Book B

Revision	Description
A	Initial release.

Contents

Tables

Chapter 1
Network Administration

This chapter discusses typical software configuration procedures as well as network administration using Annex tools and utilities. Using the Annex, you can:

- Monitor network activity.
- Monitor Annex activity.
- Secure the network.
- Manage the Annex's host table.

Monitoring Network Activity

The Annex provides three CLI commands (**netstat**, **ping**, and **arp**) to monitor network activity (for more details, see *Using the CLI Commands* on page A-121). Using the CLI commands, you can:

- Display network statistics.
- Test the network.
- Manage the ARP table.

Displaying Network Statistics

The CLI **netstat** command displays information that the Annex has obtained from the network. Using **netstat** you can display:

- Active connections.
- Ethernet statistics.
- PPP statistics.
- SLIP statistics.

- AppleTalk statistics.
- IPX statistics.
- RIP statistics.
- Routing table information.
- Route cache information.
- Dial-out route statistics.
- Rotary information.
- Filtering statistics.
- Memory statistics.
- Protocol statistics.

Active Connections

Entering the **netstat** command without arguments displays the local and remote addresses, send and receive queue sizes (in bytes), protocol, and the internal state of the protocol for all active connections. Table B-1 lists the arguments for this command.

Table B-1. Arguments for the netstat Command

Argument	Description
–A	Adds the protocol control block (PCB) addresses.
–a	Includes sockets used by server processes; can be used in combination with **–A**.

The **netstat –a** command display looks like this:

```
annex01# netstat –a

Active connections (including servers)
Proto  Recv-Q  Send-Q  Local Address  Foreign      (state)
                                      Address
tcp    0       2       annex1.telnet  test1.4759   ESTABLISHED
tcp    0       0       annex1.883     gibbs.login  ESTABLISHED
tcp    0       0       annex1.1085    ale.telnet   ESTABLISHED
tcp    0       0       annex1.1081    opus.telnet  ESTABLISHED
tcp    0       0       annex1.1022    test1.login  ESTABLISHED
tcp    211     0       annex1.953     xzyx.login   ESTABLISHED
tcp    0       0       annex1.1021    test1.login  ESTABLISHED
tcp    0       0       *.finger       *.*          ESTABLISHED
tcp    0       0       *.printer      *.*          ESTABLISHED
tcp    0       0       *.telnet       *.*          LISTEN
udp    0       0       *.bootp        *.*
udp    0       0       *.snmp         *.*
udp    0       0       *.who          *.*
udp    0       0       *.erpc         *.*
udp    0       0       *.route        *.*
```

Interface Statistics

The **netstat –i** command displays interface statistics for an Annex running on an Ethernet LAN. Table B-2 describes the hardware interface statistics for Ethernet.

Table B-2. Hardware Interface Statistics for Ethernet

Statistic	Description
Frames Received	The number of packets received from the network interface.
Frames Transmitted	The number of packets transmitted on the network interface.
Bytes Received	The number of bytes received from the network interface.
Bytes Transmitted	The number of bytes transmitted on the network interface.
CRC Errors	The number of frames received from the network interface with a bad CRC.
Alignment Errors	The number of frames received from the network interface that were both misaligned and have a CRC error.
Bad Type/ Length Fields	The number of frames received from the network interface that have an unrecognized type field (ethernet) or an illegal length field (802.3).
Buffer Drops	The number of frames received from the network interface that were good, but dropped because no buffers were available.
FIFO Drops	The number of frames received from the network interface that were lost since the local system bus was not available.
Interface Resets	The number of times the network interface has been initialized from reset; typically, one.
TX DMA Underruns	The number of times a frame transmission is terminated due to lack of data.

(continued on next page)

Table B-2. Hardware Interface Statistics for Ethernet (continued)

Statistic	Description
RX DMA Overruns	The number of times a frame reception is terminated due to lack of system bus bandwidth.
Carrier Sense Losses	The number of times a frame transmission is terminated due to loss of the Carrier Sense signal. The transceiver cable may have a short or an open.
Clear to Send Losses	The number of times a frame transmission is terminated due to loss of the Clear to Send signal.
Collisions Detected	The number of times a frame transmission is terminated due to a collision.
Max Collision Retries	The number times consecutive collisions for a frame exceed the maximum collision retry limit.

The **netstat –i** command display looks like this:

```
annex01# netstat –i

Name   Mtu   Network      Address   Ipkts  Ierrs  Opkts   Oerrs  Collis
en0    1500  132.245.66.0 worm      26563  0      15085   744    0
en0    1500  10000–20000  18062.79  1626   0      823     0      0
lo0    1536  127          127.0.0.1 0      0      0       0      0
asy2   604   18358        18062.79  0      0      0       0      0
asy16  1006  132.245.6    annex01   14770  0      7468    0      0
asy3   1500  192.9.200    zipwad    3453   0      3002    0      0

                    *** Hardware Interface Statistics ***

Ethernet Address:      00-80-2d-00-00-9b
Frames Received:       39861             Frames Transmitted:   45239
Bytes Received:        33965470          Bytes Transmitted:    29453
CRC Errors:            2                 Alignment Errors:     10
Bad Type/Length Fields:6                 Buffer Drops:         0
FIFO Drops:            1                 Interface Resets:     1
TX DMA Underruns:      241               RX DMA Overruns:      0
Carrier Sense Losses:  451               Clear to Send Losses: 0
Collisions Detected:   17526             Max Collision Retries:125
```

PPP Statistics

The **netstat –ip** *port-number* command displays a summary of a PPP interface and its current state. Table B-3 describes the fields in the **netstat –ip** command display.

The **netstat –ip** command display looks like this:

```
annex01# netstat -ip 5

                *** LCP Status ***
State           Current: Open         Prior: Ack sent
MRU             Local:                Remote:
Auth type       1500                  1500
LQM             PAP                   CHAP
ACFC            None                  None
ACCM            0x00000000            0x00000000
Magic           0x32ed028b            0x6694d55e
PFC             On                    On

                *** NCP (IPCP) Status ***
State           Current: Open         Prior: Ack sent
Options         Local:                Remote:
IP addresses    192.0.5.242 [ACP]     192.0.5.243 [ACP]
Compression     None                  None

                *** NCP (ATCP) Status ***
State           Current: Request sent Prior: Request sent

                *** NCP (IPXCP) Status ***
State           Current: Open         Prior: Ack sent
Options         Local:                Remote:
Network No      00000001              00000001
Node No         00802d00bb7f          00802d00abf6
Compression     None                  None
Routing Prot    RIP/SAP               RIP/SAP
Router Name     LM00BB7F              LM00ABF6
```

If *compression* is set, the values that appear in the display are *Max–slot–id* and *Comp–slot–id*. These values are sub-options of VJ compression.

Table B-3. Field Definitions for the netstat –ip Command

Field	Definition
local	Refers to the Annex.
remote	Refers to the peer.
[xxx]	The origin of the value for *ip–addresses*: ANX=param; REM=peer–defined; and ACP=from security server ACP dial–up addresses.
LCP and NCP (IPCP) Options	Shows the current and the prior state of the connection. Any *current* setting other than *Open* indicates the link is not up. The states are:
Closed	The layer has shut down via an administrative or peer request.
Request sent	The Annex has sent a configure request and is waiting for an answer.
ACK received	The Annex has received a configure ACK and is waiting for a configure request.
ACK sent	The Annex received and answered a configure request.
Open	Layer negotiation has completed successfully.
Closing	The link is in process of closing. The Annex has sent a terminate request and is waiting for a terminate ACK.
Security	Shows the states based on the last security messages sent and received; this field appears only in superuser mode.

(continued on next page)

Table B-3. Field Definitions for the netstat –ip Command (continued)

Field	Definition
Possible local states for PAP security:	
Initial	No PAP security has been initiated.
AREQ received	The Annex has received the Authenticate-Request message and currently is processing it.
ANAK sent	The Annex has rejected the peer's Authenticate-Request; the link will be coming down.
AACK sent	The Annex has authenticated the peer.
Possible remote states for PAP security:	
Initial	No PAP security has been initiated.
AREQ sent	The Annex has sent the Authenticate-Request message and is waiting for the response.
ANAK received	The Annex's Authenticate-Request has been rejected by the peer; the link will be coming down.
AACK received	The peer has authenticated the Annex.
Possible states for CHAP security:	
CHAP AACK Rcvd	CHAP Authentication Acknowledged Received.
CHAP AACK Sent	CHAP Authentication Acknowledged Sent.
CHAP CHAL Rcvd	CHAP Challenge Received.
CHAP CHAL Sent	CHAP Challenge Sent.
CHAP RESP Rcvd	CHAP Response Received.
CHAP RESP Sent	CHAP Response Sent.

SLIP Statistics

The **netstat –iS** command displays SLIP data after the hardware interface statistics:

```
annex01# netstat –iS

Name  Mtu   Network      Address   Ipkts  Ierrs  Opkts  Oerrs  Collis
en0   1500  192.9.200    annex1    648918 0      352845 0      0
lo0   1536  127          127.0.0.1 0      0      0      0      0
asy6  1006  192.9.200    annex1    0      0      0      0      0
asy13 256   192.9.200    annex1    0      0      0      0      0

                  *** Hardware Interface Statistics ***

Ethernet Address:      00-80-2d-00-14-3d
Frames Received:       705482            Frames Transmitted:   35283
Bytes Received:        62425605          Bytes Transmitted:    193578
CRC Errors:            0                 Alignment Errors:     0
Bad Type/Length Fields:0                 Buffer Drops:         0
FIFO Drops:            0                 Interface Resets:     1
TX DMA Underruns:      0                 RX DMA Overruns:      0
Carrier Sense Losses:  0                 Clear to Send Losses: 0
Collisions Detected:   2389              Max Collision Retries:0

        SLIP rcvr:
             intrs 0, loops 0, bytes 0, pkts 0
             bytes/intr 0, bytes/loop 0, bytes/pkt 0
             hiwaters 0, overflows 0, mbuf waits 0, mbuf kicks 0
             overruns 0, ipintrq full 0
             FRAME_ENDs 0, FRAME_ESCs 0, proto errs 0, last proto err 0
        SLIP xmit:
             intrs 13, starts 22, vectors 108, bytes 1874, pkts 13
             FRAME_ENDs 22, FRAME_ESCs 32
             bytes/intr 144, bytes/vec 17, vec/pkt 8, bytes/pkt 144
```

AppleTalk Statistics

The **netstat –i** command displays interface statistics. AppleTalk addresses display as *net.node* in hexadecimal, where *net* is 16 bits and *node* is 8 bits. ARAP interfaces display as *ara* plus the unit number. Table B-4 defines the arguments for displaying AppleTalk statistics.

