Remote Access Concentrator Software Reference

Marketing Version Number 5.1

Part No. 118360-A Rev. A
September 1997

Bay Networks

Bay Networks

4401 Great America Parkway
Santa Clara, CA 95054

8 Federal Street
Billerica, MA 01821

Copyright © 1997 Bay Networks, Inc.

The information in this document is subject to change without notice. The statements, configurations, technical data, and recommendations in this document are believed to be accurate and reliable, but are presented without express or implied warranty. Users must take full responsibility for their applications of any products specified in this document. The information in this document is proprietary to Bay Networks, Inc.

The software described in this document is furnished under a license agreement and may only be used in accordance with the terms of that license. A summary of the Software License is included in this document.

Trademarks

Bay Networks is a registered trademark and Remote Annex, BayStack, Quick2Config, System 5000, Bay Networks Press, and the Bay Networks logo are trademarks of Bay Networks, Inc.

Microsoft, MS, MS-DOS, Win32, Windows, and Windows NT are registered trademarks of Microsoft Corporation.

All other trademarks and registered trademarks are the property of their respective owners.

Restricted Rights Legend

Use, duplication, or disclosure by the United States Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Notwithstanding any other license agreement that may pertain to, or accompany the delivery of, this computer software, the rights of the United States Government regarding its use, reproduction, and disclosure are as set forth in the Commercial Computer Software-Restricted Rights clause at FAR 52.227-19.

Statement of Conditions

In the interest of improving internal design, operational function, and/or reliability, Bay Networks, Inc. reserves the right to make changes to the products described in this document without notice.

Bay Networks, Inc. does not assume any liability that may occur due to the use or application of the product(s) or circuit layout(s) described herein.

Portions of the code in this software product are Copyright © 1988, Regents of the University of California. All rights reserved. Redistribution and use in source and binary forms of such portions are permitted, provided that the above copyright notice and this paragraph are duplicated in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that such portions of the software were developed by the University of California, Berkeley. The name of the University may not be used to endorse or promote products derived from such portions of the software without specific prior written permission.

Bay Networks, Inc. Software License Agreement

NOTICE: Please carefully read this license agreement before copying or using the accompanying software or installing the hardware unit with pre-enabled software (each of which is referred to as “Software” in this Agreement). BY COPYING OR USING THE SOFTWARE, YOU ACCEPT ALL OF THE TERMS AND CONDITIONS OF THIS LICENSE AGREEMENT. THE TERMS EXPRESSED IN THIS AGREEMENT ARE THE ONLY TERMS UNDER WHICH BAY NETWORKS WILL PERMIT YOU TO USE THE SOFTWARE. If you do not accept these terms and conditions, return the product, unused and in the original shipping container, within 30 days of purchase to obtain a credit for the full purchase price

1. License Grant. Bay Networks, Inc. (“Bay Networks”) grants the end user of the Software (“Licensee”) a personal, nonexclusive, nontransferable license: a) to use the Software either on a single computer or, if applicable, on a single authorized device identified by host ID, for which it was originally acquired; b) to copy the Software solely for backup purposes in support of authorized use of the Software; and c) to use and copy the associated user manual solely in support of authorized use of the Software by Licensee. This license applies to the Software only and does not extend to Bay Networks Agent software or other Bay Networks software products. Bay Networks Agent software or other Bay Networks software products are licensed for use under the terms of the applicable Bay Networks, Inc. Software License Agreement that accompanies such software and upon payment by the end user of the applicable license fees for such software.

2. Restrictions on use; reservation of rights. The Software and user manuals are protected under copyright laws. Bay Networks and/or its licensors retain all title and ownership in both the Software and user manuals, including any revisions made by Bay Networks or its licensors. The copyright notice must be reproduced and included with any copy of any portion of the Software or user manuals. Licensee may not modify, translate, decompile, disassemble, use for any competitive analysis, reverse engineer, distribute, or create derivative works from the Software or user manuals or any copy, in whole or in part. Except as expressly provided in this Agreement, Licensee may not copy or transfer the Software or user manuals, in whole or in part. The Software and user manuals embody Bay Networks’ and its licensors’ confidential and proprietary intellectual property. Licensee shall not sublicense, assign, or otherwise disclose to any third party the Software, or any information about the operation, design, performance, or implementation of the Software and user manuals that is confidential to Bay Networks and its licensors; however, Licensee may grant permission to its consultants, subcontractors, and agents to use the Software at Licensee’s facility, provided they have agreed to use the Software only in accordance with the terms of this license.

3. Limited warranty. Bay Networks warrants each item of Software, as delivered by Bay Networks and properly installed and operated on Bay Networks hardware or other equipment it is originally licensed for, to function substantially as described in its accompanying user manual during its warranty period, which begins on the date Software is first shipped to Licensee. If any item of Software fails to so function during its warranty period, as the sole remedy Bay Networks will at its discretion provide a suitable fix, patch, or workaround for the problem that may be included in a future Software release. Bay Networks further warrants to Licensee that the media on which the Software is provided will be free from defects in materials and workmanship under normal use for a period of 90 days from the date Software is first shipped to Licensee. Bay Networks will replace defective media at no charge if it is returned to Bay Networks during the warranty period along with proof of the date of shipment. This warranty does not apply if the media has been damaged as a result of accident, misuse, or abuse. The Licensee assumes all responsibility for selection of the Software to achieve Licensee’s intended results and for the installation, use, and results obtained from the Software. Bay Networks does not warrant a) that the functions contained in the software will meet the Licensee’s requirements, b) that the Software will operate in the hardware or software combinations that the Licensee may select, c) that the operation of the Software will be uninterrupted or error free, or d) that all defects in the operation of the Software will be corrected. Bay Networks is not obligated to remedy any Software defect that cannot be reproduced with the latest Software release. These warranties do not apply to the Software if it has been (i) altered, except by Bay Networks or in accordance with its instructions; (ii) used in conjunction with another vendor’s product, resulting in the defect; or (iii) damaged by improper environment, abuse, misuse, accident, or negligence. THE FOREGOING WARRANTIES AND LIMITATIONS ARE EXCLUSIVE REMEDIES AND ARE IN LIEU OF ALL OTHER WARRANTIES EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Licensee is responsible for the security of its own data and information and for maintaining adequate procedures apart from the Software to reconstruct lost or altered files, data, or programs.

4. Limitation of liability. IN NO EVENT WILL BAY NETWORKS OR ITS LICENSORS BE LIABLE FOR ANY COST OF SUBSTITUTE PROCUREMENT; SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES; OR ANY DAMAGES RESULTING FROM INACCURATE OR LOST DATA OR LOSS OF USE OR PROFITS ARISING OUT OF OR IN CONNECTION WITH THE PERFORMANCE OF THE SOFTWARE, EVEN IF BAY NETWORKS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL THE LIABILITY OF BAY NETWORKS RELATING TO THE SOFTWARE OR THIS AGREEMENT EXCEED THE PRICE PAID TO BAY NETWORKS FOR THE SOFTWARE LICENSE.

5. Government Licensees. This provision applies to all Software and documentation acquired directly or indirectly by or on behalf of the United States Government. The Software and documentation are commercial products, licensed on the open market at market prices, and were developed entirely at private expense and without the use of any U.S. Government funds. The license to the U.S. Government is granted only with restricted rights, and use, duplication, or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c)(1) of the Commercial Computer Software—Restricted Rights clause of FAR 52.227-19 and the limitations set out in this license for civilian agencies, and subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause of DFARS 252.227-7013, for agencies of the Department of Defense or their successors, whichever is applicable.

6. Use of Software in the European Community. This provision applies to all Software acquired for use within the European Community. If Licensee uses the Software within a country in the European Community, the Software Directive enacted by the Council of European Communities Directive dated 14 May, 1991, will apply to the examination of the Software to facilitate interoperability.

Licensee agrees to notify Bay Networks of any such intended examination of the Software and may procure support and assistance from Bay Networks.

7. Term and termination. This license is effective until terminated; however, all of the restrictions with respect to Bay Networks' copyright in the Software and user manuals will cease being effective at the date of expiration of the Bay Networks copyright; those restrictions relating to use and disclosure of Bay Networks' confidential information shall continue in effect. Licensee may terminate this license at any time. The license will automatically terminate if Licensee fails to comply with any of the terms and conditions of the license. Upon termination for any reason, Licensee will immediately destroy or return to Bay Networks the Software, user manuals, and all copies. Bay Networks is not liable to Licensee for damages in any form solely by reason of the termination of this license.

8. Export and Re-export. Licensee agrees not to export, directly or indirectly, the Software or related technical data or information without first obtaining any required export licenses or other governmental approvals. Without limiting the foregoing, Licensee, on behalf of itself and its subsidiaries and affiliates, agrees that it will not, without first obtaining all export licenses and approvals required by the U.S. Government: (i) export, re-export, transfer, or divert any such Software or technical data, or any direct product thereof, to any country to which such exports or re-exports are restricted or embargoed under United States export control laws and regulations, or to any national or resident of such restricted or embargoed countries; or (ii) provide the Software or related technical data or information to any military end user or for any military end use, including the design, development, or production of any chemical, nuclear, or biological weapons.

9. General. If any provision of this Agreement is held to be invalid or unenforceable by a court of competent jurisdiction, the remainder of the provisions of this Agreement shall remain in full force and effect. This Agreement will be governed by the laws of the state of California.

Should you have any questions concerning this Agreement, contact Bay Networks, Inc., 4401 Great America Parkway, P.O. Box 58185, Santa Clara, California 95054-8185.

LICENSEE ACKNOWLEDGES THAT LICENSEE HAS READ THIS AGREEMENT, UNDERSTANDS IT, AND AGREES TO BE BOUND BY ITS TERMS AND CONDITIONS. LICENSEE FURTHER AGREES THAT THIS AGREEMENT IS THE ENTIRE AND EXCLUSIVE AGREEMENT BETWEEN BAY NETWORKS AND LICENSEE, WHICH SUPERSEDES ALL PRIOR ORAL AND WRITTEN AGREEMENTS AND COMMUNICATIONS BETWEEN THE PARTIES PERTAINING TO THE SUBJECT MATTER OF THIS AGREEMENT. NO DIFFERENT OR ADDITIONAL TERMS WILL BE ENFORCEABLE AGAINST BAY NETWORKS UNLESS BAY NETWORKS GIVES ITS EXPRESS WRITTEN CONSENT, INCLUDING AN EXPRESS WAIVER OF THE TERMS OF THIS AGREEMENT.

Revision Level History

Revision	Description
A	Initial Release.

About This Guide

If you are responsible for configuring and/or managing a Model 8000 or Model 5399 Remote Access Concentrator (RAC), you need to read this guide.

If you want to	Go to
Use the RAC Command Line Interface (CLI) commands to connect to hosts, manage jobs, display and modify parameters (via the **admin** command), and display network information	Chapter 1
Display and modify parameters using the host-based **na** utility (which is more comprehensive than the **admin** command)	Chapter 2
Use the RAC configuration parameters	Chapter 3
Obtain information about the expedited remote procedure (**erpcd**), which accesses host files, dumps RAC images, and implements the RAC host-based Access Control Protocol used for security	Chapter 4

Before You Begin

Before using this document, you must complete the following procedures. For a new RAC:

- Order your WAN (PRI or CAS) lines from your telco. Order information is provided in the Bay Networks publication *Provisioning WAN Lines for Remote Access Concentrators*.
- When ordering your WAN lines, keep a record of the service options the telco provides you with, so that you can set WAN parameters on the RAC accordingly.

- Install the RAC and boot it, as described in the appropriate hardware installation manual (for example, for the Model 8000 RAC, this is the Bay Networks publication *Installing the Model 8000 Remote Access Concentrator*).
- Do not physically connect cables to the WAN interface ports; wait until you have at least performed a minimal configuration. An alarm from an improperly configured interface could cause the telco to drop the line.

Conventions

`special type`	In examples, special type indicates system output.
`special type`	Bold special type indicates user input.
<cr>	In command examples, this notation indicates that pressing the **Return** key enters the default value.
lowercase bold	Lowercase bold indicates commands, pathnames, or filenames that must be entered as displayed.
lowercase italics	In the context of commands and command syntax, lowercase italics indicate variables for which the user supplies a value.
[ ]	In command dialogue, square brackets indicate default values. Pressing the **Return** key selects this value. Square brackets appearing in command syntax indicate optional arguments.
{ }	In command syntax, braces indicate that one, and only one, of the enclosed values *must* be entered.

|

In command syntax, a vertical line (|) separates the different options available for a parameter.

CTRL-*X*

This notation indicates a two-character sequence for control characters. To enter the control character, hold down the **Control** key (often labeled CTRL) and press the character specified by *X*.

Notes provide important information.

Warnings inform you about conditions that can have adverse effects on processing.

Cautions notify you about dangerous conditions.

Acronyms

ACP	Access Control Protocol
AUI	Attachment Unit Interface
BFS	Block File Server
BootP	Bootstrap Protocol
BRI	Basic Rate Interface
CAS	Channel Associated Signalling
CCITT	International Telegraph and Telephone Consultative Committee (now ITU-T)
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DLCMI	Data Link Control Management Interface
erpcd	expedited remote procedure call daemon
FTP	File Transfer Protocol
GUI	Graphical User Interface
IP	Internet Protocol

ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ITU-T	International Telecommunications Union–Telecommunications (formerly CCITT)
LAN	local area network
MAC	media access control
MAU	media access unit
MMP	Multisystem Multilink PPP
MP	Multilink PPP
MDI-X	media-dependent interface with crossover
NBMA	nonbroadcast multi-access
OSI	Open Systems Interconnection
PPP	Point-to-Point Protocol
PRI	Primary Rate ISDN
RIP	Routing Information Protocol
RAC	Bay Networks Remote Access Concentrator
RADIUS	Remote Authentication Dial In User Service
SMDS	Switched Multimegabit Data Service
SNMP	Simple Network Management Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
Telnet	Telecommunication Network
TFTP	Trivial File Transfer Protocol
TPE	twisted-pair Ethernet
UDP	User Datagram Protocol
WAN	wide area network

Ordering Bay Networks Publications

To purchase additional copies of this document or other Bay Networks publications, order by part number from Bay Networks Press™ at the following numbers:

- Phone--U.S./Canada: 888-422-9773
- Phone--International: 510-490-4752
- FAX--U.S./Canada and International: 510-498-2609

The Bay Networks Press catalog is available on the World Wide Web at *support.baynetworks.com/Library/GenMisc*. Bay Networks publications are available on the World Wide Web at *support.baynetworks.com/Library/tpubs*.

Bay Networks Customer Service

You can purchase a support contract from your Bay Networks distributor or authorized reseller, or directly from Bay Networks Services. For information about, or to purchase a Bay Networks service contract, either call your local Bay Networks field sales office or one of the following numbers:

Region	Telephone number	Fax number
United States and Canada	800-2LANWAN; then enter Express Routing Code (ERC) 290, when prompted, to purchase or renew a service contract 508-916-8880 (direct)	508-916-3514
Europe	33-4-92-96-69-66	33-4-92-96-69-96
Asia/Pacific	61-2-9927-8888	61-2-9927-8899
Latin America	561-988-7661	561-988-7550

Information about customer service is also available on the World Wide Web at *support.baynetworks.com*.

How to Get Help

If you purchased a service contract for your Bay Networks product from a distributor or authorized reseller, contact the technical support staff for that distributor or reseller for assistance.

If you purchased a Bay Networks service program, call one of the following Bay Networks Technical Solutions Centers:

Technical Solutions Center	Telephone number	Fax number
Billerica, MA	800-2LANWAN	508-916-3514
Santa Clara, CA	800-2LANWAN	408-495-1188
Valbonne, France	33-4-92-96-69-68	33-4-92-96-69-98
Sydney, Australia	61-2-9927-8800	61-2-9927-8811
Tokyo, Japan	81-3-5402-0180	81-3-5402-0173

Contents

Remote Access Concentrator Software Reference

About This Guide

Contents

Figures

Tables

Chapter 1
CLI Commands

Chapter 2
na Utility

Chapter 3
Configuration Parameters

Chapter 4
erpcd

Figures

Tables

Chapter 1
CLI Commands

The Command Line Interpreter (CLI) is the command interface that runs on the RAC. At the CLI, you enter commands that connect to hosts, manage jobs, display and modify parameters, and display RAC and network information.

The CLI provides two groups of commands, user and superuser. You use superuser commands for RAC administration. To access superuser commands, issue the **su** command at the CLI user prompt and enter the RAC's administrative (**su**) password (see *su on page 1-97*).

The default user-level prompt is a colon; the default superuser prompt is a # symbol:

```
annex: su
Password:
annex#
```

A CLI command can take a maximum of ten arguments.

Command Syntax

You can shorten any CLI command or host name to the minimum number of letters that make the name unique. If you do not want the RAC to interpret a host name using minimum uniqueness, enclose the name in quotation marks (""). For example, entering hosts "new" prevents ambiguities between hosts newark and new. You can enter commands and host names in lowercase, uppercase, or a combination of the two. The RAC performs any necessary case conversion.

CLI Command Descriptions

actcall

The superuser **actcall** command lists the active calls on a given RAC. Specifying a device type (internal port type) with this command displays a summary of all active calls for that particular device type. The command syntax is:

actcall [*device_type*[*number*]]

For *device_type*, specify **asy** (the default), **syn**, or **ta**. You can also specify a device type and number, such as **ta10**, for detailed call information. If you specify a number alone, it is assumed to be an **asy** device. For example, entering **10** produces the same result as entering **asy10**.

admin

The superuser **admin** command is installed and operates on the RAC, acting as a local substitute for the host-resident **na** command. Like **na**, **admin** provides subcommands for configuring and managing the RAC. These subcommands are a subset of those provided by **na**.

The **admin** syntax is:

admin [*subcommand* [*arguments*]]

Entering the **admin** command alone at a superuser CLI connection puts you in administrative mode. The CLI prompt is replaced with the *admin* prompt. To terminate the **admin** session, press the attention key or execute the **quit** subcommand.

admin Subcommand Summary

Table 1-1 summarizes the CLI **admin** subcommand set.

Table 1-1. admin Subcommand Set

Subcommand	Description
broadcast	Sends a broadcast message to one or more ports.
help (or ?)	Displays the **admin** help screen.
interface	Establishes a default interface set used in subsequent subcommands.
modem	Displays the modem type supported by the RAC.
quit	Ends the **admin** session.
reset	Resets a routing interface, global port, WAN interface, or subsystem.
set	Modifies the value of a configuration parameter.
show	Displays the current value of a configuration parameter.
wan	Establishes a default WAN set used in subsequent subcommands.

Differences Between admin and na

The CLI **admin** subcommands function like their **na** counterparts, with the following exceptions:

- **admin** subcommands function only on the local RAC.
- The **help** subcommand provides a help summary for the CLI **admin** subcommand set only. Entering **help** *subcommand_name* displays the subcommand's syntax.
- The **reset** subcommand does not reset the connection from which the **reset** is issued. To reset your port or connection, return to the CLI and issue **hangup**.
- Any subcommand issued without arguments, except **reset**, responds with an error or usage message; the commands do not prompt for missing arguments.
- The **show** subcommand displays the **annex** or **wan** parameter values for the local RAC. If there are more than 24 lines of information to display, a *more* prompt appears after the last line. Pressing **q** (for quit) returns to the *admin* prompt; the attention character terminates the **admin** session; pressing any other key continues the display.

The **na** utility is accessed from a UNIX, Microsoft® Windows 95, or Windows NT® host.

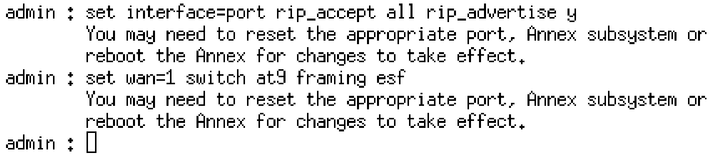
Specifying Parameters with admin Subcommands

Arguments for the **admin set**, **show**, and **reset** subcommands take parameters as arguments. With these subcommands, the parameter class is the first argument you specify. Parameter classes are **annex**, **interface**, **port**, and **wan**. With **interface** and **wan**, do one of the following:

- Define a default **interface** or **wan** *set* to which the **set**, **show**, or **reset** subcommand will apply. To do this, use the **admin interface** or **wan** subcommand.
- Define the **interface** or **wan** set within the subcommand. To do this, use the syntax:

  set parameter_class={*interface_set* | *wan_set*} *parameter_value*

Two examples follow. The first one sets two parameters for the global port interface. The second sets two parameters for WAN interface 1. Table 1-2 shows valid values for the **interface** and **wan** sets.



```
admin : set interface=port rip_accept all rip_advertise y
        You may need to reset the appropriate port, Annex subsystem or
        reboot the Annex for changes to take effect.
admin : set wan=1 switch at9 framing esf
        You may need to reset the appropriate port, Annex subsystem or
        reboot the Annex for changes to take effect.
admin : 
```

Table 1-2. interface and wan Sets

Set	Value
interface_set	**port**, to specify the global port (which includes all channels on both WAN interfaces), or **en0**, to specify the RAC’s Ethernet interface, or **all**, to specify both **en0** and the global port.
wan_set	The number of a WAN interface (**1** or **2**), or **1,2**, or **all**.

admin Subcommands

The following describes the **admin** subcommands in alphabetical order.

broadcast

The **broadcast** subcommand sends a message to modem users on specified internal asynchronous ports. The syntax is:

broadcast[=*async_port_set* | *target*] *message*

Table 1-3 describes the arguments.

Table 1-3. Arguments for the broadcast Subcommand

Argument	Description
async_port_set	Indicates the numbers of the internal asynchronous ports to which the message is to be broadcast. For example, a port set of **1, 2, 3** specifies internal ports asy1, asy2, and asy3.
target	If set to **all**, broadcasts to all asynchronous modem ports and all virtual connections. If set to **virtual**, broadcasts to all virtual CLI connections. (You cannot broadcast to a single virtual CLI connection.)
message	Specifies the message to be sent to users. If *message* requires more than one line, use the backslash (\) character at the end of each line to insert a new line.

help

The **help** (or **?**) subcommand displays a help summary for the CLI **admin** subcommand set. Entering **help** *subcommand_name* displays the subcommand syntax.

interface

The **interface** subcommand establishes a default routing *interface_set* that will be used in subsequent **admin** subcommands until another *interface_set* is specified. Grouping interfaces using an *interface_set* allows you to issue one **admin** subcommand to examine or change the parameter values for multiple interfaces. The syntax is:

interface *interface_set*

For *interface_set*, enter one of the following:

- **port**, to specify the global port (which includes all channels on both WAN interfaces)
- **en0**, to specify the RAC's Ethernet interface
- **all**, to specify both **en0** and the global port

quit

The **quit** subcommand terminates the **admin** program and returns to the superuser CLI. The syntax is:

quit

reset

The **reset** subcommand enables changes you have made to the RAC. Until you use **reset**, changes you make (using the **set** subcommand) usually become effective only after a reboot. Parameter changes to internal ports go into effect for all calls received after the reset, but do not affect current calls. Table 1-4 describes the **reset** subcommands.

⚠ Resetting a VPN (virtual private network) device or an internal (ta, asy, syn, or virtual CLI) port or modem terminates the connection on that device, port, or modem.

Table 1-4. admin reset Subcommands

Command	Description
reset all	Resets all serial and virtual CLI ports, terminating the connections on them.
reset annex all	Resets the message of the day and the Session Parameter Blocks. Also resets the security, name server, LAT, and *syslog* subsystems, as well as the customized user interface macros.
reset annex dialout	Resets the dialout subsystem by rereading the dialout section of the RAC configuration file (which defaults to **config.annex**).
reset annex lat	Resets the LAT-specific RAC parameters so that any future LAT circuits (connections) will use the new values; existing circuits will continue to use the old values. This keyword will not terminate existing LAT circuits.
reset annex macros	Rereads the customized user interface macros.

(continued on next page)

Table 1-4. admin reset Subcommands (continued)

Command	Description
reset annex modem	Rereads the **modem** section of the RAC configuration file (default is **config.annex**) and downloads the modem code from the load host.
reset annex motd	Rereads the message-of-the-day.
reset annex nameserver	Resets the name server parameters and flushes the RAC's host table.
reset annex security	Resets the security parameters and reconnects to the security host.
reset annex session	Rereads the Session Parameter Blocks from the configuration file. Existing calls are not reset. No new calls are answered while the reset is in progress.
reset annex syslog	Resets the *syslog* subsystem. The *syslog* subsystem does not use any changes made to the **syslog_port** parameter.
reset *async_port_list* reset async[=*async_port_list*]	Terminates asynchronous connections on the specified ports. For *async_port_list*, specify an individual asynchronous port number, multiple port numbers separated by commas (for example, **1,2,3**), or a range of port numbers. Separate the end points of a range with a hyphen, such as **1-10**. Parameter changes will be in effect only for calls received after the reset. Current calls (that have not been terminated) are not affected.

(continued on next page)

Table 1-4. admin reset Subcommands (continued)

Command	Description
reset interface[=*interface_set*]	Resets the interface parameters for the specified routing interfaces: **en0**, **port** (for the global port), or **all**.
reset int_modem [*modem_list*]\ [hard | soft]	Resets the hardware for each modem in the modem list, whether or not a call is active on the modem. If you specify **hard**, the modem is reset immediately, which terminates the call. If you specify **soft** (the default), the modem is reset when the call (if any) on it hangs up. For *modem_list*, specify an individual modem number, multiple modem numbers separated by commas, or a range of numbers from 1 through the total number of installed modems. Separate the ends of a range with a hyphen, for example, **1-23**.
reset serial	Terminates all asynchronous connections.
reset sync=*sync_port_list*	Terminates synchronous connections on the specified ports. For *sync_port_list*, specify an individual synchronous port number, multiple port numbers separated by commas (for example, **1,2,3**), or a range of port numbers. Separate the ends of a range with a hyphen, such as **1-10**. Parameter changes will be in effect only for calls received after the reset. Current calls (that have not been terminated) are not affected.

(continued on next page)

Table 1-4. admin reset Subcommands (continued)

Command	Description
reset ta=*ta_port_list*	Terminates ta (V.120) connections on specified ports. For *ta_port_list*, specify an individual port number, multiple port numbers separated by commas (such as **1,2,3**), or a range of port numbers. Separate the end points of a range with a hyphen (for example, **1-10**). Parameter changes will be in effect only for calls received after the reset. Current calls (that have not been terminated) are not affected.
reset virtual	Resets and terminates all virtual CLI connections.
reset vpn=*vpn_port_list*	Terminates VPN connections on specified ports, tearing down the L2TP tunnel calls that established them. For *vpn_port_list*, specify an individual VPN port number, multiple port numbers separated by commas (for example, **1,2,3**), or a range of port numbers. Separate the end points of a range with a hyphen (for example, **1-10**). Parameter changes will be in effect only for VPN calls received after the reset. Current calls (that have not been terminated) are not affected.
reset wan[=*wan_set*]	Resets all parameters for the WAN interface or interfaces (**1**,**2**, or **all**) in *wan_set*.

set

The **set** subcommand modifies configuration parameters:

set annex	Modifies RAC parameters.
set interface	Modifies interface parameters.
set port	Modifies global port parameters.
set wan	Modifies WAN parameters.
set wan b	Modifies WAN B channel parameters.
set wan ds0	Modifies WAN DS0 channel parameters.

