via bbs forwarding. When not set, the headers will be stripped (default).

mbox stats
Shows statistics on the number of logins, uplink and downlinks {JNOS40}.
Displays the current users of the system {NOS}.

mbox sysoponly [iface] [on | OFF] {JNOS}
Specify whether port is accessible only by stations with SYSOP flag set. Default is off.

mbox timer [<nnnn>] {JNOS}
Entering only the subcommand displays the forwarding setting and countdown timer value. [<nnnn>] sets the interval between forwarding attempts in seconds. Default = 0 (No forwarding).

mbox tdisc [<nnnn>] {JNOS}

This command sets the mailbox inactivity timer in seconds. If no user input has been received for nnnn seconds while connected to the mailbox, the user will be disconnected. Default = 0 = no timeout.

mbox thirdparty [on|OFF] {JNOS}

Set or display whether 3rd party traffic may be handled through the mailbox. Default is OFF.

See also 'mbox header' command.

mbox tmsg [<string>] {JNOS}

Display or set the 'telnet message'. This is the message sent to telnet connections before the login prompt is sent!

mbox trace [on|OFF] {JNOS}

This displays or sets the value of the trace flag. If set, the mailbox is verbose about certain aspects of the forwarding cycle.

mbox utc [n] {JNOS}

Display or set the offset from UTC. [n] is used to calculate the current time for the forwarding header.

mbox zipcode [<nnnnn>] {JNOS}

Set or display the system's postal zipcode. This is used in the R: line when forwarding.

These commands are used for memory allocation.

memory freelist

Display the storage allocator free list. Each entry consists of a starting segment, in hex, and a size, in decimal bytes.

memory ibufsize [<size>]

(B) Display or set the size of the buffers in the interrupt buffer pool. The size should be set to the largest type of buffer plus a header size of 8. For example: If your ax.25 is the only interface and a packet length of 512 is defined, the ibufsize should be 512 + 72 + 8 = 592 . The 72 is the ax.25 header (source , destination, 8 digipeaters, 1 control byte and 1 pid byte). Default is 600. See also the section on **INTERFACE BUFFERS**.

memory minalloc [<bytes>]

Set the minimum number of bytes to allocate per malloc() call. Setting a small value (32 or 64) might allow the realloc scheme to work more effectively by preventing excessive memory fragmentation. Default = 0, no minimum allocation size.

memory nibufs [<number>]

(B) Display or set the number of interrupt buffer pool buffers. If the number of buffers is set, the statistics in the 'memory status' display are reset for number of interrupt buffer fails. The minimum available value is set to the requested number of buffers. A rule of thumb for the number of buffers is to watch the statistics and keep a minimum of 2 free buffers. Increase or decrease as required. Default is 10. See also the section on **INTERFACE BUFFERS.**

memory sizes

Display a histogram of storage allocator requested sizes. Each histogram bin is a binary order of magnitude (i.e., a factor of 2).

memory status

Display a summary of storage allocator statistics.

```
heap size 52560, avail 12880 (24%), morecores 150, coreleft 5872
```

The first line shows the total size of the internal heap, the amount of memory available on the internal heap with the percentage of the total heap size, next the number of times memory has been requested from the Operating System, and the amount of memory the OS has left over.

```
allocs 16706, frees 16389 (diff 317), alloc fails 0, invalid frees 0
```

Next, the number of times memory has been allocated, and has been freed is shown. The difference is the number of buffers currently allocated. Alloc fails show up when the system is running out of memory resources.
Invalid frees mean that memory was overwritten, and indicates the system is about to lose sanity...

```
garbage collections yellow 0, red 0
```

Garbage collections free memory from network control structures that could not have otherwise been freed. Yellow garbage collections are started when the total available memory, i.e. avail+coreleft, becomes smaller then memthresh. Red garbage collections indicate that available memory got below memthresh/2

interrupts-off calls to malloc 0, free 0

These should never be other then 0. They indicate calls to the memory allocator with interrupts off. These requests should be handled by the interrupt buffer pool. If these values are non-zero, you most likely have a problem.

Intqlen 9, Ibufsize 600, Iminfree 9, Ibuffail 0

This line shows the current number of interrupts buffers in the interrupts buffer pool, the size of each buffer, and the minimum number of free buffers. If this last number gets close to, or becomes zero, you should increase the buffer pool size with the 'memory nibuf' command. The statistics are reset when this command is executed.

memory thresh [<size>]

(B) Displays or sets the memory threshold size in bytes. If free memory gets below this value, no more new connections can be started and no new connections will be accepted. This is an attempt to preserve the system's sanity.

Control the default transmission mode on the specified AX.25 interface. In **datagram** mode, IP packets are encapsulated in AX.25 UI frames and transmitted without any other link level mechanisms, such as connections or acknowledgments.

In **vc** (virtual circuit) mode, IP packets are encapsulated in AX.25 I frames and are acknowledged at the link level according to the AX.25 protocol. Link level (i.e. AX.25) connections are opened as necessary.

In both modes, ARP is used to map IP to AX.25 addresses.

(Currently not implemented in NOS: the defaults can be overridden with the type-of-service (TOS) bits in the IP header. Turning on the "reliability" bit causes I frames to be used, while turning on the "low delay" bit uses UI frames. The effect of turning on both bits is undefined and subject to change.)

In both modes, IP-level fragmentation is done if the datagram is larger than the interface MTU. In Virtual Circuit mode, however, the resulting datagram (or fragments) is further fragmented at the AX.25 layer if it (or they) is still larger than the AX.25 <paclen> parameter. In AX.25 fragmentation, datagrams are broken into several I frames and reassembled at the receiving end before being passed to IP. This is preferable to IP fragmentation whenever possible because of decreased overhead (the IP header isn't repeated in each fragment) and increased robustness (a lost fragment is immediately retransmitted by the link layer).

Display or set the current message of the day. This message is shown to users logging in via telnet or via connections to the system's alias. It is also shown by the nodeshell **'Motd'** command.

These commands influence netrom behavior.

netrom acktime [<milliseconds>]

(B) Displays or sets the ack delay timer, similarly to ax25 t2. Default is 8000ms (i.e. 8 seconds).

netrom alias <aliascall>

This sets the netrom alias call for this station. Other stations can connect to the ax25 callsign and to the Netrom alias (when set). The alias is broadcast with a Netrom broadcast. If netrom is not activated, you can use the **'ax25 alias'** command to set the alias callsign, such that users can still connect to the alias, even though netrom activities are not allowed.

netrom bcnodes <iface>

Initiates an immediate broadcast of nodelist on **<iface>**. Verbose behavior is controlled by the **'netrom interface'** command.

netrom bcpoll <iface>

Initiates a poll sent to the named interface. This poll will request a netrom routes broadcast from other nodes, so that the routing table can be updated. This is automatically done any time an interface is activated (or changed) for netrom. This should speed up route discovery at startup. This is NOT currently implemented in Thenet or Net/Rom nodes, but works with Data Engines running JNOS40 or with systems running JNOS v1.05 or later.

netrom choke [<milliseconds>]

(B) Display or set the time breaking a send choke. Choke is the term netrom uses for flow control conditions. Default is 180000 ms (180 seconds.)

netrom derate [ON|off]

(B) Display or set automatic derating of netrom routes on link failure. Default is on.

netrom hidden [on|OFF]

Display or set the hidden node flag. Nodes with aliases starting with the '#' character are displayed using the command 'N *'. Default is OFF.

netrom interface [<iface> <quality> [n]]   {JNOS}

Activate **<iface>** as a netrom interface. **<quality>** can be between 1 and 255. Interfaces are activated using verbose routes broadcasting by default, meaning that they broadcast all known routes. This can be changed by adding the optional **'n'** parameter to the command. Then only the system itself will be announced in a broadcast, not the known routes.

If the paclen of the interface is smaller then the netrom mtu + 20, then the netrom mtu will be set to paclen-20 . This is to avoid fragmentation, causing incompatibilities with none-NOS based Netrom nodes. See the '**ifconfig <iface> paclen**' command for more.

If **<iface>** has already been activated as a netrom interface, re-issuing the command will set **<quality>** and **[n]** to the new values. '**netrom interface**' will show the currently active netrom interface.

Each time an interface is activated or changed, a broadcast poll will be sent out on the interface. This minimizes the route discovery time at startup, and will update routes when the interface quality has changed.

netrom interface <iface> <quality> [<min-bc-quality>] {JNOS40}

Set or display the netrom interface(s) for the node. This command is essentially the same as the command for JNOS.

Only nodes of min-bc-quality and better will be broadcast in a nodes boradcast on this interface. If <min-bc-quality> is not set (no value assigned), then the 'netrom minquality' value is used. If <min-bc-quality> = 0, then no nodes are broadcast except this one (same as 'n' parameter above.)

Examples:

```
netrom interface ax0 224 180  -> nodes >= 180 are broadcast
```

```
     netrom interface ax0 224 0    -> only ourself is broadcast

          netrom interface ax0 224      -> nodes >= 'minquality' are
                                           broadcast
```

netrom irtt [<milliseconds>]
(B) Display or set the initial round trip time. Default is 45000ms, i.e. 45 seconds.

netrom kick <nrcb>
Give the control block a kick to get activity going again.

netrom load {JNOS}
Retrieves the last set of netrom destination data saved with the 'netrom save' command.

netrom minquality [<minqual>]
(B) Display or set the minimum quality for recognizing a node entry. Entry's below this value are not considered valuable for usage. Default is 50.

netrom mycall <call>
Displays or sets the call to be used by the netrom interface. Note: this is a shortcut for the '**ifconfig netrom linkaddress**' command. It defaults to the '**ax25 mycall**' value.

netrom neighbor
Display all known netrom neighbors.

netrom nodefilter <subcommands>
Manipulate node filtering.

netrom nodefilter add <neighbor> <iface>
Add **<neighbor>** on port **<iface>** to the filter table. See the '**netrom nodefilter mode**' command to determine the manner to handle node updates from <neighbor>.

netrom nodefilter mode [none|accept|reject]
Display or set the initial node filter scheme. '**none**' accepts all netrom routes and is the default. '**accept**' accepts routes only from nodes defined with the '**netrom nodefilter add**' command. '**reject**' does not accept routes from any nodes defined with '**netrom nodefilter add**'

netrom nodefilter drop <neighbor> <iface>
Delete the node **<neighbor>** on interface **<iface>** from the filter table.

netrom nodetimer [<seconds>]
(B) Display or set the interval to transmit the nodes list. If you want to use other than the default, you must first attach the netrom interface with '**attach netrom**' and then set the new nodetimer value. Default is 1800 seconds (half an hour).

netrom obsoinit [#}
(B) Display or set the initial obsolescense count for direct routes (Routes table entries) to other systems. Default is 6

netrom obsominbc [#]