Table B-4. Displaying AppleTalk Statistics using the netstat Command

Argument	Description
–i	Displays interface statistics.
–ip *port number*	Displays a specific Annex PPP interface (see *PPP Statistics* on page A-6).
–z	Displays the network zone list.

The **netstat –i** command display looks like this:

```
annex01# netstat -i

Name  Mtu   Network      Address    Ipkts  Ierrs Opkts  Oerrs Collis
en0   1500  132.245.66.0 worm       26563  0     15085  744   0
en0   1500  10000-20000  18062.79   1626   0     823    0     0
lo0   1536  127          127.0.0.1  0      0     0      0     0
asy2  604   18358        18062.79   0      0     0      0     0
```

(continued on next page)

```
          *** Hardware Interface Statistics ***
Ethernet Address:  00-80-2d-00-00
                   -9b
Frames Received:   39861          Frames             452397
                                  Transmitted:
Bytes Received:    33965470       Bytes Transmitted: 2945335
CRC Errors:        2              Alignment Errors:  10
Bad Type/Length    6              Buffer Drops:      0
Fields:
FIFO Drops:        1              Interface Resets:  1
TX DMA Underruns:  241            RX DMA Overruns:   0
Carrier Sense      451            Clear to Send      0
Losses:                           Losses:
Collisions         17526          Max Collision      125
Detected:                         Retries:

       *** IEEE 802.2 Data Link Layer Statistics ***
802.2 packets received:1    802.2 packets sent:      0
ATALK packets sent:    0    AARP packets sent:       0
ATALK packets received:0    AARP packets received:   0
Unknown 802.2 types:   0    Unknown 802.2 SAP's:     0
Unknown SNAP org codes:0    Unknown SNAP ether types:0
```

RIP Statistics

The **netstat –g** command displays RIP statistics. Table B-5 describes the field definitions for the command display.

The **netstat –g** command display looks like this:

```
annex01# netstat –g
Input packets: 19942, Output packets:   0
Interface triggers:   2, Timer events: 4818   Load trips: 0

Sources:

132.245.33.22:   4661 packets      132.245.33.34:   5632 packets
132.245.33.228:  4822 packets      132.245/33/238:  4816 packets
132.245.33.138:  9                 132.245.33.254:  1 packet

Rooting Changes: 1  Queries received: 0

Intf  Bad   Bad   Trigg.  Recv'd  Sent  Disc'd  Update  Queries
      Pkts  Rtes
en0   0     0     0       19942   0     0       22      4
```

Table B-5. Field Definitions for the netstat –g Command

Field	Definition
Intf	Displays the interface.
Bad Pkts	Displays the number of packets the interface dropped due to invalid format or data.
Bad Rtes	Displays the number of routes the interface dropped due to invalid format or data.
Trigg.	Displays the number of triggered updates transmitted over the interface. The Annex sends triggered updates whenever it changes the hop count of a route. It transmits them immediately, even if it is not yet time for one of the regular update messages to be transmitted.
Rec'd	Displays the number of packets (with or without errors) received over the interface.
Sent	Displays the number of output packets the Annex tried to send over the interface. This number includes packets that were dropped because the Annex ran out of buffers or the link's output queue was full.
Disc'd	Displays the number of input packets discarded due to protocol errors or restrictions set by configuration parameters (e.g., rip_accept).
Update	Displays the number on lines in the routing table that were modified due to packets received on that interface.
Queries	Displays the number of routing-table queries received on the interface.

Routing Table Information

The **netstat –r** command displays statistics and information about all available routes in the RIP routing table, including dial-out routes; dynamic dialing routes that do not have a phone connection established appear with a *w* at the end of the route entry. Table B-6 lists the **netstat** command arguments that display routing information. Table B-7 describes the field definitions for the **netstat –r** command display.

Table B-6. Displaying Routing Table Information using the netstat Command

Field	Definition
–r	Displays statistics and information about all available routes in the routing table. A route comprises a destination host or network and the gateway through which data is forwarded. If the dial-out route currently is not active, only *do<route number>* appears in the *Interface* field. If the route currently is active, *asy<port number>* appears in the *Interface* field.
–ra	Displays only AppleTalk routes.
–ri	Displays only IP routes.

The **netstat –r** command display looks like this:

```
annex: netstat –r

tables
Destination     NextHop        Flags  Usage   UseCount Mtr  Interface
4400 - 4499     4475.129       UHF    1       3        0    en0
Apple default   4400.22        UGF    0       0        0    en0
IP default      132.245.44.22  US     +0      0        2    en0
127.0.0.0/8     *              UI     fixed   0        2    lo0
132.245.1.0/24  132.245.44.22  UR     -114    0        3    en0
132.245.2.0/24  132.245.44.22  UR     -114    0        2    en0
132.245.9.0/24  132.245.44.22  UR     -78     36       2    en0
132.245.10.0/24 132.245.44.22  UR     -114    0        2    en0
132.245.11.0/24 132.245.44.22  UR     -114    0        2    en0
132.245.12.0/24 132.245.44.22  UR     -114    0        2    en0
132.245.22.0/24 132.245.44.22  UR     -114    0        2    en0
132.245.33.0/24 132.245.44.22  UR     +33     147      2    en0
132.245.34.0/24 132.245.44.22  UR     -114    0        2    en0
132.245.44.0/24 *              UI     fixed   8382     1    en0
bermuda         132.245.44.22  USH    -114    0        2    en0
132.245.66.0/24 132.245.44.22  UR     -114    0        2    en0
132.245.77.0/24 132.245.44.22  UR     -114    0        2    en0
```

Table B-7. IP Fields in the netstat –r Command Display

Field	Explanation
Destination	The IP address of the route's destination, followed by a slash (/), followed by the number of 1 bits, counting from left to right, in the Destination's subnet mask. For example, the */24* following the IP address *132.254.1.0* indicates a subnet mask of 24 bits (eight octets), or 255.255.255.0. (For more information, see *Entering Routes in the Remote Annex Configuration File* on page A-194.) If *IP Default* appears in the Destination field, the entry specifies the route the Annex uses if it can find no other route for a destination. If a name appears in the Destination field, the entry is for a host route; name servers do not have names for network routes. (However, the Annex does not always know a host's name.)
NextHop	The next router to which packets with the given Destination are sent. If the Destination is a local interface, this field displays an asterisk (*); interface routes have no next hop.
Flags	The following three flags:
First flag (Status) *U* *Q* *D*	 The route is valid (up) and in use. The route is valid but the interface is quiescent, i.e., the interface is not up yet or was brought down by expiration of the timer set by the **net_activity** port parameter. The route is invalid (down) and has a metric of 16 (RIP infinity). It will stay in the routing table for two more minutes so that other routers can learn that it is invalid.

(continued on next page)

Table B-7. Fields in the netstat –r Command Display (continued)

Field	Explanation
Second flag (Source) *C* *I* *R* *S*	The route was learned via an ICMP redirect. This can occur only when IP routing is disabled (by setting the **routed** parameter to **N**). The route is an interface route. The route was learned via RIP. The route is a static route, learned from a route defined in the **gateway** section of the Annex configuration file or a route entered using the CLI superuser **route** command.
Third flag *H*	The route is a hardwired static route.
Usage	A positive or negative integer indicating a route's usage. When RIP adds a route to the routing table, it sets its usage value to 0. Every time the route is used RIP adds one to the value; every thirty seconds RIP subtracts one from the value. When the routing table reaches its maximum size of 256 entries, RIP removes the route with the lowest usage value. If there is a tie, RIP removes the first route listed. The values range from -9999999, for a route that has not been used in 9.5 years, to +9999999, for a very frequently used route. Interface, hardwired, and *extremely* frequently used routes contain the word *fixed* in this field instead of a number.
UseCount	A positive integer indicating the number of times the route has been used to transmit a packet. If you subtract the value in this field from the value of *Usage*, you can determine how long a route has been in the routing table.
Mtr	The metric for the route.
Interface	The interface over which the Annex can reach the next hop.

Route Cache Information

The **netstat –C** command displays the contents of the cache route, including both static routes added from the **gateways** section of the configuration file and routes added by the **route** command. Table B-8 describes the flags for the command display.

Table B-8. Flag Descriptions for the netstat –C Command

Flag	Definition
intf *x*	An interface route, where *x* is the interface name and number, e.g., asy8. This can be a back-up route for a an interface that has a duplicate definition in the routing table. For example, if you define a subnet mask for a Proxy-ARP serial interface, and that mask is the same as the Annex's en0 subnet mask, the routes to that interface will be considered duplicates. As a result, the Annex will store the en0 interface route in the routing table and the serial interface route in the cache, thus making the serial interface unreachable. The example below shows a dial-out route, *do67*.
hardwired	Route added either by the **route –h** command or a route defined as ***hardwired*** in the **gateway** section of the Annex configuration file.

The **netstat –C** command display looks like this:

```
annex01# netstat –C

Destination      Subnet Mask     Gateway         Metric  Flags
default          0.0.0.0         132.245.33.22    1
74.68.67.0       255.255.255.0   0.0.0.0          1      intf do67
132.245.124.0                    132.245.71.72    2      hardwired
```

Dial-out Route Information

The **netstat –r** command displays statistics and information about all available routes in the routing table, including dial-out routes. If the dial-out route currently is not active, only *do<route number>* appears in the *Interface* field. If the route has been assigned to a port, either *slip <port number>* or *ppp <port number>* appears in the *Interface* field. Table B-7 on page A-15 describes the field definitions for the **netstat –r** command display.