The syntax is:

set annex *annex_parameters*

set interface [=*interface_set*] *interface_parameters*

set port *port_parameters*

set wan[=*wan_set*] | *wan_parameters*

set wan[=*wan_set*] **b**[=*channel_range*] *wan_b _parameters*

set wan[=*wan_set*] **ds0**[=*channel_range*] *wan_ ds0_parameters*

interface_set and *wan_set* must be included in the **set** subcommand or defined in an earlier **interface** or **wan** subcommand.

For *interface_set*, enter one of the following:

- **port**, to specify the global port (which includes all channels on both WAN interfaces)
- **en0**, to specify the RAC’s Ethernet interface
- **all**, to specify both **en0** and the global port

For *wan_set*, enter one of the following:

- the number of the WAN interface: **1** or **2**
- **1,2** or **all** to indicate both interfaces

When specifying parameters, use a parameter name and a value separated by a space. You can enter more than one parameter with each command; separate the parameter-value pairs with a space. If you are entering multiple parameters that require a new line, precede the new line with the backslash (\) character. Changes made to parameters take effect after booting or resetting the RAC or resetting the port, routing interface, or WAN interface.

Sample command lines for setting parameters are:

```
admin: set port mode cli
admin: set wan=1 switch_type at9 framing ESF
```

show

The **show** subcommand displays current parameter values:

show annex	Displays RAC parameters.
show interface	Displays interface parameters.
show port	Displays the global port parameters.
show wan	Displays WAN parameters.
show wan b	Displays WAN B channel parameters.
show wan ds0	Displays WAN DS0 channel parameters.

The syntax is:

show annex *keyword* | *annex_parameters*

show interface[=*interface_set*] [*keyword* | *interface_parameters*]

show port [*keyword* | *port_parameter_names*]

show wan[=*wan_set*] [*keyword* | *wan_parameters*]

show wan[=*wan_set*] **b**[=*range*] *b_channel_parameters*

show wan[=*wan_set*] **ds0**[=*range*] *ds0_channel_parameters* *interface_set* and *wan_set* must be included in the **set** subcommand or defined in an earlier **interface** or **wan** subcommand.

For *interface_set*, enter one of the following:

- **port**, to specify the global port (which includes all channels on both WAN interfaces)
- **en0**, to specify the RAC's Ethernet interface
- **all**, to specify both **en0** and the global port

For *wan_set*, enter one of the following:

- the number of the WAN interface, **1** or **2**
- **1,2** or **all** to indicate both interfaces

Each keyword in a **show** subcommand displays a subset of parameters. See *Parameters Listed by Type on page 3-5*.

arap

The **arap** command converts a CLI session into an AppleTalk Remote Access Protocol (ARAP) session. Resetting the port returns the CLI to its original mode. The syntax is:

arap

The command display looks like this:

```
annex: arap
Remote Annex switching line to ARAP.
```

arp

The superuser **arp** command displays and, optionally, modifies the IP-to-hardware address translation table used by the Address Resolution Protocol (ARP). Because the RAC builds the ARP table dynamically, you rarely need to modify it. Table 1-5 defines the arguments for this command. The syntax is:

arp [**-ads**] [*host*] [*addr*] [**temp** | **pub**]

Using either the *host* or the **-a** argument, **arp** displays a host name, if known, or a **?** in place of the host name, the Internet and Ethernet addresses, and the *time to live* (TTL) field for each entry. An example follows.

```
annex# arp -a

thirdfloor (132.245.6.65) at 00-80-2d-00-2a-c0 ttl=20
oleom (132.245.6.12) at 00-80-20-06-34-39 ttl=19
? (132.245.6.20) at 00-80-20-01-fe-b1 ttl=19
caddy (132.245.6.25) at 00-80-2d-00-22-41 ttl=20
(23f4.3e) at 08-00-4e-34-22-39 ttl=16
```

Table 1-5. Arguments for the Superuser arp Command

Argument	Description
host	Displays the current ARP table entry for that host.
addr	Displays the current ARP table entry for that address.
-a	Displays all entries in the table.
-d	Deletes the entry specified with *host*.
-s	Creates an entry for *host*, specified using either a name or Internet address, at the hardware address specified by *addr*. If you do not include **temp** or **pub**, the entry is permanent and not published.
temp	The created entry is temporary and is to be deleted after 20 minutes. Temporary entries are not published.
pub	The created entry is to be published. The RAC responds to requests for the host’s hardware address.

The **arp** command shows AppleTalk information but does not allow you to change this information. Because **arp** interprets all addresses as IP addresses, if you try to delete an AppleTalk address such as 1.123 using **arp -d**, the ARP table entry 1.0.0.123 is deleted.

bg

The **bg** (background) command puts a job into the background and displays the job number, the CLI command that created it, and an ampersand (&) to indicate that the job is in the background.

The RAC forwards output generated by the background job to the terminal, if another job is active. If another job is not active, the output is held until you bring a job back into the foreground by issuing the **fg** command. Table 1-6 describes the arguments for **bg**. The syntax is:

bg [**-d**] [**%**] [**%**, +, -, *n*, *hostname*]

Table 1-6. Arguments for the bg Command

Argument	Description
-d	Discards output from the background job to the terminal screen until the job is terminated or brought to the foreground.
%, %%, %+	Puts the most recent job (+) into the background.
%-	Puts the previous job (-) into the background.
n, %*n*	Puts job *n* into the background.
%*hostname*	Puts the job at *hostname* into the background.

You can omit **bg** from the command entry when you use one of the arguments beginning with **%**.

Entering **bg** with no arguments is the same as entering **bg%**; it puts the most recent job into the background. The **bg** command display looks like this:

```
annex: bg
2 telnet secondhost &
```

boot

The superuser **boot** command reboots the RAC and, optionally, produces a dump of the RAC’s operational code. You can set a time at which the boot is to take place. The **boot** command also sends a warning message to users attached to the RAC. Table 1-7 describes the arguments for **boot**. The syntax is:

boot [**-adhlqr**] [*time*] [*filename*] [*warning*]

Table 1-7. Arguments for the boot Command

Argument	Description
-a	Aborts any boots that are pending. Sends an abort message to all users.
-d	Causes a dump before rebooting.
-h	Causes a diagnostics boot.
-l	Boots the operational image and stores it onto local media; for use with the stand-alone file system. Only ROMs that have the self-boot option loaded support **-l**.
-q	Performs a boot without sending a warning message.
-r	Uses the last loaded image name for rebooting.

(continued on next page)

Table 1-7. Arguments for the boot Command (continued)

Argument	Description
time	Defines the time to shut down as either an offset +*MM* or +*HH***:***MM*, or a clock time *HH***:***MM*.
filename	Identifies the name of the file in which the RAC's image is maintained. If you do not enter a file name, the RAC prompts for one. If you enter a blank line, the RAC boots the image defined by the **image_name** parameter. Pressing the **Return** key at the prompt directs the RAC to boot the default file.
warning	Allows an optional message (up to 249 characters) to be sent to users. This prompt appears only if the **-q** argument is not specified.

The following command line directs the RAC to reboot an hour and fifteen minutes from the time of entry:

```
annex# boot +1:15
bootfile: <cr>
warning: Shutting down for PM
```

The RAC can request its boot file from a defined preferred load host. If that host is not defined, or does not respond, the RAC broadcasts its request and boots from the first load host to respond (assuming the **load_broadcast** parameter is set to **Y**).

Booting the RAC with a **image** file that does not exist causes the unit to hang as it searches for the image. Pressing the **Reset** switch recovers from this condition.

compact

The superuser **compact** command consolidates all valid (in-use) records to the beginning of the nonvolatile memory (EEPROM). The RAC stores configuration parameters and file system records in this memory. When the amount of free space nears depletion, the RAC logs a syslog warning message, and the **admin set** command may fail.The syntax is:

compact [-s]

The **-s** argument displays the total space and free space (in bytes) in EEPROM. The amount of free space is determined by the amount of unused space at the end of EEPROM.

Compacting can take as long as 3 minutes for an 8 K EEPROM, and 20 minutes for a 32 K EEPROM. During this time, no process can access EEPROM, including all **admin** commands and many CLI commands. You must wait for the CLI to display its prompt before continuing.

connect

The **connect** command uses the LAT protocol to connect to an advertised LAT service. This command is available only if you have enabled LAT by setting the **lat_key** parameter correctly. The syntax is:

connect *service* [*hostname* [*port*]]

If you enter the command without arguments, the **connect** command returns a *missing service name* error. If the service to which you are connecting requires a password, you are prompted for one. If you enter **connect** with only the *service* name, the command connects to the highest-rated service with that name. If you enter both the *service* name and the *hostname*, **connect** tries to establish a connection to that service on that host. Entering **connect** with *service*, *hostname*, and *port* causes the RAC to attempt to connect to the specified service on the port and host.

control

The superuser **control** command is a diagnostic tool that allows you to reset DTR and RTS or to output a short test message for a CLI port or the global port. Table 1-8 describes the arguments for **control.** The syntax is:

control *port* [**dtr-** | **dtr+** | **rts-** | **rts+**] | *port* **testmsg** [*times* | **forever**]

The following command displays the default message 10 times on CLI port 14:

```
annex# control 14 testmsg 10
Enter test message, or press Return for default: <cr>
```

If the port is not a CLI port, the command displays *Device must be in use.*

Table 1-8. Arguments for the Superuser control Command

Argument	Description
port	(Required) Specifies the index of the CLI port or the word port, which indicates the global port.
dtr-	De-asserts DTR.
dtr+	Asserts DTR.
rts-	De-asserts RTS.
rts+	Asserts RTS.
testmsg	Outputs a message to a CLI port or to a port that has been opened as a slave from a host. After the message prompt appears, pressing the **Return** key displays the default message *The quick brown fox jumped over the lazy dogs*.
times	Specifies the number of times the message is output.
forever	Displays message until a break is entered.

cp

The superuser **cp** command copies a file in the local file system. The syntax is:

cp *source_filename destination_filename*

The *source_filename* is the file to be copied; the *destination_filename* is the new file. The RAC overwrites the destination file if it exists; it reports an error if the source file does not exist.

dialout

The superuser **dialout** command displays each dialout route, along with all of the defined parameters and chat scripts for each route. The syntax is:

dialout [**do** *route_number*]

The **dialout do** *route_number* command displays only the specified route and the chat scripts that it references.

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

```
annex# dialout

Route do44:
mode: ppp                    local_address: 132.245.88.12
remote_address: 132.245.44.12net_inactivity: 10
phone_number: "92030401"     do_compression: Y
allow_compression: Y         net_inactivity_units: minutes
subnet_mask: 255.255.255.0   ppp_ncp: ipcp
rip_sub_advertise: Y         rip_sub_accept: Y
rip_advertise: all           rip_accept: all
advertise: Y
ports: 13-16
filter: in exclu proto udp src_port router netact
filter: out exclu proto udp src_port router netact
```

```
Route do131:
mode: ppp                    local_address: 132.245.88.12
remote_address: 131.110.0.13 net_inactivity: 10
phone_number: "92050111"     do_compression: Y
allow_compression: Y         net_inactivity_units: minutes
subnet_mask: 255.255.255.248 ppp_ncp: ipcp
rip_sub_advertise: Y         rip_sub_accept: Y
rip_advertise: all           rip_accept: all
advertise: Y
ports: 1-2
filter: in exclu proto udp src_port router netact
filter:out exclu proto udp src_port router netact
```

The **dialout do** *route_number* command display is as follows:

```
annex# dialout do44

Route do44:
mode: ppp            local_address: 132.245.88.12
remote_address: 132.245.44.12net_inactivity: 10
phone_number: "92030401"do_compression: Y
allow_compression: Y net_inactivity_units: minutes
subnet_mask: 255.255.255.0ppp_ncp: ipcp
rip_sub_advertise: Y rip_sub_accept: Y
rip_advertise: all   rip_accept: all
advertise: Y
ports: 13-16
filter: in exclu proto udp src_port router netact
filter: out exclu proto udp src_port router netact
```

edit

The superuser **edit** command allows you to edit any local RAC file. It provides a full screen editor that supports a small set of terminal types: vt100, wy75, and wy85 (set the **term_var** parameter to the appropriate terminal type). The syntax is:

edit *filename*

The editor supports quit, write-and-exit, page-up, page-down, and arrow keys. A menu bar at the top of the screen describes how to perform these functions.

fg

The **fg** (foreground) command resumes a job that has been suspended or placed in the background. If the RAC saved any output from the host while the job was interrupted, the output appears on the terminal immediately after you put the job in the foreground. Otherwise, nothing appears until you enter a carriage return. Table 1-9 describes the arguments for **fg**. The syntax is:

fg [**-q**] [**%**] [**%**, +, -, *n*, *hostname*]

Entering the **fg** command with no arguments puts the most recent job into the foreground. The command display looks like this:

```
annex: fg
2 telnet secondhost
```

Entering **%** is the same as entering **fg%** and returns you to the most recent job. You can omit **fg** from the command entry when you use one of the arguments beginning with **%**; the following example uses **%-** to bring the previous job to the foreground:

```
annex: %-
1 rlogin firsthost
```

Table 1-9. Arguments for the fg Command

Argument	Description
-q	Prevents a one-line message from appearing on the terminal screen when bringing a session to the foreground.
%, %%, %+	Brings the most recent job (+) to foreground.
%-	Brings the previous job (-) to foreground.
n, %*n*	Brings job *n* to the foreground.
%*hostname*	Brings the job at *hostname* to the foreground.

filter

The superuser **filter** command allows you to filter the traffic that crosses the RAC. It affects both the currently running configuration and the configuration stored in EEPROM. The **filter** command has eight subcommands: **add**, **list**, **enable**, **disable**, **delete**, **help**, **usage**, and **quit**. The syntax is:

filter [*subcommand*]

If you use the **filter** command without a subcommand, the RAC enters the filtering subsystem and displays the *filter* prompt. At this prompt, you can execute any of the eight subcommands. You return to the CLI prompt from the subsystem by executing the **quit** subcommand.

If you use the **filter** command with a subcommand, you can issue filtering subcommands directly at the superuser CLI prompt. When the subcommand completes, you are still at superuser CLI level.

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

```
annex# filter
filter:
```

The **filter list** command display looks like this:

```
annex# filter list
```

```
Num  Stat Ifname Dir  Scope Family Actions/Parameters
1    ena  en0    in   incl  ip     disc icmp/port_pair=*,nfs
2    ena  en0    in   incl  ip     disc/port_pair=*,tftp
annex#
```

Filters are applied to a particular physical interface on a RAC or to all RAC interfaces and can affect incoming or outgoing packets. An interface is the Ethernet port (specified as *en0* in commands) or the global port.

Filters are complicated and can interact in ways you might not anticipate; use them with great care.

Filters can cause performance to deteriorate significantly.

Syslogging common occurrences can flood the **syslog** file.

Syslogging *syslogs* can cause infinite loops.

Be careful when creating filters that discard packets on the Ethernet interface; filters of this type can hang the RAC.

You need superuser privileges not only to configure the RAC for filtering but also to create or modify filters.

Filter Numbers

When you **add** a filter, the RAC assigns it a number that remains associated with it until you delete the filter. The **filter** subcommand **list** displays this number, and you specify the number when you **delete**, **enable**, or **disable** a filter (see *Filter Lists* below for permissible ways to specify filters).

Filter Lists

The **delete**, **enable**, and **disable** subcommands accept a *filter_list* argument. A filter list can be a filter number, a string of filter numbers separated by commas, or a range of filter numbers. Use a dash (-) to separate the beginning and end of a range, or use it before or after a filter number. Used before a filter number, a dash indicates all defined filters up to and including that number; used after a filter number, a dash indicates all filters from that number up to and including the highest numbered filter.

Specifying an asterisk (*****), the word **all**, or a dash (-) by itself for a filter list indicates all filters. Table 1-10 shows sample subcommands using filter lists.

When you delete filters, their numbers remain unused until you add another filter; the added filter is then assigned the lowest unused number. If you invoke a subcommand with a range that includes unused numbers, the subcommand operates on the assigned filters but displays an error message for each unused number. This does not happen when you specify a group of filters by entering a number with a leading or trailing dash; in this case, unused numbers are ignored.

Table 1-10. Sample Commands using the filter_list Arguments

Argument	Description
delete 2	Deletes filter 2. A subsequent **list** subcommand will not display an entry for filter number 2.
disable 3-6	Disables filters 3, 4, 5, and 6. If one of these numbers represents a deleted filter or an existing filter associated with an inactive interface, an error message is displayed for that number; the other filters in the range are disabled.
disable 1, 3-7,10	Disables filters 1, 3, 4, 5, 6, 7, and 10. If any of these numbers represents a deleted filter or a filter for an inactive interface, an error message is displayed for that number; the other filters in the range are disabled.
disable -5	Disables filters 1, 2, 3, 4, and 5.
disable 3-	Disables all filters from filter 3 through the end of the list of all filters.
enable -	Enables all filters.
enable *	Enables all filters.
enable all	Enables all filters.
enable 5-	Enables all filters from filter 5 through the end of the list of all filters.

Filter Subcommands

This section describes the subcommands in alphabetical order.

add

The **add** subcommand adds new filter(s) and enables them in both the currently running system and EEPROM; the RAC need not be rebooted for the added filters to take effect. Table 1-11 describes the arguments for **add**. The syntax is:

add *interface direction scope* [*family*] *criteria actions*

Table 1-11. Arguments for the add Subcommand

Argument	Description
interface	Specifies the physical interface to which this filter applies. Valid values are **en0** (for Ethernet) or “*****” (for all interfaces).
direction	Specifying **input** applies the filter to incoming packets. Specifying **output** applies the filter to outgoing packets. Two filter definitions are required to apply a filter to both incoming and outgoing packets.
scope	Specifying **include** means the filter matches only those packets that meet all of the specified *criteria*. Specifying **exclude** means the filter matches only those packets that do not meet at least one of the specified criteria.
family	(Optional) Specifies the network level address family (protocol) to which the filter applies. Currently, the RAC only supports **ip**.
criteria	Specify the conditions on which the filter is based. All criteria must be met for the filter to match the packet. Specify criteria in the form: **keyword** *value*).
actions	Specifies what a filter does when all of its *criteria* match a packet. You can specify any combination of *actions*. Possible *actions* are **discard**, i**cmp**, **netact**, and **syslog**.

(continued on next page)

Table 1-11. Arguments for the add Subcommand (continued)

discard	Discards the packet. Discarding is done after any **syslog**, **icmp**, or **netact** actions are taken.
icmp	Discards the packet and sends an ICMP *destination unreachable* message.
netact	Customizes the definition of activity for a SLIP or PPP dynamic dial-out line. If one or more filters containing this action are enabled on one of these lines, only the traffic matching the filters constitutes activity. If the link is quiescent, **netact** discards the packet.
syslog	Logs the event in the system log file.

Filters that cause multiple **syslog**, **icmp**, and **netact** *actions* for the same interface are reduced to a single **syslog**, **icmp**, or **netact** action.

To add a dynamic dial-out filter, configure the **dialout** section of the RAC configuration file; you cannot add the filter using the **filter** command (see the document *Managing Remote Access Concentrators Using Command Line Interfaces*).

Table 1-12 lists valid keywords and values for the **add** subcommand's *criteria* argument. The syntax is:

keyword value

Table 1-12. Keywords for the add criteria Argument

Keyword	Value	Explanation
dst_address	{*ip_addr*[/*n*] | ***** | **-1**}	Matches the packet's destination IP address. To test only the non-host portion of the address, enter /*n* after the address, where *n* is the number of bits in the non-host portion of the subnet mask for this address. For example, **132.245.33.0/24** denotes a mask of 255.255.255.0, which matches destination addresses on network 132.245.33.0. (If you list the filter, 132.245.33.0/24 appears as the destination address.) To match all addresses, enter **-1** or "*****" instead of an address.
dst_port	{*pnum* | *sname* |***** | **-1**}	Matches the TCP or UDP destination port. Specify the port as a decimal number (*pnum*) from **1** through **65535** or as a standard service name (*sname*), such as **finger**, **ftp**, **nfs**, **rlogin**, **smtp**, **telnet**, or **tftp**. Specifying **-1** or "*****" matches all port numbers. For a list of service names and their corresponding port numbers, see Table 1-13.

(continued on next page)

Table 1-12. Keywords for the add criteria Argument (*continued*)

Keyword	Value	Explanation
src_port	{*pnum* | *sname* | ***** | **-1**}	Matches the TCP or UDP source port number. Specify the port as a decimal number (*pnum*) from **1** through **65535** or as a standard service name (*sname*), such as **finger**, **ftp**, **nfs**, **rlogin**, **smtp**, **telnet**, or **tftp**. Specifying **-1** or "*****" matches all port numbers. For a list of service names and their corresponding port numbers, see Table 1-13.
src_address	{*ip_addr*[/*n*] | ***** | **-1**}	Matches the packet's source IP address. To match only the non-host portion of the address, enter /*n* after the address, where *n* is the number of bits in the non-host portion of the subnet mask for this address. For example, **132.245.33.0/24** denotes a mask of 255.255.255.0, which matches destination addresses on network 132.245.33.0. (If you list the filter, 132.245.33.0/24 appears as the destination address.) To match all addresses, enter **-1** or "*****" instead of an address.

(continued on next page)

Table 1-12 Keywords for the add criteria Argument (*continued*)

address_pair	{*ip_addr1*[/*n*] | ***** | **-1**}
	{*ip_addr2*[/*n*] | ***** | **-1**}
	(Enter both addresses on the same line; separate them with a space)
		Matches packets passing in either direction between two specified IP addresses. To match only the non-host portion of an address, enter /*n* after the address, where *n* is the number of bits in the non-host portion of the subnet mask for this address. For example, **132.245.33.0/24** denotes a mask of 255.255.255.0, which matches destination addresses on network 132.245.33.0. (If you list the filter, 132.245.33.0/24 appears as the destination address.)
		To match all packets to or from a given address, enter one *ip_addr* and then specify ***** or **-1** for the other. For example, *** 132.254.33.2** and **132.254.33.2 -1** match all packets to or from 132.254.33.2.
		Restriction: if you use the **address_pair** keyword, you cannot use the keyword **dst_address** or **src_address**.

(continued on next page)

Table 1-12. Keywords for the add criteria Argument (continued)

port_pair	{*p1 p2*| *s1 s2* |*****|**-1**}	Matches packets passing in either direction between the two specified TCP or UDP port numbers (*p1* and *p2*) or standard service names (*s1* and *s2*), such as **finger**, **ftp**, **nfs**, **rlogin**, **smtp**, **telnet**, or **tftp**. Use a space to separate the port numbers or names. To match all packets to or from a given port number, enter one port number or service name and specify **-1** or ***** for the other. For a list of service names and their corresponding port numbers, see Table 1-13. Restriction: if you use the **port_pair** keyword, you cannot use the **dst_port** or **src_port** keyword.
protocol	{*protonum*|*protoname*}	Matches the transport protocol in the packet. Valid protocol numbers range from **1** to **255**. Or, specify a protocol name, such as **tcp**, **udp**, or **icmp**. If no protocol is given but a port is specified (**dst_port**, **src_port**, or **port_pair**), the port specification applies to both TCP and UDP packets. **Warning**: A command such as the following can cause infinite loops: **filter add asy1 output include\** > **protocol icmp icmp**

Table 1-13 shows the standard service names you can supply for name *s1* and *s2* in Table 1-12.

Table 1-13. Standard Service Names

Service Name	Port Number
domain	53
ercp	121
finger	79
ftp	21
name	42
nfs	2049
nntp	119
rlogin	221
route, routed, router	520
rtelnet	107
sftp	115
smtp, mail	25
snmp	161
telnet	23
tftp	69
time	37
who, login	513

Multiple service names shown on the same line in Table 1-13 are synonyms. Using any one of them in a filter implies using the other. However, when you list the filter using the **list** subcommand, you will see only the first service name.

Add Subcommand Examples

Because the NFS and TFTP protocols do not support password protection, you may want to use filtering to prevent hosts on an external network from using those protocols to access files on your internal network. To do this, pick a RAC to act as a firewall between the local and external network and create filters on it to block NFS and TFTP traffic. For example, you could create the following two filters, which prevent TFTP or NFS packets from crossing RAC interface *en0*:

```
annex# filter
filter: add en0 input include protocol udp port_pair\
> nfs * icmp
filter: add en0 input include protocol udp port_pair\
> tftp * icmp
```

Note the following about the preceding sample filters:

- Both filters apply only to packets arriving on RAC interface *en0*. To apply a filter to another interface, specify a second filter for that interface, or specify ***** instead of *en0*, thereby blocking the protocol on all interfaces.
- Both filters match packets whose network protocol family is IP. Because the **family** argument is optional (IP is assumed), the examples omit it.
- Both filters specify **protocol** as UDP because UDP is the transport-level protocol on which NFS and TFTP operate.
- The *port_pair* argument in each filter specifies that the filter applies to any UDP packet that contains NFS or TFTP in its source or destination protocol field.
- When one of these filters matches a packet, the RAC discards the packet and sends the ICMP message *destination unreachable, communication administratively prohibited* to the originator of packet. To discard the packet without sending a message, specify **discard** instead of **icmp**.

The following example creates a filter that logs the arrival of every IP packet on the Ethernet interface (*en0*). The example omits the network protocol family because it is optional; IP is assumed.

```
filter: add en0 input include src_addr * syslog
```

Logging events requires configuration (see the document *Managing Remote Access Concentrators Using Command Line Interfaces* for more information).

The following example allows packets to and from 132.254.100.2 and 132.254.100.3 to be forwarded over interface *en0*; all other packets are discarded.

```
filter: add en0 input exclude address_pair 132.254.100.2\
> * discard
filter: add en0 input exclude address_pair 132.254.100.3\
> * discard
```

The following example allows UDP and ICMP packets to and from 132.254.100.2 and 132.254.100.3 to be forwarded over interface *en0*; all other packets are discarded.

```
filter: add en0 input exclude address_pair 132.254.100.2 *\
> protocol icmp discard
filter: add en0 input exclude address_pair 132.254.100.3 *\
> protocol icmp discard
filter: add en0 input exclude address_pair 132.254.100.2 *\
> protocol udp discard
filter: add en0 input exclude address_pair 132.254.100.3 *\
> protocol udp discard
```

delete

The **delete** subcommand deletes filters from both the currently running system and nonvolatile memory. The syntax is:

delete *filter_list*

See *Filter Lists on page 1-27* for an explanation of the *filter_list* argument.