(B) Display or set the minimum obsolescense count a route should have to be included in routes broadcasts originating from this system. Default is 4.

netrom obsotimer [<seconds>]
(B) Display or set the time a node obsolescence count gets decremented. If you want to use other than the default, you must first attach the netrom interface with '**attach netrom**' and then set the new obsotimer value. Default is 1800 seconds.

netrom promiscuous [OFF|on]
(B) Enables nodes with a path quality greater than defined with minquality. If on, all nodes are received regardless of nodefilter mode. Default is off.

netrom qlimit [<nnnn>]
(B) Display or set the maximum queue limit for choke to occur. Similar to ax25 window. Default is 512 bytes.

netrom reset <nrcb>

Remove the control block. You can find the control block with the **'netrom status'** or **'socket'** commands.

netrom retries [<nn>]

Display or set the maximum number of retries on connect, disconnect or data. Default is 3.

netrom route <subcommands>

Netrom routing commands. Routes can be marked as 3 types in the various route displays:

'P' - a permanent route set with the **'netrom route add'** command
'B' - a route received through a nodes broadcast from a neighbor node
'R' - a recorded route from an incoming packet from a not previously known netrom node

netrom route add <alias> <call> <iface> <quality> <neighbor>

Add a permanent netrom route. The new route is to netrom system **<alias>** with call **<call>**, and the route is on interface **<iface>** with quality **<quality>** via the neighbor **<neighbor>**.

netrom route add salem af7s-1 port1 178 k7uyx-1

A route to a direct neighbor looks like:

netrom route add crv k7uyx-1 port1 192 k7uyx-1

netrom route drop <destination> <neighbor> <iface>

Delete the netrom route to call **<destination>** via neighbor **<neighbor>** on **<iface>**.

netrom route info [<destination>]

Display the route a packet would take to get to **<destination>**. If <destination> is not given, information about all known netrom nodes is displayed. 'netrom route info' and 'netrom route info *' are equivalent commands.

netrom status
Display all netrom connections.

netrom tdisc [secs]
(B) Display or set the Netrom Link "redundancy" timer. Value is in seconds. When no data exchange has happened during this time the link is reset and closed. Default is 900 seconds (15 minutes).

netrom timertype [exponential|LINEAR]
(B) Displays or sets the type of backoff used on netrom retries. Default is **linear**.

netrom ttl [<hops>]
(B) Display or set the maximum number of hops a frame might take before being discarded. Default is 10.

netrom window [<frames>]
(B) Display or set the size of the sliding window. This is the largest send and receive window we might negotiate. Default is 2.

Displays the netrom serial interface statistics. This is only valid if the serial port was attached in NRS mode. See the **'attach'** command for more.

Invoke a device-specific control routine. A simple **'param <iface>'** will give a list of available parameters, and their current values, for the interface **<iface>**. **<param>** can be the literal description of the parameter.

On the serial interface, this sends control packets over the serial port. Example, **'param port1 txdelay 255'** will set the keyup timer (type field = txdelay) on the KISS TNC configured as port1 to 2.55 seconds (255 x .01 sec).

On the radio ports, all timing parameters use a 10ms tick count, e.g. 30 means 300ms. Some commonly used options are:

slottime - the channel access slottime

maxwait   - maximum time to defer transmissions

txdelay	- keyup delay before sending data
persist	- the csma persistence (range 0-255)
maxkey	- maximum time to allow transmitter to be keyed (range 0-65000)
txtail	- time to keep transmitter keyed up after end of packet

NOTE: the **txtail** parameter is NOT settable, but is always 4 characters duration at the port's speed. The value shown is calculated from the speed given when the interface is first attached. When you use the trick previously mentioned to allow interchanging of DE1200 and DE9600 modems, the txtail shown when a DE9600 modem is used is WRONG. However, the actual value used is still correct.

{JNOS40)

Set a new sysop password. 30 chars maximum. Default is '0123456789' .

Verify a tcp host is alive. <host> is the address to ping, 'timeout' is option and is in seconds. Default timeout = 30.

popmail <subcommands>

popmail addserver <host> [<seconds>] [hh:mm-hh:mm] <protocol> <mailbox> <username> <password>

Add hostP as a pop server. When seconds is given, a timer is started to query the host with that interval for mail. If not specified no quering to the pop host will be started. You have to do that manualy with a kick. When hh:mm is given then only in that exact timeframe are queries to the host made (allowed). Protocol is either POP2 or POP3, depending on the mail service the host is providing.

Note: pop2 is superceded by pop3.

Mailbox is the mailbox name on the host where mail has to be picked up.

Username and password are this system's validation parameters for the host.

Note: On entering this command the host name is looked up. If nonexistent, an error message is displayed.

popmail dropserver <host>

Drops host from the list of pop servers to be queried. All references to the entry are deleted from the current system.

popmail kick <host>

Starts a pop session with host to retrieve mail. This command is needed when no interval is specified with the popmail addserver command.

popmail list

Lists the current popmail server table.

popmail quiet <yes|no>

Displays or sets the notification of new mail ariving via pop.

popmail trace <level>

Displays or sets the trace level of pop sessions. Current trace levels are:

0 - No tracing

1 - Serious errors reported

2 - Transient errors reported

3 - session progress reported

Note that tracing only goes to the log file.

{JNOS40}

Displays the serial port and internal radio ports statistics.

If the serial port is used as a network interface, it will display port statistics. For receive, it shows the number of received characters, the high count in the receive buffer and the number of characters that were dropped due to buffer overruns. The buffer in question here is the receiver ring buffer that is setup in the attach command (see 'attach 1'). If the high value is close to the buffer size, of there are overrun characters, increase the buffer size. (i.e. re configure the system with a new eprom) For transmit, it shows the number of characters transmitted, and the number of packets that are waiting to be sent.

It will show the type of modem connected to each radio port, and other variables. Variables shown are:

Modem type: A is DE1200 and compatible, B is DE9600 and compatible, C is external K9NG etc., and D is loop back modem.
State: Indicates driver state, where

0=IDLE	Transmitter off, no data pending
1=DEFER	Deferring transmit
2=KEYUP	Permission to keyup the transmitter
3=KEYWT	Transmitter switched on, in txdelay wait time

```
     4=ACTIVE Transmitter on, sending data
                    5=FLUSH  CRC sent - attempt to start next frame
                    6=TAIL   End of transmission, sending tail
Queued:             the number of packets still waiting to be
       transmitted.
Sent:               the number of packets transmitted.
Txerr:              the number of maximum key-up failures.
Rcvd:               the number of valid packets received.
Rxerror:            the number of erroneous packets received.
Rxspace:            the number of failed attempts to allocate a
       receive buffer.
Exints:             the number of External interrupts, this includes
       the next two.
Brkints:            the number of receiver break-abort interrupts.
Dcdints:            the number of Data-Carrier-Detect interrupts.
Spints:             the number of Special interrupts. (Spints - Error
       should equal Rcvd !)
```

Display process status information. The first line shows the time the system has been running, the active stack segment, and the interrupt stack usage. Next it displays all processes in the system. The fields are as follows:
PID - Process ID (the segment of the process descriptor).

SP - The current value of this process' stack pointer.

stksize - The size of the stack allocated to this process.

maxstk - The apparent peak stack utilization of this process. This is done in a somewhat heuristic fashion, so the numbers should be treated as approximate. If this number is close to the stksize figure, the system is likely to crash. **Please notify the author if you examine such a situation.** (The program should be recompiled to give the process a larger allocation when it is started.)

event - The event this process is waiting for, if it is not runnable.

fl - Process status flags. There are three: I (Interrupts enabled), W (Waiting for event) and S (suspended). The I flag is set whenever a task has executed a pwait() call (wait for event) without first disabling hardware interrupts. Only tasks that wait for hardware interrupt events will turn off this flag; this is done to avoid critical sections and missed interrupts. The W flag indicates that the process is waiting for an event; the 'event' column will be non-blank. Note that although there may be several runnable processes at any time (shown in the 'ps' listing as those without the W flag and with blank event fields) only one process is actually running at any one instant (The Refrigerator Light Effect says that the **'ps'** command is always the one running when this display is generated.)

Sets the remote server password. Maximum 30 chars. Default is '0123456789' The remote server is compatible with the client in the NOS.EXE program. The remote server provides a way to remote reboot the system without having to connect to it. You need to address the remote server with the remote client in the NOS.EXE program. Please see the documentation of your favorite flavor of NOS.EXE for the syntax for the remote command. Sending a valid password with a command will cause the command to be executed.
Currently the server will execute the following commands:
reset - restarts the system with a 'coldboot'; i.e. all variables in battery backed ram are reset to their default values

exit - restarts the system with a 'warm boot'; i.e. all variables in battery backed ram remain the same.

{jnos40}
This will restart the system. This is an immediate command ! (i.e. no 'are you sure? warning...) The system will 'warm-boot'. Thus all variables in battery backed ram will keep their value as last set.

Both JNOS and JNOS40 now have RIP-2, a newer version of RIP. RIP-2 is not part of the "distribution compile" of JNOS from WG7J, but it is included in the JNOS40 distribution.

The commands given here are used for RIP. After this list of commands is the list for RIP-2. The RIP-2 implementation includes compatibility with RIP-1. The sets of commands are separated here to improve clarity.

rip accept <gateway>
Remove the specified gateway from the RIP filter table, allowing future broadcasts from that gateway to be accepted.

rip add <hostid> <seconds> <flags>
Add an entry to the RIP broadcast table. The IP routing table will be

sent to **<hostid>** every interval of seconds. If **<flags>** is specified as 1, then "split horizon" processing will be performed for this destination. That is, any IP routing table entries pointing to the interface that will be used to send this update will be removed from the update. If split horizon processing is not specified, then all routing table entries except those marked "private" will be sent in each update. (Private entries are never sent in RIP packets). If flags is 2, the broadcast will also advertise a route to the system itself. Flags are accumalative, ie a value of 3 will mean both "split horizon" and "me too". See also the '**route**' command.

Triggered updates are always done. That is, any change in the routing table that causes a previously reachable destination to become unreachable will trigger an update that advertises the destination with metric 15, defined to mean "infinity".