The **netstat –i** command displays the dial-out route's interface name. A truncated view of the command display looks like this:

```
annex01# netstat –i

Name Mtu  Network    Address       Ipkts Ierrs Opkts Oerrs Collis
en0  1500 132.245.33 132.245.33.50 16    0     16    0     0
lo0  1536 127        127.0.0.1     0     0     0     0     0
do1  1500 1          132.245.33.90 4     0     4     0     0
```

Rotary Information

The **netstat –R** command displays all rotaries configured for the Annex. Table B-9 describes the field definitions for the command display. The **netstat –R** command display looks like this:

```
annex01# netstat –R

Rotary name Address                Proto  Camp Flags Annex ports
oemandy1    *.telnet               telnet ask       11
conan_33    *.telnet               telnet ask       16
borneo1     192.9.200.250.telnet   telnet ask       1
brazil7     192.9.200.253.6003     telnet ask       7
annex3      *.6103                 telnet ask       8,13,15
```

Table B-9. Field Definitions for the netstat –R Command

Field	Definition
Rotary name	Displays the name of the rotary.
Address	Displays the auxiliary address, if assigned, or an asterisk (*), indicating the rotary has the same address as the server.
Proto	Displays the assigned protocol.
Camp	Displays the camp-on options: ***ask***, ***always***, or ***never***.
Flags	Displays ***I*** if the rotary is invisible.
Annex ports	Displays the port(s).

Filtering Statistics

The **netstat –f** command displays filtering statistics. The statistics are cumulative for the Ethernet ports, i.e., changing filters does not reset the counters. The counters for a SLIP and PPP line reset each time the connection resets. Table B-10 describes the field definitions for the command display.

The **netstat –f** command display looks like this:

```
annex01# netstat -f

Int     In-hits    Out-hits     Drop        ICMP       Syslog
en0     0          0            0           0          0
asy1    0          0            0           0          0
asy2    0          0                        0          0
```

Table B-10. Field Definitions for the netstat –f Command

Field	Definition
Int	Displays the interface.
In-hits	Displays the number of packets that matched an input filter.
Out-hits	Displays the number of packets that matched an output filter.
Drop	Displays the number of discarded filtered packets.
ICMP	Displays the number of filtered packets that sent an ICMP message.
Syslog	Displays the number of filtered packets that were syslogged.

Memory Statistics

The **netstat –m** command displays statistics for the memory management routines:

```
annex01# netstat –m
1127/3599 mbufs in use:
        7 mbufs allocated to data
        2 mbufs allocated to packet headers
        9 mbufs allocated to socket structures
        14 mbufs allocated to protocol control blocks
        3 mbufs allocated to routing table entries
        2 mbufs allocated to socket name
        2 mbufs allocated to interface address
          64 mbufs allocated to incoming network i/f packets
        1024 mbufs allocated to SPD Layer RX Data/Status
899 Kbytes allocated to network (31% in use)
0 requests for memory denied
```

Protocol Statistics

The **netstat –s** command displays statistics for the following protocols: ICMP, UDP, TCP, IP, TMux, LAT, and DDP. The LAT statistics display only if the correct **lat_key** value is set; TMux statistics display only if the **tmux_enable** parameter is set to **Y**; DDP statistics display only if the correct **option_key** value is set. A truncated view looks something like this:

```
annex01# netstat –s

    tcp:
     3097 data packets sent
     394865 packets sent
                  309577 data packets (1011910 bytes)
                  87 data packets (22401 bytes) retransmitted
    udp:
     0 incomplete headers
     0 bad data length fields
     0 bad checksums
     2755 no listening port
     77148 packets received
     956 packets sent

    ip:
     613422 total packets received
     0 bad header checksums
     4 output packets we did frag
     5 output fragments we created
```

(continued on next page)

```
icmp:
     2359 calls to icmp_error
     0 errors not generated 'cuz old message too short
     0 errors not generated 'cuz old message was icmp
     Output histogram:
       destination unreachable: 2358
     Input histogram:
       echo reply: 41
tmux:
     65 packets from upper levels
       0 TMUX packets sent
       0 not suitable to TMUX
       0 dropped by TMUX
       65 not able to TMUX
     0 packets from IP
       0 encapsulated packets received
       0 TMUX checksum fails
       0 TMUX other fails
     1 TMUX ENQ packets sent
lat:
     241 Total run messages received
     228 Total run messages transmit
     56382 Total service messages recv.
     3796 Total service messages used
ddp:
     0 short header packets received
     13838 long header packets received
     12120 no checksum
     1 packet too short
     5 not enough data
     13671 packets forwarded
     0 packets encapsulated
```

Testing the Network

The superuser CLI **ping** command tests and measures the LAN. Also, it can isolate a single-point hardware or software failure. The **ping** command sends out an Internet Control Message Protocol (ICMP) echo request packet each second, or until input from the terminal terminates the command. After completing, **ping** displays a summary of all echo replies received. This display includes a calculation of the time, in milliseconds, that it takes to return the message (if the number of data bytes is 8 or greater). Table B-11 lists the arguments for this command The syntax is:

ping [–**artv**] *host* [*databytes* [*count*]]

Table B-11. Arguments for the ping Command

Argument	Description
–a	Generates AppleTalk Echo Protocol (AEP) echo request packets to a target node.
–r	Bypasses the normal routing table and sends the message directly to a host on an attached network. An error returns if the host is not on a directly attached network.
–t	Traces the path of a packet from the local host to the destination host and back, displaying information about each router in the path. This option allows you to see whether a packet arrived at and/or returned from its remote destination and, if not, where it stopped. The option is based on the Traceroute facility described in RFC 1393 (see *Using the –t (traceroute) Option* on page A-165 for more details). You can use **–t** with the **–r** and/or **–v** argument(s), but not with **–a**.

(continued on next page)

Table B-11. Arguments for the ping Command (continued)

Argument	Description
–v	Displays the IP and ICMP packet headers for the reply from the host.
host	The host, router, or Annex to which the **ping** is sent.
databytes	The number of bytes of data in the ICMP Echo Request message. The default is **56**.
count	The number of ICMP Echo Request messages to be sent to the **ping** destination. The default is unlimited. When invoked with **–t**, **ping** ignores the *count* argument.

The **ping** command display looks like this:

```
annex01# ping caddy
PING caddy: 56 data bytes
64 bytes from 132.245.6.25: icmp_seq=0. time=37. ms
64 bytes from 132.245.6.25: icmp_seq=1. time=12. ms
64 bytes from 132.245.6.25: icmp_seq=2. time=12. ms
64 bytes from 132.245.6.25: icmp_seq=3. time=12. ms
----caddy PING Statistics----
4 packets transmitted, 4 packets received, 0% packet loss
round-trip (ms) min/avg/max = 12/20/37
```

For more details, see *ping* on page A-161.

Managing the ARP Table

The Address Resolution Protocol (ARP) maps Internet addresses to hardware addresses. Hosts implementing ARP maintain a translation table for these address mappings. When an Annex receives a request for a host that does not have a translation entry in the ARP table, it broadcasts for the hardware address. The superuser CLI **arp** command displays and modifies entries in this translation table.

Since the Annex automatically builds the ARP table dynamically, you rarely need to modify the table. You can use **arp** to modify the table for hosts that do not implement ARP, enabling communications between the host and the Annex. Using **arp**, you can delete a specified entry and/or create an entry for a host.

A created entry is permanent unless it is defined as temporary, in which case the entry is deleted after 20 minutes. An entry defined as published causes the Annex to respond with its hardware address for the specified host, even though the IP address is not the Annex's. Publishing a hardware address for another host frequently is done to route data to a host connected to the Annex through a SLIP or PPP link.

Monitoring Annex Activity

The CLI commands assist in monitoring Annex activities (see *Using the CLI Commands* on page A-121 for more information). These activities include:

- Logging user and Annex activities.
- Displaying user activity.
- Displaying Annex statistics.
- Monitoring serial line activity.

Logging User and Annex Events

The Annex provides two mechanisms for logging events: host-based security and a 4.3BSD-style **syslog** daemon (see *Using Remote Annex Security* on page A-421 for details on host-based security and ACP).

Host-based Security Logging

Host-based security provides logging capabilities that maintain audit trails of user activity. The security server logs each event as a message to its ACP log file. Security logging is enabled automatically when host-based security is enabled for an Annex (using the Annex parameter **enable_security**).

Refer to the *Remote Annex Server Tools for Windows NT® User Guide* for information about host-based security logging in the Windows NT® environment.

Events are logged to the security server that responded to the security request, either granting or denying access requests. When using back-up security servers, the ACP log file is located on each server.

To change the name and/or format of the ACP log file, see *Modifying the Supplied Security Application* on page A-546.

Each logged message in the ACP log file contains the following fields:

- *IP address of the Annex.*
- *Sequence number.*
- *Port number.*
- *Date.*
- *Time.*
- *Module.*
- *Event.*

- *Packets in.*
- *Packets out.*
- *Bytes in.*
- *Bytes out.*
- *Protocol-dependent information*
- *Username*.

All fields are separated by colons and are encoded for use by UNIX utilities that sort, merge, select, or filter streams.