Dial-out filters can be deleted but will be re-installed after a reboot of the RAC or a reset of the dial-out subsystem.

disable

The **disable** subcommand disables enabled filters in the currently running system. The filters remain defined in nonvolatile memory, but packets are not matched against them. When you reboot your RAC, the disabled filters are re-enabled. The syntax is:

disable *filter_list*

The following example disables filters 2 through 4:

```
annex# filter
filter: disable 2-4
filter 2 disabled
filter 3 disabled
filter 4 disabled
```

See *Filter Lists on page 1-27* for an explanation of the *filter_list* argument.

Dial-out filters can be disabled but will be re-enabled after a reboot of the RAC or a reset of the dial-out subsystem.

enable

The **enable** subcommand immediately enables disabled filters in the currently running system. Otherwise, disabled filters are not enabled until the RAC reboots or until the port resets. A filter can be disabled only by the **disable** command. The syntax is:

enable *filter_list*

See *Filter Lists on page 1-27* for an explanation of the *filter_list* argument.

The following example enables filters 2 through 4:

```
annex# filter
filter: enable 2-4
filter 2 enabled
filter 3 enabled
filter 4 enabled
```

help

The **help** subcommand displays information about one or more **filter** subcommands. Entering **help** with no arguments displays information about all of the subcommands. Entering **help** and the name of a subcommand displays an explanation of that subcommand. For a one-line syntax, see *usage on page 1-40*.

list

The **list** subcommand displays the defined filters, along with their status (enabled or disabled) and assigned number (used by **enable**, **disable**, and **delete**). Table 1-14 describes the arguments for **list**. The syntax is:

list [**-eia**]

Table 1-14. Arguments for the list Command

Argument	Description
-e	Lists the filters stored in nonvolatile memory instead of the filters in the currently running system. Using **list -e** eliminates the status column from the display because the enabled/disabled status is not saved in nonvolatile memory.
-i	Sorts the output by interface name, instead of sorting by filter number.
-a	Sorts the output by action, instead of sorting by filter number.

If you do not specify **-e**, **list** displays only filters that are associated with active interfaces or that were acquired from **acp_dialup**.

quit

The **quit** subcommand exits the filtering subsystem and returns control to the CLI.

usage

The **usage** subcommand displays the syntax for one or more **filter** subcommands. Entering **usage** with no arguments prints syntaxes for all the subcommands. Entering **usage** and the name of a subcommand prints the syntax for that subcommand.

For more detailed information about one or more subcommands, see *help on page 1-39*.

Using Include and Exclude in a Filter

You configure a RAC filter to either **include** or **exclude** particular types of packets, based on whether or not the packet types match specified *criteria*. Including certain types means the filter does not affect any other packet type; excluding certain types means only other types are affected by the filter. The actual effect a filter has depends on the *actions* you specify for it, such as **discard**.

The *criteria* of a single **include** or **exclude** filter are logically ANDed. For an **include** filter, this means that a packet must match all the filter's *criteria* in order for the filter's *actions* to be taken. For an **exclude** filter, it means a packet must match all of the *criteria* for the *actions* not to be taken.

Multiple **include** filters for the same interface that specify the same *actions* are logically ORed. For example, if one **include** filter for *asy2* specifies that TFTP packets are to be discarded, and another **include** filter for *asy2* specifies that NFS packets are to be discarded, then any packet whose type is either TFTP *or* NFS is discarded; all other packets are accepted on interface *asy2*.

Multiple **exclude** filters for the same interface that specify the same *actions* are logically ANDed. For example, if one **exclude** filter for *asy2* specifies IP address 132.254.45.1 and **discard**, and another **exclude** filter for *asy2* specifies IP address 132.254.55.2 and **discard**, then any packet whose destination address does not match 132.254.45.1 and does not match 132.254.55.2 is discarded; that is, only packets addressed to either 132.254.45.1 or 132.254.55.2 are accepted on interface *asy2*.

In general, use **include** filters to perform an action (such as discard) on only a few types of packets. Use **exclude** filters to exempt only a few types of packets from a particular action.

If both **include** and **exclude** filters are defined for the same interface, the **exclude** filters take precedence. However, mixing **include** and **exclude** filters on the same interface is strongly discouraged, because the interactions are subtle and can be confusing even for a networking expert.

hangup

The **hangup** command terminates all of your jobs, resets the CLI for the port, and drops the modem control signal DTR; it restores the default terminal characteristics defined for the port. Also, entering the **hangup** command at the CLI prompt disconnects a virtual connection to another RAC. The syntax is:

hangup

If you have any open jobs on the CLI, the **hangup** command lists them and prompts for permission to terminate each job before completing the command. For example:

```
annex: hangup
Following background job(s) will be terminated:
-1 telnet mouse
+2 telnet 132.245.6.35
Do you still wish to hangup (y/n)[n]: y
Terminating jobs, resetting line and disconnecting.
```

If you are using a modem, configure it to respond when the DTR signal is dropped.

histcall

The superuser **histcall** command displays the call history summary. The syntax is:

histcall [*index*]

The call history statistics are maintained in first-in, first-out (FIFO) order. Using the command without specifying an index displays a summary of the call history table. Specifying an index number with this command displays a detailed record of a call's history.

help

The **help** or **?** command provides online help. The syntax is:

help [*command*]

? [*command*]

Entering **help** or **?** with a CLI command as the argument (for example, **help hosts**) displays a short description of that command and its syntax. Entering **help** or **?** without an argument displays a summary of all CLI commands and macros available on the current port.

help -m

The superuser **help -m** command displays a list of all macros and their assigned *port_set* for the current RAC. The syntax is:

help -m [*macro_name*]

The **help -m** command display looks like this:

```
annex01# help -m

Name    Assigned Ports            Description
==========================================================
1       1,12                     :Menu 1
2       1-16,v                   :Read EMAIL
3       1-16,v                   :Command disabled
4       1-16,v                   :Another who command
==========================================================
init_cli 7                       :Set stty commands
init_cli 2,4,6,8,10,12,14,v      :Another who command
init_cli 1,3,5,16                :Dedicated port macro
init_psr 2,4,6,10                :Port 10 information
annex01#
```

The **help -m** [*macro_name*] display looks like this:

```
annex01# help -m menu2
Macro Name: menu2  Description: Menu 2
Command List (command access restrictions apply):
     bg, boot, fg hangup, help, hosts, jobs, menu, pg,
     procs, rlogin, telnet, who
Assigned Ports: 1,12
Functional Text:
(No available data)
```

In the previous sample display, the *Command List* field applies to menus only: it lists the commands available from the menu. The following conditions may restrict command access:

- Superuser commands are not available from a user level CLI.
- The aliases listed may not be available from a given port.
- Command masks may apply.
- Other restrictions may apply.

hosts (User Level)

The **hosts** command displays the names and addresses of hosts and other RACs listed in the RAC host table (known hosts). The command also displays any status information that a host broadcasts. Table 1-15 describes the arguments for **hosts**; Table 1-16 describes the *status* field in the **hosts** command display. The syntax is:

hosts [**-qns**] [*host*][*ip_address*]

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

```
annex: hosts

Host Name  Status   Users    Load    Internet Addr
alpha      --       --       --      132.245.6.65
neon       up       12       0.40    132.245.6.30
calvin     down     16       3.55    132.245.6.128
hobbes     down ?    8       2.01    132.245.6.1
```

Table 1-15. Arguments for the hosts Command

Argument	Description
-q	Displays only the names of known hosts.
-n	Displays data from the RAC's list of name server hosts, rather than the list of all hosts, as well as the default domain and domain search list contents.
-s	Displays the name and Internet address of the currently connected security host.
host	Displays information for *host*. Specify *host* as a name or an IP address. IEN-116 name servers cannot do reverse address queries. Specifying an IP address succeeds only if the address is in the local host table.

Table 1-16. Status Field Definitions

Field	Definition
--	The host does not broadcast status information.
up	The host broadcasted within the last several minutes.
down?	Six minutes have elapsed since the host’s last broadcast.
down	More than 12 minutes have elapsed since the host’s last broadcast.

hosts (Superuser Level)

The superuser **hosts** command provides information about hosts and name servers. Table 1-17 describes the arguments for **hosts**. The syntax is:

hosts [**-qaffn**] [*host...*]

Table 1-17. Arguments for the Superuser hosts Command

Argument	Description
-q	Displays only the names of known hosts.
-an *host*	Adds new name servers to the name server table; these entries are not saved in EEPROM and are lost when the RAC is rebooted. The syntax is: **hosts -an** *host* [*protocol* [*max_retry* [*time_out_retry* [*base* [*multiplier*]]]]] All omitted values are set to defaults: *time_out_retry* is measured in minutes; *base* is in milliseconds (ms); and *multiplier* is in tenths of seconds.
-f *host*	Flushes that host from the host table. Entering the command without *host* flushes all entries except the RAC’s own entry.
-ff *host*	Deletes permanent entries loaded from the **gateway** section of the configuration file.

(continued on next page)

Table 1-17. Arguments for the Superuser hosts Command (continued)

Argument	Description
-fn	Flushes all name servers.
-fn *ip_address*	Flushes all name servers of the given IP address.
-n	Lists all name servers.
host	Displays information for *host.* Specify *host* as a name or an IP address. IEN-116 name servers cannot do reverse address queries. Specifying an IP address succeeds only if the address is in the local host table.

ipx

The **ipx** command configures a CLI port for IPX usage, while leaving the port **mode** set to **cli**. This command allows IPX administrators to take full advantage of security features such as SecurID and Enigma. When a Fastlink II user in terminal mode logs into a RAC CLI port, the RAC authenticates the user according to the value of the **cli_security** parameter and the configuration of RAC security parameters.

Although **ipx** is a user-level command, only the superuser **help** command displays information about it.

jobs

The **jobs** command displays information about all current jobs (or sessions). The syntax is:

jobs

The **jobs** command displays the CLI command used to create the job. A plus sign (+) displayed with the job indicates the most recently active job; a minus sign (-) indicates the previously active. The **jobs** command display looks like this:

```
annex: jobs
-1 telnet firsthost
+2 rlogin secondhost
 3 telnet thirdhost
```

kill

The **kill** command terminates a connection and ends a job. The RAC accepts up to four arguments to kill multiple jobs. Table 1-18 describes the arguments for **kill**. The syntax is:

kill [**%**] [**%**, +, -, *n*, *hostname*] ...

Entering the **kill** command without arguments kills the most recent job and displays the job number and the CLI command that created it:

```
annex: kill
2 [terminated] rlogin secondhost
```

Table 1-18. Arguments for the kill Command

Argument	Description
%, %%, %+	Kills the most recent job (+).
%-	Kills the previous job (-).
n, %n	Kills job *n*.
%*hostname*	Kills the job at *hostname*.

lock

The **lock** command prevents unauthorized use of your port. The **lock** command prompts for a password, and denies access to the port until that password is entered. The syntax is:

lock [*time_out*]

A *Key* prompt appears after the port is locked and remains until you enter the correct password, as shown in this example:

```
annex: lock
Key:
Again:
Remote Annex port 3 locked
Key:
```

The password is never displayed with the *Key* or *Again* prompts.

The **lock** command permits you to define a *time_out*. This is the amount of time in minutes that the port is locked. When the time limit is exceeded, the RAC resets and unlocks the port (like **hangup**). For example:

```
annex: lock 60
```

Entering the RAC's administrative password, or resetting the port, unlocks the port.

ls

The superuser **ls** command displays the image name along with revision information for the operational image stored in the self-boot ROM. The syntax is:

ls

The **ls** command displays three fields from each file:

- The size in bytes
- The last modified date
- The file name

The self-boot image file name is a special case: the image’s revision information is also displayed. Because the directory is part of the file system, it is displayed along with the other files (the directory’s name is “.”).

modem

The superuser CLI **modem** command displays information about the RAC internal modems. Using the **modem** command and its arguments you can display status information that includes:

- Type of modem.
- Configuration settings for a type of modem
- Whether or not a modem is allocated

You can also use the **modem** command to make a failed modem available.

modem Command Syntax

modem [**-almusv**[*number-range*]]

When no arguments are specified, the command lists the names of the internal modem types in use for this RAC. Modems are in use if they are defined in the RAC configuration file and also defined by the **type_of_modem** parameter in an SPB. If no modem types are defined in the configuration file, the **modem** command displays BAY_5399_DEFAULT.

Table 1-19 describes the arguments for the **modem** command.

Do *not* enter a space between one or more **-mus** arguments and the *number-range*.

Table 1-19. Arguments for the modem Command

Argument	Description
-a	Lists the names and configuration settings for the internal modem types defined in the RAC configuration file, excluding those not in use for this machine (that is, not defined by the **type_of_modem** parameter in an SPB). If no modem types are defined in the configuration file, the command displays BAY_5399_DEFAULT.
-l	Lists the names of all modem types defined in the configuration file, including those not in use on this machine. (Modem types are not in use unless they are also defined by the **type_of_modem** parameter in an SPB.) If no modems are defined in the configuration file, the command displays BAY_5399_DEFAULT.

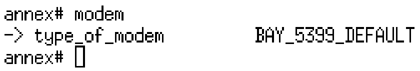
(continued on next page)

Table 1-19. Arguments for the modem Command (continued)

Argument	Description
-m[*number-range*]	Displays whether each modem specified in the *number-range* is *Allocated* (assigned from the modem pool) or *Unallocated* (free within the modem pool). If *number-range* is not specified, this information is displayed for all internal modems.
-u[*number-range*]	Makes failed modems available. The command has no effect on any modems already available, busied out, or in use. If *number-range* is not specified, all failed modems are made available. Failed modems result from diagnostics that run at boot time.
-s[*number-range*]	Displays a digital modem's status blocks. If *number-range* is not specified, this information is displayed for all internal modems.
-v	Displays a string obtained from the modem image, indicating the version, date, and checksum of the internal modem set, followed by modem image status. The normal status is running; any other status typically indicates a fault. For self-booting RACs, the string displayed indicates an internal modem version. If you contact Bay Networks Technical Support because of modem problems, you will be asked to supply the version information displayed by this command.

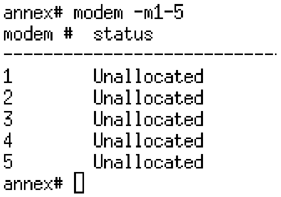
Examples

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



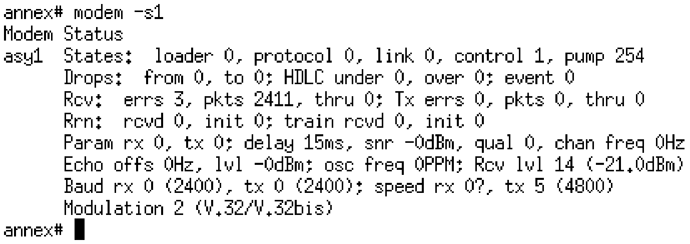
```
annex# modem
-> type_of_modem          BAY_5399_DEFAULT
annex# []
```

The following **modem -m** command display shows that modems 1 through 5 are unallocated (available in the modem pool):



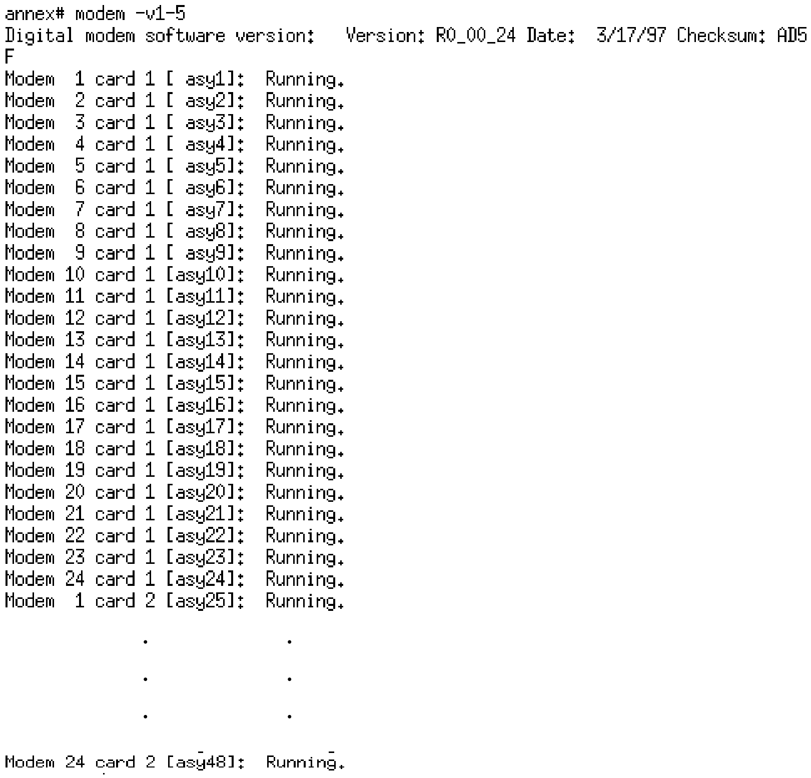
```
annex# modem -m1-5
modem #  status
----------------------------
1        Unallocated
2        Unallocated
3        Unallocated
4        Unallocated
5        Unallocated
annex# []
```

The next example shows the status block for modem 1:



```
annex# modem -s1
Modem Status
asy1  States:  loader 0, protocol 0, link 0, control 1, pump 254
      Drops:  from 0, to 0; HDLC under 0, over 0; event 0
      Rcv:  errs 3, pkts 2411, thru 0; Tx errs 0, pkts 0, thru 0
      Rrn:  rcvd 0, init 0; train rcvd 0, init 0
      Param rx 0, tx 0; delay 15ms, snr -0dBm, qual 0, chan freq 0Hz
      Echo offs 0Hz, lvl -0dBm; osc freq 0PPM; Rcv lvl 14 (-21.0dBm)
      Baud rx 0 (2400), tx 0 (2400); speed rx 0?, tx 5 (4800)
      Modulation 2 (V.32/V.32bis)
annex# █
```

This example displays the version of the internal modem set in use, along with status information for each modem:



```
annex# modem -v1-5
Digital modem software version:   Version: R0_00_24 Date:  3/17/97 Checksum: AD5
F
Modem  1 card 1 [ asy1]:  Running.
Modem  2 card 1 [ asy2]:  Running.
Modem  3 card 1 [ asy3]:  Running.
Modem  4 card 1 [ asy4]:  Running.
Modem  5 card 1 [ asy5]:  Running.
Modem  6 card 1 [ asy6]:  Running.
Modem  7 card 1 [ asy7]:  Running.
Modem  8 card 1 [ asy8]:  Running.
Modem  9 card 1 [ asy9]:  Running.
Modem 10 card 1 [asy10]:  Running.
Modem 11 card 1 [asy11]:  Running.
Modem 12 card 1 [asy12]:  Running.
Modem 13 card 1 [asy13]:  Running.
Modem 14 card 1 [asy14]:  Running.
Modem 15 card 1 [asy15]:  Running.
Modem 16 card 1 [asy16]:  Running.
Modem 17 card 1 [asy17]:  Running.
Modem 18 card 1 [asy18]:  Running.
Modem 19 card 1 [asy19]:  Running.
Modem 20 card 1 [asy20]:  Running.
Modem 21 card 1 [asy21]:  Running.
Modem 22 card 1 [asy22]:  Running.
Modem 23 card 1 [asy23]:  Running.
Modem 24 card 1 [asy24]:  Running.
Modem  1 card 2 [asy25]:  Running.

              .              .

              .              .

              .              .


Modem 24 card 2 [asy48]:  Running.
```

more

The superuser **more** command provides a read-only mechanism for reviewing files in the local file system. The syntax is:

more *filename*

The file is displayed from the beginning to the end. This command pauses after every 23 newline characters and prompts you to press a key. Pressing **q** or the attention key cancels the command; pressing any other key displays the next page of the file.

mv

The superuser **mv** command renames a file in the local file system. The syntax is:

mv *source_filename destination_filename*

The *source_filename* is the name of the existing file; the *destination_filename* specifies the new file name. The RAC overwrites the destination file if it exists; it reports an error if the source file does not exist.

netstat

The **netstat** command displays statistics and information that the RAC has obtained from the network. The command is similar to the UNIX **netstat** command in format and display, but offers additional options. Table 1-20 describes the arguments for **netstat**. The syntax is:

netstat [**-Aab**[**mp#**]**CfgimnpQRrSsTtx**[**-i** | **-r** [*network_number*] | **-sS** [*server_name*] | **? | -m**] **z**] *port*

Because the RAC is a multitasking system, this command can produce misleading information if the underlying data structures are changing rapidly.

The display format varies according to the options selected and the network protocols implemented for the RAC. Entering **netstat** without arguments displays the local and remote addresses, the send and receive queue sizes (in bytes), the protocol, and the internal state of the protocol for all active connections.

Addresses are displayed as either *host***.***port* or *network***.***port*. The latter form is displayed if a socket's address does not include a specific host address. Known host names are displayed; otherwise, the Internet addresses are displayed. Unspecified or wildcard addresses and ports appear as an asterisk (*).

Table 1-20. Arguments for the netstat Command

Argument	Description
-A	Displays the default information along with the address of any associated protocol control blocks.
-a	Displays the state of all sockets, including those used by server processes.
-b[mp#]	Displays statistics for all MP bundles or for a specified bundle, such as **mp3**. Virtual Private Network Links whose dial-in links terminate on other RACs are displayed as such.
-C	Displays the contents of the route cache.
-i	Displays the state of the hardware interfaces, for example, AppleTalk, SLIP, PPP, as well as a dial-out route's interface name.
-ia *port*	Displays statistics for a specific RAC AppleTalk interface.
-ip *port*	Displays the current state of a PPP interface.
-iQ	Displays interface queues.
-iS	Displays the state of the hardware interfaces plus additional information about the SLIP interfaces.
-f	Displays filtering statistics.
-g	Displays RIP statistics.
-m	Displays statistics for memory buffer allocation.
-n	Displays all network addresses as numbers rather than names or symbols; can be used in combination with **-A**, **--a**, **-i**, **-r**, **-t**. It displays the IP addresses and TCP ports in dotted-decimal notation.
-R	Displays information about rotaries.
-r	Displays the routing table, including dial-out routes.
-ra	Displays only AppleTalk routes.
-ri	Displays only IP routes.

(continued on next page)

Table 1-20. Arguments for the netstat Command (continued)

Argument	Description
-s	Displays network protocol statistics. LAT statistics are displayed only if the correct **lat_key** value is set.
-rs	Displays routing statistics.
-T	Displays information about VPN tunnel interfaces.
-t	Displays the default active connection information along with the attached device name.
-x	Displays information about IPX.
-xi	Displays information about RACs currently in use for dial-in or LAN-to-LAN routing.
-x?	Displays information about using the **netstat -x** command.
-xm	Displays information about the amount of memory available in the large and small IPX buffer pools.
-xr	Displays the routes defined in the RAC's IPX routing table.
-xr *network_number*	Displays the RAC route for that network.
-xs	Displays server names, types, and addresses.
-xs *server_name*	Displays information for the specified server (the *server_name* argument is case-sensitive).
-xS	Displays the RAC route for each server.
-xS *server_name*	Displays the RAC route for the specified server (the *server_name* argument is case-sensitive).
-z	Displays the network zone list for AppleTalk.

passwd

The superuser **passwd** command changes the RAC's administrative password. The RAC does not echo passwords. Pressing the **Return** key after the prompts for the new password sets the password back to its default. The syntax is:

passwd

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

```
annex# passwd
Current password:
New password:
Confirm new password:
```

If the RAC is configured with an IP address, the default administrative password is the IP address.

If the RAC is not yet configured with an IP address and the administrative password has not been modified (either via this command or via the RAC parameter **password**), the default password is a null string ("").

If the RAC is not configured with an IP address and boots via MOP, IPX, or from flash ROM, the default password is a null string (""); entering a carriage return at the *Password* prompt places you in superuser mode.

ping

Use the superuser **ping** command to determine whether a remote host, router, or RAC can be reached, and to view statistics about packet loss and delivery time. The **ping** command sends an Internet Control Message Protocol (ICMP) Echo Request message to elicit an ICMP Echo Response from the specified host, router, or RAC. The command prints output for each response returned. Table 1-21 describes the arguments for **ping**.

The syntax is:

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

Table 1-21. Arguments for the Superuser ping Command

Argument	Description
-a	Generates AppleTalk Echo Protocol (AEP) echo request packets to a target node. Displays the time the packet took to turn around.
-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. This option can ping a local host through an unlisted interface in the routing table.
-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 returned from its remote destination and, if not, where it stopped. The option is based on the Traceroute facility described in RFC 1393. For more information, see *Using the -t (traceroute) Option on page 1-62*. Table 1-22describes the fields displayed by this option. You can use **-t** with the **-r** and **-v** options, but not with the **-a** option.
-v	Displays the IP and ICMP packet headers for the reply from the host.
host	The host, router, or RAC 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 pings to be sent to the destination. The default is unlimited. When invoked with the **-t** option, **ping** ignores the *count* argument.

Each Echo Request includes a timestamp if the number of data bytes is greater than 8. This timestamp calculates the round-trip time and is returned unchanged in the Echo Response. The default packet size is 64 bytes, 56 of which are for data and 8 of which are for header information. You can change the number of data bytes using the *databytes* argument.

Unless you use the **-t** option or the *count* argument, **ping** continually sends one request per second, and displays a line of output for every response. Entering any character from the keyboard stops **ping**. The *count* argument allows you to send a limited number of requests. When **ping** stops, it displays a brief summary.

The following is a partial **ping** display, which the user has stopped by entering a keyboard character (not shown); the *PING Statistics* appear the character is entered.

```
annex# 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
```

The following is a sample **ping** display for a host that does not answer, which the user stops by entering a keyboard character; the *PING Statistics* appear after the character is entered.

```
annex# ping zinc
PING zinc: 56 bytes of data
---- zinc PING Statistics ----
15 packets transmitted,0 packets received, 100% packet loss
```

If the host named *zinc* (in the previous example) is on another network, **ping** displays the following:

```
annex# ping zinc
PING zinc: 56 bytes of data
---- zinc PING Statistics ----
Host is unreachable... Received from 132.254.55.11
annex#
```

In the preceding example, *132.254.55.11* is the IP address of the router connecting the local network to other networks.

Sample Displays Using the -a and -v Options

The following is a sample **ping -a** display for a Macintosh:

```
annex# ping -a 03fe.88
PING xenna: 56 data bytes
---- zinc PING Statistics ----
64 bytes from 03fe.88: aep_seq=0. time=7. ms
64 bytes from 03fe.88: aep_seq=1. time=5. ms
```

The following is a sample **ping -v** display:

```
annex# ping -v 132.245.55.222 56 1
PING 132.245.55.222: 56 data bytes
64 bytes from 132.245.55.222: icmp_type=0 (echo reply)
x00: x00400045
x04: x00000f8b
x08: x000001ff
x0c: xde37f584
x10: xdf37f584
x14: x6de10000
x18: x00000171
x1c: x2d458f5d
x20: x0001d11e
x24: x0b0a0908
x28: x0f0e0d0c
x2c: x13121110
icmp_code=0
64 bytes from 132.245.55.222: icmp_seq=0. time=5. ms
---- 132.245.55.222 PING Statistics ----
1 packets transmitted, 1 packets received, 0% packet loss
round-trip (ms) min/avg/max = 5/5/5
annex#
```

In the preceding display, the 4-byte hexadecimal numbers in the line beginning *x00* through the line beginning *x10* represent the IP header, while the remaining hexadecimal numbers represent the ICMP header and data. The bytes in the header lines are displayed in reverse order, so read them from right to left.