Note that for RIP packets to be sent properly to a broadcast address, there must exist correct IP routing and ARP table entries that will first steer the broadcast to the correct interface and then place the correct link-level broadcast address in the link-level destination field. If a standard IP broadcast address convention is used (e.g. 44.26.0.0 or 44.26.255.255) then chances are you already have the necessary IP routing table entry (unusual subnet or cluster-addressed networks may require special attention!) However, an '**arp add**' command will be required to translate this address to the appropriate link level broadcast address; For example, **arp add 44.255.255.255 ax25 qst-0**
for an AX25 packet radio channel. (If there are multiple AX25 interfaces, make a unique address for each interface.)

rip drop <dest>
Remove an entry from the RIP broadcast table.

rip kick {JNOS}
Immediate command to send a rip update.

rip merge [on|off]
(B) This flag controls an experimental feature for consolidating redundant entries in the IP routing table. When rip merging is enabled, the table is scanned after processing each RIP update. An entry is considered redundant if the target(s) it covers would be routed identically by a less "specific" entry already in the table. That is, the target address(es) specified by the entry in question

must also match the target addresses of the less specific entry and the two entries must have the same interface and gateway fields. For example, if the routing table contains

```
Dest            Len      Interface       Gateway   Metric  P Timer  Use
44.2.3.4        32       ax0             44.96.1.2    1    0   0     0
44.1.2.3        24       ax0             44.96.1.2    1    0   0     0
```

then the first entry would be deleted as redundant since packets sent to 44.2.3.4 will still be routed correctly by the second entry. Note that the relative metrics of the entries are ignored.

rip refuse <gateway>
Refuse to accept RIP updates from the specified **<gateway>** by adding the gateway to the RIP filter table. It may be later removed with the **'rip accept'** command.

rip request <gateway>
Send a RIP Request packet to the specified **<gateway>**, causing it to reply with a RIP Response packet containing its routing table.

rip status
Display RIP status, including a count of the number of packets sent and received, the number of requests and responses, the number of unknown RIP packet types, and the number of refused RIP updates from hosts in the filter table. A list of the addresses and intervals to which periodic RIP updates are being sent is also shown, along with the contents of the filter table.

rip trace [0|1|2]
(B) This variable controls the tracing of incoming and outgoing RIP packets. Setting it to 0 disables all RIP tracing. A value of 1 causes changes in the routing table to be displayed, while packets that cause no changes cause no output. Setting the variable to 2 produces maximum output, including tracing of RIP packets that cause no change in the routing table.

rip ttl <seconds>
(B) Displays or sets the time to live timer to **'seconds'**. Normal time-out value is 240 seconds. This is not the ttl in a rip broadcast (16 = infinite). Set this timer before starting rip. Change this timer only in cooperation with your surrounding nodes. Default is 240 seconds.

End of RIP-1 commands.

****

The following text is provided by N0POY who did the NOS implementation of RIP-2.
This document covers the implementation of RIP-2 (RFC 1388) in NOS. Specifically the WG7J version of NOS. RIP-2 is an enhanced version of the RIP protocol (RFC 1058). RIP and RIP-2 are an interior gateway protocol (IGP). RIP-2 for NOS was implemented by Jeff White, N0POY.
This documentation is for the beta release V0.9 of RIP-2

RIP-2 Features

The NOS implementation implements all features of the normal RIP protocol (RFC 1058) and all features of the RIP-2 protocol (RFC 1388) except multicasting (which NOS does not currently implement) and Route Tags (NOS does not implement any EGPs).

Features include:

Routing Domains

Authentication

Proxy routing

Filtering of naughty nodes

Optional refusal of a default route

Enhanced logging and tracing

Route subnet masks correctly maintained

Optional refusal to accept older RIP version broadcasts

Mixing of RIP-1 and RIP-2 support

NOS RIP COMMANDS

RIP ACCEPT <gateway>

The RIP ACCEPT command resumes the acceptance of RIP broadcasts from a specific node given in the <GATEWAY> field.

```
RIP ACCEPT 192.55.248.1  or
RIP ACCEPT skeggi.tcman.ampr.org
```

RIP ADD <DEST> <INTERVAL> [<FLAGS>] [<RIPVER>] [AUTH <PASSWORD>] [RD <routing domain>]

The RIP ADD command adds a node to the list of stations that are to be broadcast to with the local nodes routing table.

<DEST> is the destination node, usually a broadcast address.
<INTERVAL> is the number of seconds between broadcasts.
<FLAGS> are the RIP flags used (see below for the flags), it is a hexadecimal number.
<RIPVER> is the version of the RIP broadcasts. This may be a 1 or 2.
The AUTH identifier preceeds the authentication password to be included with the RIP broadcasts to this destination.
The RD identifier preceeds the routing domain number. This number must range from 0 to 65535.

The authentication fields and routing domain fields are only valid with RIP-2 broadcasts. The password must be 16 characters or fewer. Printable ASCII characters are recommended, but not required.

RIP FLAGS

0x01 Do 'split horizon' processing

0x02 Include ourselves in the routing broadcast

0x04 Broadcast RIP packets (default type)

0x08 Multicast RIP packets (not implemented) (RIP-2)

0x10 Poisoned Reverse on

0x20 Authentication data to be included in broadcast (RIP-2)

Recommend flags are Split Horizon, and Poisoned Reverse or 0x11. Authentication and routing domain data entered here only applies to the outgoing RIP broadcasts. See RIP AUTHADD and RIP AUTHDROP for entering acceptable passwords and routing domains.

Example:

```
RIP ADD SKEGGI.TCMAN.AMPR.ORG 30 0x31 2 AUTH frodo RD 2

RIP ADD BIGGUS.TCMAN.AMPR.ORG 300 0x11 1
```

RIP PROXY <SRC> <DEST> <INTERVAL> [<FLAGS>] [AUTH <PASSWORD> [RD <ROUTING DOMAIN>]

The RIP PROXY command adds a node to the list of stations that are to be broadcast to with the local nodes routing table.

<SRC> is the node that the broadcast will "point" to.
<DEST> is the destination node, usually a broadcast address.
<INTERVAL> is the number of seconds between broadcasts.
<FLAGS> are the RIP flags used (see below for the flags), it is a hexadecimal number.
The AUTH identifier preceeds the authentication password to be included with the RIP broadcasts to this destination.
The RD identifier preceeds the routing domain number. This number must range from 0 to 65535.

The authentication fields and routing domain fields are only valid with RIP-2 broadcasts. The password must be 16 characters or fewer. Printable ASCII characters are recommended, but not required.

RIP FLAGS

0x01	Do split horizon processing
0x02	Include ourselves in the routing broadcast
0x04	Broadcast RIP packets (default type)
0x08	Multicast RIP packets (not implemented) (RIP-2)
0x10	Poisoned Reverse on
0x20	Authentication data to be included in broadcast (RIP-2)

Recommend flags are Split Horizon, and Poisoned Reverse or 0x11.

Authentication and routing domain data entered here only apply to the outgoing RIP broadcasts. See RIP AUTHADD and RIP AUTHDROP for entering acceptable passwords and routing domains.

Proxy RIP is tricky, complex and not needed for normal use. Do NOT use proxy rip unless you understand what you are doing. Proxy RIP's primary use would be to advertise routes to another machine that is aquiring routing information via another routing protocol. See RFC 1388 for further details.

RIP DROP <dest> [<DOMAIN>]

RIP DROP removes a routing broadcast entry. If a RIP-2 broadcast was entered, the correct routing domain needs to be entered, since it is possible to broadcast multiple routing domains to the same address.

Example:

```
RIP DROP SKEGGI.TCMAN.AMPR.ORG 2
```

RIP AUTHADD <interface> <routing domain> [<password>]

RIP AUTHADD adds an acceptable routing domain and optionally a password to a specific interface.

Example:

```
RIP AUTHADD ax0 2 frodo

RIP AUTHADD en0 3
```

RIP AUTHDROP <interface> <routing domain>

RIP AUTHDROP removes an acceptable routing domain (and password if any) from a specific interface.

Example:

```
RIP AUTHDROP ax0 2
```

RIP REJECT <version>

RIP REJECT is used to ignore older RIP broadcasts, as they may cause undesirable routing table alterations. The version number is the version number and below that are ignored. RIP version 0 (XNS RIP) is always ignored. The default is 0.

To ignore RIP-1 broadcasts: RIP REJECT 1 would do the job.

RIP FILTER <ON|OFF>
RIP FILTER will cause advertisements to the default route (0.0.0.0) to be tossed and ignored. By default this is off.

This can serve as a LID filter. Default routes should NOT be advertised, unless there is a specific reason (ie this machine is a gateway to the rest of the Internet).

RIP MERGE <ON|OFF>
RIP MERGE will cause overlapping routing entries to be merged into one routing entry.

For example N0BEL.TCMAN.AMPR.ORG is a route to 192.133.30.0/28, and 192.133.30.16/28, with merging on this would become a single entry of 192.133.30.0/27.

RIP REFUSE <gateway>
RIP REFUSE will reject all RIP broadcasts from the GATEWAY station. RIP ACCEPT is the opposite. By default all stations are accepted.

RIP REQUEST <GATEWAY>
RIP REQUEST asks the gateway station to send a routing table now, rather than waiting for periodic updates.

RIP STATUS
RIP STATUS will display various statistics for RIP-1 and RIP-2, RIP broadcasts, RIP refusals, and acceptable Interface, Domain and Password combinations. It also displays the refusing version level. The DEFAULT interface is for every interface. Thus unless removed, and RIP-2 broadcast with a domain of 0 does not require a password and will be accepted.

RIP TRACE <level> [<FILE>]

RIP TRACE will begin tracing RIP operations. The higher the level, the more detailed the logging. Level 9 is the useful maximum, with level 0 (the default) being no logging. If a file is specified, logging will go to that file, else logging appears on the console.

RIP TTL <time-To-LIVE>
RIP TTL sets the time-to-live before RIP entries expire from the

routing tables. The default should work for almost all cases.

End of RIP-2 Description

With no arguments, '**route**' displays the IP routing table.

route add <desthostid>[/bits] default <iface> [<gatewayhostid> | direct] [metric]

NOTE: Attempting tcp connections to an address without an existing route fails immediately.

This command adds an entry to the routing table. It requires at least two more arguments, the **desthostid** of the target destination and the name of the interface **<iface>** to which its packets should be sent. If the destination is not local, the gateway's hostid should also be specified. (If the interface is a point-to-point link, then **<gatewayhostid>** may be omitted even if the target is non-local because this field is only used to determine the gateway's link level address, if any. If the destination is directly reachable, **<gatewayhostid>** is also unnecessary since the destination address is used to determine the interface link address). If **<rspf>** is used and the system is a switch / router to multiple routes, the keyword '**direct**' can be used instead of a **<gatewayhostid>** to set the metric higher than the default of 1. This way routes advertised by other rspf stations can be cheaper and get selected. If '**direct**' is given but **<metric>** not, an new algorithm is used to set the metric dependent on the number of subnet mask bits.
The optional **/bits** suffix to the destination host id specifies how many leading bits in the host id are to be considered significant in the routing comparisons. If not specified, 32 bits (i.e., full significance) is assumed. With this option, a single routing table entry may refer to many hosts all sharing a common bit string prefix in their IP addresses. For example, ARPA Class A, B and C networks would use suffixes of /8, /16 and /24 respectively. E.g. the command

```
          route add 44/8 ax0 44.64.0.2
```

causes any IP addresses beginning with "44" in the first 8 bits to be routed to 44.64.0.2; the remaining 24 bits are "don't-cares".