When more than one host functions as a security server, the log files can be merged and sorted by the date and time fields. Following is a sample log file:

```
132.245.11.11:420b02bb:#01:950626:003015:cli hook:login:moseley
132.245.11.11:420b02bc:#01:950626:003015:telnet:login:132.245.77.1:23:moseley
132.245.11.11:420b02bd:#02:950626:010620:ipx:login:djones
132.245.11.11:420b02be:#02:950626:010900:ipx:logout:djones
132.245.11.11:420b02be:#02:950626:010900:ipx:acct:191:190:29486:12577:djones
132.245.11.11:420b02bf:#01:950626:011456:telnet:logout:132.245.77.1:moseley
132.245.11.11:420b02c0:#01:950626:011502:cli hook:logout:moseley
132.245.11.11:420b02c0:#01:950626:011502:cli hook:acct:0:0:1021:143882:moseley
132.245.11.11:420b02c1:#04:950626:012317:rlogin:logout:132.245.33.7:mildram
132.245.11.11:420b02c2:#04:950626:012317:rlogin:logout:132.245.33.16:mildram
132.245.11.11:420b02c3:#04:950626:012317:cli hook:logout:mildram
132.245.11.11:420b02c3:#04:950626:012317:clihook:acct:0:0:10286:196301:mildra
132.245.11.11:420b02c4:#01:950626:012420:cli hook:login:mildram
132.245.11.11:420b02c5:#01:950626:012421:rlogin:login:132.245.33.7:513:mildram
132.245.11.11:420b02c6:#01:950626:013758:rlogin:logout:132.245.33.7:mildram
132.245.11.11:420b02c9:#01:950626:064309:telnet:login:132.245.77.1:23:tetreault
132.245.11.11:420b02ca:#02:950626:064948:cli hook:login:mcgillivray
132.245.11.11:420b02cb:#02:950626:064949:telnet:login:132.245.77.1:23:mcgill
132.245.11.11:420b02cc:#01:950626:065207:telnet:logout:132.245.77.1:tetreault
132.245.11.11:420b02ce:#01:950626:070102:cli hook:login:pearson
132.245.11.11:420b02cf:#01:950626:070109:telnet:login:132.245.77.1:23:pearson
132.245.11.11:420b02d0:#02:950626:070434:telnet:logout:132.245.77.1:mcgillivray
132.245.11.11:420b02d1:#02:950626:070436:cli hook:logout:mcgillivray
132.245.11.11:420b02d1:#02:950626:070436:cli hook:acct:0:0:1063:69595:mcgill
132.245.11.11:420b02d2:#01:950626:071048:telnet:logout:132.245.77.1:pearson
```

Events are written continuously to the ACP log file. To prevent this file from overwhelming the file system on the hosts, and still obtain the record information for generating reports, move and compress the file at regular intervals. The size of your network, the number of Annexes, and the amount of activity generated at each Annex determines the frequency for moving and compressing the file.

Events written while using ARA or the dial-back security feature have their own messages:

- **bad access code**

  Users entered an unidentified access code for the defined username – the login was terminated.

- **call-back**

  Users logged in with a known username and access code – the Annex calls back a pre-defined phone number (this log can be generated by any of the features that perform a call-back, including dial-back security and ARA); possible status values are **request**, **ok**, **no answer**, or **no device**.

Including the call-back message, the new messages generated by ACP while processing ARA logins are:

- **login**

  User is authenticated and session is started.

- **logout**

  Session exited via user hang-up, time-out, or administrator **reset**.

- **reject**

  Authentication failed.

New log messages are generated by the **acp_userinfo** file parser if an error is detected when processing the **acp_userinfo** file (see *Using include Files in the acp_userinfo File* on page A-477 for more details).

The demand dial and modem code include debug level syslog information that provides progress, status, and failure information. This information appears in the following format:

```
Apr 2 1:53:42 annex.site.com ppp[323]: asy15 configuring dynamic dial interface
Apr 2 1:54:06 annex.site.com ppp[323]: asy15 type_of_modem is "Optima96"
Apr 2 1:54:06 annex.site.com ppp[323]: asy15 use cli modem command to verify
modem information for "Optima96"
Apr 2 1:54:06 annex.site.com ppp[323]: asy15 attempting to activate dynamic
dial interface
Apr 2 1:54:06 annex.site.com ppp[323]: asy15 sending reset string "ATZ"
Apr 2 1:55:36 annex.site.com ppp[323]: asy15 send/expect timed out
(numeric result codes expected)
Apr 2 1:55:36 annex.site.com ppp[323]: asy15 sending reset string "ATZ"
```

Event Logging Using syslog

The Annex can log events for a system running a 4.3BSD-style **syslog** daemon or syslog to a serial port on the Annex. The Annex parameter **syslog_port** defines the port to which logged messages are sent (for more details, see *Using Event Logging* on page A-37 and *syslog_port* on page A-108). The logged message includes:

- The date and time of the event.
- The name or IP address of the Annex on which the event occurred.
- The name of the event and PID of the Annex process.
- A description of the event.

In the following example, on May 5, at 9:19 a.m., a user named *Worth* on port 8 of *annex01* issued the **rlogin** command to host *galago*.

```
May 5 9:19:03 annex01 cli[598]:Job-Begin:8:rlogin
galago:Worth
```

The information display differs, depending on the event. In the following example of a typical message, a time server updates the Annex's time. The time server host's address displays in hexadecimal longword. Times are expressed in hexadecimal as the number of seconds since 00:00:00 January 1, 1970.

```
Jan 5 9:56:5 annex timed[38]:adjusting time from host
      5fc809c0: old=25bf1398, new=25bf1399, delta=1
```

The next example shows a user on port 9 of *annex* issuing the **telnet** command to access another Annex.

```
May 5 8:56:3 annex telnet_cmd[35]:Telnet-Begin:9:telnet
annex1
```

The next example shows a request for the printer on *annex* through the port server.

```
May 5 8:17:5 annex
rdr[39]:Port-Begin:14:RDP:LPRt10:Actg:ager
```

You can create audit trails and accounting reports for the Annex and its serial ports by sorting and merging log entries.

Displaying User Activity

When the CLI **who** command is issued for an Annex, it displays the user name, the jobs the user is running, when the connection began, any idle time, and the source of the connection. This command also displays current users on other Annexes, and on other hosts, if those hosts have **fingerd** running for **who** *user@host*.

The **who** command display looks like this:

```
annex01# who

Port   What  User   Location     When    Idle Address
1      CLI   bob    Ext 528      8:44am       [local]
2      CLI   ---    ---          9:02am       [local]
4      LPD   ---    ---          9:45am       oaxaca
6      ARAP  cobb   P-01-03-con  9:59am       [local]
16     PSVR  cody   lpq port     10:00am :43  support
v1     CLI   ellis  Ext 632      10:00am :41  192.9.200.133
v2     CLI   carey  ---          10:43am      192.9.200.60
```

When the command is issued for a 4.3BSD host, the display is the same as for the **finger** command executed at the host. Using the **who** command, you can obtain a significant amount of information on users and their activities in the network. For example:

- All users connecting to or from a specific host(s).
- A single user or a group of users connected to the Annex.
- All users connected to specific port or virtual CLI.
- A specific user (**who** *user@host*) or all users (**who** *@host*) logged into a specific host.

Using abbreviations, you can display a range of hosts or user names.

Displaying Annex Statistics

The CLI **stats** command displays general Annex statistics, or statistics for one or more serial ports (see *stats* on page A-184 for more details). A typical **stats** command display for a Remote Annex on an Ethernet network looks like this:

```
annex: stats

S/W Version: Remote Access Rx.x Build #2: Thu Sep 14 20:37:27 EDT 1995
H/W: Remote Annex 4000          H/W Rev: 36. ROM Rev 0811.
Comm: eth-aui&twi/64asy/1par     Mem: 5mDRM/64kEEPRM/16kSL1/16kSL2
Boot from: 132.245.88.5         Date: Thu Sep 21 13:27:50 1995 EDT
Image: oper.46.enet             Uptime: 15 hours 48 mins
Inet addr: 132.245.88.170       Subnet mask: 255.255.255.0
Ethernet addr:00-80-2d-00-b4-42 Broadcast addr: 132.245.88.255
Default domain: <unknown>
     CPU current/average = 1%/0%  procs active/max/limit = 87/88/800
     rescheds = 0/32  switches = 48/109401  activates = 49/109722
Loading:
     CPU current/average = 1%/0%  procs active/max/limit = 87/88/800
     rescheds = 0/32  switches = 48/109401  activates = 49/109722
Mbufs:
     total=5400  free=3273  minimum free=3200  denied=0
Serial Ports:
     Total bytes:  rcv'd=24982  xmt'd=5934
     Errors:  parity=0  framing=0  fifo overruns=0
Parallel Ports:
     Total bytes:  xmt'd=0
Memory:
     total=5242880 avail=3894424 free=2073480 min free=1782488
     fails=0
annex:
```

The **stats –s** command displays statistics for all serial ports:

```
annex01# stats -s

P#   Control      Speed  CharTx  CharRx ParityOverru Framing
     Lines                                      n
1    none         38400     255 0      0     0      0
2    CTS RTS       4800     255 0      0     0      0
3    none         19200     255 0      0     0      0
4    DTR DCD DSR  38400  176715 4123   0     0      0
5    DTR DCD DSR   9600  937802 7864   0     0      0
6    idle         idle   0       0      0     0      0
:
total                    1118837 11987  0     0      0
```

The **stats –p** command displays statistics for all parallel ports:

```
annex01# stats -p

P#    Type   CharTx   Status
1     CT     576      selected, paper error, busy
2     DP     1318     selected
total        1894
```

The superuser **stats –c** command clears all serial line statistics to zero.

The **stats –m** command displays statistics for active control lines, but displays the modem controls for inactive control lines rather than displaying *idle*.

```
annex01# stats -m

P#  Control Lines      Speed CharT CharRx Parit OverrunFraming
                             x           y
1   CTS RTS DTR DCD DSR9600  0     0     0     0       0
2   CTS TRS DTR DCD DSR9600  0     0     0     0       0
3   cts RTS DTR dcd dsr9600  0     0     0     0       0
4   cts RTS DTR dcd dsr9600  0     0     0     0       0
5   cts RTS DTR dcd dsr9600  0     0     0     0       0
:
64  cts RTS DTR dcd dsr9600  0     0     0     0       0
```

Monitoring Serial Line Activity

The Annex provides two superuser CLI commands that display information about the state of the Annex's serial ports: **control** and **tap** (see *control* on page A-141 and *tap* on page A-205 for more details).

The superuser CLI **control** command is a diagnostic tool that, for a specified port, allows you to set DTR and RTS or output a short test message. The superuser CLI **tap** command accesses (wire taps) a serial port from a terminal.

The **tap** command will not work with PPP.