Using the -t (traceroute) Option

The **ping -t** command sends only one ICMP Echo Request. This request, called the *outbound packet*, contains an IP *traceroute* option and a traceroute hop count of 0. If an outbound packet crosses routers on the path to its destination, each router increments the hop count by 1, forwards the packet, if possible, and returns a traceroute message to the originator (Figure 1-1 illustrates an outbound packet that crosses two routers). This message indicates whether or not the packet was forwarded. If so, the message contains the incremented hop count and information about the outbound interface over which the packet was forwarded. If the packet could not be forwarded, the router discards it, **ping -t** terminates, and the traceroute message contains 0s in place of interface information.

If an outbound packet reaches its destination, the destination node sends an ICMP Echo Response, called the *return packet*, to the router from which it received the outbound packet. The destination node copies the traceroute option from the outbound packet to the return packet and sets the return packet’s hop count to 0. If the return packet passes through the routers in the path back to the **ping -t** source, each router increments the hop count by 1, forwards the packet, if possible, and returns a traceroute message to the **ping -t** source (see Figure 1-1).

The traceroute message indicates whether or not the packet was forwarded. If so, the message includes the incremented hop count and information about the interface over which the packet was forwarded. If the packet could not be forwarded, the router discards it, **ping -t** terminates, and the traceroute message contains zeros in place of interface information.

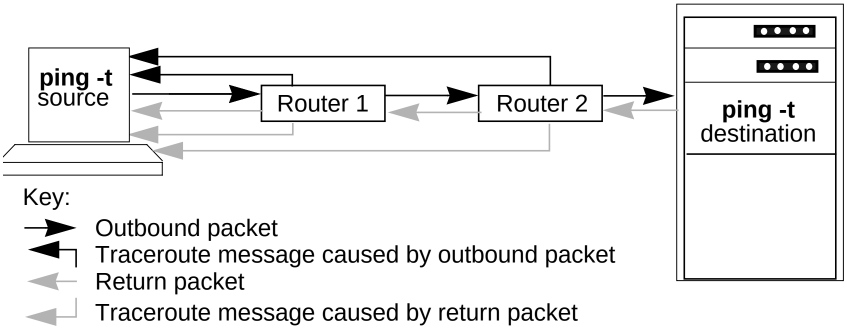

ping -t
source
Router 1
Router 2
ping -t
destination
Key:
Outbound packet
Traceroute message caused by outbound packet
Return packet
Traceroute message caused by return packet

Figure 1-1. ping -t (traceroute) Operation

Using the information carried in the outbound packet, along with the return packet and the traceroute messages, **ping -t** displays the path of the packets and the characteristics of the routing interfaces along the way and back. If a packet cannot be forwarded, **ping -t** locates the failure. Table 1-22 describes the fields displayed by **ping -t**.

Table 1-22. Fields Displayed by the ping -t Command

Field	Definition
Dir	The direction in which the ICMP packet is heading. The greater than (>>>) symbols indicate an outbound packet heading towards the **ping -t** destination. The less than (<<<) symbols indicate a return packet heading back towards the **ping -t** source. The asterisks (***) indicate that a router could not forward the packet. In this case, the router discards the packet and **ping -t** terminates.
Router	The IP address of the router interface over which the outbound or return packet was forwarded.
Hops	The number of routers that the outbound or return packet has crossed. If the count skips a hop (for example, goes from 4 to 6), a traceroute message was lost, probably due to network congestion.
Speed	The speed, in bits per second, of the interface over which the outbound or return packet was forwarded. If the packet could not be forwarded, **ping -t** displays a 0 in this field.
MTU	The maximum transmission unit (in bytes) of the interface over which the outbound or return packet was forwarded. The MTU is the largest packet size the interface can forward without fragmenting the packet. If the packet cannot be forwarded because its size exceeds the MTU and its header indicates not to fragment, **ping -t** displays a 0 in this field.

The sample topology shown in Figure 1-2 is assumed by the **ping -t** examples that follow.

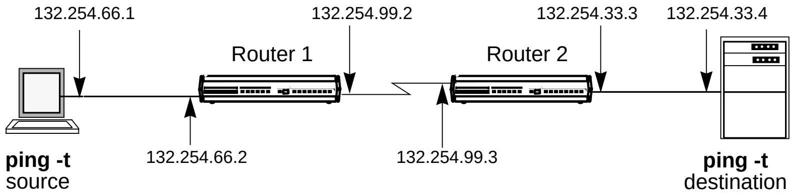


Figure 1-2. Topology for ping -t Examples

Given the topology in Figure 1-2, the **ping -t** command displays output such as the following when a traceroute packet passes successfully to the **ping -t** destination and back. (Table 1-23 explains each line of the display.)

The line numbers at the right of this example are for reference only; they are not part of the actual display.

```
annex# ping -t 132.254.33.4
PING hobbes: 56 data bytes                  line 1

 Dir     Router          Hops   Speed (b/s)   MTU      line 2
 >>>     132.254.99.2    1      19200         1024     line 3
 >>>     132.254.33.3    2      10000000      1500     line 4
 <<<     132.254.99.3    1      19200         1024     line 5
 <<<     132.254.66.2    2      10000000      1500     line 6
 64 bytes from 132.254.33.4: time=10. ms               line 7
```

Table 1-23. ping -t Command Display

Line	Description
line 1	Indicates that **ping -t** has started.
line 2	Contains the display header.
line 3	Indicates that Router 1 was the packet's first hop on the path to the **ping -t** destination. The interface over which Router 1 forwarded the outbound packet has an IP address of 132.254.99.2, a speed of 19200 bits per second, and can transmit packets of up to 1024 bytes in length without fragmenting them.
line 4	Indicates that Router 2 was the packet's second hop on the path to the **ping -t** destination. The interface over which Router 1 forwarded the outbound packet has an IP address of 132.254.33.3, a speed of 10,000,000 bits per second, and can transmit packets of up to 1500 bytes in length without fragmenting them.
line 5	Indicates that Router 2 was the return packet's first hop on the way back to the **ping -t** source. The interface over which Router 2 forwarded the packet has an IP address of 132.254.99.3, a speed of 19200 bits per second, and can transmit packets of up to 1024 bytes in length without fragmenting them.
line 6	Indicates that Router 1was the return packet's second hop on the way back to the **ping -t** source. The interface over which Router 1 forwarded the packet has an IP address of 132.254.66.2, a speed of 10,000,000 bits per second, and can transmit packets of up to 1500 bytes in length without fragmenting them.
line 7	Indicates the **ping -t** source has received the return packet and that the round-trip took 10 milliseconds.

In the following example, the second router is unable to forward the outbound packet, as indicated by the asterisks (***) under the *Dir* heading. Note that the hop count remains at *1* because the packet crossed only one router.

```
annex# ping -t 132.254.33.4
PING hobbes: 56 data bytes

 Dir     Router          Hops    Speed (B/s)      MTU
 >>>     132.254.99.2    1       19200            1024
 ***     132.254.33.3    1       0                0
```

ppp

The **ppp** command allows a user at a remote host to dial into a modem attached to the RAC and convert the CLI port to a PPP interface. Resetting the port returns it to CLI mode. The syntax is:

ppp

The command display looks like this:

```
annex: ppp
Switching to PPP and starting LCP and ATCP negotiations.
```

If you execute the **ppp** command, and the port is not configured properly for PPP, the terminal displays *This port cannot be used for PPP.*

Although **ppp** is a user-level command, only the superuser **help** command displays information about it.

You cannot apply the minimum uniqueness feature to **ppp**.

procs

The superuser **procs** command displays information about RAC processes in a tabular format. It is used for debugging RAC software. Table 1-24 describes the arguments for **procs**; Table 1-25 describes the fields in the **procs** command display; and Table 1-26 describes the RAC processes. The syntax is:

procs [**-ir**] [**-p***pid*] [**-d***dev*]

Table 1-24. Arguments for the Superuser procs Command

Argument	Description
-i	Displays statistics on time spent in interrupt routines.
-r	Displays only processes that are currently running on the RAC.
-p*pid*	Displays only processes for the specified *pid*.
-d*dev*	Displays only processes attached to the specified *dev*. Entering a ? displays processes without an attached device.

A typical display looks like this:

```
annex# procs
  PID  PPID SF  STACK SSIZ  USPTR IP CP SIG  CTIME  CPU TIME    DEV NAME
    0     0 W0 3f1740 1000 3f2660  0  0   0      0  0:00.284      ? root
 1281     0 W0  1bd20  23c  1beec  0  0   0      1  0:00.269      ? watcher
 1282     0 S0  1ad98  c3c  1b8f4 12 12   0      1  0:09.232      ? routed
 1283     0 W0  1a838  43c  1abd4 12 12   0      1  4:54.655      ? syslogd
 1284     0 S0  1a188  43c  1a52c 12 12   0 2f2c1d  0:00.005      ? timed
 1285     0 S0  19d48  43c  1a080 12 12   0 2f2c1d  0:00.001      ? erpcd
 1286     0 S0  190f0  13c  19184 12 12   0 2f2c1d  0:00.000      ? nerpcd
 1287     0 S0  1e988  63c  1ef38 12 12   0 2f2c1d  0:08.663      ? cmb_d
 1288     0 S0  1d948 103c  1e870 12 12   0 2f2c1d  0:00.386      ? snmpd
   .
   .
   .
  1387  1301 S0  26da8  93c  275c0 12 12   0 369ffa  0:00.000  ptml1 telnetd_rdr
  1389  1387 Sc  2a420  63c  2a970 12 12   0 369ffa  0:00.001  ptml1 telnetd_wri
  1390     0 X0  28ee0  b8c  29648 12 12   0 369ffe  0:00.006   pts1 cli
```

Table 1-25. Superuser Procs Command Display

Command	Description
PID	Process ID in decimal.
PPID	Parent process ID in decimal.
S*x*	Status: **S** (sleeping), **W** (semaphore wait), **R** (runnable), **X** (executing; always the CLI process executing **procs**), **E** (event wait), and **Z** (zombie, waiting for parent to collect exit status).
*x*F	Flag bits: **0x01** (system mode), **0x02** (process pre-empted), **0x04** (SIGCLD sent to parent on exit), **0x08** (go to zombie on exit).
STACK	System stack pointer in hexadecimal.
SSIZ	Stack size in hexadecimal.
USPTR	User stack pointer in hexadecimal.
IP	Initial priority at process creation in decimal.
CP	Current priority in decimal.
SIG	Pending signals for process in hexadecimal.
CTIME	Creation time of the process. Calculated as the number of seconds since January 1, 1970, expressed as the last 6 hexadecimal digits.
CPU TIME	CPU time used by the process, in minutes, seconds, and milliseconds.
DEV	Device: **?** for system processes, port number for a physical port, **v** followed by a number for a virtual CLI.
NAME	The name of the process.

Table 1-26. RAC Processes

Process	Purpose
acplog	Reliable ACP (TCP based)
acppr	Polls main host to see when primary comes back in service.
adm_timer	Watches serial ports for activity (idle timer, inactivity timer).
arap	ARAP line client.
atalkd	AppleTalk daemon.
at_echoer	AppleTalk ping handler.
at_nbp	AppleTalk name binding protocol.
at_rtmp	AppleTalk routing table maintenance protocol.
at_zip	AppleTalk zone information protocol.
bootpd	Bootpd server for BootP over SLIP/PPP.
ccp_engine	PPP data compression (CCP).
cli	One per active CLI.
cmb_d	5000 series hub daemon.
connect_rdr connect_wtr	One pair per active **connect** command.
dhcpd	DHCP relay/client.
dp_mon	Listens for dedicated port requests.
dyn_dial	Dial-out route in progress.
erpcd	Listens for incoming **erpcd** requests.
fingerd_lis	User information for listener.
httpd	HyperText Transfer Protocol.
line_adm	Port and virtual line administrator.
lpd	Listens for **aprint** commands.

(continued on next page)

Table 1-26. RAC Processes (continued)

Process	Purpose
nerpcd	RAC server dial-back security.
netdattimer	Ages the host table.
ping	**ping** command.
ppp	PPP line client.
p_srvr_conv	Prompts for CLI and rotary access.
reset_mach	Listens for the **reset all** command.
rlogin_rdr rlogin_wtr	One pair for each active **rlogin** command.
root	The initial process.
routed	Listens for and transmits RIP messages.
ROUTER_main	IPX routing information protocol.
rwhod	Listens for **rwho** requests.
slip	A SLIP line client.
snmpd	Listens for SNMP commands and requests.
syslogd	Syslog forwarding daemon.
syslog_port	Logs messages to the port specified by the **syslog_port** parameter.
telnet_cmd telnet_rdr	One pair per active **telnet** command.
telnetd_lis	Listens for incoming Telnet requests.
telnetd_rdr telnetd_wri	One pair per active incoming Telnet session.
watcher	Maintains the watchdog timer.
timed	Maintains the RAC time-of-day clock.
wan_manager	WAN interface call control.

queue

The **queue** command displays information about queued HIC requests or removes a particular HIC request from the queue. It is available only after LAT is configured. Table 1-27 describes the arguments for **queue**. The syntax is:

queue [[[**-h** *hostname*] [**-s** *service*] [**-p** *port*]], [**-r** *entry_id*] [**-v**]]

Table 1-27. Arguments for the queue Command

Argument	Description
-h	Displays only the entries originating from *hostname.*
-s	Displays only the entries requesting *service.*
-p	Displays only the entries requesting connection to *port.*
-r	Removes the entry associated with *entry_id* from the queue. Do not combine this argument with another argument.
-v	Displays the *service_name* and the *port_number* for each queued service that is available.

Entering the command without arguments displays all the requests in the queue. For each entry, **queue** displays the *service_name* and the *port_number* requested (if specified), the host requesting the service, the *entry_id* assigned to each queued request, the time (in minutes) that the request has been waiting in the queue, and the request’s position in the queue.

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

```
annex: queue

position                             port  time
in queue host (from)   service (to)  (to)  (min)  entry id
1        vax_marketing lab_printer   10    17     538
3        vax_sales     laser_printer 8     11     384
4        vax_marketing modem_pool    -     8      82
7        annex_lab     modem_test    2     2      611
```

In the previous example, queue positions 2, 5, and 6 are not displayed because they are being used for non-LAT requests.

The following example shows a display using **queue -h** *host_name*:

```
annex: queue -h vax_marketing

position                                     port   time
in queue host (from)    service (to)   (to)   (min)  entry id
1        vax_marketing  lab_printer    10     17     538
4        vax_marketing  modem_pool     -      8      82
```

The following example shows a display using **queue -r** *entry_id*:

```
annex: queue -r 538
Entry 538: removed
```

The following example shows a display using **queue -v**:

```
annex: queue -v

Service Name         Ports
TERMINAL             2, 3
WPVAX                3, 4, 12
No entries found.
```

rlogin

The **rlogin** command connects to the specified host using the **rlogin** protocol. The syntax is:

rlogin *host* [**-l** *user_name*]

The **-l** *user_name* argument logs you into the remote host under that *user_name*; otherwise, it sends the port’s *user_name* or prompts for *user_name*.

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

```
annex: rlogin slowpo
login:
```

rm

The superuser **rm** command deletes one or more files in the local file system. The syntax is:

rm *filename* ...

The RAC reports an error if a specified file does not exist, and continues with the next file name in the list.

route

The superuser **route** command adds routes to and deletes routes from the RIP route cache and active routing table.

The gateway address specified in the **route** command must be the *remote_address* of the PPP or SLIP link, not the *local_address*.

Table 1-28 describes the arguments for **route**. The syntax is:

route [**-fF**] **add** [**-s**] *dest mask gateway* [*metric*]

route [**-fF**] **add default** *gateway* [*metric*]

route [**-fF**] **delete** [**default** | *dest*]

Added routes are either *temporary* or *hardwired*.

- A temporary route does not age but a RIP route can replace it.
- A hardwired route does not age and a RIP route cannot replace it.

If you are using **telnet** to connect to the RAC, deleting the route leading to the host to which you are connected breaks the connection.

For more details on routing, see the document *Managing Remote Access Concentrators Using Command Line Interfaces*.

Table 1-28. Arguments for the Superuser route Command

Argument	Description
-f	Flushes the temporary routes from the routing table and route cache.
-F	Flushes the hardwired and interface routes from the routing table and route cache. An interface route is a route to a network directly connected to the RAC. RAC RIP automatically enters these routes into the routing cache and table.
-fF	Flushes all routes from the routing table and cache.
add	Adds a route to the route cache. It also adds the route to the routing table if the *gateway* argument specifies an address that is directly reachable on an active interface.
-s	Specifies a hardwired route that RIP cannot replace.
default	Specifies the default route. In general, using **route** to add or delete a default route can have unpredictable results. The only time you can safely use **route** to add a default route is when a default route is not defined in the configuration file and the RAC is not receiving RIP updates.
dest	Specifies the destination address of the route.
mask	Specifies the subnet mask to apply to the destination address.
gateway	Specifies the IP address of the gateway (router) that is to be the next hop for the route. This address must be on a network directly attached to the RAC.
metric	Specifies the number of hops to the destination. Values range from **1** through **15**; the default is **1**.
delete	Deletes a route (temporary, hardwired, or interface) from the route cache and the routing table.

services

The **services** command displays information about available LAT services that have been advertised by LAT hosts. The format of this display depends on the arguments and information that you supply on the command line. Table 1-29 describes the arguments for **services**; Table 1-30 describes the command display. The syntax is:

services [**-vh**] [*service_name* [*host_name*]]

Entering the **services** command without arguments displays a summary of available LAT services on the network. Available services are restricted by group codes (see *group_value on page 3-35*). The summary typically includes the *service name*, *status*, and *service identification*.

Table 1-29. Arguments for the services Command

Argument	Description
-v	Displays the expanded view of LAT services on the network. The expanded view includes the *service name*, *rating*, *service identification*, *host name*, *host identification*, *host status*, and *facility number* of the advertising host. If multiple services have the same name, the summary includes only the service with the highest rating.
service_name	Displays a summary of all services having that *service name*, regardless of the service's rating. This summary displays the *host name* field of the requested service rather than the *service name* field.
service_name *host_name*	Displays a summary of the service having that *service_name* on the host specified by *host_name*.

(continued on next page)

Table 1-29. Arguments for the services Command (continued)

Argument	Description
-h	Displays a summary of all available LAT services by host.
-vh	Displays the expanded view of all available LAT services by host.
-h *host_name*	Displays a summary of all the services available from the host specified by *host_name*.
-vh *host_name*	Displays the expanded view of all services available for the host specified by *host_name*.

If multiple services have the same name, the summary includes only the service of the highest rating. For example:

```
annex: services

Local Server Name : ALPHA

Service Name        Host Status         Service Id
TERMINAL            Reachable           LAT server
DA08                Unreachable         LAT server
LAT_00802D0018B6    Reachable           Modem
WPVAX               Reachable           SYS$ANNOUNCE
```

The following shows sample output from **services -v**:

```
annex: services -v terminal

Local Server Name: ALPHA

Service Name : TERMINAL
Service Id   : LAT server      Rating     : 9
Host Name    : WPVAX           Host Id    : 3f
Host Status  : Reachable       Facility # : 0
```

The following shows sample output from **services -h**:

```
annex: services -h wpvax

Local Server Name : ALPHA

Host Name : WPVAX

   Service Name        Host Status         Service Id
   TERMINAL            Reachable           LAT server
```

The following shows sample output from **services -vh**:

```
annex: services -vh wpvax

Local Server Name: ALPHA

Host Name    : WPVAX                       Host Id      : 3f
Host Status  : Reachable                   Facility #   : 0

   Service Name : TERMINAL
   Service Id   : LAT server               Rating       : 9
```

Table 1-30. The services Command Display

Field	Definition
Service Name	The name of the service being offered.
Service Identification	The service’s use or purpose.
Rating	An integer typically indicating the number of resources (ports) available for the service on the indicated host.
Host Name	The name of the host offering the service.

(continued on next page)

Table 1-30. The services Command Display (continued)

Field	Definition
Host Identification	The host’s location or other special characteristics of the host offering the service.
Host Status	The status of the host offering the service.
Facility Number	An integer indicating the facility number or the host number of the host offering the service.

slip

The **slip** command allows a user at a remote host to dial into a modem attached to the RAC and convert the CLI port to a SLIP interface. Resetting the port returns it to CLI mode. The syntax is:

slip

The command display looks like this:

```
annex: slip
Username: ellis
Password:
Switching to SLIP.
Remote Annex address is 132.245.254.65. Your address is
132.254.6.90.
```

If you execute the **slip** command, and the port is not configured properly for SLIP, the terminal displays *This port cannot be used for SLIP* (for more details on using a SLIP link, see the document *Managing Remote Access Concentrators Using Command Line Interfaces*).

You cannot use the minimum uniqueness feature with the **slip** command.

stats

The **stats** command displays RAC statistics. Table 1-31 describes the arguments for **stats**. The syntax is:

stats [**-csm** [*ports*][*time*] | [**-tpoT**]]

Table 1-31. Arguments for the stats Command

Argument	Description
-c	Clears all serial line statistics.
-s	Displays statistics for all serial ports. You can enter a single port (**-s5**) or a range of ports (**-s5-10** or **-s5,7,9-12**). This argument also displays control line status in which an asserted signal appears in uppercase letters and a de-asserted signal appears in lower case letters.
-s *time*	Displays serial line statistics, pausing between each display the number of seconds specified by *time*. Entering the attention character, attention string, or the **Break** key aborts the display.
-m	Displays statistics for active control lines much the same as **-s**, but displays the modem controls for inactive control lines (that is, unattached slave lines) rather than displaying *idle*.
-m *time*	Displays statistics for both active and inactive control lines, pausing between each display the number of seconds specified by *time*.
-o	Displays the status of the RAC keyed options and disabled modules.

(continued on next page)

Table 1-31. Arguments for the stats Command (continued)

Argument	Description
-p	Displays parallel printer statistics for all connected printers, including the printer type, number of characters transmitted, and printer status. You can specify the printer by number.
-t	Displays statistics for the SNMP trap table.
-T	Displays statistical information about the WAN network interface.

If you specify a time interval, the RAC ignores an attention string that contains multiple characters.

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

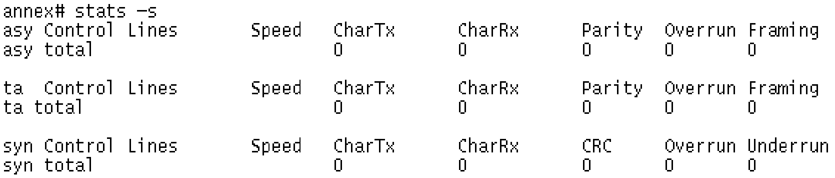
```
annex: stats
S/W: Remote Access I14.0.12         Build #1: Sat Mar 22 00:26:16 EST 1997
H/W: 5399/Turbo, MLB Rev 132.4      ROM Rev: 110C
Ports: eth 2wan 48mod 64syn/ta      Clock Source: Interface 1
Memory: 8MB RAM 64KB EE 256KB SLC1 256KB SLC2 2MB FLSH
Boot from: 132.245.33.7             Date: Mon Mar 24 14:18:40 1997 EST
Image: oper/oper.64.enet/I14.0.12   Uptime: 3 hours 37 min.
Inet addr: 132.245.11.106           Subnet mask: 255.255.255.0
Ethernet addr: 00-80-2d-05-58-2e    Broadcast addr: 132.245.11.255
Primary NS: 132.245.33.7            Domain: xylogics.com
QUICC Ver: 130
IPX Frame Type: EthernetII          IPX Network Number: 11
Apple: Node 6121.139  Router 6100.22Zone: macip1
CPU Load: cur 0%, avg 0%            SLC CPU: slot 1/c 0%, slot 2/c 0%
Procs:    current 43, max 50, limit 1280
Tasking:  rescheds 0/0, switches 176/269783, activates 176/269809
Mbufs:    total 9000, free 8792, min free 8742
Memory:   total 8MB, avail 6.2MB, free 2.3MB, min 2.1MB

Port type   Receive  Transmit  R Frames  T Frames  R Errors  T Errors
   asy         0 by      0 by         0         0         0         0
   syn         0 by      0 by         0         0         0         0
   ta          0 by      0 by         0         0         0         0
   ctl       23.7KB    36.4KB         0         0         0         0
annex:
```

If the RAC is operating without IP protocols (that is, with the **inet_addr** parameter is set to 255.255.255.255), the **stats** command omits the *Inet addr, Subnet mask,* and *Broadcast addr* fields.

If the RAC is operating without IP on the Ethernet interface, but IP protocols are enabled (that is, the **subnet_mask** parameter is set to 255.255.2555.255), the **stats** command shows <*unused*> in place of the subnet mask value and omits the *Broadcast addr* field.