When an IP address to be routed matches more than one entry in the routing table, the entry with largest 'bits' parameter (i.e., the "best" match) is used. This allows individual hosts or blocks of hosts to be exceptions to a more general rule for a larger block of hosts.

The special destination **'default'** is used to route datagrams to addresses not matched by any other entries in the routing table; it is equivalent to specifying a **/bits** suffix of /0 to any destination hostid. Care must be taken with **'default'** entries since two nodes with default entries pointing at each other will route packets to unknown addresses back and forth in a loop until their time-to-live (TTL) fields expire. (Routing loops for specific addresses can also be created, but this is less likely to occur accidentally).

There is one built-in interface: loopback. Loopback is for internal purposes only.

Here are some examples of the route command:

```
# Route datagrams to IP address 44.0.0.3 to SLIP line #0.
# No gateway is needed because SLIP is point-to point.

route add 44.0.0.3 sl0

# Route all default traffic to the gateway on the local Ethernet
# with IP address 44.0.0.1

route add default ec0 44.0.0.1

# The local Ethernet has an ARPA Class-C address assignment;
# route all IP addresses beginning with 192.4.8 to it

route add 192.4.8/24 ec0

# The station with IP address 44.0.0.10 is on the local AX.25 channel

route add 44.0.0.10 ax0

#An encapsulation link to 192.4.8.12 where the subnet 44.64.0.0 is
accessible. The Internet does not know
 #where we are but we just use them with what they know:
route add 44.64.0.0/16 encap 192.4.8.12 4
```

route addprivate <dest hostid>[/bits] | default <iface> [<gateway hostid> [<metric>]]

This command is identical to **'route add'** except that it also marks the new entry as private; it will never be included in outgoing RIP updates. It will also not be shown in the nodeshell **'IProute'** command.

route drop <dest hostid>
Delete an entry from the table. If a packet arrives for the deleted address and a default route is in effect, it will be used.

RSPF is the Radio Shortest Path First protocol. Each station listens for RRH (Router to Router Hello) messages. When such a RRH message is received, it will figure out if the link is bi-directional by pinging the other station. The protocol is described in the RSPF 2.1 specification.

rspf interface <interface> <quality> <horizon>
<interface> is the required interface rspf should use. **quality** is from 1 to 127, **horizon** is between 1 to 255. End nodes should have the quality set to 1. Immediate nodes normally set the quality to 8. The normally used value for horizon is 32.

rspf mode [vc | datagram | none]
(B) Display the preferred mode for RSPF. Modes are **vc** (Virtual Circuit) and **datagram**. **none** resets the preferred mode.

rspf rrhtimer [seconds]
(B) Display or set the rrh timer value.

rspf suspecttimer [<seconds>]
(B) Display or set the suspect timer value.

rspf timer [<seconds>]
(B) Display or set the update timer value.

{JNOS}

Without arguments, displays the list of current sessions, including session number, remote TCP or AX.25 address, the address of the TCP or AX.25 control block, and the session swap mode under the 'Sw' heading. An asterisk (*) is shown next to the current session. Entering a session number as an argument to the session command will put you in converse mode with that session. If the Telnet session; entering a blank line at this point puts you in converse mode with server is enabled, the user is notified of an incoming request and a session number is automatically assigned. The user may then select the session normally to converse with the remote user as though the session had been locally initiated.

session [[<session#>] [flowmode [on | off]]] {JNOS}

flowmode' displays or enables / disables setting of *more* handling for < session#>. This is handy for long directory listings coming from an ftp session, for example . Escaping to command mode before issuing the dir command and entering "session # flowmode on" gives a page at a time to look at. At any time you can escape out again and switch flowmode off. Note that a ftp session has it's own flow command now built in. See FTP commands later in this manual.

session [swap [E|X|M|F]] {JNOS}

Display or set the current session swap mode as Ems, Xms, Memory, or File.
session screens when swapped out (ie. not the active session) can now be saved in 4 different modes which can save conventional memory.

Modes are:
0 - in EMS (if available), this is now the default mode.
1 - in XMS (if available and no EMS) (Note: NOT! functional yet)
2 - in conventional memory, the previous method. Used if EMS/XMS doesn't exist.
3 - in a temporary file. (Use only if you have a ramdisk for tmpfiles, or a fast harddisk with cache...)

You can force options 1, 2 and 3 from the command line starting NOS with the -m# option, where # is 1, 2 or 3.

{JNOS}

Suspends JNOS and executes a sub-shell ("command processor" under MS-DOS). When the sub-shell exits, Nos resumes (under MS-DOS, enter the exit command).

Note: see the COMSPEC environment variable.

Background activity (FTP servers, etc.) is also suspended while the subshell executes. Note that this will fail unless there is sufficient unused memory for the subshell and whatever command the user tries to run. When shelled out, Mailbox Operator connects and ttylink incoming connections are refused. A 'system unattended' message is sent to the "connector" of that socket.

This is a shorthand for the various '**kick**' subcommands. This one searches the socket for correct type and kicks the transport layer.

These commands are used for the Simple Message Transport Protocol service (that is, mail).

smtp batch [yes | no]

If set smtp will batch the commands into one frame. When off only one command is sent and a response is waited for. Some old and flaky smtp servers cannot handle more than one command at a time. NOS can handle multiple. If you are not hindered by an old smtp server, setting batch reduces bandwidth.

smtp bidcheck [ON | off]

Sets or displays status of smtp bidchecking. Default is ON.

smtp gateway [<hostid>]

Displays or sets the host to be used as a "smart" mail relay. Any mail sent to a host not in the host table will instead be sent to the gateway for forwarding.

smtp kick

Run through the outgoing mail queue and attempt to deliver any pending mail. This command allows the user to "kick" the mail system manually. Normally, this command is periodically invoked by a timer whenever NOS is running.

smtp kill [<jobid>]

Kill <jobid> and delete the message.

smtp list

List the current jobs. An "L" means locked and in progress. It is wise to add in autoexec.bat a "del /spool/mqueue/*.lck" command. As of JNOS 1.10 (x16), all '.lck' files are automatically deleted on startup.

smtp maxclients [<count>]

Displays or sets the maximum number of simultaneous outgoing SMTP sessions that will be allowed. The default is 10. Reduce <count> if network congestion is a problem.

smtp mode [queue | ROUTE]

Sets the smtp delivery mode. If 'queue', all messages are left in /spool/rqueue for external forwarding and handling. If 'route', messages are handled and, if for local, appended to a mailbox, or if remote they are forwarded. Default = 'route'

smtp quiet [YES | no]

Enables or disables the message that new mail arrived at this system.

smtp t4 [<seconds>]

Displays or sets a t4 timer for smtp sessions so that they will disconnect after a period of inactivity and prevent lockups. Default = 0, i.e., no disconnect for timeout.

smtp timer [<seconds>]

Displays or sets the interval, between scans of the outbound mail queue. For example, smtp timer 600 will cause the system to check for outgoing mail every 10 minutes and attempt to deliver anything it finds, subject of course to the smtp maxclients limit. Setting a value of zero disables queue scanning altogether. This value is recommended for standalone IP gateways that never handle mail, since it saves wear and tear on the floppy disk drive. Default = 0

smtp trace [<value>]

Displays or sets the trace flag in the SMTP client, allowing you to watch SMTP's conversations as it delivers mail. Zero (the default) disables tracing.

smtp usemx [yes | NO]

Displays or sets a flag enabling or disabling MX record lookups. This can be enabled if a domain server is available in the near distance (reachable). It should be disabled (default) if no domain server is in reach to satisfy the MX query. Note that MX record handling is very limited in NOS. If an answer from a domain name server comes in it is taken to be the destination.

Without an argument, displays all active sockets, giving their index and type, the address of the associated protocol control block and the and owner process ID and name. If the index to an active socket is supplied, the status display for the appropriate protocol is called. For example, if the socket refers to a TCP connection, the display will be that given by the 'tcp status' command with the protocol control block address.

(B) Start the specified servers. Currently supported are telnet, ax25, netrom, convers, remote, rip. These are automatically started at startup (i.e. no 'start xyz' needed in the autoexec 'file'.)

Displays information on JNOS40: When started, how long running, etc.

(B) Stop the specified server, rejecting any further remote connect requests. Existing connections are allowed to complete normally.

The sysop command allows you to configure the ip address of a remote system that nodeshell user sysop chat attempts should be routed to. The port number defaults to 87, i.e. the tcp ttylink port. If configured, and a node shell user gives the **'Sysop'** command, a tcp/ip link will be tried to establish to the remote system.

For example, I have the Data Engine setup to link to my own PC's ttylink listener, thus when a user tries to talk to the sysop, I get a keyboard to keyboard link coming in.

sysop 44.26.1.17

Tail displays the last 20 (twenty) lines from <filename>. This is most useful for looking at the last screen full of entries into the log file.

These commands are used for the Transmission Control Protocol service.

Notes:

{JNOS} Attempting outgoing connections to addresses without an existing route result in Error number 19.

{JNOS40} Outgoing connection attempts to addresses without an existing route are terminated immediately.

tcp irtt [<milliseconds>]
(B) Display or set the initial round trip time estimate, in milliseconds, to be used for new TCP connections until they can measure and adapt to the actual value. The default is 5000

milliseconds (5 seconds). Increasing **irtt** when operating over slow channels will avoid the flurry of re-transmissions that would otherwise occur as the smoothed estimate settles down at the correct value. Note that this command should be given before servers are started in order for it to have effect on incoming connections.

TCP also keeps a cache of measured round trip times and mean deviations (MDEV) for current and recent destinations. Whenever a new TCP connection is opened, the system first looks in this cache. If the destination is found, the cached IRTT and MDEV values are used. If not, the default IRTT value mentioned above is used, along with a MDEV of 0. This feature is fully automatic, and it can improve performance greatly when a series of connections are opened and closed to a given destination (e.g. a series of FTP file transfers or directory listings).

tcp kick <tcb_addr>

If there is unacknowledged data on the send queue of the specified TCB, this command forces an immediate retransmission. **<tcb addr>** can be found with the '**tcp status**' command.

tcp maxwait [<msec>]

Set or show the maximum time for retry timeout in milliseconds. Default = 0, no maximum.

tcp mss [<size>]

(B) Display or set the TCP Maximum Segment Size in bytes that will be sent on all outgoing TCP connect request (SYN segments). This tells the remote end the size of the largest segment (packet) it may send. Changing MSS affects only future connections; existing connections are unaffected. See also the section **ON MTU**, etc.

tcp reset <tcb_addr>

Deletes the TCP control block at the specified address.

tcp retries [<num>]

(B) Display or set the number of retries before a tcp connection will be reset. Default is 16. This is useful to eliminate idle connections that have not been properly shut down. Default = 0, there is no maximum, i.e. a connection will never retry out.

tcp rtt <tcb_addr> <milliseconds>

Replaces the automatically computed round trip time in the specified TCB with the rtt in milliseconds. This command is useful to speed up recovery from a series of lost packets since it provides a manual bypass around the normal backoff retransmission timing mechanisms.

tcp status [<tcb_addr>]
Without arguments, displays several TCP-level statistics, plus a summary of all existing TCP connections, including TCB address, send and receive queue sizes, local and remote sockets, and connection state. If **<tcb addr>** is specified, a more detailed dump of the specified TCB is generated, including send and receive sequence numbers and timer information.

tcp syndata [yes | no]
(B) Display or set the tcp syn + data piggybacking flag. Some tcp systems cannot handle syn + data together.

tcp timertype [linear | exponential]
(B) Display the current setting or set the timer type backoff algorithm. Default is linear.

tcp trace [yes | no]
(B) Display or set the tcp trace flag on or off.

tcp window [<size>]
(B) Displays or sets the default receive window size in bytes to be used by TCP when creating new connections. Existing connections are unaffected.