Using **tap**, you can:

- Observe the output to the port. The command also displays keystrokes entered from your terminal as output to the port you are tapping as if they had been entered on the port.
- Find out exactly what users are seeing on their terminals from a remote location.
- Provide on-line advice and instructions to users at their terminals.
- Monitor traffic in both directions on the port, especially incoming special conditions, such as line breaks and special characters.

Under certain circumstances, the order of displayed data may not match the actual time sequence of the events. All input and output data is displayed. Special characters and control line changes are stored in a limited buffer. If these changes occur too rapidly, they may be lost.

Managing the Host Table

The host table contains this information for each host:

- Host name.
- Aliases (if any).
- IP address.
- Multiple IP addresses (if any).
- System status (if the entry is updated by RWHO).
- Load factor (if the entry is updated by RWHO).
- Number of users (if the entry is updated by RWHO).

The CLI **hosts** command displays all entries in the host table. The Annex can build and update the host table from RWHO messages and from responses to DNS and/or IEN-116 queries. Entries are updated according to information received. Information for a host will be updated if new information received is different from what is currently in the host table. The Annex considers information from a DNS server the most reliable source; it considers an IEN-116 as the next reliable source; and it considers RWHO broadcasts as the least reliable source. Thus, information from a DNS server always updates current information received from either an IEN-116 server or an RWHO broadcast; information from an IEN-116 server always updates current information received from an RWHO broadcast.

IEN-116 servers are not supported in a Windows NT® environment. As a result, an Annex in a Windows NT® environment never considers host table information from an IEN-116 server.

The Annex also deletes entries. The criteria for deletion depend on the source of the entry. Each DNS response includes a time to live (TTL). When an entry reaches its full life (default=60 minutes), the DNS server is queried again. If a DNS server recognizes the name, the entry is re-entered in the host table; otherwise, it is deleted. The Annex keeps track of how often each IEN-116 host table entry is referenced. If a name server entry has not been used for 32 days, it is deleted.

The Annex expects to receive an RWHO message from a host at least every six minutes; if no message is received in that time period, the host table status entry for that host is changed to *down?*. If there is no message for 12 minutes, the status is changed to *down*, and if no message is received for 60 minutes, the entry is removed from the table.

If the host table acquires a new entry after it is full, the Annex deletes the oldest, least-used entry to make room for the new one. If the host table is too small, it frequently changes. Increasing the size of the host table using the Annex parameter **host_table_size** reduces these changes.

Other tools for managing the host table are:

- The CLI **hosts –n** and **hosts –f** commands.

  The hosts **–n** command displays name server information; **hosts –f** flushes all, or specified, entries in the host table.

- The **na** or CLI **admin** command **reset annex nameserver**.

  The **reset annex nameserver** command resets all name server parameters discussed in this section and flushes all entries from the host table.

Flushing the host table and resetting the name server does not remove down-loaded entries from the **gateway** section of the Annex configuration file.

Additionally, the **gateway** section of the configuration file permits a line entry containing a host name that is associated with an IP Address. This entry is identical to the **/etc/hosts** file entry, except aliasing is not supported. When the Annex boots, it adds this host name entry to the host table. Each entry lives in the host table until a nameserver overrides the information or until the administrator resets the Annex nameserver using the **na** or CLI **admin** commands. For more information, see *Loading the Host Table from the Configuration File* on page A-357.

Disabling Software Modules

The Annex parameter **disabled_modules** allows you to disable individual software modules to free memory space. If you enter more than one module, separate module names using commas. Valid options are **admin**, **atalk**, **dialout**, **edit**, **fingerd**, **ftpd**, **ipx, lat**, **nameserver**, **ppp**, **slip**, **snmp**, **tn3270**, **tstty**, **vci**, **all**, or **none**. The default is **vci** (disables the Annex VMS interface).

The syntax for disabling several modules is:

```
set annex disabled_modules lat,snmp,ppp,slip
```

You should exercise extreme caution when disabling modules:

- If **disabled_modules** is set to a value other than **none** and **server_capability** includes the operational image, no modules are disabled; a syslog message announces this override.
- The **vci** option disables the Annex interface for VMS environments along with the following commands: **backwards**, **change**, **clear**, **crash**, **define**, **disconnect**, **forwardlis**, **forward**, **list**, **logout**, **resume**, **set**, **show**.
- If **lat_key** is invalid and **server_capability** is set to **none**, the LAT code is freed for use by the system.
- Disabling LAT also disables the CLI commands **services**, **connect**, and **queue**.
- Disabling **admin** and **snmp** can cause problems if host-based **na** is not available. To change parameters in this case, return to monitor mode, erase the parameters in non-volatile memory, and reconfigure the Annex.

Typical Configuration Problems

Each Annex hardware platform provides a hardware installation guide that contains troubleshooting information. Many problems that occur after an Annex is running are due to improper configuration of the Annex or a host. The following subsections describe the symptoms of several common configuration problems.

Sessions not Terminated

Several situations can leave a session open.

- On CLI ports, the **hangup** command may not disconnect a modem or a switch. On CLI login ports, a modem, telephone, or switch disconnection (de-asserting DCD) may not terminate the CLI connection or UNIX session. Thus, the next port user finds a CLI connection with jobs already active and does not receive a security prompt or receives a shell prompt without logging in.
- A port configured as autobaud may retain the baud rate of the previous session.
- The port server session may not be terminated if you try to use an outgoing Annex port as a front-end to another host (or to connect to a modem or switch), and the interface at the other end drops DCD (see *Modems* on page A-99 for more information on using modems).

If any of these situations occur:

- Make sure the Annex port parameters are set correctly.
- Check the cable wiring, and pay close attention to the wiring of the Annex's DCD, DSR, and DTR control lines.

The superuser CLI **stats**, **tap**, and **control** commands provide useful information.

When changing parameters using **na** or **admin**, remember to use the **reset** command after entering the new values.

Connection Delays When Using Name Servers

Annex users may notice connection delays under certain circumstances. If **name_server_1** and **name_server_2** are defined, and **name_server_1** is down or does not exist, there will be a 15–30 second delay until **name_server_2** resolves the name during a connect to a host using **rlogin** or **telnet**. If both name servers are down or they do not exist, there will be up to a 45 second delay. If the host to which the user ID is trying to connect is in not in the RWHO host table, an error occurs; the terminal displays a message informing the user that the name server is unreachable.

Hosts not Appearing in Hosts Display

The Annex **hosts** command should list any hosts that broadcast RWHO packets if the configuration parameter **rwhod** is set to **Y**. If you expect to see a host in the **hosts** display and it does not appear, wait several minutes and then re-issue the **hosts** command before assuming there is a problem; the time between broadcasts can vary. Before proceeding, verify that the host not appearing in the **hosts** display is sending RWHO packets correctly by entering **ruptime** on another host on the network, or by checking that the host in question is running **rwhod**.

If the host is sending RWHO packets correctly, incompatible broadcast addresses may be causing the problem. Originally, a broadcast packet used a host address of all *zeros* (*network.0*). Later refinements required a change to the broadcast address, specifying a host address of all *ones* (*network.255*). A host configured with a *network.255* address will accept *network.0* broadcasts. Hosts configured with *network.0* addressing will not see *network.255* broadcasts. You can configure the Annex for either method of addressing by setting the **broadcast_addr** parameter.

Wrong Host Address in Host Table

The Annex assumes that the host described in the data part of the RWHO packet sent the packet, and the IP header's *source-Internet-address* field contains the host's address. Usually, this assumption is correct because routers do not forward broadcast packets. Some RWHO daemons do forward RWHO packets.

You can turn off RWHO at the Annex by setting the **rwho** parameter to **N**. RWHO entries are not added to the Annex's host table.

Network Logins to BSD Hosts are Invisible

An Annex user can **rlogin** or **telnet** to a host, but the pseudo-terminal does not show up in a **who** command display. This problem is caused by a mismatch between pseudo-terminals configured in the **/dev** directory and pseudo-terminal entries in **/etc/ttys**. Update the **/etc/ttys** file to contain the proper number of pseudo-terminals as indicated by the actual device entries in **/dev**.

All Network Ports are in Use

The **rlogin** or **telnet** command is rejected after the user name is entered in response to the login prompt. The error message *all network ports in use* indicates that all available pseudo-terminals are in use. On BSD hosts, update **/etc/ttys** and create more pseudo-terminals in **/dev**.

Chapter 2
Simple Network Management Protocol (SNMP)

This chapter describes the Simple Network Management Protocol (SNMP) and the SNMP agent provided by the Annex. This chapter includes the following sections:

- *SNMP Protocol Overview*
- *Setting up the Annex for SNMP*
- *Standard MIB Support*
- *Annex Parameter vs. Annex Private Enterprise MIB*

SNMP Protocol Overview

SNMP is a heavily used management protocol. It operates over the User Datagram Protocol (UDP), which is part of the TCP/IP protocol suite. SNMP provides an easier and more efficient means of managing the Annex.

- The SNMP protocol can send queries to the SNMP agents located in each Annex.
- Each SNMP agent collects information about its Annex and provides that information to the Network Management Station running the Annex. The agent process acts as a server in a typical client-server model.
- Management Information Bases (MIBs) located on the SNMP Network Management Station describe the information that comes from the agents.

SNMP Management Stations

An SNMP Network Management Station is a dedicated or shared network device that is the client in the client-server model. The management station can run an application specifically written for the Annex and its MIBs (e.g., the Xylogics graphical user interface, Annex Manager), or a generic application that communicates with other non-Xylogics devices (e.g. SunNet Manager,™ HP/OpenView, ™ NetView for AIX™). The generic application must include the definitions of the MIBs supported by the Annex.

The SNMP agent processes **get**, **set**, **get-next** commands, returns a response indicating the command's success or failure, and returns the requested data for the **get** and **get-next** commands (*SNMP Commands* on page A-46 describes these commands in greater detail).

Message Delivery

SNMP messages are encapsulated in UDP datagrams. The UDP layer does not guarantee delivery. The Annex uses a timeout and retry mechanism to guarantee the SNMP command's delivery. If a timeout occurs, the Annex does not know if the agent did not receive the command or if the agent's response was lost.