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

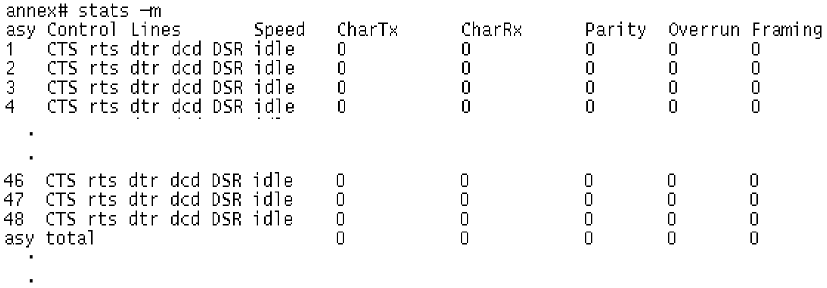


```
annex# stats -s
asy Control Lines       Speed   CharTx      CharRx      Parity  Overrun Framing
asy total                       0           0           0       0       0

ta  Control Lines       Speed   CharTx      CharRx      Parity  Overrun Framing
ta total                        0           0           0       0       0

syn Control Lines       Speed   CharTx      CharRx      CRC     Overrun Underrun
syn total                       0           0           0       0       0
```

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



```
annex# stats -m
asy Control Lines       Speed   CharTx      CharRx      Parity  Overrun Framing
1   CTS rts dtr dcd DSR idle    0           0           0       0       0
2   CTS rts dtr dcd DSR idle    0           0           0       0       0
3   CTS rts dtr dcd DSR idle    0           0           0       0       0
4   CTS rts dtr dcd DSR idle    0           0           0       0       0
  .
  .
46  CTS rts dtr dcd DSR idle    0           0           0       0       0
47  CTS rts dtr dcd DSR idle    0           0           0       0       0
48  CTS rts dtr dcd DSR idle    0           0           0       0       0
asy total                       0           0           0       0       0
  .
  .
```

The **stats -o** command displays the status of the keyed options and the disabled modules:

```
annex01# stats -o
KEYED OPTIONS:

 LAT:                      keyed on but disabled by disabled_modules
 Atalk:                    keyed off
 tn3270:                   keyed off
 dialout/RIP/filtering:    keyed off
 IPX:                      keyed on but disabled by loader

MODULES DISABLED:
     atalk, dialout, ipx, lat, tn3270, vci
```

In the previous command display, IPX is disabled by both the **disabled_modules** parameter and by the loader. In this case, the loader takes precedence; removing IPX from **disabled_modules** will not enable it.

stats -c

The superuser **stats -c** command clears the serial line statistics to 0. You can enter a single port (**-c5**) or a range of ports (**-c5-7**).

stats -T

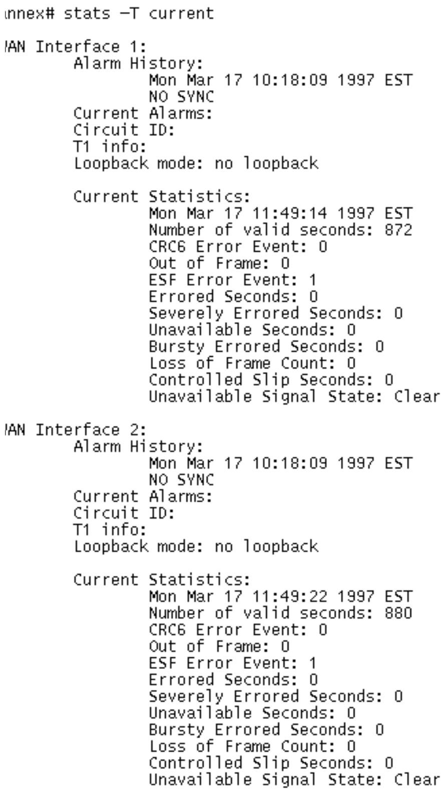
The superuser **stats -T** command displays an extensive set of CAS/PRI line statistics, alarm indications, and other line performance monitoring information. This information is obtained through the RAC implementation of the Facilities Data Link (FDL) protocol and from the PRI module. Table 1-32 describes the arguments for **stats -T**. The syntax is as follows:

annex: **stats -T** {**clear**|**current**|**total**|**all**}

Table 1-32. Arguments for the stats -T Command

Argument	Description
current	Displays CAS/PRI statistics information for the current 15-minute interval.
total	Displays a summary of CAS/PRI statistics information for the last 24 hours. This option is available only if the **wan** *fdl_type* parameter is set to **att**.
all	Displays the CAS/PRI statistics for each valid interval. There are up to 96 intervals (15 minutes per interval) for a 24-hour period. The most recently recorded interval is displayed as the highest-numbered one. This option is available only if the **fdl_type** parameter is set to **att**. See *fdl_type on page 3-33*.
clear	Clears the alarm condition saved in the history buffer. Also clears the previous 24-hour statistics. Once the alarm history buffer is cleared, the next alarm event is captured and stored in the history buffer. Subsequent alarm events will not overwrite the history buffer.

Table 1-33 describes the fields that **stats -T** displays. The following is a sample display:



```
nnex# stats -T current

AN Interface 1:
       Alarm History:
               Mon Mar 17 10:18:09 1997 EST
               NO SYNC
       Current Alarms:
       Circuit ID:
       T1 info:
       Loopback mode: no loopback

       Current Statistics:
               Mon Mar 17 11:49:14 1997 EST
               Number of valid seconds: 872
               CRC6 Error Event: 0
               Out of Frame: 0
               ESF Error Event: 1
               Errored Seconds: 0
               Severely Errored Seconds: 0
               Unavailable Seconds: 0
               Bursty Errored Seconds: 0
               Loss of Frame Count: 0
               Controlled Slip Seconds: 0
               Unavailable Signal State: Clear

AN Interface 2:
       Alarm History:
               Mon Mar 17 10:18:09 1997 EST
               NO SYNC
       Current Alarms:
       Circuit ID:
       T1 info:
       Loopback mode: no loopback

       Current Statistics:
               Mon Mar 17 11:49:22 1997 EST
               Number of valid seconds: 880
               CRC6 Error Event: 0
               Out of Frame: 0
               ESF Error Event: 1
               Errored Seconds: 0
               Severely Errored Seconds: 0
               Unavailable Seconds: 0
               Bursty Errored Seconds: 0
               Loss of Frame Count: 0
               Controlled Slip Seconds: 0
               Unavailable Signal State: Clear
```

Table 1-33. Fields in the stats -T Command Display

Field	Description
Alarm History	Displays the first alarm that occurred after a boot or a **stats -T clear** command, preceded by the time at which the alarm occurred.
Current Alarms	When NO SYNC is displayed, the CAS/PRI engine has lost frame synchronization and enters the Red Alarm state. When nothing is displayed for Sync, the CAS/PRI engine is detecting frame synchronization.
	When LOSS OF SIGNAL is displayed, the CAS/PRI engine is not detecting pulses on the CAS/PRI network interface receiver. When *LOSS OF SIGNAL* is not displayed, the CAS/PRI network interface is receiving pulses. The loss of signal condition causes the CAS/PRI engine to transmit AIS (all ones unframed) on the network interface.
	When BLUE is displayed, the CAS/PRI engine is receiving AIS (all ones unframed) from the network. When *BLUE* is not displayed, the CAS/PRI engine is not receiving AIS.
	When RED is displayed, the CAS/PRI engine has detected loss of frame synchronization. During this condition, the CAS/PRI interface is sending a Yellow Alarm to the network. When *RED* is not displayed, the CAS/PRI engine is in frame synchronization.
	When YELLOW is displayed, the CAS/PRI engine is receiving a Yellow Alarm from the network. When *YELLOW* is not displayed, the CAS engine is not receiving a Yellow Alarm from the network.

(continued on next page)

Table 1-33. Fields in the stats -T Command Display (continued)

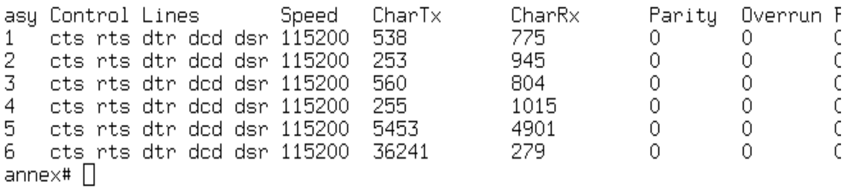
Field	Description
Circuit ID	The CAS/PRI engine’s Circuit ID displayed from the **tni_circuit_id** parameter.
Loopback mode	The loopback status has the following possible states: **Loopback** -- Loopback is in progress. **No Loopback** -- There is no loopback in progress. The test LED on the front panel is not illuminated.
Number of Valid Seconds	Part of the **current** report that indicates the number of seconds for which statistics data has been collected.
CRC6 Error Event	A CRC6 error occurs when the six-bit CRC field calculated by the customer installation, based on the incoming DS1 signal, does not agree with the CRC field contained in the DS1 signal received from the network.
Out of Frame	An Out of Frame (OOF) event begins when any two of four consecutive frame synchronizing bits are received from the network interface. An OOF state ends when reframe occurs.
ESF Error Event	An Extended Superframe CRC error.
Errored Seconds	An Errored Second is a second with one or more ESF error events, that is, one or more CRC error events or one or more Out of Frames (OOFs).

(continued on next page)

Table 1-33. Fields in the stats -T Command Display (Continued)

Field	Description
Severely Errored Seconds	A Severely Errored Second is a second with 320 or more CRC error events or one or more OOFs.
Unavailable Seconds	Unavailable Seconds is a count of 1-second intervals during which service is unavailable.
Bursty Errored Seconds	A Bursty Errored Second (BES) is a second with more than one, but less than 320, CRC error events.
Loss of Frame Count	Loss of Frame Count is the number of times that frame synchronization has been lost.
Controlled Slip Seconds	A Controlled Slip Second is a second with one or more Controlled Slips. A Controlled Slip is the occurrence of a replication or deletion of a DS1 frame by the receiving terminal.

The following sample display reports statistics for the first six asynchronous ports:



```
asy Control Lines       Speed   CharTx      CharRx      Parity  Overrun F
1   cts rts dtr dcd dsr 115200  538         775         0       0       (
2   cts rts dtr dcd dsr 115200  253         945         0       0       (
3   cts rts dtr dcd dsr 115200  560         804         0       0       (
4   cts rts dtr dcd dsr 115200  255         1015        0       0       (
5   cts rts dtr dcd dsr 115200  5453        4901        0       0       (
6   cts rts dtr dcd dsr 115200  36241       279         0       0       (
annex# 
```

stty

Use the **stty** command to display and change port parameters that control terminal characteristics, CLI connection options, and special characters. The syntax is:

stty [*parameter* [*value*]]

Table 1-34 describes the parameters that can be set using the **stty** command.

Parameter changes are valid only for the current CLI session.

The **stty** command displays the current parameter settings:

```
annex: stty
Annex port pts3 (virtual CLI port 3):

-newlin echo -ilower -olower crtcera crtlera tabs
cera     lera     wera     ldisp    flush    tesc
[^?]     [^U]     [^W]     [^R]     [^O]     [^]]
user: rsmith, prompt: %a%c
break lbreak -ixany broadcast climask7
iflow: eia, oflow: eia, isize: 1, fwdtimer: 5, term: XTERM attn: ^A
rows: 22
baud: 9600, bchar: 8, stopb: 1, parity: none, control_lines: both
-imask7
signals: dcd+ dsr+ cts+
annex:
```

You can modify these parameters using the **stty** command. Rebooting the RAC, resetting the port, or issuing a **hangup** command resets the parameters to their original values. Depending on the type of parameter, you enter a new value as follows:

- Enter a minus sign (-) before the parameter name to indicate off, or enter only the parameter name to indicate on. For example, entering **break** defines the **Break** key as an attention signal; entering **-break** indicates the key is not defined as an attention signal.
- Enter one value from a list of available values. For example, the **baud** parameter requires that you select from a list of numeric values.
- Enter the parameter name followed by a single character, which is often a control character, such as the **lera** (line erase) argument followed by CTRL-U. (The **stty** command displays this value as ^U.) Specify control characters in one of two ways:
    - Enter a circumflex (^) followed by the desired character. For or example, enter **^ c** to indicate CTRL-C.
    - Enter the special character. For example, hold down the **Control** key while you type **C**.
- Enter a string, such as the string **martha** for the **user** parameter.

To *undefine* (turn off) a parameter that requires a value, enter one of the following values along with the parameter:

- **undef** (or **u**)
- **none** (or **n**)
- The two characters ^ and **@** (indicating a null string) for parameters that require control characters
- Quotation marks ("") for the parameters **prompt**, **term**, and **user** (this is the only way to turn off these parameters)

Table 1-34. Setting Parameters Using stty

Parameter	Description
attn *string*	Defines a control character sequence as an attention character or string. Entering this character or string returns you to the CLI prompt. Make sure the selected control character sequence is not one that is used with any host applications. If neither **break** nor **lbreak** are defined, and you do not define an attention character or string, you must log off the host to return to the CLI prompt. The default for a serial connection is a null string (""); the default for a virtual CLI connection is CTRL-A.
back *string*	Defines a control character sequence as a backward switch character or string. Entering this character or string reopens the next lower numbered session, already established at your port, from within the current session without returning to local mode. Make sure the selected control character sequence is not one that is used with any host applications. To clear an existing switch, enter a null string (""). On virtual ports, the switch is limited to one character; on physical ports, the switch string can contain from 1-16 characters.
baud *value*	Sets the serial line speed for input and output. Possible values are **75**, **110**, **134.5**, **150**, **200**, **300**, **600**, **1200**, **1800**, **2000**, **2400**, **3600**, **4800**, **7200**, **9600**, **19200**, **38400**, **57600**, and **115200**. The default is **9600**.
bchar *value*	Defines the data bits per character, not including the start bit, stop bits, or the parity bit (if any). Possible values are **5**, **6**, **7**, or **8**. The default is **8**.
break -break	Defines a break as an attention signal. Generally, the break is generated by a key labeled **Break**. Specifying **-break** turns off **Break** as an attention signal.

(continued on next page)

Table 1-34. Setting Parameters Using stty (continued)

Parameter	Description
broadcast -broadcast	Enables the terminal to display RAC administrative messages. Specifying **-broadcast** prevents any display of administrative messages. The default is **broadcast**.
cera *character*	Sets the character-erase character. The default is **Delete** (CTRL-?).
climask7 -climask7	Masks CLI input to seven bits. When climask7 is disabled (**-climask7**), 8-bit ASCII input is expected.
crt	Sets the following defaults for video terminals: **crtcera**, **crtlera**, **tabs**, **echo**, **-newlin**, **-ilower**, and **-olower**.
crtcera -crtcera	Controls how both the character erase (cera) and the word erase (wera) are echoed. Specifying **crtcera** echoes the erase characters in an appropriate way for a video terminal; the previous character (or word) appears as if it has been erased. The default is crtcera. Specifying **-crtcera** echoes the erase characters in a way appropriate for a hardcopy terminal. The first erase character is echoed as a backslash (\ ) followed by the deleted character. Each additional use of the erase character deletes and displays another character. The first character typed (other than the erase character) echoes a backslash (/) and the character; typing asdf<*Delete Delete*>g is echoed as asdf\fd/g.
crtlera -crtlera	Controls how the line erase (**lera**) is echoed. Setting **crtlera** erases all characters on the line and moves the cursor back to the beginning of the line. Setting **-crtlera** echoes the line erase character in a way that is suitable for hardcopy terminals. It echoes the erase character and a new line, so that the deleted line is still visible but the print head is positioned at the beginning of the next line. The default is **crtlera**.

(continued on next page)

Table 1-34. Setting Parameters Using stty (continued)

Parameter	Description
echo -echo	Controls the echoing of characters as they are typed. Setting **-echo** turns off echoing. The default is **echo**.
flush *character*	Sets the flush character. When pressed, this character flushes your output buffer. The default is CTRL-O.
forw *string*	Defines a control character sequence as a forward switch character or string. Entering this character or string reopens the next available higher numbered session, already established at your port. Make sure the selected control character sequence is not one that is used with any host applications. To clear an existing switch, enter a null string (""). On virtual ports, the switch is limited to one character; on physical ports, the switch string contain up to 16 characters.
fwdtimer *time*	Defines the time in hundredths of a second for the forwarding timer. This timer causes data read from a serial port and stored in a buffer to be forwarded to a host when it expires.
iflow *argument*	Specifies the method the RAC uses to stop input from the terminal if the RAC's input buffer is about to overflow. The default is **bell**. Possible values are:
none	Specifies no flow control; characters are lost if the buffers overflow.
eia	Selects hardware flow control.
xonoff	Specifies XON/XOFF flow control. The RAC sends XOFF when its buffers are nearly full and sends XON when the buffer level reaches a safe level. This is known as in-band flow control.
bell	The RAC rings the terminal bell when its input buffer is full.

(continued on next page)

Table 1-34. Setting Parameters Using stty (continued)

Parameter	Description
ilower -ilower	Controls case conversion for characters sent from the terminal to the RAC. Use **ilower** for older terminals without lowercase characters. The RAC converts typed uppercase characters to lowercase. The default, **-ilower**, does not change case.
imask7 -imask7	Enables clearing the eighth bit of received characters. This parameter has no effect on transmitted characters. When the **data_bits** parameter is set to **5**, **6**, or **7**, **imask7** has no effect. Specifying **-imask7** does not clear the eighth bit. The default is **-imask7**.
ixany -ixany	Specifies that typing any character restarts the output to the screen that was stopped with the XOFF character. The default is **-ixany**.
lbreak -lbreak	Defines a long break as an attention signal; **-lbreak** turns off long break as an attention signal. The default is **lbreak**.
ldisp *character*	Sets the reprint line character. The default is CTRL-R.
lera *character*	Sets the line erase character. The default is CTRL-U.
newlin -newlin	Defines the terminal as a new line terminal, which sends a carriage return followed by a line feed when Return is pressed and displays a carriage return followed by a line feed when a line feed character is received. The default is **-newlin**.

(continued on next page)

Table 1-34. Setting Parameters Using stty (continued)

Parameter	Description
oflow *argument*	Specifies the terminal’s method for stopping output from the RAC. The default is **xonoff**. Possible values are:
none	No flow control; characters are lost if the buffers overflow.
eia	Selects hardware flow control.
xonoff	Specifies XON/XOFF flow control. When the RAC receives XOFF, it stops sending output to the terminal; when it receives XON, it starts sending output again. Setting **ostopc** to CTRL-S for XOFF and **ostartc** to CTRL-Q for XON allows you to type these characters to stop and start output before data scrolls off the screen.
both	Specifies both XON/XOFF and CTS/RTS flow control. Both flow controls are independent. Data flows out of the port only if CTS is high and the last received XON/XOFF character was XON. Receiving XOFF or dropping CTS stops output flow from the RAC to the device.
bell	Has the same effect as setting the parameter to **none**.
olower -olower	Controls case conversion for characters sent from the RAC to the terminal. Specifying **olower** converts lowercase characters to uppercase. The default, **-olower**, does not change case.

(continued on next page)

Table 1-34. Setting Parameters Using stty (continued)

Parameter	Description
ostartc *character*	Sets the restart output character. The default is CTRL-Q.
ostopc *character*	Sets the stop output character. The default is CTRL-S.
parity *argument*	Type of parity checked on input and generated on output. Possible values are **none**, **even**, or **odd**. The default is **none**.
prompt *code*	Changes the CLI prompt. The prompt is specified as alphanumeric characters and embedded formatting codes that are expanded when the prompt is displayed. The formatting codes consist of a percent symbol (%) followed by a single lowercase character. The default is %a%c (*annex:*). The formatting codes are:
%a	The string *annex*.
%c	A colon followed by a space.
%d	The current date and time in standard UNIX format, such as Mon Mar 14 13:59:42 1997.
%i %j	The RAC’s IP address. A new line character; skip to the beginning of the next line.
%l	The location defined for the port or the string *port nn*.
%n	The RAC’s name or IP address.
%p	The port number.

(continued on next page)

Table 1-34. Setting Parameters Using stty (continued)

Parameter	Description
%r	The string *port*.
%s	A space.
%t	The current time in 24-hour format, such as 13:59:42.
%u	The user name defined for the port or a null string.
stopb *argument*	Number of stop bits. Possible values are: 1, 1.5, and 2. The default is 1.
tabs -tabs	Specifies replacing output tab characters with spaces. If the terminal does not have hardware tab support, use **-tabs**. The default is **tabs**.
term *string*	Identifies the type of terminal using the CLI connection. The value should be a valid terminal type for the host.
tesc *character*	Sets the Telnet escape character. This character returns you to the *telnet* prompt when you are in a Telnet session with the remote host. The default is CTRL-].
user *name*	Sets the user name for the connection.
wera *character*	Sets the word erase character. The default is CTRL-W.

su

When entered in superuser mode, the **su** command returns you to the user CLI and requires no password. When entered at the user level, the su command prompts you for a password and, if the password is valid, displays the superuser prompt *(annex#*, by default).

If the RAC is configured with an IP address, the default administrative (**su**) password is the IP address in dotted-decimal notation.

If the RAC is not yet configured with an IP address and the administrative password has not been modified (using the RAC parameter **password** or the CLI **passwd** command), the password is a null string ("").

If the RAC is not configured with an IP address and boots via MOP, IPX, or from flash ROM, the default password is a null string (""), and entering a carriage return at the *Password* prompt places you in superuser mode.

tap

The superuser **tap** command displays input and output on a device attached to a specific port. Any input from the port performing the tap is inserted into the tapped port's input stream. Table 1-35 describes the arguments for **tap**. The syntax is:

tap [**-afksvx**] *port*

Entering the **tap** *port* command without additional arguments displays any output to the port on the terminal. Keystrokes from the terminal are interpreted as if they were typed on the port. After establishing a tap, the terminal displays the message *Warning: This port is being tapped*, unless the **-s** argument was entered. To stop a tap, enter a break command to return to the CLI prompt, and then issue the CLI **kill** for the **tap** job.

The **tap** command creates a RAC job in the same way as the **telnet** and **rlogin** commands. You can enter a break to return to the CLI prompt and you can then execute other CLI commands. However, when **tap** is not the active job, all activity on the tapped port is suspended.

Flow control on the tapping port affects the tapped port. Suspending output on the tapping port also stops output on the tapped port.

The **-k** and **-v** arguments allow you to use **tap** as a limited software line monitor. You can monitor traffic in both directions, including incoming special conditions (such as line breaks, flow control, and characters with special interpretations).

The **who** command displays a tap on a port only when it is invoked locally in superuser mode.

Table 1-35. Arguments for the Superuser tap Command

Argument	Description
-a	Use ANSI enhanced display mode escape sequences instead of angle brackets for highlighting all input displayed by **-k**. For printing characters, the escape sequence 033 133 061 155 octal (ESC [1 m) is output instead of <; the escape sequence 033 133 060 155 (ESC [0 m) is output instead of >. Both the escape sequences and the angle brackets are output for special characters. These escape sequences can begin and end a graphic rendition such as bold or reverse video.
-f	This argument forces a tap to occur. It also allows the user to tap across several calls on one internal port.
-k	Displays input and output data, but the order of the displayed data may not match the actual time sequence of the events. Special characters and control line changes are stored in a limited buffer; if they occur too rapidly, some may be lost. Angle brackets distinguish input from output. Additional information also appears in angle brackets:
	Control characters: <^J> for line feed, <^I> for tab, and so on.
	Special characters: the characters defined as special for the tapped port, such as flow control or attention characters, display as <spcl ^S>, <spcl ^Q>, and so on.
	Line breaks: <break> and <break end>.
	Control line state changes: <rts-> and <rts+>, <dcd-> and <dcd+>, and so on.
-s	Do not display a warning message after establishing the tap. This option is useful when the device connected to the port is not a terminal, and the message might interfere with its normal operation.

(continued on next page)

Table 1-35. Arguments for the Superuser tap Command (continued)

Argument	Description
-v	Display output from the tapped port in verbose mode. Control codes (000 to 037 octal) display as ^*X*. DEL (177octal) are displayed as ^**?**. Codes greater than 177 octal are displayed with **M-** preceding their 7-bit representation. For example, 012, the code for line feed, displays as ^J; 212 displays as M-^J.
-x	Display hexadecimal codes for all characters. Use this option with either **-k** or **-v**.
port	The number of the port to be tapped. You can specify **asy1** through **asy***n* and **ta1** through **ta***n*, where *n* is the total number of asy or ta ports. You can also specify **ctl1** (for CLI on the console port), ctl2 (for WAN interface 1) and **clt3** (for WAN interface 2).

telnet

The **telnet** command establishes a **telnet** connection between two ports on two machines. Table 1-36 describes the arguments for this command; Table 1-37 describes the commands that you can execute using **telnet**. The syntax is:

telnet [**-lrst**] [*host* [*port*]]

Table 1-36. Arguments for the telnet Command

Argument	Description
-l	In *character-at-a-time* mode, if neither side negotiates for echo, **telnet -l** directs the RAC to send a LF character to the terminal for each CR received. Do not execute a **telnet -l** command if **stty echo** is turned on.
-r	Requests *raw* mode. In raw mode, **telnet** passes data between the terminal and a TCP connection in line-by-line mode.

(continued on next page)

Table 1-36. Arguments for the telnet Command (continued)

Argument	Description
-s	Requests *silent* mode. Prevents the RAC connection from sending progress or termination messages unless an unexpected error occurs. Use **-s** in combination with **-r** and **-t**, or alone; it is most useful in macros to hide the Telnet interface from the user.
-t	Opens a *transparent* TCP connection to the specified port in which the Telnet protocol is not used. The Telnet escape character is ignored.
host	Opens a connection to that host.
port	Allows you to enter a TCP port number on the specified host to which **telnet** makes a connection.

If you enter the command without arguments, **telnet** enters command mode and displays the Telnet prompt. After connecting to the remote host, **telnet** prints a message called a connect banner, which displays the Telnet escape character. If pressed, the Telnet escape character returns you to the *telnet* prompt, where you can enter **telnet** commands. For example:

```
annex: telnet topsy
Trying...
Connected to topsy
Escape character is “^]”.
4.3 BSD UNIX (topsy)
login:
```

After establishing the connection, **telnet** is in input mode. Input mode supports either *character-at-a-time* or *line-by-line* mode. In character-at-a-time mode, the RAC immediately sends each typed character to the host, where character echoing and line editing occur.

In line-by-line mode, the RAC retains input until you press either **Return** or an interrupt character. In this mode, character echoing and line editing are performed locally at the RAC. Using the **telnet mode** command, you can change both the mode and where echoing occurs.

You can send a Break to the remote host by using either the regular break or the long break key. This allows you to send a Break sequence using a local break, rather than using the Telnet **send brk** command. To do this, you must turn off the regular and/or long break as the CLI attention character (see *stty on page 1-88*).

If a foreign port is not specified, **telnet** defaults to port 23. The local port is chosen to represent the user location as follows:

- If the user connects to the RAC via a serial port through a modem or a terminal, the local port is chosen as 10000 + *port**100 + *sequential*, where *port* is the serial line number (1 to 99), and *sequential* is a number (0 to 99) that distinguishes connections and is chosen sequentially.
- If the user connects to the RAC via the network by using **telnet** *racname*, the local port is chosen as 10000 + *sequential*, where *sequential* is a number (0 to 99) that distinguishes connections and is chosen sequentially. If all local ports from 10000 to 10099 are in use, a random unused port in the range 20000 to 29999 is chosen.

Table 1-37. Commands You Can Enter Using telnet

Command	Description
close	Closes a Telnet session and returns to the CLI prompt.
display	Displays the state of the toggle arguments and the definitions of the special characters.
mode *type* [*echo*]	Specifies the input mode. The value for *type* is specified as line for line-by-line and character for character-at-a-time. The *echo* argument specifies whether echoing is performed by the RAC (local_echo) or by the host (remote_echo). The defaults for echo are remote_echo for character mode and local_echo for line mode.
open [-rt] *host* [*port*]	Opens a connection to the specified host. You can enter either the host's name or its Internet address. If a port is not specified, Telnet connects to the default Telnet port (23). The **-r** argument turns off all Telnet protocol interpretation; **-t** opens a transparent TCP connection to the specified port (you must specify the port number).
quit	Closes any open Telnet session and exits Telnet.
status	Displays the current status of Telnet.