Controls packet tracing by the interface drivers. Specific bits enable tracing of the various interfaces and the amount of information produced. Tracing is controlled on a per-interface basis; without arguments, 'trace' gives a list of all defined interfaces and their tracing status. Output can be limited to a single interface by specifying it, and the control flags can be change by specifying them as well.

NOTE: {JNOS40} Trace works ONLY when the serial port is used as CONSOLE, i.e. with AUX pushed in.

The flags are given as a hexadecimal number which is interpreted as follows:

B - Broadcast filter flag. If set, only packets specifically addressed to this node will be traced; broadcast packets will not be displayed.

T - Controls type of tracing:

0 - Protocol headers are decoded, but data is not displayed

1 - Protocol headers are decoded, and data (but not the headers themselves) are displayed as ASCII characters, 64 characters/line. Unprintable characters are displayed as periods.

2- Protocol headers are decoded, and the entire packet (headers AND data) is also displayed in hexadecimal and ASCII, 16 characters per line.

I - Enable tracing of input packets if 1, disable if 0

O - Enable tracing of output packets if 1, disable if 0

Example:

```
# Trace all incoming packets on port1 and display with

# headers
```

trace port1 0111

Show the status of active udp control blocks

Send a message to a particular user. **<message>** is the message, if "more then one word, put it in quotes." **<username|sock#>** can be either the user name of a nodeshell user, or a valid socket number. The latter allows you to send a message to a network conference user etc.

E.g.: 'write wg7j "this is a test!"'

The message will be shown as:

```
     '<bell>*** Message from node-op: this is a test!'
```

Send a message to all nodeshell or conference call users. E.g. when you've changed some things that require remote rebooting, you can warn users of the shutdown.

```
ARRL Computer Networking Conference Proceedings
    Available from ARRL HQ, Newington CT.
    Send mail to info@arrl.org for an automatic response pointing at
    more information about the ARRL.
    Some of these papers are available online in the directory
    ucsd.edu:/hamradio/packet/tcpip/docs.

    This list is not exhaustive; there are many other interesting
    articles, but these are the ones most relevant to NOS and TCP/IP.

    NOS Overviews and Documentation

     NOS Command Set Reference
               Ian Wade G3NRW           10th (1991)

     NOSVIEW: The On-Line Documentation Package for NOS
               Ian Wade G3NRW           11th (1992)

     The KA9Q Internet (TCP/IP) Package: A Progress Report
               Phil Karn KA9Q           6th (1987)

     Amateur TCP/IP: An Update
               Phil Karn KA9Q           7th (1988)

     Amateur TCP/IP in 1989
               Phil Karn KA9Q           8th (1989)

    Services and Protocols

     The Design of a Mail System for the KA9Q Internet protocol
               Bdale Garbee, N3EUA      6th (1987)
               Gerard van der Grinten, PA0GRI

     Finger - A User Information Lookup Service
               Michael T. Horne, KA7AXD      7th (1988)

     Callsign Server for the KA9Q Internet Protocol Package
               Doug Thom, N6OYU         8th (1989)
               Dewayne Hendricks, WA8DZP

     The Network News Transfer Protocol and its Use in Packet Radio
               Anders Klemets, SM0RGV           9th (1990)
```

```
 A Routing Agent for TCP/IP: RFC 1058 Implemented for the KA9Q
 Internet Protocol Package               7th (1988)
           Albert G. Broscius, N3FCT

 Thoughts on the Issues of Address Resolution and Routing in
 Amateur Packet Radio TCP/IP Networks
           Bdale Garbee, N3EUA      6th (1987)

 Another Look at Authentication
           Phil Karn KA9Q           6th (1987)

 LZW Compression of Interactive Network Traffic
           Anders Klemets, SM0RGV           10th (1991)

 PACSAT Protocol Suite -- An Overview
           Harold Price, NK6K       9th (1990)
           Jeff Ward, G0/K8KA

 BULLPRO -- A Simple Bulletin Distribution Protocol
           Tom Clark, W3IWI         9th (1990)

Macintosh

 KA9Q Internet Protocol Package on the Apple Macintosh
           Dewayne Hendricks, WA8DZP     8th (1989)
           Doug Thom, N6OYU

 Status Report on the KA9Q Internet Protocol Package for the
 Apple Macintosh
           Dewayne Hendricks, WA8DZP     9th (1990)
           Doug Thom, N6OYU

 Higher Speed Amateur Packet Radio using the Apple Macintosh
 Computer
           Doug Yuill, VE3OCU       10th (1991)

Network design

 The Implications of High-Speed RF Networking
           Mike Chepponis, K3MC          8th (1989)
           Glenn Elmore, N6GN
           Bdale Garbee, N3EUA
           Phil Karn, KA9Q
           Kevin Rowett, N6RCE

 Design of a Next-Generation Packet Network
           Bdale Garbee, N3EUA      8th (1989)
```

```
More and Faster Bits: A Look at Packet Radio's Future
          Bdale Garbee, N3EUA      7th (1988)

Physical Layer Considerations in Building a High Speed Amateur
Radio Network
          Glenn Elmore, N6GN       9th (1990)

Spectral Efficiency Considerations for Packet Radio
          Phil Karn, KA9Q               10th (1991)

     This should be considered to be required reading.

MACA - A New Channel Acess Method for Packet Radio
          Phil Karn, KA9Q               9th (1990)

A Duplex Packet Radio Repeater Approach to Layer One
Efficiency
          Robert Finch, N6CXB      6th (1987)
          Scott Avent, N6BGW

A Duplex Packet Radio Repeater Approach to Layer One
Efficiency, Part Two
          Scott Avent, N6BGW       7th (1988)
          Robert Finch, N6CXB


Network Implementation

Packet Radio at 19.2 kB -- A Progress Report
          John Ackermann, AG9V          11th (1992)

Implementation of a 1Mbps Packet Data Link
          Glenn Elmore, N6GN       8th (1989)
          Kevin Rowett, N6RCE

Hubmaster: Cluster-Based Access to High-Speed Netowrks
          Glenn Elmore, N6GN       9th (1990)
          Kevin Rowett, N6RCE
          Ed Satterthwaite, N6PLO

Recent Hubmaster Networking Progress in Northern California
          Glenn Elmore, N6GN       9th (1990)
          Kevin Rowett, N6RCE

The 56 kb/s Modem as a Network Building Block: Some Design
Considerations
```

```
          Barry McLarnon, VE3JF         10th (1991)


Digital Networking with the WA4DSY Modem - Adjacent Channel
and Co-Channel Frequency Reuse Considerations
          Ian McEachern, VE3PFH         10th (1991)

A Full-Duplex 56kb/s CSMA/CD Packet Radio Repeater System
          Mike Chepponis, K3MC          10th (1991)
          Lars Karlsson, AA6IW

A High Performance, Collision-Free Packet Radio Network
          Phil Karn KA9Q           6th (1987)

Adaptation of the KA9Q TCP/IP Package for Standalone Packet
Switch Operation
          Bdale Garbee, N3EUA      9th (1990)
          Don Lemley, N4PCR
          Milt Heath
```

Hardware

```
The KISS TNC: A Simple Host-to-TNC Communications Protocol
          Mike Chepponis, K3MC          6th (1987)
          Phil Karn, KA9Q

The Ottawa Packet Interface (PI) A Syncrhonous Serial PC
Interface for Medium Speed Packet Radio
          Dave Perry, VE3IFB       10th (1991)

HAPN-2: A Digital Multi-Mode Controller fo the IBM PC
          John Vanden Berg, VE3DVV 11th (1992)

The PackeTen system - The Next Generation Packet Switch
          Don Lemley, N4PCR        9th (1990)
          Milt Heath
```

The following commands are available to the users connected to the mailbox. This file is available separately as mboxcmds.txt.

AREA

The Area command lists the mail areas that contain messages you may read.

A gives a short listing, whereas
AF gives a full listing with descriptions (if available)
AN shows areas that have new mail since you last logged off.

To read messages in one of the areas, type 'A <areaname>'. You will then be told how many new, not previously listed messages there are in this area.

You can send mail to any of the listed areas as 'S <areaname>'

BYE

The Bye command is used to exit from the NOS MBOX. This will close your mailbox file and remove any messages that you have deleted with the K[ill] command.

CONNECT

The Connect command has the following modes:

C[onnect] [port] [callsign] [<digipeater> . . .] connects to station 'callsign' on interface 'port', possibly via digipeaters 'digipeater...' (note the lack of 'via'!)

C[onnect] [node] connects over netrom to a remote node with 'node' as either node-call or node-alias

DOWNLOAD

The download command will begin sending a file from the mailbox to you.

CONV [<channel>]

(if available) puts you in convers mode. This is a roundtable discussion feature. 'channel' allows specifying the conference channel you wish to join. Channel default = 0.

D[ownload] [/][<path_name>/]filename sends a plain ASCII text file.

DU [/][<path_name>/]filename downloads binary files converted to UUENCODED ASCII.

You will need the "uudecode" utility to convert this ASCII file back to binary. Source code, in various languages, for uudecode can be downloaded from this system. Look for uudecode.bas, uudecode.pas, and uudecode.c.

The optional path_name may be included along with the filename if the desired file is not in the current directory (you can determine this using the W[hat] command). Please note that the character used to separate the path and filename is a "/" (forward slash).

ESCAPE

This character is what will be used if you want to exit from the current session. For instance, if you have started a "chat" session, and you don't get any response from the operator after waiting a few minutes, you can enter the escape character, followed by a <RETURN> or <ENTER>, and the session will be terminated. You will then be returned to the MBOX prompt.

E[scape] entered by itself will display the character that is currently set as the escape character.