The SNMP agent can generate an unsolicited trap command and send it to one or more network addresses. Receivers of traps, i.e., trap hosts, do not respond to the SNMP agent (for more details, see *Defining Trap Hosts and Traps* on page A-44).

The Annex supports only the cold-start, link-up, and link-down traps defined in MIB-II.

Configuring the Annex for SNMP

Before an SNMP network management application can monitor or manage the Annex, you must define certain configuration data, including the SNMP agent and related Annex parameters.

Configuring the SNMP Agent

Entries in the **gateway** section of the configuration file, which is downloaded during Annex initialization, both enable the SNMP agent and define the operating characteristics of the SNMP daemon that controls the SNMP agent (for more details on creating and using the configuration file, see *Parsing the Configuration File* on page A-345).

The **gateway** section of the configuration file contains four optional keywords for configuring the Annex SNMP agent:

- **community**
- **traphost**
- **contact**
- **location**

The following subsections detail each of these keywords as well as the Annex parameters required for use with SNMP.

A sample entry in the **gateway** section of the configuration file looks like this:

```
annex 132.245.6.34
     host 132.245.1.01 gateway 132.245.7 metric 1 hardwired
     net 132.245.9.0 gateway 132.245.2.3 metric 1 hardwired
     snmp contact john smith ext 370
     snmp location computer room
end
snmp community public
snmp traphost 132.245.6.50
```

Defining the Community String

Each SNMP message contains a community string in its header. The receiving SNMP agent tries to match the message's string with an existing community string list. If there is no match, the SNMP agent discards the message without responding to the sender.

The keyword **community** defines an SNMP community name from which the Annex responds to requests. At system start-up, the SNMP agent requires at least one community string to be defined in the configuration file. If the file does not contain a community string, the Annex defaults to the community name *public* (unless SNMP is disabled in the Annex parameter **disabled_modules**). There is no notion of read-only or read-write communities.

You can specify up to four SNMP community names in the **gateway** section of the configuration file, but each community requires a separate line. The Annex adds these communities to the SNMP agent's community table. The syntax is:

snmp community *name*

Defining Trap Hosts and Traps

The Annex employs two methods for defining the host addresses it uses when generating SNMP trap messages.

- The first method defines up to ten static trap hosts in the configuration file using the SNMP trap host syntax.
- The second method loads the trap hosts (if any) from the configuration file into the Trap Host Table (i.e., the anxTrapHostTable objects in the proprietary MIB). You can modify this table by adding or deleting trap hosts. However, the changes you make directly through the table will be lost when the Annex reboots. If you want your changes to be permanent, you must use the Annex configuration file.

Traps are unsolicited administrative messages generated by SNMP agents on the network. The keyword **traphost** defines the host to which SNMP traps are sent. For the Annex to generate traps, one or more trap host addresses must be defined in the **gateway** section of the configuration file along with the SNMP community string. All generated trap messages use the first community string defined in the configuration file (if the file does not contain a community string, the Annex defaults to *public).*

You can specify up to ten static trap hosts in the configuration file, but each host requires a separate line. Specify the trap host using its IP address (RFC 1157 provides more details on communities and traps). Table B-12 describes the supported SNMP traps. The syntax is:

snmp traphost *ipaddr*

Table B-12. Supported SNMP Traps

Trap	Description
coldstart	Upon initialization of the SNMP agent at boot time.
link-up	Upon initialization of each network interface.
link-down	Upon de-configuration of any network interface.

Defining the Contact String

The keyword **contact** defines the object that identifies the person responsible for managing the Annex, as supported by MIB-II. The syntax is:

snmp contact *string*

The *string* can include information about how to contact the person; e.g., *M. Law, x 370.*

Defining the Location String

The keyword **location** defines the object that describes the Annex's location; e.g., *computer room*. The syntax is:

snmp location *string*

Defining the disabled_modules Parameter

The Annex parameter **disabled_modules** allows you to turn off certain features during Annex software initialization (e.g., enter *LAT, PPP, SLIP* to turn these features off). If you disable SNMP, the Annex will discard all SNMP messages it receives. By default, the SNMP agent on the Annex is enabled (for more details, see *disabled_modules* on page A-57).

Defining the allow_snmp_sets Parameter

The Annex's default setting for the **allow_snmp_sets** parameter does not permit parameter value changes because the SNMP **set** command's header transmits the community string in clear text, which may be a security risk. To modify parameters through SNMP, you must first set **allow_snmp_sets** to **yes** using the **na** utility or the **admin** command. You cannot set this parameter using SNMP (for more details, see *allow_snmp_sets* on page A-45).

SNMP Commands

The SNMP agent software in the Annex supports the SNMP commands **get**, **get-next**, **set**, and **trap** as defined in RFC 1157. Table B-13 describes these commands.

Table B-13. SNMP Commands Supported by the Annex

Action	Description
get	Retrieves the value of a specific object from one of the supported MIBs.
get-next	Traverses the MIB tree to retrieve the next object's management information.
set	Modifies the values of MIB objects. The Annex private enterprise MIB and several objects in the standard MIBs allow you to configure the Annex from an SNMP management station on the network rather than using the **na** utility or CLI **admin** command.
trap	Asynchronously reports significant events.

When the **allow_snmp_sets** parameter is enabled, the Annex accepts SNMP **set** commands from any source and processes them. When disabled, the Annex rejects all SNMP **set** commands; the Annex SNMP agent returns the error *no such name* for the first object in the **set** command (for more details, see *allow_snmp_sets* on page A-45).

The specifics of using the SNMP commands are management station-dependent (see your SNMP management station documentation). The MIB definitions in the files provided in the directory /*annex_root*/**src/snmp** must be compiled and included in your management station database before you can manage the Annex.

Using SNMP set to Send Commands to the Annex

The private enterprise MIB objects allow you to change the configuration of the Annex or its ports. These configuration changes do not take effect until the Annex is rebooted or the port is reset.

Using the SNMP **set** command, you can broadcast a message, reset a port or subsystem, and reboot the Annex.

- To broadcast a message, use SNMP **set** to write the message to the MIB object **anxcBcastMsg** and then **set** the broadcast type to the MIB object **anxcBcast**.
- To reset an Annex subsystem, use SNMP **set** to write the desired type (**all**, **macros**, **motd**, **nameserver**, **security**) to the MIB object **anxcReset**.
- To reset all printer, serial, or virtual ports on the Annex as a group, use SNMP **set** to write the desired value to the MIB object **anxcReset**.
- To reset a single serial port, use SNMP **set** to write the appropriate value to the character MIB object **charPortReset** (defined in RFC 1316) that corresponds to the serial port to reset.
- To reboot the Annex, **set** the desired image name to the MIB object **anxcBootImage** and **set** any boot warning message to the MIB object **anxcBootMsg**. For a delayed boot, **set** the boot time to the MIB object **anxcBootTime**. Then **set** the boot type to the MIB object **anxcBoot.**

To change the Annex's configuration using **set**, SNMP must be enabled at boot time. Make sure the argument **snmp** is not disabled in the **disabled_modules** parameter. For more details, see *disabled_modules* on page A-57).

You cannot configure filters through SNMP. For details on filtering, see *Enabling Filtering* on page A-251.

Standard MIB Support

The Annex supports the following standard MIBs:

- MIB-II (defined in RFC1213).
- Character MIBs (defined in RFCs 1316, (Character MIB), 1317 (RS232-like MIB), and 1318 (Parallel Printer MIB).
- Ethernet MIB (defined in RFC 1398).
- RIPv2 MIB (defined in RFC 1389).
- AppleTalk MIB (defined in RFC 1243).

Some standard MIB objects are not valid or meaningful for the Annex. For detailed information, see *Annex Restrictions on Standard MIBs* on page A-51.

Most Annex parameters do not map to standard MIB objects. Instead, they map to MIB objects in a proprietary (or private enterprise) MIB specific to the Annex. The private MIB also contains objects that provide status and statistics information to the network manager (see *Annex Parameters vs. Annex Private Enterprise MIB* on page A-58).

This section explains the relationship between the Annex and standard MIBs, listing the exceptions and restrictions placed on standard MIBs by the Annex SNMP agent. This section includes:

- *MIB Object Hierarchy*
- *Describing and Naming Objects*
- *Annex Restrictions on Standard MIBs*

MIB Object Hierarchy

MIBs define the hierarchy of managed objects. MIB objects represent data that the Annex can retrieve or configuration information that it can modify.

Describing and Naming Objects

RFC 1155 (*Structure and Identification of Management Information for TCP/IP-based internets*) describes the layout and encoding of exchanged data objects.

The SMI (Structure of Management Information) uses the ISO standard ASN.1 (Abstract Syntax Notation One) to define a method for describing a hierarchical name space for managed information. Each object has:

- A name (also referred to as an Object Identifier (OID)).
- A syntax and an encoding. In addition to the basic integer and octet string data types, several special types are defined (e.g, *IP Address*, *Network Address*, *Counter*, *Gauge*, *TimeTicks)*. RFC 1212 (Concise MIB Definitions) is an easier-to-read form used in most standard MIBs today. It is used to define the Annex private enterprise MIB.

Annex Restrictions on Standard MIBs

The Annex SNMP Agent does not use all objects in the supported standard MIBs. This section lists the supported standard MIBs and outlines the differences between the Annex parameters and specific standard MIB objects. Table B-14 lists the supported standard MIBs.

Table B-14. Standard MIBs Supported by the Annex

MIB Name	RFC Number
MIB-II	RFC 1213
Character MIB	RFC 1316
RS-232 MIB	RFC 1317
Parallel Printer MIB	RFC 1318
AppleTalk MIB	RFC 1243
RIP version 2 MIB	RFC 1389
Ethernet MIB	RFC 1398

RFC 1213 MIB-II Restrictions

The Annex supports RFC1213's *system*, *interfaces*, *at*, *ip*, *icmp*, *tcp*, *udp*, and *snmp* groups. It does not support the *egp* group. In addition, some individual objects have the restrictions outlined in Table B-15.