(continued on next page)

Table 1-37. Commands You Can Enter Using telnet (continued)

Command	Description
send [*arguments*]	Sends one or more special character sequences to the remote host. Valid arguments are:
ao	Sends the Telnet *Abort Output* sequence, causing the remote host to stop sending output to your terminal.
ayt	Sends a Telnet *Are You There* sequence.
brk	Sends a Telnet *Break* sequence.
ec	Sends a Telnet *Erase Character* sequence.
el	Sends a Telnet *Erase Line* sequence.
escape	Sends the current Telnet escape character.
ga	Sends a Telnet *Go Ahead* sequence.
ip	Sends a Telnet *Interrupt Process* sequence.
nop	Sends a Telnet *No Operation* sequence.
synch	Sends a Telnet *Synch* sequence.
?	Displays help information for the **send** command.
set [*special character*]	Sets the Telnet special characters. (The *special characters* that **set** accepts can also be designated by their *stty* counterparts (aliases). For example, specify *escape* as **set escape ^]** or **set esc ^]**.) Setting the value for a *special character* to **U** turns off that character's function. Use the **display** command to display the *special character* assignments. *Special characters* are:
eof	Sets the **eof** character to be sent to the host (if Telnet is operating in line-by-line mode).
erase	Sets the erase character that, when entered, sends the **send ec** command (if the Telnet session is in **localchars** mode and in character-at-a-time mode). The initial value is taken from the **stty cera** character.

(continued on next page)

Table 1-37. Commands You Can Enter Using telnet (continued)

Command	Description
escape	Sets the Telnet escape character used to enter command mode. The initial value is taken from the **stty tesc** character.
flushoutput	Sets the **flushoutput** character that, when entered, sends the **send ao** command (if the Telnet session is in **localchars** mode). The initial value is CTRL-O.
interrupt	Sets the interrupt character that, when entered, sends the **send ip** command (if the Telnet session is in **localchars** mode). The initial value is CTRL-C.
kill	Sets the **line erase** character that, when entered, sends the **send el** command (if the Telnet session is in **localchars** mode and in character-at-a-time mode). The initial value is taken from the **stty lera** character.
quit	Sets the **quit** character that, when entered, sends the **send brk** command (if the Telnet session is in **localchars** mode). The initial value is CTRL-\.
?	Displays help for the **set** command.
toggle *argument*	Toggles (turns on and off) arguments that control how **telnet** responds to events. The **display** command displays the current value. Arguments are:
binary	Toggles **telnet** binary mode. Using binary mode eliminates some translation errors that can occur with carriage return and line feed characters. The initial setting is not to use binary mode.

(continued on next page)

Table 1-37. Commands You Can Enter Using telnet (continued)

Command	Description
crlf	Toggles carriage return-line feed mode. When this argument is specified, a carriage return received from the serial port is encoded as a Telnet protocol carriage return-line feed end-of-line sequence. When the argument is not specified, a carriage return received is encoded as a Telnet protocol carriage return-null carriage return sequence. Additionally, when using the **-r** flag, enabling this mode causes the Telnet encoder to send a carriage return-line feed for a received carriage return and to delete a subsequent line feed from the input stream if it is the next received character. When disabled, all characters received are sent without this translation.
crmod	Toggles carriage return mode. When enabled, most carriage return characters received from the host are translated into a carriage return followed by a line feed. This mode is used when the host sends only a carriage return without a line feed. The initial value disables carriage return mode.
localchars	Toggles the local recognition of Telnet special characters. When enabled, the special characters are recognized by the RAC and are mapped into appropriate Telnet control sequences. The initial setting is *will map* when the mode is line-by-line, and *won't map* when the mode is character-at-a-time. You can toggle **localchars** between *will map* for both modes, *won't map* for both modes, and the initial setting of *will*/*won't map* based on mode.

(continued on next page)

Table 1-37. Commands You Can Enter Using telnet (continued)

Command	Description
options	Toggles the displaying of internal Telnet protocol processing. The initial value is no display.
?	Displays help information for the **toggle** command.
? [*command*]	Displays help information. Without arguments, **telnet** prints a help summary. If you specify a command, **telnet** prints help information for that command.

tn3270

The **tn3270** command is a variation of Telnet that allows you to log on to an IBM host from an ASCII terminal attached to the RAC. The IBM host to which you connect can be running either the Virtual Machine/ Conversational Monitor System (VM/CMS) or the Multiple Virtual Systems (MVS).

While you are connected to an IBM host, your ASCII terminal emulates an IBM 3278 (Model 2) full-screen terminal. This is the only member of the IBM 3270 family of terminals that the RAC **tn3270** command supports. Table 1-38 describes the arguments for **tn3270**. The syntax is:

tn3270 [*host* [*port*]]

Table 1-38. Arguments for the tn3270 Command

Argument	Description
host	Opens a connection to *host*. Specify *host* as either an IP address (in dotted-decimal notation) or a host name. If you use a host name, it must be defined in the RAC's host table or available from the RAC's name server. See *hosts (Superuser Level) on page 1-45* and *name_server_1 on page 3-60*.
port	Specifies the number of the protocol port on *host* to which **tn3270** connects. The default is the standard Telnet port (23).

This product supports only the U.S. ASCII character set.

After opening a connection to a host, **tn3270** displays a connect banner. The remote host then displays its own connect banner and prompts you to log on:

```
annex: tn3270 132.345.254.3
Trying...
Connected to 132.245.254.3
Escape char is “^]”

VIRTUAL MACHINE/CMS

                     VM/CMS

Fill in your USERID and PASSWORD and press Enter
USERID ===>
PASSWORD ===>
```

In the previous example, note the line that displays the escape character. Use this character to enter **tn3270** command mode from within a logon session. To change this escape character, execute the CLI **stty** command with the **tesc** argument before executing the **tn3270** command.

A second escape character is defined in the **map3270** file. You can use this escape character instead of the one that appears when a connection is opened.

Entering the **tn3270** command puts the RAC in **tn3270** command mode and displays the *tn3270* prompt:

```
annex: tn3270
tn3270:
```

In **tn3270** command mode, you can execute the commands shown in Table 1-41.

ASCII Terminal Requirements and Setup

An ASCII terminal connected to the RAC must have the following characteristics to use **tn3270**:

- Cursor addressing and movement. If a terminal does not have this feature, **tn3270** denies access to the IBM host and displays the error message *Terminal must have cursor addressing capability.*
- A screen size of at least 24 lines and 80 columns. If the screen is smaller, **tn3270** denies access to the IBM host and displays the error message *Terminal must have at least 24 lines and 80 columns*.

tn3270 uses the standard UNIX file **termcap** to determine whether or not a particular terminal type has these features. The network administrator must make **termcap** accessible to **tn3270** and ensure that the terminals using **tn3270** have terminal types listed in this file (see *Configuration Checklist on page 1-119*).

Print Screen and Transparent Mode

The RAC **tn3270** has two features not available with the Berkeley version of **tn3270** on which it is based. These are:

- The IBM print-screen function
- The ability to turn transparent mode on or off

Transparent mode is useful for running file transfer programs such as Kermit. The command you use to turn transparent mode on or off depends on the IBM host. This feature requires no special configuration.

The print-screen feature lets you dump a screen from the IBM host session to a printer. To do this, enter the RAC key sequence that is mapped to the IBM **LPRT** key.

The network administrator must configure the print-screen feature. This includes mapping a RAC key sequence to the IBM **LPRT** key (see *Terminal Emulation on page 1-111* and *Configuration Checklist on page 1-119*). It also involves setting the **na/admin** port parameters **printer_name** and **printer_host**. These parameters have user level equivalents, **printer** and **printhost**, that can be set using the CLI **stty** command. Table 1-39 describes both sets of parameters.

Table 1-39. Print-Screen Parameters Set Using stty and na/admin

stty	na/admin	Explanation
printer *name*	printer_name *name*	Specifies the name of the printer to which screen dumps are to be sent. This name must be listed in the **/etc/printcap** file on the host specified as **printhost** or **printer_host**.
printhost *ip_addr*	printer_host *ip_addr*	Specifies the IP address of a host running a Berkeley-style line printer daemon (**lpd**) server and configured to accept print requests from the RAC.

Terminal Emulation

To make an ASCII terminal emulate a 3278 terminal, **tn3270** simulates the special 3278 keys. For example, a 3278 terminal has a key labeled EEOF that erases the contents of the current field from the location of the cursor to the end of the field. An ASCII terminal does not have this key.

To simulate the 3278 keys, **tn3270** maps them to keys sequences you can enter from an ASCII terminal. The key sequences **tn3270** uses depend on the RAC terminal type and are defined in the standard UNIX file **/etc/map3270**. So that **tn3270** can access the file, you must copy it into (or create a link to it in) the directory on the load host that contains the RAC operational image (see *Configuration Checklist on page 1-119*).

Each entry in a **map3270** file begins with the names of the terminal types to which it applies. A vertical bar (|) separates one type from another. Following the terminal types are the key-sequence definitions, grouped by function.

Figure 1-3 shows part of a **map3270** entry that applies to six terminal types (-*avt*, *vt100*, and so on). Each definition begins with a reserved name, such as *enter*, identifying the IBM key or function. Not all key names listed in a **map3270** file refer to actual keys on a 3278 keyboard (Table 1-40 describes the key names).

```
avt | vt100 | vt100nam | pt100 | vt125 | vt102
enter = ’^m’;
clear = ’^z’ | ’\EOM’;

#for tn3270 print-screen function
lprt = ’\Ep’;

nl = ’^?’;
tab = ’^i’;
btab = ’^b’;
left = ’^h’ | ’\E[D’;
right = ’^l’ | ’\E[C’;
```

Figure 1-3. Portion of a Sample map3270 File

The reserved name in a **map3270** definition is followed by an equal sign (=) and one or more ASCII key sequences, each of which is enclosed in single quotation marks. Within a **map3270** ASCII sequence, a caret (^) introduces a control character (or a control-character sequence), and ESC is represented as backslash E (\E).

If more than one sequence can be used for the same IBM function, a vertical bar separates the sequences. For example, in the **map3270** file shown in Figure 1-3, the IBM **clear** function is defined as either of the following sequences: ^z or \EOM.

From an ASCII terminal in **tn3270** emulation mode, you enter key sequences as follows:

- To enter a character that does not print (that is, a character preceded by ^ in **map3270**), hold down the **CTRL** key while you type the character. For example, to enter ‘^z’, hold down the **CTRL** key while typing **z**. If there is a second character in the key sequence, hold down the **CTRL** key, type the first character, release the **CTRL** key, and then type the second character. For example, to enter the sequence ‘^pp’, hold down the **CTRL** key while typing **p**, then release **CTRL** and type another **p**. CTRL sequences are not case-sensitive; ‘^p’ and ‘^P’ generate the same ASCII code.

  The notation ‘^^’ indicates that you press the **CTRL** key while you enter a caret (^).

- To enter an escape character (that is, a character preceded by \E in **map3270**), press the **ESC** key, release it, and then enter each character in the sequence. For example, to enter the three-character sequence ‘\EOM’, press the **ESC** key, release it, and then type an uppercase **O** followed by an uppercase **M** (ESC sequences are case-sensitive).

For more information about how to set up a VT100-type terminal to use special keys, see *Configuration Checklist on page 1-119*.

You will almost never need to type a three-character escape sequence, because most of these sequences are mapped to special keys such as the those on the numeric keypad. For example, on a VT220 terminal that has *Keypad* mode set to *Application*, you can send the sequence \EOM by pressing the **Enter** key.

If **map3270** does not contain an entry for the ASCII terminal you are using, or if there is no **map3270** file, **tn3270** uses the defaults shown in Table 1-40. This table:

- Separates key-sequence choices with the word “or”
- Uses CRTL- and ESC instead of ^ and \E
- Does not enclose key sequences in single quotes

For more information, see the *Berkeley UNIX* manual pages for **map3270**.

Table 1-40. Default Key Mappings for tn3270

IBM 3270 Key Name	ASCII Key Sequence	Description
Command Keys		
ENTER	CTRL-m	Enter
CLEAR	CTRL-z	Clear screen
LPRT	CTRL-p p or CTRL-p P	Print screen
Cursor Movement Keys		
NL	CTRL-n	New line
TAB	CTRL-i	Tab
BTAB	CTRL-b	Back tab

(continued on next page)

Table 1-40. Default Key Mappings for tn3270 (continued)

IBM 3270 Key Name	ASCII Key Sequence	Description
Cursor Movement Keys (continued)		
LEFT	CTRL-h or ESC D or ESC [D or ESC O D	Cursor left
RIGHT	CTRL-l or ESC C or ESC [C or ESC O D	Cursor right
UP	CTRL-k or ESC A or ESC [A or ESC O A	Cursor up
DOWN	CTRL-j or ESC B or ESC [B or ESC O B	Cursor down
HOME	ESC h	Cursor home
Editing Keys		
DELETE	CRTL-d	Delete character
EEOF	CTRL-e	Erase end of field
EINP	CTRL-w	Erase input
INSRT	ESC<space>	Insert
Program Function Keys		
PFK1 - PFK9	ESC 1 - ESC 9 or ESC [1 - ESC [9 or ESC O 1 - ESC O 9	PF1 - PF9 keys
PFK10	ESC 0 or ESC [0 or ESC O 0	PF10 key
PFK11	ESC - or ESC [- or ESC O -	PF11 key
PFK12	ESC = or ESC [= or ESC O =	PF12 key
PFK13	ESC ! or ESC [! or ESC O !	PF13 key
PFK14	ESC @ or ESC [@ or ESC O @	PF14 key
PFK15	ESC # or ESC [# or ESC O #	PF15 key

(continued on next page)

Table 1-40. Default Key Mappings for tn3270 (continued)

IBM 3270 Key Name	ASCII Key Sequence	Description
PFK16	ESC $ or ESC [$ or ESC O $	PF16 key
PFK17	ESC % or ESC [% or ESC O %	PF17 key
PFK18	ESC ^ or ESC [^ or ESC O ^	PF18 key
PFK19	ESC & or ESC [& or ESC O &	PF19 key
PFK20	ESC * or ESC [* or ESC O *	PF20 key
PFK21	ESC (or ESC [(or ESC O (	PF21 key
PFK22	ESC) or ESC [ ) or ESC O)	PF22 key
PFK23	ESC _ or ESC [_ or ESC O _	PF23 key
PFK24	ESC + or ESC [+ or ESC O +	PF24 key
Program Attention Keys		
PA1	CTRL-p 1	PA1 key
PA2	CTRL-p 2	PA2 key
PA3	CTRL-p 3	PA3 key
Local Control Keys		
ESCAPE	CTRL-c	Telnet escape
FLINP	CTRL-x	Flush input
MASTER_RESET	CTRL-g	Unlock and redisplay
RESHOW	CTRL-v	Redraw screen
RESET	CTRL-t	Unlock keyboard

(continued on next page)

Table 1-40. Default Key Mappings for tn3270 (continued)

IBM 3270 Key Name	ASCII Key Sequence	Description
DP	ESC d or ESC [d or ESC O d	Duplication character
FM	ESC f or ESC [f or ESC O f	Field mark character
FERASE	CTRL-u	Field erase
SYNCH	CTRL-r	Synchronize with user
TREQ	CTRL-a	Test request
XOFF	CTRL-s	Suspend output to screen
XON	CTRL-q	Resume output to screen

tn3270 Command Mode

Table 1-41 describes the commands you can issue in **tn3270** command mode. Enter this mode in one of two ways:

- By entering the escape character while you are connected to a host (this suspends the host session)
- By issuing the CLI **tn3270** command with no arguments

The **tn3270** commands are a subset of the **telnet** commands (see Table 1-36).

Table 1-41. Commands Used in tn3270 Command Mode

Command	Description
close	Closes the connection to the remote host and returns you to the CLI prompt. On the RAC, this method of ending a connection is equivalent to using **quit** (see *Ending a tn3270 Session on page 1-119*).
open [*host* [*port*]]	Opens a connection to *host*. Specify *host* as either an IP address (in dotted-decimal notation) or a host name. If you use a name, it must be defined in the RAC's host table or available from the RAC's name server. If you do not specify a host, **tn3270** prompts you for it by displaying "to:". If you do not specify a port, **tn3270** connects to the default Telnet port (23). **Note:** If your RAC is already connected to a host via **tn3270**, you cannot use **open** to make a tn3270 connection to another host. To do that, enter the CLI attention string (defined using the CLI **stty** command), and then re-invoke **tn3270** from the CLI.
quit	Closes the connection to the remote host and returns you to the CLI prompt. On the RAC, **quit** is equivalent to **close** (see *Ending a tn3270 Session on page 1-119*).
status	Displays the status of the current **tn3270** connection.
<CR>	Entering a carriage return (and nothing else) at the *tn3270* prompt resumes the suspended session. (You suspend a session by entering the escape character.)
? [*command*]	Displays information about one or all of the commands described in this table. If you specify *command*, **?** displays information about that particular command. Issued without an argument, **?** displays summaries of all commands.

Ending a tn3270 Session

To end a **tn3270** session, execute the CMS or MVS **LOGOFF** command. This closes the connection in the most orderly fashion and returns you to the CLI command level.

If you cannot issue a **LOGOFF** (for example, because the remote host is not accepting commands), enter the **tn3270** escape character and then execute a **close** or **quit** command. Either of these commands closes the connection and returns you to the CLI prompt.

Configuration Checklist

To configure your network to use **tn3270** for the first time:

1. **Create one or more user accounts on the IBM host(s) to be accessed.**
2. **Verify that the IBM hosts allow** telnet-tn3270 **access.**
3. **Use** na **or** admin **to set the RAC port parameter** term_var **to the appropriate terminal type for the global RAC port.**
4. **If your RAC boots from a load host that has the standard UNIX files** /etc/termcap **and** /etc/map3270**, copy those files into the directory that contains the RAC's operational image.**

   If the RAC uses **erpcd** to boot, the operational image is in the load-host directory **/usr/spool/erpcd/bfs**. If the RAC uses **tftp** to boot, the image is in the **tftp** directory.

   When **tn3270** is invoked, it queries the load host for **map3270** and **termcap**. If the host cannot supply the files, **tn3270** searches for the files in the manner specified by the RAC's **load_dump_sequence** parameter.

During this process, **tn3270** broadcasts for **map3270** and **termcap** - even if the RAC **load_broadcast** parameter is set to **N**.

If **tn3270** follows the configured load-dump sequence but still cannot find **map3270** and **termcap**, it uses default, compiled versions of the files that are built into the RAC's operational image. The default **map3270** file is shown in Table 1-40. The default **termcap** file contains definitions for the following terminal types: VT100; WYSE Models 75 and 85; and IBM 3151. (For **term_var**, specify these as **vt100**, **wy75**, **wy85**, and **ibm3151**. See Step 3.)

Another RAC cannot act as a boot-server for **map3270** and **termcap** because a boot server returns only files in its cache, and these two files are not cached.

5. **Configure each terminal you specified in Step 3 to transmit 7-bit (rather than 8-bit) control codes, to match the 7-bit codes in** termcap **and** map3270**.**

   If your terminal has a keypad, you may also want to configure the terminal to take advantage of the **map3270** keypad mappings. For example, if you are using a RAC **term_var** of **vt100** but the terminal has a numeric keypad (which an actual VT100 does not), you can configure the terminal as *VT200 7bit, VT300 7bit,* or *VT400 7bit, (*which do support keypads). The model you select depends on the capabilities of the terminal. Consult the programmer's manual for the terminal being emulated. For example, if you are configuring a VT100 to emulate a VT220, use the *VT220 DEC Programmer's Reference Manual*.

Use the following procedure to perform the two functions just described:

- Use the terminal’s setup utility or edit the **termcap** file to set the terminal’s control codes to 7 bits and (optionally) to specify a terminal model number that supports a numeric keypad.

  VT100 emulation mode always uses 7 bits, although not all setup utilities indicate that it does.

  To access the setup utility, press the appropriate key (typically labeled *Setup*). Then choose the option that lets you select a value for the emulation mode parameter. (On VT200s, VT300s, and VT400s, the option is labeled *General,* and the parameter you select is *Mode*. For *Mode*, select the value *VT400 7 bit*, *VT300 7 bit*, or *VT200 7 bit.)*

  Instead of using the setup utility, you can set the emulation mode by editing the **termcap** file.

  To set the emulation mode to VT200 7 bit, enter the following string in an *is* control sequence in the portion of **termcap** that corresponds to the terminal’s **term_var** port parameter:

  ```
  \E[62;1”p
  ```

  To set emulation mode to VT300 7 bit, enter:

  ```
  \E[63;1”p
  ```

  Do not confuse control sequence length with data bits (which can also be set to 7 or 8 using the setup utility). The latter is a hardware parameter that specifies the number of bits per character the terminal transmits.

  The emulation of one ASCII terminal by another should not be confused with the emulation of an EBCDIC, IBM 3278 terminal by an ASCII terminal running tn3270.

- In the section of **termcap** that corresponds to the **term_var** port parameter for the terminal, enter the string `=\E=` in an *is* control sequence to enable Keypad Application mode when a user invokes **tn3270**. This allows the user to enter escape sequences by pressing the single keypad keys listed in **map3270**.

  Because **tn3270** does not support the *rs* control sequence, the keypad remains in Application mode when the user ends a host session.

6. **If you plan to use the IBM print-screen function:**

   - Use **na** or **admin** to set the RAC port parameters **printer_host**, **printer_name**, and **user_name** for each port that is to use **tn3270**. Set the first two parameters as explained in Table 1-39 and make sure that **user_name** is set to a non-null value.
   - Configure the UNIX host specified by **printer_host** to allow print requests from the RAC. See the manual pages for the host's line-printer daemon (**lpd**).
   - If your RAC obtains **map3270** from a host that is running Berkeley UNIX, modify **map3270** to include the ASCII key sequence for the IBM print-screen key (**LPRT**). Because the print-screen function is not supported by Berkeley **tn3270**, LPRT is not included in Berkeley **map3270** files.

7. **Make sure that the RAC's name is either listed in the RAC's host table or available from the RAC's name server; see *hosts (Superuser Level) on page 1-45*, *name_server_1 on page 3-60*, and *name_server_2 on page 3-60*).**

8. **Use the** admin **command** reset **to reset the port you have configured (see *admin on page 1-2* for more details).**

wan

The superuser **wan** command displays information about and controls certain functions of the WAN interfaces. The syntax is:

wan [*argument*]

When issued with no arguments, the **wan** command displays statistics for both WAN interfaces.

Table 1-42. Arguments for the wan Command

Argument	Description
interface *number*	Establishes a default WAN interface or interfaces for subsequent **wan** commands. Valid values for *number* are: **1** **2** **1,2** **1-2** This argument can also be specified as **interface=***number* in the **wan** command line (for example, **wan interface=1** or **wan interface=1,2**).
loopback	Places the specified WAN interfaces in loopback mode for testing.
b ds0 channel	Each of these three arguments displays information about all active calls on the specified WAN interfaces. Information displayed includes call setup data, SPB and internal port names associated with calls, and session durations.

(continued on next page)

Table 1-42. Arguments for the wan Command (continued)

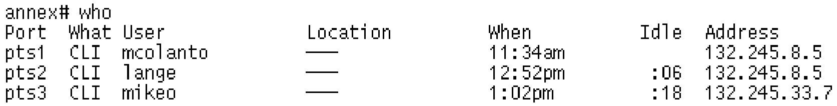
Argument	Description
calls *action direction*	Controls whether calls are accepted or rejected on the specified WAN interfaces: *action* must be **allow** or **stop**; *direction* must be **incoming**, **outgoing**, or **all**.
busyout [channel=*channel_no*]	Forces the specified WAN interface(s) to return a busy signal to the telco Central Office for the specified channel numbers (*channel_no*). Valid values for *channel_no* are **1** through the total number of B or DS0 channels, a range of channel numbers separated by a hyphen, or a series of channel numbers separated by commas. An example is: **wan interface=1 busyout channel=1-9** Issued without the **channel** sub-argument, **busyout** displays a list of the channels currently busied out. Calls remain busied out until you reboot the RAC or use the **unbusyout** argument.
unbusyout [channel=*channel_no*]	Stops the RAC from sending busy signals to the telco Central Office for the specified channel numbers. Valid channel numbers are the same as those allowed for the **busyout** *channel_no* argument.

who

The **who** command displays information about current RAC users. This command also displays current users on other RACs, and on remote hosts, if those hosts have **fingerd** running for **who** *@host*. The syntax is:

who [[**h=**]*host* | [**u=**]*user* |[**p=**]*port* | *@host* | *user@host* | **-l** *@host*]

If you enter the command without arguments, **who** displays a list of all RAC users:



```
annex# who
Port  What User          Location        When          Idle  Address
pts1  CLI  mcolanto      ——              11:34am             132.245.8.5
pts2  CLI  lange         ——              12:52pm        :06  132.245.8.5
pts3  CLI  mikeo         ——              1:02pm         :18  132.245.33.7
```

Table 1-43 describes the arguments for this command; Table 1-44 describes the information that the **who** command displays.

Table 1-43. Arguments for the who Command

Argument	Description
[h=]*host*	A single host or multiple hosts displayed in the *Address* column, or with a job entry in the *User* column. Abbreviating a host name displays any host whose name can be expanded. For example, specifying **bo** selects **borneo**, **bolo**, and **bonzo**. To avoid potential ambiguity with other strings (such as a user name), enter the host name with the type. For example, entering **h=bo** avoids displaying information for user **bobby**.
[u=]*user*	A single user or group of users with entries appearing in the *User* column. Using an abbreviated user name displays any expandable user name. For example, specifying **k** selects both **kevin** and **kathryn**.

(continued on next page)

Table 1-43. Arguments for the who Command (continued)

Argument	Description
[p=]*port*	A specific port with an entry in the *Port* column; for example, **2** indicates Port 2, and **v2** displays Virtual Port 2. If you enter **v**, all virtual CLI ports are displayed. Optionally, you can enter the type; for example, **p=2** or **p=v2**.
@*host*	All users at the specified host. If *host* is a 4.3BSD system, the display is the **finger** command. If *host* is a RAC, the display is the **who** command.
user@*host*	A specific user at the specified host. If *host* is a 4.3BSD system, the display is the same as the **finger** *user* command. If *host* is a RAC, the display is the same as the **who** *user* command.
-l @*host*	All users at the specified host. If *host* is a 4.3BSD system, the display is the same as the **finger -l** command. If *host* is a RAC, the display is the same as the **who** command.

Table 1-44. who Command Display

Field	Description
Port	The number for the serial port; **v***nn* indicates a VCLI connection.
What	One of the following connection types:
CLI	The port is defined as a CLI port and was opened by a device connected to the port or by a connection from a host as a virtual CLI.
LPD	The port is defined as a RAC line printer daemon (**lpd**).
PSVR	The port is defined as a slave port and was opened via a network connection to the port server.
DP	The port is defined as dedicated.
PPP	The port is defined as a PPP interface.
SLIP	The port is defined as a SLIP interface.
ARAP	The port is defined as an ARAP interface.

(continued on next page)

Table 1-44. The who Command Display (continued)

User	Displays each user's name and current jobs. Three dashes indicate that no name is defined.
Location	Displays a location defined for the port.
When	Displays the time that the port was opened from the RAC time-of-day clock.
Idle	Displays the amount of time (hours and minutes) since the last activity on the port.
Address	Displays the source of the connection. The name or network address indicates the host or RAC originating the connection; *local* indicates a serial port.

write

The **write** command allows the user to enter a text file into the RAC file system.