E[scape] [<new_escape_character>] Enter "escape" followed by a <SPACE> and the character that will become the new escape character. This must be a single typed character (the <CTRL> key may be used in addition).

```
EXAMPLES
           escape ^Z     (the ASCII character <CTRL>Z)
           escape X      (the character "x" is the new escape)
```

FINGER

The finger command retrieves personal information about users of a system.

F[inger] displays a list of known users on the current system.

F[inger] [<user_name>] display information about if and when the user last logged in, as well as any information which may be set in the user's finger-file.

F[inger] [<user_name>][@<host>] Perform the same functions detailed above on another TCP/IP host connected to the network.

To get a list of the users on a remote system, enter "finger" followed by a <SPACE> and an "@", then the host name. To get information about a remote user, insert the user name before the "@".

```
EXAMPLES
finger           (list the known users on this system)
f sysop          (list info about the local user "sysop")
f @wg7j          (list the known users at host "wg7j")
f johan@wg7j     (display info about "johan" at host "wg7j")
```

HELP

Get on-line assistance for user commands

? displays a list of the commands that have help descriptions available for them:

```
     area       bye       connect     download  escape    finger
     help       info      jheard      kill      list      mboxuser
     nodes      nroutes   operator    ports     read      send
     telnet     upload    verbose     xpert     what      zap
```

H <command> Displays help for a specific command.

Example: Display the help text for the command 'connect'.
'h connect'

IHEARD

The IHeard command shows the **tcp/ip** systems recently heard.

I[heard] Show tcp/ip activity for all ports.

I[heard] [<port>] Show tcp/ip activity for <port>.

For ax.25 interfaces (ports), show all tcp/ip activity heard, even when this system was not involved in it. For other interfaces, show those systems that we actively routed packets for (ie. systems that talked to us.)

IPROUTE

IP[route] shows the available TCP/IP routes the system has configured. It shows the interfaces and gateways involved in the routes, and also the expiration timer (if applicable).

This could be a LONG list if the system has a lot of ip routes. Please ask the sysop for more about the information given in the display.

JHEARD

The jheard command will display a list of all the station callsigns that have been received as sending packet traffic on the channel, the time since the station was heard last, and the total number of packets received.

J[heard] displays the "heard" list for all interfaces.

J[heard] [<port>] displays a list of the stations heard on a particular channel. See the Ports command for determining which channel is heard on which port.

Warning: if this system has been on the air for very long, and the channels are very active, the "heard" list could be extremely long.

KILL

The **kill** command allows you to delete messages from the current mailbox (if you have been given that permission by the operator).

K[ill] <message_number> [<message_number> . . .] deletes the specified messages. If no message number is supplied, the current message is deleted. The message numbers you may select from can be displayed with the "L[ist]" command. The second parameter on each line of the list is the <message_number>

KM will delete ALL read messages in the area.
KU will "un-kill" a message that was previously marked for deletion.

The kill command only applies to messages in the current mail "area". The current mail area can be checked and modified with the "A[rea]" command.

LIST

L[ist] [<starting_msg_number> [<ending_msg_number>]] prints a list of the messages from the current mailbox (or "area"). For each message, the list contains the subject header line, the time and date it was created, who it is from, how many bytes long it is, and whether or not it has been read.

You may include an optional "starting_msg_number" from which to begin displaying the list. If you specify a starting msg number, then you may also specify an ending number as well. This will limit the display for you in case there are a large number of messages in a particular "area" mailbox.

L by itself will display the headers for all unread messages, if any.
LA list all messages, read or unread
LL display the last <number> of message headers.
LM is the same as 'L'
LB list all bulletins
LS [subject] list messages in the current 'area' with [subject] in the subject line.
LT list all traffic
L> xxx list all messages that have the string 'xxx' in the To: address
L< xxx list all messages that have the string 'xxx' in the From: address

Mailbox USERS

M will display a list of all the current users, how they connected, and their current activity.

ML will list all past users of the system, when they were last on and how many times they've connected.

ML n will show the last n users of the system

ML call will list when 'call' last logged in

MS will give some info on the number of messages handled since the system has been up

NODES

N prints a list of NetRom nodes that are known to this system and for which the nodeids do not begin with '#'.
N * will give info on all known nodes including "hidden" nodes (those with IDs beginning with '#').
N <nodename> displays information about routes (paths) available to <nodename>

NR[oute] command will list all known NetRom neighbour stations, with a listing of the path quality to them, number of destinations the neighbour knows and the obsolescense count.

'>' in front indicates that the route has been used in the past 60 seconds

OPERATOR

O[perator] allows you to "talk" keyboard-to-keyboard with the operator of this NOS system if the system is attended.

When you wish to terminate the chat session, type the escape character on your keyboard, and then press <ENTER> or <RETURN>. The default escape character is "CTRL-X", which means to hold down the <CTRL> key and press the <X> key simultaneously. This escape character may be changed to whatever you prefer by using the "E[scape]" command.

PING <host>
Check of <host> is alive. Returns RTT.

PORTS

P[orts] prints a list of AX.25 interfaces (ports) that are installed in this system. A description of the port is also given if one has been setup for that port. These ports can be used to make outgoing AX.25 connections with the "C[onnect]" command.

QUERY

Q <call> [<call> . . .] If available, this queries the BuckMaster CDRom callbook server for info about the calls given. More then one call per query is allowed.

READ

Read a message (or messages) from the current mail area.

or,
R[ead] # or,
R[ead] <msg_number> [<msg_number> . . .] To read a specific message, you may either type "read #" or just the number by itself. If there is a specific list of messages you are interested in (determined by the use of the L[ist] command, for instance), you can enter the list of message numbers (separated by spaces) on the "read" command-line. You can also simply advance sequentially through the messages by just pressing the <ENTER> or <CR> key. This will display the next message in order. The "read" command displays only an abbreviated portion of the mail headers. If you want to display all the header lines, use the V[erbose] command instead.

RH <msg_number> [<msg_number> . . .] show all the headers of the messages.
RM display without interruption all unread messages.

```
EXAMPLES:
     read 3 5         (Display only messages 3 and 5)
     4                (Display message 4)
     <CR>             (Display next message)
```

SEND

The send command allows you to enter a message and send it to a user at either this system, or some other system on the network.

S[end] <user>[@ <host>] [< <from_addr>] [$<bulletin_id>]   Send a message to <user>. The system will prompt for "Message Subject" and "Text" fields. The "from_addr" and "bulletin_id" fields are for special use and won't be covered here.

SP <user>[@ <host>] (Send Personal) As above, but only the addressee (<user>) may read the message from the mailbox.

SB <user>[@ <host>] (Send Bulletin) As above, but ANY <user> may read the message from the mailbox. <User> is usually a category rather than an individual stationid when sending bulletins

SR [msg_number] "reply" to either the current message or the message number specified. The subject will be copied and the reply will be sent to the address it was sent from.

SF <user>[@ <host>] [< <from_addr>] [$<bulletin_id>] Forward a copy of the current message to the user specified.

SC <user>[@ <host>] [< <from_addr>] [$<bulletin_id>] Send a message to more than one user. The system will prompt with "Cc: ", which allows you to add more users to be sent 'carbon copies' of the message. Separate users on the Cc: line with commas.

```
     EXAMPLES
          send kf7xx            (Send a message to the local user,
                                   kf7xx)
          s kf7xx @ wb7xxx      (Send a message to kf7xx at the
                                   wb7xxx host)
          sr 3                  (Reply to message number 3)
          sf n7aaa%n7bbb@w7ccc  (Forward current msg to n7aaa at
                                   n7bbb via w7ccc)
          sc wg7j               (Send with Carbon copy to others)
           Cc: ka7ehk, n7dva@n7dva
```

TELNET

T[elnet] <hostname> [<port_number>] Initiate a TCP connection from the JNOS mailbox out across the network to another host. This allows an AX.25 user with nothing more than a terminal and TNC to gain access to the TCP/IP network.

By including the optional port_number, you can connect to any TCP server at the given host. The default is to be connected to the "telnet" server, which in the case of NOS software, is the MBOX.

To quit the session at any time, enter the escape character. (<CTRL>X, the default, can be changed with the E[scape] command).

UPLOAD

U[pload] [/][<path_name>/]<filename> Transfer an ASCII file from your system onto disk at this host. You may also specify a full path_name containing a specific directory in which to deposit the new "upload". All uploads can only go into the directory that you logged into, or into another directory under the current one.

The transfer proceeds line-by-line until the file is sent and you enter either a "<CTRL>Z" or "/ex" as the first item on a blank line.

EXAMPLES
```
     upload kepler.txt
     u /public/satelite/oscar13.txt
```

VERBOSE
This command allows you to read a message (or messages) from the current mail area, and it includes all the header lines for display. The R[ead] command operates the same way, but with abbreviated header lines.

V[erbose] <msg_number> [<msg_number> . . .] View a specific message or a list of messages with all headers.

VM, 'verbose mine' Display, without interruption, all unread messages in the area.

WHAT
W[hat] [/][<path_name>] Generate a sorted directory listing of the current directory or the one specified by the optional path_name. The listing includes the filename (or subdirectory name if there is a "/" appended), the file size in bytes, creation time, and date.

EXAMPLES
```
               what      (Displays a directory listing of the
                              "current" dir)
               w /nos/pub    (Display a list of files contained in
                              the "/nos/pub" dir)
```

XPERT

The Xpert command toggles the prompts that the system gives

X - toggles the prompt between using long and short prompts.
XA - toggles the 'current area' indication on or off.
XN - toggles the 'netrom id' prompt on or off
XM - shows the number of lines before -more- prompting occurs in lists
XM n - sets the number of lines ...

The states of the above are remembered at logout and used at next login.

ZAP

Z[ap] [/][<path_name>/]<filename> The zap command allows you to delete a file in the current directory of one you specify with the optional path_name. Use of this command requires that permission be granted by the operator of this system.

EXAMPLES
```
     zap myfile.txt            (Deletes myfile.txt in the current
dir)
     z /nos/mydir/myfile.txt   (Deletes myfile.txt in /nos/mydir)
```

End of Appendix A: JNOS User Commands

This is a list of user commands that can be given when a user is connected to the JNOS40 nodeshell. Commands marked with **(*)** behave differently depending on the type of connection and are discussed in the section **JNOS40 AND THE USER** (See JNOS40 Configuration Guide). Only the **CAPITAL** letters need to be given, and the commands are not case sensitive.
NOTE: These commands can be altered using the 'mbox alias' command.

?? - Gives names of all JNOS40 commands

? x - Gives extended help on command 'x'

Bye (*) - disconnect from nodeshell

Connect (*) - Connect, has a few options:

C <node> [d][e]	connect over netrom to <node>
C <port> <call> [digis] [e]	connect to <call> on <port> via [digis]

NOTE: List [digis] WITHOUT the 'via' and don't use commas:

C PORT CALL DIGI1 DIGI2 DIGI3

NOTE: the optional **[e]** enables escape checking during the connection.