Table B-15. RFC 1213 MIB-II Objects

Object Name	get/set Restrictions	Read Object Limitations
ifAdminStatus	read only	Returns only *up (1)* and *down (2)*
ifOperStatus	none	Returns only *up (1)* and *down (2)*
atEntry	Cannot create new rows	none
ipRouteEntry	Cannot create new rows	none
ipRouteProto	none	Returns only *local (2)*, *icmp (4)*, and *rip (8)*
ipRouteType	none	Returns only *invalid (2)*, *direct (3)*, *indirect (4)*
ipNetToMediaEntry	Cannot create new rows	none
ipNetToMediaType	Writes only *invalid (2)*, *dynamic (3)*, and *static (4)*	Returns only *dynamic(3)* and *static(4)*

RFC 1243 AppleTalk MIB Restrictions

The Annex does not support the *llap*, *rtmp*, *kip*, *zip*, and *nbp* groups. It supports the *aarp*, *atport*, *ddp*, and *atecho* groups with the restrictions listed in Table B-16.

Table B-16. RFC 1243 AppleTalk

Object Name	Restrictions	Read Object Limitations
atportType	Read only	None
atportNetStart	Not supported	None
atportNetEnd	Not supported	None
atportNetAddress	Not supported	None
atportStatus	Read only	None
atportZone	Read only	None
atportIfIndex	Read only	None
ddpOutRequests	Not supported	None
ddpInLocalDatagrams	Not supported	None
ddpNoProtocolHandlers	Not supported	None
ddpBroadcastErrors	Not supported	None
ddpShortDDPErrors	Not supported	None
ddpHopCountErrors	Not supported	None

RFC 1389 RIPv2 MIB Restrictions

The Annex supports *rip2GlobalGroup*, *rip2IfStatTable*, and *rip2IfConfTable*. It does not support *rip2PeerTable*. Table B-17 describes additional restrictions.

Table B-17. RFC 1389 RIPv2 MIB Objects

Object Name	Restrictions	Read Object Limitations
rip2IfStatStatus	Read only	None
rip2IfConfDomain	Not supported	None
RipIfConfAuthKey	Not supported	None
ripIfConfStatus	Read only	None

RFC 1398 Ethernet MIB Restrictions

The Annex supports RFC 1398's *dot3StatsTable* and *dot3CollTable* with the restrictions outlined in Table B-18.

Table B-18. RFC 1398 Ethernet MIB Objects

Object Name	Restrictions	Read Object Limitations
dot3StatsSQETestErrors	Not supported	None
dot3StatsInternalMac ReceiveErrors	Not supported	None

RFC 1316 Character MIB Restrictions

The Annex supports the *char* group with the restrictions outlined in Table B-19.

Table B-19. RFC 1316 Character MIB Objects

Object Name	Restrictions	Read Object Limitations
charPortAdminStatus	Read only	Returns only *enabled (1), disabled (2), off (3)*
charPortOperStatus	None	Returns only *up (1), down (2), active (5)*
charPortInFlowType	Supports only *none (1), xonXoff (2),* and *hardware (3)*	None
charPortOutFlowType	Supports only *none (1), xonXoff (2),* and *hardware (3)*	None
charPortAdminOrigin	Read only	None
charPortName	Read only	None
charPortSessionMaximum	Maximum value is *16*	None
charSessKill	Read only	None
charSessState	None	Returns only *connected (2)*
charSessConnectionId	None	Returns only null
charPort objects for virtual ports	Read only, read-write objects apply only to physical ports	None

RFC 1317 RS-232 MIB Restrictions

The Annex supports this MIB with the restrictions described in Table B-20.

Table B-20. RFC 1317 RS-232 MIB Objects

Object Name	Restrictions	Read Object Limitations
rs232PortInSpeed rs232PortOutSpeed	Setting one sets both. Can set only to a supported value (see *Setting Port Speed* on page A-57).	None
rs232AsyncPortParity	*none (1)*, *mark (4)*, or *space (5)* all map to *none (1)*	Returns only *none (1)*, *odd (2)*, or *even (3)*
rs232AsyncPortStopBits	*dynamic (4)* maps to *one (1)*	Returns only *one (1)*, *two (2)*, or *one-and-half (3)*
rs232SyncPortTable	Not supported	

Setting Port Speed

The rs232PortInSpeed, rs232PortOutSpeed, and rs232AsyncPortAutobaud are related for the Annex's asynchronous ports.

- To set the port to autobaud, you must first set rs232AsyncPortAutobaud to *enabled (1)* and then set either rs232PortInSpeed or rs232PortOutSpeed to *zero*. Setting the port speed to zero when rs232AsyncPortAutobaud is disabled results in a bad value error.
- To disable autobaud, you must first set the port speed to a non-zero value, and then set rs232AsyncPortAutobaud to *disabled(2)*. This ensures that the port is not left in a state without a declared speed.

For example, you can set a port to 9600/autobaud by setting *9600* in rs232PortInSpeed or rs232PortOutSpeed and then setting rs232AsyncPortAutobaud to *enabled(1)*.

RFC 1318 Parallel Printer MIB Restrictions

The Annex supports:

- *paraNumber*.
- *paraPortTable*.
- *paraInSigTable*.

The Annex does not support:

- *paraOutSigTable*.

Annex Parameters vs. Annex Private Enterprise MIB

The private enterprise MIB file provides the object descriptions for the hardware, software, ports, parameters, and commands groups. The Annex software distribution provides this information in the file */annex_root/* **src/snmp/anx-rx.x**. Copies of the standard MIBs reside in the */ annex_root*/**src/snmp** directory.

Most of the configuration parameters are provided as objects with read-write access permission in the Annex private enterprise MIB. A number of these parameters can be found in the standard MIBs that the Annex SNMP agent supports.

Most MIB object names for the Annex parameters in the private enterprise MIB are preceded by the string "*.iso.org.dod.internet.private.enterprises.xylogics.annex.*".

One of the many exceptions is the object corresponding to the **image_name** parameter. The MIB object name for **image_name** is preceded by the string "*.iso.org.dod.internet.private.enterprises.xylogics.annex.annexcmds.*".

MIB Prefixes

All MIB object names have a prefix that indicates the MIB in which it is defined; Table B-21 lists these prefixes and their corresponding MIB. Table B-22 lists the configuration parameters and the corresponding MIB object names.

There are other settable MIB objects included in the standard MIBs supported by the Annex SNMP agent. The read-only objects defined in the various MIBs allow the SNMP management station to monitor many Annex variables.

Table B-21. Prefixes for MIB Object Names

Prefix	Corresponding MIB
anx	Annex private enterprise MIB
rs232	rs232 MIB
char	character MIB

Configuration Parameters vs. MIB Objects

Table B-22 lists the Annex configuration parameter and the corresponding MIB Object.

Table B-22. Configuration Parameter vs. MIB Object Name

Configuration Parameter	MIB Object
acp_key	anxAcpKey
allow_snmp_sets	** not applicable **
a_router	anxAppleTalkRouter
authoritative_agent	anxAuthAgent
broadcast_addr	anxBcastAddr
cli_prompt	anxCliPrompt
config_file	anxConfigFile
daylight_savings	anxDaylightSavings
default_zone_list	anxAppleTalkDefZones
disabled_modules	anxDisabledModules
enable_security	anxEnableSecurity
host_table_size	anxHostTableSize

(continued on next page)

Table B-22. Configuration Parameter vs. MIB Object Name (continued)

Configuration Parameter	MIB Object
image_name	anxDefaultImageName
inet_addr	anxInetAddr
ipencap_type	anxIpEncapType
ip_forward_broadcast	anxIpFwdBcast
lat_key	anxLatKey
load_broadcast	anxLoadBcast
load_dump_gateway	anxLoadDumpGateway
load_dump_sequence	anxLoadDumpSeq
loose_source_route	anxLooseSrcRoute
max_vcli	anxMaxVcli
min_unique_hostnames	anxMinUniqueHostNames
motd_file	anxMotdFile
name_server_1	anxNameServer1Type
name_server_2	anxNameServer2Type
nameserver_broadcast	anxNameServerBcast
network_turnaround	anxNetTurnAround
node_id	anxAppleTalkNodeId
option_key	anxOptionKey
password	anxPassword
pref_dump_addr	anxPrefDumpAddr
pref_load_addr	anxPrefLoadAddr

(continued on next page)

Table B-22. Configuration Parameter vs. MIB Object Name (continued)

Configuration Parameter	MIB Object
pref_name1_addr	anxNameServer1Addr
pref_name2_addr	anxNameServer2Addr
pref_secure1_host	anxSecurServer1Addr
pref_secure2_host	anxSecurServer2Addr
rip_auth	anxRipAuth
rip_routers	anxRipRouteList
routed	anxRouted
rwhod	anxRwhod
security_broadcast	anxSecurBcast
server_capability	anxServerCap
subnet_mask	anxSubnetMask
syslog_facility	anxSysLogFacility
syslog_host	anxSysLogHost
syslog_mask	anxSysLogMask
syslog_port	anxSysLogPort
tcp_keepalive	anxTcpKeepAlive
tftp_dump_name	anxTftpDumpName
tfpt_load_dir	anxTftpDirName
time_broadcast	anxTimeBcast
timezone_minuteswest	anxTimeZone
vcli_password	anxVcliPassword
vcli_security	anxVcliSecurity
zone	anxAppleTalkZone

LAT-specific Configuration Parameters vs. MIB Objects

Table B-23 lists the LAT-specific configuration parameters and the corresponding MIB object names. The string "*.iso.org.dod.internet.private.enterprises.xylogics.annex.*" precedes the MIB object names.

Table B-23. LAT-specific Configuration Parameters vs. MIB Object Name

LAT-specific na Parameter	MIB Object
circuit_timer	anxCircuitTimer
facility_num	anxFacilityNum
group_value	anxLatGroupVal
keep_alive_timer	anxKeepAliveTimer
lat_queue_max	anxLatQueueMax
retrans_limit	anxReXmitLimit
server_name	anxServerName
service_limit	anxServiceLimit
sys_location	anxLatLocation
vcli_groups	anxLatVcliGroupVal

LAT Statistic Objects

Table B-24 lists the LAT statistic objects; these objects provide the same information available in the CLI **netstat** command. The following string precedes the MIB object names: "*.iso.org.dod.internet.private.enterprises.xylogics.annex.*".