Chapter 2
na Utility

The **na** utility provides subcommands for managing the RAC. These subcommands allow you to:

- Set and display the operating characteristics of the RAC.
- Reboot or reset the RAC, reset internal ports, and reset sessions.
- Broadcast administrative messages to RAC users.

The **na** utility is stored on and accessed from a UNIX, Windows 95, or Windows NT host.

The RAC stores the parameters set using **na** in nonvolatile memory (EEPROM). After a reboot or a reset, the RAC updates its run-time parameters with the EEPROM parameters changed using **na**. The **na** utility can communicate with the RAC only when the RAC is running its operational code.

You can run **na** interactively or provide it with input through a file or pipeline. You can create a script file containing **na** subcommands to configure a RAC. This script file can save the configuration information for a specific RAC and, when required, restore the configuration.

This chapter shows subcommand names, parameter names, and keywords in their long forms. Examples of **na** subcommands sometimes appear without the interactive subcommand prompt, and with embedded comments that describe the functions being performed. This format resembles the appearance of **na** scripts; the portion of the script entered at the terminal in response to the subcommand prompt appears in bold type.

Subcommand Notation

Interactive **na** sessions allow you to enter **na** subcommands with or without arguments or parameters. If you enter the subcommand without arguments or parameters, **na** prompts for them. The conventions for an interactive session are:

- You can abbreviate subcommands and parameter names to the minimum number of characters that uniquely distinguish the name from any other name that could appear in the same context.
- Type a new-line character to end a subcommand entry. To continue an entry onto the next line, type a backslash (\) character immediately preceding the new-line character.
- To enter a space as an argument, enclose it in quotation marks (" "). Otherwise, the space is assumed to be a delimiter.
- The UNIX interrupt character (usually CTRL-C) returns you to the subcommand prompt.

Additionally, **na** permits comments when the pound (#) character is present at the beginning of a comment line. All characters between the # and the next new line are ignored. Table 2-1 describes the supported arguments for **na**.

Table 2-1. Common Arguments for the na Subcommands

Argument	Description
annex_identifier	A symbolic name or an IP address assigned to a RAC: `lab` or `132.245.254.38` or `0xC0.0x9.0xC8.0x64`
annex_list	A list of one or more *annex_identifiers* separated by commas: `support,132.245.254.42,lab`

annex_parameters	A list of one or more RAC parameters and values separated by white space (space, tab, new line):
	`pref_load_addr 132.245.254.66\`
	`pref_dump_addr 132.245.254.66`

interface_identifier	Either **en0** or **port**.
interface_parameters	A list of one or more routing interface parameters, with or without values, separated by white space (space, tab, newline): `rip_sub_advertise Y`
interface_set	A list of one or more *interface_identifiers* separated by semicolons. An *interface_set* can include interfaces on different RACs: `en0@132.245.254.42;en0@132.245.44.98`
port	The global port, which includes both WAN interfaces.
port_parameters	A list of one or more global port parameters, with or without values, separated by white space: `input_flow_control eia`
wan_set	The number of a WAN interface (**1** or **2**), or **1,2**, or **all**.

Subcommands

Table 2-2 lists the **na** subcommands; the following sections describe them.

Table 2-2. The na Subcommands

Subcommand	Description
annex	Defines a default *annex_list* used with subsequent subcommands.
boot	Boots the RAC.
broadcast	Sends a broadcast message to one or more users on internal asynchronous ports.
copy	Copies configuration parameters.
dumpboot	Boots the RAC and produces a dump.
echo	Writes the remainder of the line to the standard output.
help (or ?)	Displays help for subcommands and parameters.
interface	Defines a default *interface_set* used with subsequent subcommands.
password	Defines a default administrative password used to communicate with a RAC.
port	Specifies the global port.
quit	Terminates **na**.
read	Reads and executes a script file.

(continued on next page)

Table 2-2. The na Subcommands (continued)

Subcommand	Description
reset	Resets an internal port, interface, or subsystem.
set	Defines or modifies the value of a parameter.
show	Displays the current value of a parameter.
wan	Defines a default *wan_set* for use with subsequent subcommands.
write	Writes the current configuration to a script file.

After installing **na** on a UNIX host, type **na** at a terminal connected to this host. No arguments or command line options are available.

```
% na
Annex network administrator Rx.x
command:
```

Seven of the **na** subcommands use standard UNIX superuser protection. They are **boot**, **broadcast**, **copy**, **dumpboot**, **read**, **reset**, and **set**. Only a superuser at the host can execute these subcommands.

Although the prompt you see after you issue the **na** command is *command*, this manual uses the term *subcommand* to refer to the commands you issue within **na**.

annex

The **annex** subcommand establishes a default *annex_list* of one or more RACs. This list is used in subsequent **na** subcommands to identify the RACs to which the subcommands apply. By grouping several RACs into a single list, you can then issue one **na** subcommand for the entire group of RACs. The syntax is:

annex *annex_list*

The following example creates an *annex_list* containing one RAC with the Internet address 132.245.6.40:

```
command: annex 132.245.6.40
```

The following example creates an *annex_list* containing two RACs (one specified by its IP address, the other by its name):

```
command: annex 132.245.6.40,frontlobby
```

The following example shows how **na** prompts for missing arguments:

```
command: annex
enter default annex list: 132.245.6.40,frontlobby
```

The following **annex** subcommand displays a message identifying the specified RAC, its Internet address, its software version, and the number of each type of port and WAN interface on it:

```
command: annex 132.245.6.1
132.245.6.1: 132.245.44.98: Annex-RAC Rx.x, 48 async, 64
sync, 64 ta ports, and 2 WAN interfaces
```

The following **annex** subcommand causes a RAC to prompt for an administrative password, provided that the password has been set and security has been enabled:

```
command: annex frontlobby
Password for 132.245.6.40 <frontlobby>
frontlobby:  RAC Rx.x
```

If you use the **password** subcommand to define a default password for this **na** session, and that password matches the RAC administrative password, no password prompt appears and normal processing continues (see *password on page 2-16*).

The password is not echoed when entered using the **annex** subcommand. If you enter an incorrect password, **na** prompts for the correct one. If you enter an incorrect password a second time, **na** drops the RAC from the *annex_list*. If a RAC in the list does not respond, **na** ignores that RAC and prints a status message:

```
132.245.6.1: Not responding
Warning:132.245.6.1 has been dropped from the list
```

The **na** utility drops a RAC from the *annex_list* if its name could not be translated to an Internet address, if it does not respond because it is down, or if the wrong Internet address was entered using the **annex** subcommand.

boot

The **boot** subcommand reboots all RACs in the *annex_list* and, optionally, produces a dump of the RAC's memory, including the operational code. You can set a time at which the boot is to take place. The **boot** subcommand can send a warning message to users attached to the RAC. Table 2-3 lists the supported arguments for the **boot** subcommand. The syntax is:

boot [**-adhlq**] [[+] [*HH*:*MM*]] [*annex_list*] [*filename*] [*warning*]

When the RAC reboots, it terminates all active connections.

If you try to boot a RAC with less than 2 MB of system RAM, the Net, Load, and Status 8 indicators flash until the **Reset** button is pressed.

If you try to boot with a non-existent image file name, the RAC hangs as it searches for the image. You must press the **Reset** button to recover.

Table 2-3. Arguments for the boot Subcommand

Argument	Description
-a	Aborts any delayed boots that are pending.
-d	Causes a dump before rebooting.
-h	Returns to the ROM monitor prompt if the RAC is in Test mode.
-l	Boots the operational image and stores it on local media; for use with the stand-alone file system. Only ROM revisions 0600 and greater with the self-boot option loaded support **-l**. After a **boot -l** is executed, the **ls** command may not show the newly loaded image.
-q	Causes a boot without sending a warning message.
HH:MM	The exact clock time for the boot; for example, 15:15 indicates 3:15 p.m.
+HH:MM	The number of hours and minutes before the boot takes place; for example, +2:15 indicates a boot will occur in two hours and fifteen minutes.
annex_list	Specifies the RACs to be booted. If you do not include an *annex_list*, the subcommand prompts for it. Pressing the **Return** key accepts the default *annex_list* defined by the **na annex** subcommand.
filename	Identifies the name of the RAC image file. If you do not enter a file name, the RAC prompts for one. Pressing the **Return** key at the prompt directs the RAC to boot the default *filename*. This file name is the value of the RAC *image_name* parameter, or, if that is not set, the name **oper.63.enet**.
warning	Allows you to enter a message up to 249 characters long. Warning messages are sent out to users periodically. If you do not specify a time delay or message, the **boot** subcommand generates an automatic warning message.

The following sample **boot** subcommand specifies a boot in one hour and fifteen minutes:

```
command: boot +1:15
annex list (return for default): thirdfloor, 132.245.6.40
filename (return for default): <cr>
warning: Shutting down for PM
```

The RAC can request its boot file from a defined preferred load host. If that host is not defined, or does not respond, the RAC broadcasts its request and boots from the first load host to respond.

broadcast

The **broadcast** subcommand sends a message to modem users on specified internal asynchronous ports on the identified RACs. The syntax is:

broadcast [=*async_port_set* | =*keyword* [**@***annex_identifier*]] *message*

The *async_port_set* argument indicates the numbers of the internal asynchronous ports to which the message is to be broadcast. For example, a port set of 1, 2, 3 specifies internal ports asy1, asy2, and asy3. If the *message* requires more than one line, using the backslash (\) character at the end of each line inserts a new line. Table 2-4 lists the available keywords.

Table 2-4. Supported Keywords for the broadcast Subcommand

Keyword	Description
all	Broadcasts to all asynchronous modem ports and all virtual connections.
virtual	Broadcasts to all virtual CLI connections (you cannot broadcast to a single virtual CLI connection).

copy

The **copy** subcommand requires superuser privileges.

The **copy** subcommand copies a given set of parameters from one RAC (or global port) to another RAC (or global port). Table 2-5 defines each **copy** subcommand. The syntax is:

copy annex *annex_identifier annex_list*

copy interface *interface_name***@***annex_identifier interface_set*

copy port@*annex_identifier*

Table 2-5. Descriptions of the copy Subcommand

Subcommand	Description
copy annex	Copies from the RAC specified by *annex_identifier* to *annex_list* all annex-type parameters except the IP address, the administrative password, the access control protocol key, LAT key, option key, and the virtual CLI password.
copy interface	Copies to *interface_set* all interface parameters from the interface specified by *interface_name*.
copy port	Copies all global port parameters except the port password from the current RAC to the RAC specified by *annex_identifier*.

For example, to copy global port parameters from the current RAC to the RAC at 132.245.6.55:

```
command: copy port@132.245.6.55
```

dumpboot

The **dumpboot** subcommand requires superuser privileges. When the RAC dumpboots, it terminates all active connections.

The **dumpboot** subcommand performs a dump of every RAC specified in the *annex_list* and then reboots the RAC. You can set the boot time, and the **dumpboot** subcommand sends a warning message to users attached to the RAC. Table 2-6 describes the arguments for **dumpboot**. The syntax is:

dumpboot [**-aq**] [[+] [*HH*:*MM*]] [*annex_list*] [*filename*] [*warning*]

The following is an example of the **dumpboot** subcommand:

```
command: dumpboot
annex list (return for default): backhall
filename (return for default): <cr>
warning: Diagnostic testing
```

The RAC sends the dump to a defined preferred dump host. If that host is not defined or does not respond, the RAC broadcasts its dump request and dumps to the first host that responds.

Table 2-6. Arguments for the dumpboot Subcommand

Argument	Description
-a	Aborts any delayed dump boots that are pending.
-q	Performs a boot without sending a warning message.
HH:MM	The exact clock time for the dump boot; for example, 15:15 indicates 3:15 p.m.
+HH:MM	The number of hours and minutes before the boot takes place; for example, +2:15 indicates a boot will occur in two hours and fifteen minutes.
annex_list	Specifies the RACs for which dumps and boots are to be performed. If you do not include *annex_list*, the subcommand prompts for it. Pressing the **Return** key accepts the default *annex_list.*
filename	Identifies the name of the RAC image file. If you do not enter a file name, the RAC prompts for one. Pressing the **Return** key at the prompt directs the RAC to boot the default file name. The RAC requests the boot file from a preferred load host if it is defined and available; otherwise, it broadcasts a boot request.
warning	Allows you to enter a message up to 249 characters long. Warning messages are sent out to users periodically. If you do not specify a time delay or message, the **dumpboot** subcommand generates an automatic warning message.

echo

The **echo** subcommand is used in script files to write its argument to standard output. The **write** subcommand automatically puts **echo** subcommands in the script file it writes. For more information about the **write** subcommand, see *write on page 2-25*. The syntax for **echo** is:

echo *message*

help

The **help** (or **?**) subcommand displays online help information about **na**. Entering **help** without arguments displays a list of **na** subcommands. Table 2-7 defines the arguments for **help**. The syntax is:

help [*subcommand_name* | *parameter_name* | ***** | **syntax**]

Table 2-7. Arguments for the help Subcommand

subcommand_name	Displays the subcommand syntax, along with a description of the subcommand and its arguments.
parameter_name	Displays the legal values for that parameter.
*	Displays available information for all subcommands and parameters.
syntax	Displays the syntax for all subcommands.

For example, to get help for the **boot** subcommand:

```
command: help boot
boot           Syntax: boot [-adlq][[+][HH:][MM]]\
                       [<filename>] [<warning>
```

To get help for the **timezone_minuteswest** parameter:

```
command: help timezone_minuteswest
timezone_minuteswest (annex parameter):
Minutes west of GMT: an integer
```

Entering **help** followed by the first letter or first few letters of the subcommand or parameter name displays all entries beginning with the string. The following example shows an abbreviated display:

```
command: help t

telnet_escape (serial port parameter):
escape character to use with the telnet command: a character

term_var (serial port parameter):
Terminal variable: a string, maximum sixteen characters

time_broadcast (annex parameter):
broadcast for time server to use if none found:
Y or y to enable; N or n to disable

timezone_minuteswest (annex parameter):
Minutes west of GMT: an integer

toggle_output (serial port parameter):
character used to toggle output: a character
```

interface

The **interface** subcommand establishes a default routing *interface_set* used in subsequent subcommands until another interface is specified. Grouping interfaces using an *interface_set* allows you to issue one **na** subcommand to examine or change the parameter values for multiple interfaces. The syntax is:

interface interface_set [annex_identifier]

For *interface_set*, specify **en0** (for the Ethernet interface), **port** (for the global port), or **all** (for both), followed by *annex_identifier.* If you do not identify a specific RAC by specifying *annex_identifier,* all RACs in the current *annex_list* are used. An *interface_set* referring to the default *annex_list* is updated when a new **annex** subcommand is executed.

This example defines the default *interface_set* as the global port on the RAC whose IP address is 132.254.6.34. Specifying the global port indicates the interface set is for WAN calls of protocol types SLIP and PPP:

```
command: interface port@132.254.6.34
```

This example defines the default *interface_set* as interfaces **en0** on the same RAC, plus **port** on the RAC whose IP address is 132.254.35.120:

```
command: interface en0@132.254.6.34;port@132.254.35.120
```

This example defines the default *interface_set* as the global port on every RAC in the default *annex_ list*:

```
command: interface port
```

password

The **password** subcommand allows you to define a default password for the current **na** session. This subcommand is useful when administering several RACs with the same password. The syntax is:

password [*password*]

If you enter the subcommand without giving the password, the RAC prompts for one, but does not echo it:

```
command: password
password:
```

When accessing a RAC with security enabled using the **annex** subcommand, **na** will try to match the RAC's default password with the administrative password. If they match, access is authorized automatically; if they do not match, **na** prompts for the RAC-specific administrative password. Enter a password for a given RAC only once during an **na** session, even if the default *annex_list* is changed.

quit

The **quit** subcommand terminates the **na** program. The syntax is:

quit

read

The **read** subcommand requires superuser privileges.

The **read** subcommand reads a script file that contains **na** subcommands. The **na** program executes these subcommands as if they were entered at a terminal in interactive mode. Use **read** to restore a RAC configuration that has been lost, or to copy parameter settings from one RAC to another. The syntax is:

read *filename*

You can create script files using a text editor or the **write** subcommand.

If you plan to use LAT, set the **lat_key** parameter manually (using **na** or **admin**) and reboot the RAC before executing the **read** subcommand.

The **read** subcommand loads parameters even if the subsystem is disabled.

The following sample script file, called **testscript**, modifies RAC parameters:

```
# standard parameters for RACs on our network
set annex pref_load_addr 132.245.6.63
set annex pref_dump_addr 132.245.6.63
set annex load_broadcast Y
set annex name_server_1 dns
set annex pref_name1_addr 132.245.6.9
set annex cli_prompt “%n%s%p%c”
set annex daylight_savings usa
set annex enable_security Y
set annex vcli_security Y
set annex syslog_mask all
set annex syslog_host 132.245.6.9
```

Use this script as follows:

```
command: annex thirdfloor,frontlobby,backhall
command: read testscript
```

reset

The **reset** subcommand enables changes you have made to all of the RACs in the default *annex_list* or in the *annex_list* specified with the **reset** subcommand. Unless you use **reset**, changes you make (using the **set** subcommand) usually become effective only after a reboot. Table 2-8 describes the **reset** subcommands.

Resetting a VNP device or an internal (ta, asy, syn, or virtual CLI) port or modem terminates the connection on that device, port, or modem.

Table 2-8. The reset Subcommands

Subcommand	Description
reset all	Resets all serial ports and virtual CLI connections.
reset annex[=*annex_list*] all	Resets the message of the day and Session Parameter Blocks. Also resets the security, name server, LAT, and *syslog* subsystems, as well as customized user interface macros.
reset annex[=*annex_list*] dialout	Resets the dialout subsystem by re-reading the **dialout** section of the RAC configuration file.
reset annex[=*annex_list*] lat	Resets the LAT-specific RAC parameters so that any future LAT circuits (connections) will use the new values; existing circuits will continue to use the old values. This keyword will not terminate existing LAT circuits.
reset annex[=*annex_list*] macros	Re-reads the customized user interface macros.
reset annex[=*annex_list*] motd	Re-reads the message-of-the-day.
reset annex[=*annex_list*]\ nameserver	Resets the name server parameters and flushes the RAC’s host table.

(continued on next page)

Table 2-8.The reset Subcommands (continued)

Subcommand	Description
reset annex[=*annex_list*] security	Resets the security parameters and reconnects to the security host.
reset annex[=*annex_list*] session	Re-reads the Session Parameter Blocks from the configuration file. Existing calls are not reset. No new calls are answered while the reset is in progress.
reset annex[=*annex_list*] syslog	Resets the *syslog* subsystem. The *syslog* subsystem does not use any changes made to the **syslog_port** parameter.
reset *async_port_list* reset async[=*async_port_list*]	Terminates asynchronous connections on the specified ports. For *async_port_list*, specify an individual asynchronous port number, multiple port numbers separated by commas (for example, **1,2,3**), or a range of port numbers. Separate the end points of a range with a hyphen, for example, **1-10**. Any parameter changes that apply to asynchronous calls will be in affect for calls received after the reset. Current (unterminated) calls are not affected.
reset interface[=*interface_set*]	Resets the interface parameters for the routing interfaces -- **en0**, **port** (for the global port), or **all** -- in *interface_set* on the RACs specified in the default *annex_list*.

(continued on next page)

Table 2-8. The reset Subcommands (continued)

Subcommand	Description
reset int_modem [*modem_range*]\ [hard | soft]	Resets each modem in the modem range, whether or not a call is active on the modem. If you specify **hard**, the modem is reset immediately. If you specify **soft** (the default), the modem is reset when the call (if any) on it hangs up. For *modem_range*, specify individual modem numbers separated by commas, or a range of numbers from 1 through the total number of installed modems. Separate the end points of a range with a hyphen, for example, **1-23**.
reset serial	Resets the serial networking parameters for the global port.
reset sync[=*sync_port_list*]	Terminates synchronous connections on the specified ports. For *sync_port_list*, specify an individual synchronous port, (for example, **1**), multiple ports separated by commas (for example, **1,2,3**), or a range of port numbers. Separate the end points of a range with a hyphen (for example, **1-10**). Any parameter changes that apply to synchronous calls will be in affect for calls received after the reset. Current (unterminated) calls are not affected.

(continued on next page)

Table 2-8. The reset Subcommands (continued)

Subcommand	Description
reset virtual	Resets all virtual CLI connections.
reset vpn=*vpn_port_list*	Terminates VPN connections on specified ports, tearing down the L2TP tunnel calls that established them. For *vpn_port_list*, specify an individual VPN port number, multiple port numbers separated by commas (for example, **1,2,3**), or a range of port numbers. Separate the end points of a range with a hyphen (for example, **1-10**). Parameter changes will be in affect only for VPN calls received after the reset. Current calls (that have not been terminated) are not affected.
reset wan[=*wan_set*]	Resets all parameters for the WAN interfaces (**1**,**2**, or **all**) in *wan_set*.

set

The **set** subcommand requires superuser privileges.

The **set** subcommand modifies RAC configuration parameters:

set annex	Modifies RAC parameters
set interface	Modifies interface parameters
set port	Modifies global port parameters
set wan	Modifies WAN parameters
set wan b	Modifies WAN B channel parameters
set wan ds0	Modifies WAN DS0 parameters

The syntax is:

set annex [=*annex_list*] *annex_parameters*

set interface [=*interface_list*] *interface_parameters*

set port *port_parameters*

set wan[=*wan_set*] | *wan_parameters*

set wan[=*wan_set*] **b**[=*channel_range*] *wan_b _parameters*

set wan[=*wan_set*] **ds0**[=*channel_range*] *wan_ ds0_parameters*

interface_set and *wan_set* must be included in the **set** subcommand or defined in an earlier **interface** or **wan** subcommand.

For *interface_set*, enter one of the following:

- **port**, to specify the global port (which includes all channels on both WAN interfaces)
- **en0**, to specify the RAC's Ethernet interface
- **all**, to specify both **en0** and the global port

All parameters require a name and a value separated by a space. A space is required between parameter-value pairs. You can enter more than one parameter with each subcommand. If you are entering multiple parameters that require a new line, precede the new line with the backslash (\) character. Changes made to parameters take effect after booting or resetting the RAC or the global port.

Sample subcommand lines for setting parameters are:

```
command: set port mode cli
command: set wan=1 switch_type at9 framing ESF
```

show

The **show** subcommand displays current RAC, interface, global port, or WAN parameters:

show annex	Displays RAC parameters
show interface	Displays interface parameters
show wan	Displays WAN parameters
show wan b	Displays WAN B channel parameters
show wan ds0	Displays WAN DS0 parameters

The syntax is:

show annex[=*annex_list*] [*keyword* | *annex_parameters*]

show interface[=*interface_list*] [*keyword* | *interface_parameters*]

show port [*keyword* | *port_parameters*]

show wan[**=1|2|all**] [*keyword* | *wan_parameters]*

show wan b[=*range*] *b_channel_parameters*

show wan ds0[=*range*] *ds0_channel_parameters*

show wan[**=1|2|all**] **b|ds0**[=*channel_range*] *b* | *ds0_parameters*

Each keyword in a **show** subcommand displays a subset of parameters. See *Parameters Listed by Type on page 3-5*.

wan

The **wan** subcommand establishes a default *wan_set* used in subsequent subcommands until another *wan_set* is specified. Grouping WAN interfaces using a *wan_set* allows you to issue one **na** subcommand to examine or change the parameter values for multiple WAN interfaces. The syntax is:

wan 1 | **2** | **all**

If you do not identify a specific RAC using the @ symbol and a name or IP address when entering the *wan _set*, all RACs in the current *annex_list* are used. A *wan_set* referring to the default *annex_list* is updated if a new **annex** subcommand is issued. Specifying **all** sets the default *wan_set* to WAN 1 and WAN 2.

To define the default *wan_set* as t WAN 1 on the RAC whose IP address is 132.254.6.34, enter:

```
command: wan 1@132.254.6.34
```

write

The **write** subcommand creates a script file from the configuration data for a specific RAC. You can modify this script file using any text editor. Use the **write** subcommand either to back up the current RAC's configuration or copy it to multiple RACs. After you write a script file, executing the **read** subcommand activates the RAC parameter settings contained in the file (for more details, see *read on page 2-17*). The syntax for **write** is:

write *annex_identifier filename*

For security reasons, the following basic RAC and port parameters are written to the script file as comments: **acp_key**, **lat_key**, **password**, **port_password**, **ppp_password_remote**, **rip_auth**, and **vcli_password**.

Since the **inet_addr** parameter uniquely identifies the RAC's location in the network, it is not written to the script file and it is not restored during a **read**. You must set this parameter manually.

You can remove the pound sign (#) from the parameters written as comments in the script file, enter valid data for their settings, and issue a **read** subcommand to copy or restore these parameters to another RAC.

Entering passwords as plain text in the script file poses a possible security risk for your system. Take appropriate precautions against unauthorized access of this file.

The **write** subcommand writes **set annex**, **set port**, **set interface**, and **set wan** subcommands to the script file for each **annex**, **port**, **interface**, and **wan** parameter. The **write** subcommand also includes **echo** subcommands in the script file. When the script is executed using the **read** subcommand, the arguments to the **echo** subcommand are written to the standard output, indicating the progress of the **read**.

The following is an example of the **write** subcommand:

```
command: write 132.245.6.101 fronthall.script
```

The following example uses the **write** and **read** subcommands to install a new RAC and to create a backup copy of a RAC configuration. The first line writes configuration data for the RAC *thirdfloor* to a file named **thirdfloor.prm**. The data from *thirdfloor* is copied to the new RAC specified in the *annex_list* defined using the **annex** subcommand.

```
command: write thirdfloor.prm
command: annex 132.245.6.40
command: read thirdfloor.prm
```

Following is an excerpt from the script file **fronthall.script**:

```
# annex 132.245.6.101

echo setting annex parameters
set annex pref_load_addr 132.245.6.75
set annex pref_dump_addr 132.245.6.75
set annex load_broadcast Y
set annex image_name “”
set annex subnet_mask 255.255.255.0
set annex authoritative_agent Y
echo setting serial port parameters global port
set port data_bits 8
set port stop_bits 1
set port parity none
set port do_compression N
set port slip_allow_compression Y
set port slip_no_icmp Y
set port slip_tos Y
```

Chapter 3
Configuration Parameters

The configuration parameters define the operating characteristics for a given RAC. The following tools allow you to set and display these parameters:

- The host-based **na** utility, which sends requests to the RAC to **read**, **set**, **reset**, **show**, or **copy** configuration parameters (see *na Utility on page 2-1*).
- The CLI **admin** command, a counterpart of **na**, which runs locally on the RAC (see *admin on page -2*).

This chapter includes the following sections:

- *Entering Parameter Values*
- *Resetting Parameter Values to the Defaults*
- *Parameters Listed by Type*
- *Parameter Descriptions*

Entering Parameter Values

The conventions for entering parameter values depend on the type of information the parameter defines.