NOTE: The optional [d] DISABLES the 'stay here' feature for NETROM circuits.

CONV [<channel>] Access the conference bridge (if available)

Escape - set or show the current escape character

E on - set the default usage of the escape char to 'on' for this user.

E off - set the default to off. 'e' in connect command will turn it on for that connection only.

Finger - finger, either 'f user', 'f user@host' or 'f @host' is valid
valid local users are:
mem - shows memory statistics

```
        ps          - shows the process status display
        last        - shows past users of nodeshell
        stats       - shows link statistics
        heard       - shows heard calls on all interfaces
        conf        - shows conference bridge users

        All others will show the current users.
```

Heard   - shows the heard list. This lists the calls heard during the past 30 minutes.
          'H port' will show ALL heard calls for 'port' only.

Info - list the info message. The sysop could set this to node location, frequency, height or whatever.

IHeard   - lists the recently heard tcp/ip stations.

IProute   - lists the Internet Protocol routes

Last - list the last few users of the system, and how long ago they connected

Motd - shows the 'message of the day', set by the node-operator

Nodes   - show the known nodes.
N name   -show info on node 'name'
N *   - show info on all nodes
The latter two show the path quality, the obsolescence count, the neighbor to reach the node, and the type of route. Three types of routes exist:
P are permanent routes (like netrom and bpq)
B are broadcasted routes (from nodes-broadcasts)
R are recorded routes, from netrom packets passed through the system.

Operator - if configured, will attempt a keyboard session with a remote sysop

Port - show all ports, with some info on them (if set by the sysop)

PIng <host> - determine if <host> is alive. Returns RTT.

Routes   - show all netrom routes

Telnet   - telnet to ip address with optional port number. The port used defaults to the 'telnet' port, i.e. port 23.
An optional port number can either be a number, e.g. '3600', or a name, e.g. 'convers'.

Port names currently recognized are 'telnet', 'ttylink', and 'convers'   E.g.: 't wg7j', 't wg7j 23' and 't wg7j telnet' are all the same !

TTylink   - tty-link to address. Same as t, but to the tty-link port.

Users   - show current users and their status

Version   - display software version info

There are also some **hidden** commands, not shown in the ? list:

Data - shows some data on usage and interface statistics

LInks   - shows current AX.25 and NETROM connections

UPtime - shows system's uptime.

***** LINKED TO**
          - interprets callsign changes as done by some version of NOS.EXE and by TexNet.

@   - attempt sysop mode. The sysop password can be set with 'password pwdstring', and defaults to '0123456789'. When hitting the @, you get a 5 digit challenge. You should answer with the 5 corresponding letters in the password string (**The first letter is number 0!**). You may enter as many characters as you wish, but the first five characters must be the correct response to the challenge. Two lines are required to complete the response - the second line can be a <cr> only.

The node does not reply if the response to the challenge is incorrect.
The node uses the prompt SYSOP> while in sysop mode.

End of Appendix B: JNOS40 User Commands

NOS supports a number of versions of the **attach** command to deal with different hardware. We'll discuss three of them here: **asy**, used for serial port connections; **pi**, used to connect to the Ottawa PI card; and **packet**, used to interface to hardware supporting the FTP, Inc., packet driver protocol. As usual, this discussion covers the basics; see the NOS reference manual for details on all the many options.

Hosts normally have a separate IP address for each interface. If you are running more than one interface, you can include that interface's IP address (in [xx.xx.xx.xx] form) at the end of the attach command.

The asy version provides an interface to a standard PC serial port. The syntax is:

attach asy <ioaddr> <vector> <mode> <if> <bufsize> <mtu>   <speed>

In English, these parameters are:

ioaddr -- the address of the COM port being used. COM1 is usually **0x3f8** and COM2 is usually **0x2f8**. COM3 and COM4 aren't standardized; using them will require looking at the documentation for your serial card, and probably some experimentation.

vector -- the IRQ used by the hardware. COM1 is usually **4**, and COM2 is usually **3**. Again, COM3 and COM4 vary.

mode -- this specifies the nature of the interface. **ax25** is for a connection to a KISS TNC, **slip** for a hardwired connection to another host, **ppp** for a dial-up connection, and **nrs** is for attaching a NOS station to a NetRom node.

if -- the interface name. The convention is to use **ax0**, **ax1**, etc., for KISS interfaces.

bufsize -- the buffer for incoming data, in bytes. Usually a value of **1024** is more than sufficient for a 1200 baud channel.

mtu -- the maximum transmission unit size, in bytes. See the discussion in the main text on this subject.

speed -- the speed of the serial (not radio) link, in baud. The best setting for this will depend on the speed of your computer, but generally two to four times the radio speed is adequate.

Some sample **attach asy** commands are:

```
# COM1, KISS TNC as ax0, MTU 256, 4800 BAUD
attach asy 0x3f8 4 ax25 ax0 1024 256 4800

# COM2, KISS TNC as ax1, MTU 256, 2400 BAUD
attach asy 0x2f8 3 ax25 ax1 1024 256 2400
```

```
        # SLIP link, COM1 as sl0, MTU 256, 9600 BAUD
        attach asy 0x3f8 4 slip sl0 1024 256 9600
```

The Ottawa PI card is a plug-in board for PCs designed for high-speed performance. It has two ports, one DMA driven for high speed and the other interrupt driven. The attach syntax is:

attach pi <ioaddr> <vector> <DMA chn> <mode> <name> <bufsize> <mtu> <speed a> <speed b>

A sample attach command (using the PI's default jumper settings) is:

attach pi 380 7 1 ax25 pi0 1750 1024 0 1200

In this example, the interface name for the DMA port is "pi0a" and the second port is "pi0b". Because the port a speed is 0, the PI card expects the modem to provide its own clocking. The PI attach syntax is explained in the manual provided with the card.

Finally, the **packet** interface is used to connect to ethernet cards and other hardware that supports the FTP, Inc. "packet driver" standard. There's a packet driver for the PI card. The syntax is:

attach packet <ioaddr> <vector> <if> <bufsize> <mtu>

In this case, **ioaddr** and **vector** need to match those used for the packet TSR that supports the hardware. **bufsize** is the number of packets (not bytes) that may be outstanding. For ethernet, the standard **mtu** is 1500.

End of Appendix C: Designing Attach Commands

The following is a list of the user permission values required for the FTPUSERS file.

```
Name:           value  (hex)
FTP_READ           1      0x1  /* Read files */
FTP_CREATE         2      0x2  /* Create new files */
FTP_WRITE          4      0x4  /* Overwrite or delete existing files
*/
AX25_CMD           8      0x8  /* AX.25 gateway operation allowed */
TELNET_CMD        16     0x10  /* Telnet gateway operation allowed */
NETROM_CMD        32     0x20  /* NET/ROM gateway operation allowed */
SYSOP_CMD         64     0x40  /* Remote sysop access allowed */
EXCLUDED_CMD     128     0x80  /* This user is banned from the BBS */
```

```
            /* 256 and 512 are used in PPP*/

NO_SENDCMD     1024    0x400  /* Disallow send command */
NO_READCMD     2048    0x800  /* Disallow read command */
NO_3PARTY      4096   0x1000  /* Disallow third-party mail */
IS_BBS         8192   0x2000  /* This user is a bbs */
IS_EXPERT     16384   0x4000  /* This user is an expert */
NO_CONVERS    32768   0x8000  /* Disallow convers command */
NO_ESCAPE     65536  0x10000  /* Default is no escape char */
NO_LISTS     131072  0x20000  /* No lists displayed from mailbox */
NO_LINKEDTO  262144  0x40000  /* Disable '*** linked to'  */
```

To set options, simply add values.
Format in /ftpusers file is:

<name> <password> [<drive:></rootdir>;</root2>] <#perms>]...

<name> is the userid, normally a callsign for amateur radio use.

The <name> "univperm" should be included in the ftpusers file. "univperm" allows anyone not otherwise found in the ftpusers file to logon with "guest" status.

If <password> is set to '<string>', then <string> must be used.
If <password> is set to '*', then any entry will satisfy password.

User can be given access to several drives and directories with varying permissions. These are all given on one line.

<drive:> is the drive letter for each drive to which the user is being given access.

</rootdir> is the highest directory in the system tree the user may access. It becomes the users root directory. Subdirectories under </rootdir> may be accessed by the user. More than one </rootdir> may be given per drive.

<#permissions> is the sum of the decimal OR hexadecimal values which defines what the user is allowed to do while logged onto the system.

You may provide access to more than one set of drives and directories with different permissions for each set. This allows a user to access a personal directory with complete read/write/delete access and a public directory with read permissions only, or any other combination you may desire.

```
univperm * /public 138283  (or 0x21c2b)
```

gives anyone not otherwise known login permission as a guest who can read or create (upload) new files on FTP connections, access ax25 or netrom stations, but has no mbox send, read, 3rd_party, or list functions.

```
wg0b doug c:/wg0b 0x407f /public;/nts ox407b
```

defines two different setps of permissions for three different areas.

End of Appendix D, FTP User Permissions
The rewrite file is placed in the 'spool' subdirectory. If you don't feel comfortablewith the rewrite mechanism, please refer to or read the 'mailbox.txt' distributed with this document. That document, written by NQ0I and SM0RGV, explains the bbs well. Credit goes to those gentlemen.
The following is by no means the best or only way to configure your system's rewrite mechanism. It is simply the way I run it, and is shown as an example only. My system tends to not take a lot of bulletins, to keep the load down (most are old anyway), but you might decide to do things differently.
Now to the way WG7J 'runs mail'. First thing I is catch all Internet style (ie. SMTP targeted) mail, and make sure that those messages go as is. Lines 1-4 take care of this by catching most of the top-level Internet domain names.

```
        *@*.edu $1@$2.edu
        *@*.com $1@$2.com
        *@*.gov $1@$2.gov
        *@*.org $1@$2.org
```

Next is an example of catching some things you don't want; here in Oregon some-one pumps in daily astronomical stuff. By the time it gets to my system it's way old :-(By rewriting it to 'refuse', the bbs will send a 'NO' as if it already has receive it. Same for some other things the wormhole bbs's are trying to forward to me.

```
        astro@* refuse
        *@dist9 refuse
        *@allin refuse
        *@okipn refuse
        *@allil* refuse
        msys@* refuse
        fbb@* refuse
        mods@* refuse
        *@ww refuse
```

I want users to be able to send mail to sysop on my system without it being forwarded elsewhere. I take care of this by rewriting it to the 'wg7j' area (ie. my private mail area)

```
        sysop wg7j
        sysop@wg7j* wg7j
```

Next I send everything else that comes in for sysops to the 'sysop' area. That way I can participate in receiving and forwarding stuff like 'sb sysop@allor' etc...

```
        sysop@* sysop
```

I place anything addresses to specific mail areas as setup with the '/spool/areas' files into those mailboxes

```
        tcpip@* tcpip
        wanted@* wanted
        want@* wanted
        need@* wanted
        sale@* sale
        4sale@* sale
        trade@* sale
        dx@* dx
        humor@* humor
        jokes@* humor
        happy@* humor
        races@* races
        fcc@* fcc
        amsat@* amsat
        arrl@* arrl
        ares@* ares
        swap@* sale
        nasa@* nasa
```

Then the same thing for the @-distribution names:

```
        *@nasa nasa
        *@amsat amsat
        *@ares* ares
        *@arrl arrl
        *@arl arrl
        *@pnw pnw
        *@allor* allor
        *@allusw allusw
        *@allus* allusa
```

NOTE: if you follow this style, it is important that the lines above are kept in that order (TO sorting FIRST, then AT sorting!!) Otherwise something like 'amsat@allusa' will end up in the 'allusa' area instead of the 'amsat' area.