Table B-24. LAT Statistic Objects

MIB Object Name	Description
anxLatRecvRunMsgs	total received run messages
anxLatXmitRunMsgs	total transmitted run messages
anxLatRecvSlots	total received slots
anxLatXmitSlots	total transmitted slots
anxLatRecvBytes	total received bytes
anxLatXmitBytes	total transmitted bytes
anxLatDupMsgs	total duplicate messages
anxLatRexmitMsgs	total retransmitted messages
anxLatBadCircuitMsgs	total bad circuit messages
anxLatBadSlotMsgs	total bad circuit slots
anxLatAcceptHostInits	total accepted host-initiates
anxLatRejectHostInits	total rejected host-initiates
anxLatMultipleNodes	total multiple nodes seen
anxLatCreatedCircuits	total created circuits
anxLatCreatedSessions	total created sessions

(continued on next page)

Table B-24. LAT Statistic Objects (continued)

MIB Object Name	Description
anxLatRecvFrames	total received frames
anxLatXmitFrames	total transmitted frames
anxLatIllegalFrames	total illegal frames
anxLatCircuitTimeouts	total circuit time-outs
anxLatXmitSvcMsgs	total transmitted service messages
anxLatRecvSvcMsgs	total received service messages
anxLatUsedSvcMsgs	total used service messages

TMux-specific Annex Parameters vs. MIB Objects

Table B-25 lists the TMux-specific Annex parameters and their corresponding MIB object names.

Remote Annex Server Tools For Windows NT® does not support TMux.

The following string precedes the MIB object names: "*.iso.org.dod.internet.private.enterprises.xylogics.annex.*".

Table B-25. TMux- specific Parameters vs. MIB Objects

TMux Parameter	MIB Object Name
tmux_delay	anxTmuxDelay
tmux_enable	anxTmuxEnable
tmux_max_host	anxTmuxMaxHost
tmux_max_mpx	anxTmuxMaxMpx

IPX-specific Annex Parameters vs. MIB Objects

Table B-26 lists the IPX-specific Annex parameters and their corresponding MIB object names.

The following string precedes the MIB object names: "*.iso.org.dod.internet.private.enterprises.xylogics.annex.*".

Table B-26. IPX-specific Parameters vs. MIB Objects

IPX Parameter	MIB Object Name
ipx_do_checksum	anxIpxDoChecksum
ipx_dump_password	anxIpxDumpPasswd
ipx_dump_path	anxIpxDumpPath
ipx_dump_username	anxIpxDumpUsername
ipx_file_server	anxIpxFileServer
ipx_frame_type	anxIpxFrameType

T1-specific Annex Parameters vs. MIB Objects

Table B-27 lists the T1-specific Annex parameters and their corresponding MIB object names.

The following string precedes the MIB object names: "*.iso.org.dod.internet.private.enterprises.xylogics.annex.*".

Table B-27. T1- specific Parameters vs. MIB Objects

T1 Parameter	MIB Object Name
alarmsyslog	anxt1AlarmSyslog
bypass	anxt1EngineBypass
map	anxt1ChanMap
ring	anxt1ChanRing
sigproto	anxt1ChanSigProto
t1_info	anxt1Info
tdi_distance	anxt1DiiDistance
tdi_framing	anxt1DiiFraming
tdi_line_code	anxt1DiiLineCode
tni_line_buildout	anxt1Ds1LineBuildout
tni_ones_density	anxt1Ds1OnesDensity

Interface Parameters vs. MIB Objects

Table B-28 lists the interface parameters and the corresponding MIB object names. The string "*.iso.org.dod.internet.mgmt.mib-2.interfaces.*" precedes the MIB object names.

Table B-28. Interface Parameters vs. MIB Objects

Interface Parameter	MIB Object
rip_accept	interfaceRipAccept
rip_advertise	interfaceRipAdvertise
rip_default_route	interfaceRipDefRoute
rip_horizon	interfaceRipHorizon
rip_recv_version	interfaceRipRecvVersion
rip_send_version	interfaceRipSendVersion
rip_sub_accept	interfaceRipSubAccept
rip_sub_advertise	interfaceRipSubAdvertise

Asynchronous Port Parameters vs. MIB Objects

Table B-29 lists the asynchronous port parameters corresponding to the MIB object names. Table B-30 lists the PPP and SLIP port parameters and the corresponding MIB object names. Table B-25 lists the T1-specific parameters and the corresponding MIB object names.

- All asynchronous port private MIB object names are preceded by the string "*.iso.org.dod.internet.private .enterprises.xylogics.annex.ports.portTable.portEntry.*" and appended by the port instance number.
- The following string precedes the object names that are in the rs232 MIB:
  "*.iso.org.dod.internet.mgmt.mib-2.transmission.rs232.*"
- The string "*.iso.org.dod.internet.mgmt.mib-2.char.*" precedes the MIB object names that are in the charlikeMIB.

Table B-29. Asynchronous Port Parameters vs. MIB Object Names

Async Port Parameter	MIB Object
allow_broadcast	anxpAllowBcast
arap_v42bis	anxpArapV42bis
at_guest	anxpAtGuest
at_nodeid	anxpAtNodeid
at_security	anxpAtSecurity
attn_string	anxpAttnChar
authorized_groups	anxpLatAuthGroupVal
backward_key	anxpBackwardKey
banner	anxpBanner
broadcast_direction	anxpBcastDirection
char_erase	anxpCharErase
cli_imask7	anxpCliImask7
cli_inactivity	anxpCliInactivity
cli_security	anxpCliSecurity
connect_security	anxpConnectSecurity
control_lines	anxpCtrlLines
data_bits	rs232AsyncPortTable.rs232AsyncPortEntry.rs232 AsyncPortBits
forward_key	anxpForwardKey
line_erase	anxpLineErase

(continued on next page)

Table B-29. Asynchronous Port Parameters vs. MIB Object Names (continued)

Async Port Parameter	MIB Object
location	anxpLocation
long_break	anxpLongBreak
dedicated_address	anxpDedicatedAddr
dedicated_arguments	anxpDedicatedArgs
dedicated_port	anxpDedicatedPort
echo	anxpEcho
erase_char	anxpEraseChar
erase_line	anxpEraseLine
erase_word	anxpEraseWord
forwarding_count	anxpForwardCount
forwarding_timer	anxpForwardTimer
hardware_tabs	anxpHardwareTabs
imask_7bits	anxpImask7Bits
inactivity_timer	anxpInactivityTimer
input_flow_control	charPortTable.charPortEntry.charPortInFlowType
input_is_activity	anxpInputIsActivity
input_start_char	anxpInputStartChar
input_stop_char	anxpInputStopChar
ixany_flow_control	anxpIxanyFlowCtl

(continued on next page)

Table B-29. Asynchronous Port Parameters vs. MIB Objects (continued)

Async Port Parameter	MIB Object
latb_enable	anxpLatbEnable
map_to_lower	anxpMapToLower
map_to_upper	anxpMapToUpper
max_session_count	charPortTable.charPortEntry. charPortSessionMaximum
mode	anxpMode
modem_var	anxpModemVar
need_dsr	anxpNeedDsr
newline_terminal	anxpNewLineTerm
net_inactivity	anxpNetInactivity
net_inactivity_units	anxpNetInactivityUnits
output_flow_control	charPortTable.charPortEntry. charPortOutFlowType
output_is_activity	anxpOutputIsActivity
output_start_char	anxpOutputStartChar
output_stop_char	anxpOutputStopChar
parity	rs232AsyncPortTable.rs232AsyncPortEntry. rs232AsyncPortParity
phone_number	anxpPhoneNumber
port_password	anxpPortPassword
port_server_security	anxpPortServerSecurity

(continued on next page)

Table B-29. Asynchronous Port Parameters vs. MIB Objects (continued)

Async Port Parameter	MIB Object
ppp_ipx_network	anxpPppIpxNetwork
ppp_ipx_node	anxpPppIpxNode
prompt	anxpPrompt
ps_history_buffer	anxpPsHistory
redisplay_line	anxpRedisplayLine
reset_idle_time_on	anxpResetIdleTimer
short_break	anxpShortBreak
tcp_keepalive	anxpTcpKeepAlive
speed	rs232PortTable.rs232PortEntry.rs232PortInSpeed rs232PortTable.rs232PortEntry.rs232PortOutSpeed
stop_bits	rs232AsyncPortTable.rs232AsyncPortEntry. rs232AsyncPortStopBits
tcp_keepalive	anxpTcpKeepAlive
telnet_crlf	anxpTelnetCRLF
telnet_escape	anxpTelnetEscape
term_var	anxpTermVar
tn3270_printer_host	anxpTn3270PrinterHost
tn3270_printer_name	anxpTn3270PrinterName
toggle_output	anxpToggleOutput
type	anxpType
user_name	anxpUserName

Table B-30. PPP and SLIP Port Parameters vs. MIB Objects

PPP/SLIP Port Parameter	MIB Object
allow_compression	anxpAllowCompression
dialup_addresses	anxpPppDialupAddr
do_compression	anxpDoCompression
local_address	anxpNetLocalAddr
metric	anxpNetMetric
ppp_acm	anxpPppAcm
ppp_mru	anxpPppMru
ppp_ncp	anxpPppNcp
ppp_password_remote	anxpPppPasswdRemote
ppp_security_protocol	anxpPppSecurityProto
ppp_username_remote	anxpPppUserRemote
remote_address	anxpNetRemoteAddr
slip_allow_dump	anxpSlipAllowDump
slip_load_dump_host	anxpSlipLoadDumpHost
slip_mtu_size	anxpSlipMtuSize
slip_no_icmp	anxpSlipNoIcmp
slip_ppp_security	anxpSlipSecure
slip_tos	anxpSlipTos
subnet_mask	anxpSlipSubnetMask

Numerics

A

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

W