Next I will catch anything destined for my bbs that hasn't been already caught by a previous rule. At this point, this <should> only be private mail.

```
        *@wg7j* $1
```

Then I will catch any mail destined for the bbs's I forward to and place it in their mailbox to be forwarded.

```
        *@wa7tas* wa7tas
        *@wa7shp* wa7shp
        *@w0rli* w0rli
        *@n7dxt* n7dxt
```

I place anything destined for a few in-state (ie OR) bbs's that are north of me into the 'north' mailbox. They get forwarded north-ward (see forward.bbs)

```
        *@n7hae* north
        *@n7vyn* north
        *@n7koj* north
        *@n7pwf* north
        *@wa6gfp* north
        *@n7jqk* north
        *@ka7agh* north
        *@kb7dbd* north
```

Then I take all local NTS traffic and places it in it's own area.

```
        *@97321* ntslocal
        *@9733* ntslocal
        *@97370* ntslocal
        *@97389* ntslocal
```

Other in-state NTS goes into the right direction.

```
        *@98* north
        *@970* north
        *@971* north
        *@972* north
        *@9730* wa7shp
```

All out-of-state NTS traffic gets placed into the 'nts' area for forwarding

```
        *@ntswa* north
        *@nts* nts
```

The idea is, that by rewriting every in-state bbs north of me into the north area, everything in-state left has to go south ! (Luckily, N7DXT, who gets my south traffic, is forgiving and will send my mistakes north anyway !)

```
        *@*.or* south
```

A few other states that go south:

```
        *@*.ca* south
        *@*.az* south
        *@*.tx* south
```

These next states go to K9IU in Indiana via the wormhole:

```
        *@*.in* indy
        *@*.oh* indy
        *@*.mi* indy
        *@*.ky* indy
        *@*.tn* indy
```

And lonesome KE7KD in Reno get the Nevada traffic:

```
        *@*.nv* nevada
```

Send all remaining North American mail north (to w0rli, who has an HF port...)

```
        *@*.eu north
```

Catch two more continents:

```
        *@*.oc south
        *@*.as north
```

And finally, I will catch anything that is left at this point. It puts it in the 'check' area. The idea here is that I can manually check the 'check' area and adjust '/spool/rewrite' accordingly and append that mail to the right mailbox file so it goes out. 'check' is actually an alias, that sends a copy of the message to both the 'check' area and my private mailbox, so that i will know right away when something unknown has shown up.

```
        *@* check
```

End of Appendix E - REWRITE

During converse mode with an FTP server, everything typed on the console is first examined to see if it is a locally known command. If not, the line is passed intact to the remote server on the control channel. The following commands are executed locally. Note that this generally involves other commands being sent to the remote server on the control channel. The commands to and from the remote server are indicated in most cases to allow understanding the flow between the two systems.

dir [<file> | <directory> [<local file>]]
Without arguments, 'dir' requests that a full directory listing of the remote server's current directory be sent to the terminal. If one argument is given, this is passed along in the LIST command. This can be a specific file or subdirectory that is meaningful to the remote file system. If two arguments are given, the second is taken as the local file into which the directory listing should be put instead of being sent to the console. The PORT command is used before the LIST command is sent.

get <remote file> [<local file>]
Tells the remote server to send <remote file>. <local file>, if given, is the name of the file as saved on the local machine. Otherwise the file will have the same name as on the remote machine. The PORT and RETR commands are sent on the control channel.

ls [<file> | <directory> [<local file>]]
'ls' is identical to the 'dir' command except that the "NLST" command is sent to the server instead of the "LIST" command which results in an abbreviated directory listing, one showing only the file names themselves without any other information.

mget <file> [<file> ...]
Fetch a collection of files from the server. File names may include wild card characters which will be interpreted and expanded into a list of files by the remote system using the NLST command. Filenames will be the same on the local system as they were on the server.

mkdir <remote directory>
Create a directory on the remote machine. You can do this only if permissions associated with your logon userid contained in the ftpusers file allow it.

mput <file> [<file> ...]
Send a collection of files to the server. File names may include

wild card characters and they will be expanded by the local machine into a list of files to be sent. Filenames will be the same on the server as on the local system.

put <local file> [<remote file>]
Asks the remote server to accept data and create the file named in the first argument. The second argument, if given, will be the name of the file on the remote machine; otherwise the file will have the same name as on the local machine. The PORT and STOR commands are sent on the control channel.

rmdir <remote directory>
Deletes <remote directory> on the remote machine. You must have the appropriate permissions associated with your login userid to be able to use this command.

type [a | i | l <bytesize>]

Tells both the local client and remote server the type of file that is to be transferred. The default is 'a', which means ASCII (i.e., a text file). Type 'i' means image, i.e., binary. In ASCII mode, files are sent as varying length lines of text in ASCII separated by cr/lf sequences. In IMAGE mode, files are sent exactly as they appear in the file system. ASCII mode should be used whenever transferring text between dissimilar systems (eg. UNIX and MS-DOS) because of their different end-of-line and/or end-of-file conventions. When exchanging text files between machines of the same type, either mode will work but IMAGE mode is usually faster. When exchanging raw binary files (executables, compressed archives, etc) IMAGE mode must be used.

Type 'l' (logical byte size) is used when exchanging binary files with remote servers having oddball word sizes (eg. DECSYSTEM-10s and 20s). Locally it works exactly like IMAGE, except that it notifies the remote system how large the byte size is. Bytesize is typically 8. The type command sets the local transfer mode and generates the TYPE command on the control channel.

verbose [0 | 1 | 2 | 3]
Set or display the level of message output in file transfers. This command is entered after the ftp session is started.
Verbose 0 gives the least output and verbose 3 the most as follows:

0 - Display error messages only.
1 - Display error messages plus a one-line summary after each transfer giving the name of the file, its size, and the transfer time and rate.

2 - Display error and summary messages plus the progress messages generated by the remote FTP server. (This setting is default.)
3 - Display all messages. In addition, a "hash mark" (#) is displayed for every 1,000 bytes sent or received.

If a command is sent to the remote server because it is not recognized locally, the response is always displayed regardless of the setting of verbose. This is necessary for commands like 'pwd' (display working directory) which would otherwise produce no message at all if verbose were set to 0 or 1.

End of Appendix F: FTP Session Commands

This file contains the instructions for forwarding to AX.25 BBSs. The file contains the instructions to determine destination, route, and areas to be forwarded as a series of forwarding 'records' with records being separated by a line containing two or more hyphens. The general layout for a forwarding record is:

```
--------
BBS callsign
Route
Connection commands
Areas to be forwarded  <one per line>
--------               <end of record>
```

Where:

BBS callsign is the ordinary call of the remote BBS.

On the same line you may put a list of intervals when forwarding is to take place. List items are separated by commas. Each interval is a four digit number where the first two digits are the beginning hour of the interval and the last two digits are the last hour of the interval.

Example: k0hyd 0006, 1515

Forwarding to k0hyd will occur during hours 00, 01, 02, 03, 04, 05, 06 and 15. Outside of these times the timer will not cause mail to be forwarded to k0hyd. The default forwarding interval is 0023 which means once each hour.

Route

First indicate the type of protocol to be used. This is either ax25, netrom or tcp. Next is information required to make the connection. For example, ax25 needs to have the port and the destination named while netrom and tcp only require the destination name without the port.

The following will all establish connections of the types indicated to k0hyd:

```
ax25 09 k0hyd-10
netrom scksbb
telnet k0hyd
```

To avoid any misunderstanding: The preceding three lines are examples of route statements. Only one type should be used for a route to any one station.

Connection commands

Connection commands are the script which may be required to complete the path and 'logon' to the destination BBS and come after the connection route in the record. These commands are entered one per line and take the form of a dot (period) followed by the command(s) which will be transmitted after the connection defined in the first line of the connection route is established.

A netrom connection route and connection command to are easy to show.

```
netrom kswch   <- Connect to netrom node
 .c k0hyd-10   <- Downlink to k0hyd
```

However, since k0hyd-10 is also netrom node 'scksbb', this script could more easily be written

```
netrom scksbb
```

Note: Some nodes have difficulty with lines that have the dot at the very beginning of the line. It is good practice to indent one space as is shown above in order to avoid such problems.

AX25 connections through digipeaters need to have the digipeater route entered in autoexec.nos. You cannot enter the digipeaters as part of the route in this file. Use the 'route add' command.

Areas to be forwarded

This is a list of areas in the SPOOL\MAIL directory which will be forwarded to the remote BBS. There is one entry per line and as many entries as desired may be used. Areas may be personal (callsign) or public (areaname).

An entry ,callsign' will cause the file SPOOL\MAIL\callsign.TXT to be scanned and unread messages to be sent to the remote BBS. The messages are then deleted from the file.

To forward public areas, place a line containing the areaname record. In a manner similar to the private areas, the program will search the 'areaname.txt' file and any messages which have not been forwarded to the destination BBS in question will be sent. These messages (bulletins) will NOT be deleted after forwarding. Public areas are ones which have their names entered in the 'areas' file which is a listing of the area names and a description for each area.

The format of the 'areas' file is:

areaname description

As a matter of convenience to your users, 'areaname' should be short but clear. This is because the areaname must be typed in full when changing areas. Descriptions can be any length, but the combination of areaname and description should fit on one line. If you have more than about twenty areas, when the user enters the 'AF' command the first ones will scroll off the top of the screen which can be frustrating.

Changing the recipient address

The 'To:' header line in a message is not changed by the alias or rewrite functions, but occasionally it may be desireable to alter the address used by the "S" command while forwarding. You can accomplish this in the forward.bbs file by appending the new address on the same line as the areaname. This would look like:

area new_address

This command will replace the originally typed destination with the string new_address.

End of Appendix G - Forward.BBS