- For parameters requiring an IP address, specify the address in dotted decimal notation as a decimal number (**0** through **255**), a hexadecimal number, or a combination of both, for example, 192.9.200.100, 0xC0.0x9.0xC8.0x64, or 192.9.200.0x64.
- For parameters requiring yes/no input, use either **Y** or **N** (uppercase or lowercase).

- For parameters that define passwords, the **na**/**admin** subcommand **show** displays only **"<set>"** or **"<unset>"**; it never displays the values. If you forget a password after setting it, you can reset it only by using the ROM monitor **erase** command to erase all of the RAC's nonvolatile memory.

  > Saving the configuration to a file (using the **write** command) prevents having to reconfigure the RAC if nonvolatile memory is erased.

- Parameters that require a string for input allow a maximum of 16 characters, unless otherwise specified. If you enclose these strings in quotation marks, space characters are allowed.

Resetting Parameters to Defaults

Each configuration parameter, except the RAC's IP address, has a default value. Using the **na** or **admin** subcommand **set**, you can return any parameter to its default setting. The syntax depends on the parameter class: **annex**, **interface**, **port**, or **wan**.

Resetting annex Parameters

- **set annex** *annex_parameter* **0**

  The **set annex** *annex_parameter* **0** subcommand resets parameters that require a numeric value. For example, to set **pref_dump_addr** to its default, 0.0.0.0, enter:

  ```
  command: set annex pref_dump_addr 0
  ```

- **set annex** *annex_parameter* **""**

  The **set annex** *annex_parameter* "" subcommand resets all parameters that require a string value; these parameters default to either a null string ("") or **"<unset>."** For example, to set **image_name** to its default, a null string (""), enter:

  ```
  command: set annex image_name ""
  ```

- **set annex** *annex_parameter* **default**

  The **set annex** *annex_parameter* **default** subcommand resets all other parameters. These parameters are set by choosing an option from a known list. For example, to set **output_ttl** to its default, **64**, enter:

  ```
  command: set annex output_ttl default
  ```

Resetting Port Parameters

Each port parameter applies to all internal ports of the type(s) listed in the parameter description: asynchronous (*asy* and *ta*) and/or synchronous (*syn*).

To reset a global port parameter to its default value, use the **set port** subcommand:

- set port *port_parameter* **^@**

  The **set port** *port_parameter* ^**@** subcommand resets parameters that have single-character default values. Enter the default value as a two-character sequence consisting of the circumflex character (^) followed by the at sign (**@**). For example, to set **erase_word** to its default value, ^W, enter:

  ```
  command: set port erase_word ^@
  ```

- set port *port_parameter* **""**

  The **set port** *port_parameter* "" subcommand resets parameters that require a string; these parameters default to either a null string ("") or **"<unset>."** For example, to set **user_name** to its default, the null string, enter:

  ```
  command: set port user_name ""
  ```

- set port *port_parameter* default

  The **set port** *port_parameter* **default** subcommand resets all parameters that require an option from a known list. For example, to set **forwarding_timer** to its default, **off**, enter:

  ```
  command: set port forwarding_timer default
  ```

Resetting interface Parameters

The **set interface** subcommand resets an interface parameter to its default value. The syntax depends on the type of value (described in *Resetting Port Parameters on page -3*).

- **set interface** *interface_parameter* **^@**
- **set interface** *interface_parameter* **""**
- **set interfac**e *interface_parameter* **default**

Resetting WAN Parameters

The **set wan** subcommand resets a WAN parameter to its default value. The syntax depends on the type of value (described in *Resetting Port Parameters on page -3*).

- **set wan** *wan_parameter* **^@**
- **set wan** *wan_parameter* **""**
- **set wan** *wan_parameter* **default**

Resetting All Parameters

The ROM monitor **erase** command resets all RAC parameters to the defaults (see the *Model 5399 Remote Access Concentrator Hardware Installation Guide* or *Installing the Model 8000 Remote Access Concentrator*). This command erases all parameters, including the RAC's IP address. After executing **erase**, you must re-enter the RAC's IP address and reconfigure the RAC.

Parameters Listed by Type

The RAC configuration parameters are grouped by type: annex, interface, port, and wan. Parameters within these groups are further divided by function. Each function has an associated keyword (for example, nameserver, security, and time). The **na**/**admin show** subcommand accepts these keywords as arguments.

- Table 3-1 lists the keywords and associated parameters that the **show annex** subcommand displays.
- Table 3-2 lists the keywords and associated parameters that the **show interface** subcommand displays.
- Table 3-3 lists the keywords and associated parameters that the **show port** subcommand displays.
- Table 3-4 lists the keywords and associated parameters that the **show wan** subcommand displays.
- Table 3-5 lists the keywords and associated parameters that the **show wan b** (or **show wan ds0**) subcommand displays.

In this chapter, annex parameters are the parameters that you set or show using the **set annex** or **show annex** subcommand. RAC parameters are all of the configuration parameters, including annex, interface, port, and WAN parameters.

If you are using **na**, the LAT-related parameters are visible only when the **lat_key** parameter contains the correct key value (for more details, see *lat_key on page 3-46*).

Table 3-1. Keywords for the show annex Subcommand

Keyword	Parameters
all	Displays all annex parameters.
generic	inet_addr, subnet_mask, pref_load_addr, pref_dump_addr, load_broadcast, broadcast_addr, load_dump_gateway, load_dump_sequence, image_name, motd_file, config_file, authoritative_agent, routed, server_capability, disabled_modules, tftp_load_dir, tftp_dump_name, ipencap_type, ip_forward_broadcast, tcp_keepalive, option_key, session_limit, output_ttl, fail_to_connect, mmp_enabled
vcli	max_vcli, cli_prompt, vcli_security, vcli_password, vcli_inactivity
nameserver	nameserver_broadcast, rwhod, pref_name1_addr, pref_name2_addr, name_server_1, name_server_2, host_table_size, min_unique_hostnames
security	enable_security, security_broadcast, pref_secure1_host, pref_secure2_host, network_turnaround, loose_source_route, acp_key, password, lock_enable, passwd_limit, chap_auth_name, max_chap_chall_int, auth_protocol, enable_radius_acct, radius_auth_port, radius_acct_port, radius_secret, radius_timeout, radius_retries, radius_acct_level, radius_port_encoding
time	time_broadcast, daylight_savings, timezone_minuteswest, time_server
syslog	syslog_mask, syslog_host, syslog_facility, syslog_port

(continued on next page)

Table 3-1. Keywords for the show annex Subcommand (continued)

Keyword	Parameters
mop	pref_mop_host, mop_password, login_prompt, login_password, login_timer
lat	lat_key, facility_num, server_name, sys_location, lat_queue_max, service_limit, keep_alive_timer, circuit_timer, retrans_limit, group_value, vcli_groups, multicast_timer, multisessions_enable
appletalk	a_router, default_zone_list, node_id, zone
router	rip_routers, rip_auth
ipx	ipx_file_server, ipx_frame_type, ipx_dump_username, ipx_dump_password, ipx_dump_path, ipx_do_checksum
tmux	tmux_enable, tmux_max_host, tmux_delay, tmux_max_mpx
dhcp	pref_dhcp1_host, pref_dhcp2_host, dhcp_bcast
snmp	allow_snmp_sets, call_begin_trap, call_end_trap_inc, unexpected_trap_inc, bpv_threshold, oof_threshold, es_threshold, diallink_trap_enable, call_history_limit, cv_threshold, esf_threshold, ses_threshold, uas_threshold, bes_threshold, lofc_threshold, css_threshold, ds0_error_threshold, modem_error_threshold

Table 3-2. Keywords for the show interface Subcommand

Keyword	Description
all	Displays all routing parameters for the specified interface(s): rip_send_version, rip_horizon, rip_next_hop, rip_sub_accept, rip_accept, rep_recv_version, rip_default_route, rip_sub_advertise, rip_advertise.

Table 3-3. Keywords for the show port Subcommand

Keyword	Parameters
all	Displays all asynchronous port parameters.
generic	mode, location, term_var, prompt, cli_interface, speed, data_bits, stop_bits, parity, max_session_count, allow_broadcast, broadcast_direction, imask_7bits, cli_imask7, banner, tcp_keepalive, default_session_mode, dedicated_arguments, resolve_protocol
flow	input_flow_control, input_start_char, input_stop_char, output_flow_control, output_start_char, output_stop_char, ixany_flow_control, need_dsr, v120_mru
timers	forwarding_timer, forwarding_count, cli_inactivity, inactivity_timer, input_is_activity, output_is_activity, reset_idle_time_on, long_break, short_break
security	user_name, cli_security, connect_security, port_server_security, port_password, ipso_class, ipx_security
vci	login_port_password, login_timeout
editing	attn_string, echo, telnet_escape, telnet_crlf, map_to_lower, map_to_upper, char_erase, line_erase, hardware_tabs, erase_char, erase_word, erase_line, redisplay_line, toggle_output, newline_terminal, backward_key, forward_key
serial	local_address, address_origin, metric, slip_ppp_security, net_inactivity, do_compression, allow_compression, net_inactivity_units

(continued on next page)

Table 3-3. Keywords for the show port Subcommand (continued)

Keyword	Parameters
slip	subnet_mask, slip_mtu_size, slip_no_icmp, slip_tos, address_origin, net_inactivity, net_inactivity_units, do_compression, allow_compression, net_inactivity, slip_ppp_security, metric, local_address
ppp	local_address, metric, slip_ppp_security, do_compression, allow_compression, net_inactivity_units, address_origin, ppp_acm, ppp_mru, ppp_security_protocol, ppp_username_remote, ppp_password_remote, ppp_ncp, ppp_ipx_network, ppp_ipx_node, ppp_sec_auto, mp_mrru, mp_endpoint_address, mp_endpoint_class, ipcp_unnumbered
appletalk	at_guest, at_nodeid, at_security, arap_v42bis
tn3270	printer_host, printer_name
lat	authorized_groups, latb_enable, multisessions_enable

Table 3-4. Keywords for the show wan Subcommand

Keyword	Parameters
all	Displays all parameters for the specified WAN interface(s): switch_type, buildout, fdl_type, num_b_channels, analog_encoding, framing, line_code, dnis, ani, digit_width, inter_digit, digit_power_1, digit_power_2 busy_signal_bits, local_phone_number, auto_busyout_enable remote_address, ipx_network, ipx_node, sigproto, ringback

Table 3-5. Keywords for the show wan b or show wan ds0 Subcommand

Keyword	Parameters
all	Shows all parameters for the specified channel(s) on the specified WAN interface(s): remote_address, ipx_network, ipx_node, sigproto, ringback

Parameter Descriptions

The parameter descriptions that follow are in alphabetical order.

The port parameters in this section refer to the internal ports (*asy*, *syn*, and *ta*) supported by the RAC and are also referred to as global port parameters. In addition, these parameters can be included in Session Parameter Blocks (SPBs) within the RAC configuration file. Parameters set in an SPB override their global parameter equivalents. For information about internal ports and SPBs, see *Managing Remote Access Concentrators Using Command Line Interfaces*.

All parameters that can be used for asynchronous (*asy*) ports can also be used for *ta* ports.

You do not configure the general asynchronous (gsy) port by setting parameters. Instead, you edit the **dialout** section of the RAC configuration file; the default file is named **config.annex**. For information on how to do this, see *Managing Remote Access Concentrators Using Command Line Interfaces*.

a_router

This AppleTalk parameter specifies the Ethernet address of the network's A_Router. The RAC uses this value as a hint at startup. When a Routing Table Maintenance Protocol (RTMP) message arrives from this Ethernet address, the RAC gleans the AppleTalk DDP address from the packet and tries to talk to the AppleTalk router. The address is a hexadecimal Ethernet address, such as 00-7F-12-33-44-55. The default is **00-00-00-00-00-00**.

acp_key

This annex parameter defines the encryption key used to exchange messages between the RAC and the ACP security server. This parameter works only when **enable_security** is set to **Y** and a security server is defined. The security server maintains the encryption key for each RAC in the **acp_keys** file. The RAC and the security server can communicate only when this parameter's value matches the RAC's value in the ACP security server's **acp_keys** file. The default is a null string (""), which the RAC displays as **"<unset>."**

This parameter does not affect RADIUS. For information on the RADIUS encryption key, see *radius_secret on page 3-79.*

address_origin

This asynchronous port parameter determines where the RAC looks to find the local and remote IP addresses to use for the end points of a PPP/IPCP link. Table 3-6 describes the options. The default is **local**.

Table 3-6. Valid Options for address_origin Parameter

Option	Description
auth_server	The RAC uses values determined by the authentication server. When the authentication protocol, as specified by the annex **auth_protocol** parameter, is ACP, the RAC uses the values defined in the **acp_dialup** file (see *Managing Remote Access Concentrators Using Command Line Interfaces*). When the authentication protocol is RADIUS, the RAC uses the remote address defined by the RADIUS Framed-IP-Address attribute. RADIUS can return three IP addresses for this attribute: 255.255.255.255, which indicates that the PC can decide on the addresses; 255.255.255.254, which indicates that the RAC should use DHCP; any other address is the actual address to be used. For the remote address when the authentication protocol is RADIUS, the RAC uses the address defined via the Annex-Local-IP-Address attribute.
local	The RAC uses values set by the **local_address** and **remote_address** parameters. It ignores any address in **acp_dialup** and any address defined by RADIUS.
dhcp	The RAC contacts a DHCP server to obtain a remote address dynamically on behalf of the remote client. This value is valid only when the port is effectively in PPP mode (that is, the mode was set to ppp or the ppp command was executed when the port was in CLI mode). DHCP is not supported with SLIP. In this case, local address is the IP address of the RAC’s Ethernet interface. When dhcp is specified, the RAC ignores any address in the acp_dialup file and any value supplied by RADIUS.

allow_broadcast

This asynchronous port parameter allows an asynchronous port to receive administrative broadcast messages generated by the **boot** and **broadcast** commands. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

allow_compression

This asynchronous/synchronous port parameter allows the RAC to use TCP header compression on a SLIP or PPP line. Header compression occurs only if the other side of the serial link initiates compression. A **Y** enables this parameter; an **N** disables it. The default is **N**.

allow_snmp_sets

This annex parameter determines whether or not the RAC accepts and processes SNMP **set** commands. When this parameter is disabled, the RAC rejects all SNMP **set** commands; the RAC SNMP agent returns the error *no such name* for the first object in the **set** command. A **Y** enables this parameter; an **N** disables it. The default is **N**.

Enabling **allow_snmp_sets** configures the RAC to accept SNMP **set** commands from any source. SNMP Version 1 is inherently not secure. SNMP bypasses the RAC's security measures. If security is a concern, the administrator should consider:

- Changing the community string default from *Public* to a less common string name
- Establishing filters on a firewall router to block SNMP traffic from outside the local network

analog_encoding

This WAN parameter specifies the encoding type used for modem calls. Valid values are:

- **a_law** (used in Europe)
- **mu_law** (used in the U.S.)
- **auto** (the default, which uses **a_law** or **mu_law** as appropriate)

Typically, you do not need to change this parameter. To check that the correct value is in use, execute the **wan** command from the superuser CLI.

ani

This WAN parameter applies only to R1 and R2 CAS interfaces. It specifies whether or not the telco is providing the Automatic Number Identification (ANI) service. You can enter the number provided by ANI into the **calling_no** field of an SPB. This allows you to customize how the RAC handles calls based on where the calls originated. Valid **ani** values are **Y** if the telco is providing ANI service, or **N** if the service is not being provided. The default is **N**.

arap_v42bis

This asynchronous port parameter enables V.42bis compression during an ARAP session. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

at_guest

This asynchronous port parameter allows guests to log in to an AppleTalk session. When **at_guest** is enabled and a client requests guest access, the RAC asks ACP for user name guest privileges. A **Y** enables this parameter; an **N** disables it. The default is **N**.

at_nodeid

This asynchronous port parameter defines the node ID hint used for an ARA client during connection establishment. This parameter value is an AppleTalk address in the form *net.node*. The valid *net* values are **0** through **65534**. The valid *node* values are **0** through **254**. The default is **0.0**.

at_security

This asynchronous port parameter turns on ACP service for an AppleTalk session on this port. When both **at_security** and **enable_security** are enabled, the RAC uses ACP to get security information about the client, including authentication, logging, and zone access. If **at_security** is not enabled, the RAC uses only local security. A **Y** enables this parameter; an **N** disables it. The default is **N**.

attn_string

This asynchronous port parameter defines a control character sequence that returns users to the CLI prompt. Users can define a temporary control character sequence using the CLI **stty attn** command; **stty attn** "" disables the sequence. The default is no control character sequence, displayed as ""; the default for virtual CLI connections is **CTRL-A** (^A).

If you are running a **stats** [**-sm** [*ports*] | [*time*]] command with a defined time interval, the RAC ignores an attention string with multiple characters.

auth_protocol

This annex parameter defines the authentication protocol that the RAC uses. Valid values are **acp** and **radius**. The default is **acp**.

For RADIUS authentication to work, the following parameters must be set (in addition to **auth_protocol**):

- **enable_security** must be set to Y. (This is also true for **acp**.)
- **pref_secure1_host** must be set to the IP address of the host on which the authentication server runs.
- **radius_auth_port** must be set to the UDP port on which the RADIUS server is configured to listen. The standard port is **1812** (or **1645** for older versions that pre-date RFC 2059).
- **radius_secret** must be set to the secret shared by the server and the RAC.

authoritative_agent

When this annex parameter is enabled, the RAC sends an ICMP Address Mask reply to a host that broadcasts a subnet mask request. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

authorized_groups

This asynchronous port parameter specifies the LAT protocol remote group codes that are accessible to users on a given RAC port. You can enter **all**, **none**, a series of numbers from **0** through **255** separated by commas (for example, **1, 5,7**), or a range of numbers from **0** and **255** separated by dashes (for example, **1-5,200-255**) followed by **enabled** or **disabled**. The default is **all disabled**.

auto_busyout_enable

This WAN parameter is used for CAS interfaces only. It determines whether or not the RAC busies-out remaining DS0 channels when the last available modem has been used (because, for example, modems have failed or you have fewer than the maximum number of modems installed). Valid values are **Y** (busies out DS0s) or **N** (does not busy out DS0s). The default is **N**.

If you leave this parameter disabled, you lose calls when there are fewer modems than DS0s. However, if you enable this parameter; a busy signal is delivered to the user and a syslog is generated to notify you of the event.

Once you busy out DS0s, they remain busied out until the RAC is rebooted, or until you issue the CLI superuser **wan unbusyout** command (see *wan on page 1-123*).

autodetect_timeout

This asynchronous/synchronous port parameter specifies the number of seconds that the RAC waits for **auto_detect** mode to identify an incoming call as PPP. If the number of seconds is exceeded or the user enters a carriage return before the call is detected as PPP, the RAC places the user in CLI mode. Valid values are **1-60**. The default is **30**.

backward_key

This asynchronous port parameter specifies a character or string that reopens the next lower numbered session (already established at your port) from within the current session without returning to local mode. When defining this value, use a unique, unused character (such as Control-B) or a string of characters. To clear an existing setting, enter a null string (""). The default is no control character sequence.

On virtual (*telnet*) ports, the **backward_key** value is limited to one printable or Control character. If you try to set this value to more than one character, the setting is ignored and the previous value is restored.

On physical (as opposed to virtual) ports, a **backward_key** string can range from 1 to 16 characters.

banner

This asynchronous port parameter controls whether or not the RAC banner and message of the day (motd) appear on CLI ports. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

This parameter has an effect only at the CLI level.

bes_threshold

This annex parameter specifies the number of Bursty Errored Seconds that must occur within 15 minutes on a WAN module before wanBESThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

broadcast_addr

This annex parameter defines the IP address for RAC broadcasts. It is recommended that you set a subnet broadcast address, if possible. In this case, set the subnet portion of the broadcast address to match the RAC subnet address, as determined by the RAC subnet mask, and set the host portion of the broadcast address to all ones. For example, if the RAC subnet address is 132.254.9.0, and the RAC subnet mask is 255.255.255.252, you should set the broadcast address to 132.254.9.3. To calculate this, subtract the subnet mask from 255.255.255.255. Thus, in the previous example, you subtract 255.255.255.252 from 255.255.255.255 to arrive at 0.0.0.3.

If your network is not subnetted, you can specify a network broadcast address. In this case, you set the network portion of the broadcast address to match the RAC network address, as determined by the intrinsic mask for the network class. You set the host portion of the broadcast address to all ones.

Finally, you can set a limited broadcast address of 255.255.255.255 that reaches all nodes on the subnet. However, if you have more than one subnet on the same physical cable, the RAC broadcasts to all nodes on all of the subnets. This can cause problems if some of the subnets or nodes do not recognize the broadcast.

The default for **broadcast _addr** is **0.0.0.0**, which RAC RIP routing does not support (because most hosts do not recognize it).

broadcast_direction

This asynchronous port parameter defines the direction in which an administrative broadcast message is sent on a port. The options are **network** or **port**; the default is **port**. This parameter is valid only for a **slave** port (defined by the **mode** parameter).

If you specify **network**, the RAC sends administrative broadcast messages out the network side of the connection to the initiator. If you specify **port**, the RAC sends broadcast messages out the port side of the connection.

buildout

This PRI/CAS line parameter is applicable to RACs with internal CSUs. It is a string defining the CSU transceiver line provided by the telephone company. Valid values are **0db**, **7.5db**, **15db**, **22.5db**. The default is **0db**.

busy_signal_bits

This WAN DS0 parameter indicates the type of busy signal to transmit when a DS0 is busied out. Valid values are **00**, **01**, **10**, and **11**. The default is **11**. These apply to the *a* and *b* bits of a CAS frame: **10** turns on the *a* bit, **01** turns on the *b* bit, **11** turns on both bits, **00** turns off both bits.

DS0s can be busied out manually or automatically. To manually busy-out DS0s, set **sigproto** to **none** or use the **wan busyout** command (see *wan on page -123*). Or, you can arrange to have remaining DS0s busied-out automatically when the last available modem is used (see *auto_busyout_enable on page 3-17*).

bpv_threshold

This annex parameter specifies the number of Bipolar Violation or Line Code Violation errors that can occur on a WAN module in a 15-minute interval before the SNMP wanBpvThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables wanBpvThreshTrap.

call_begin_trap

This annex parameter enables or disables SNMP traps on call-begin events. Setting this parameter to **Y** enables traps; **N** disables them. The default is **Y**.

call_endtrap_inc

This annex parameter specifies the number of call-end events after which a call-end SNMP trap is sent. Valid values are **0** through **65535**. (The higher the value, the more memory used.) The default is **0**, which disables call-end traps.

call_history_limit

This annex SNMP parameter specifies the maximum number of call history records that the RAC saves. Valid values are **0** through **65535**. The default is **0**, which disables the saving of call history records.

chap_auth_name

This annex parameter defines the character string that is used as the *Name* field entry when issuing a CHAP challenge over a PPP link. The minimum string length is one character and the maximum string length is 16 characters; the default is **chap**.

Change the value of this parameter:

- If you want the remote end of the PPP link to choose a secret token that depends on the received name
- If you want to achieve the highest level of security

char_erase

When this asynchronous port parameter is enabled, the RAC echoes both the character erase and the word erase characters for a video terminal; that is, the previous character (or word) looks as if it has been erased. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

When **char_erase** is disabled, the RAC echoes the erase characters for a hardcopy terminal. It echoes the first erase character as a backslash (\) followed by the deleted character. Each additional use of the erase character deletes and displays another character. The first character typed (other than the erase character) echoes a slash (/) and the character; for example typing “**asdf**<Delete><Delete>**g**” echoes as “**asdf\fd/g**.”

This parameter has an effect only at the CLI level.

circuit_timer

This annex parameter defines the time interval in tens of milliseconds between the transmission of LAT packets (for example, if you enter **9**, the time interval will be 90 milliseconds). Valid values range from **1** to **25**. The default value is **8** (80 milliseconds).

cli_imask7

When this asynchronous port parameter is enabled, the RAC masks CLI input to 7 bits. The RAC masks input only at the CLI. When **cli_imask7** is disabled, the RAC expects 8-bit ASCII input. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

cli_inactivity

This asynchronous port parameter specifies the amount of time in minutes that the RAC remains idle before disconnecting a CLI session from a port. Unlike the **inactivity_timer**, this timer does not disconnect a CLI session with active jobs. Valid values range from **0** (or **off**) to **255** minutes (or **immediate)**. The default is **0**.

Entering **0** (or **off**) disables the timer; entering **255** (or **immediate**) causes the RAC to disconnect as soon as it exits from its last job.

cli_interface

This asynchronous port parameter allows you to control the prompt that appears for VMS or UNIX environments. Valid values are **vci** and **uci**. The default is **uci**.

When set to **vci**, the *Local>* prompt is displayed followed by the *Username>* prompt; the **uci** setting provides a standard UNIX interface (with prompts defined by the **cli_prompt** and **prompt** parameters).

cli_prompt

This annex parameter defines the RAC prompt for all CLI users. This parameter uses formatting codes consisting of the percent character (%) and a single lowercase letter. You can combine up to 16 of these codes (for example, *%a%c*). You can also enter text that will appear in the prompt as long as the entry as a whole does not exceed 32 characters. The default prompt is %a%c (the string *annex:*). Table 3-7 lists the formatting codes.

Table 3-7. Formatting Codes for RAC Prompts

Code	Expansion
%a	The string *annex*
%c	A colon followed by a space
%d	The current date and time in the following format: Mon Mar 14 13:59:42 1997
%i	The RAC’s IP address
%j	A new line character; go to the beginning of the next line
%n	The RAC’s name, if known, or the IP address
%r	The string *port*
%s	A space
%t	The current time in 24-hour format
%u	The user name defined for the port; if none, a null string

cli_security

This asynchronous port parameter enables user authentication by the host-based security server for all CLI connections. When this parameter is disabled, you cannot use any RAC security mechanism other than the administrative password for CLI ports. A **Y** enables this parameter; an **N** disables it. The default is **N**.

When **cli_security** is enabled and ACP security is being used, the RAC logs PPP/SLIP logins/logouts to the ACP log file.

config_file

This annex parameter defines the file name for the configuration file maintained on the load host. This file contains information about gateways, macros, and services; it must reside in the **/usr/spool/erpcd/bfs** directory. The default file name is **config.annex**.

connect_security

This asynchronous port parameter enables or disables the host-based security policy for using **telnet** and **rlogin** to access the network from the RAC CLI. If **connect_security** is enabled, the user must receive authorization to connect to a host on the network. If the security protocol is ACP, the file **/install-directory/acp_restrict** is scanned to authorize a connection. If the security protocol is RADIUS, the Users file on the RADIUS server is scanned. If authorization is not granted, the connection is not made. A **Y** enables this parameter; an **N** disables it. The default is **N**.

css_threshold

This annex parameter specifies the number of Controlled Slip Seconds that must occur within 15 minutes on a WAN module before an SNMP wanCssThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

cv_threshold

This annex parameter specifies the number of CRC6 Error Event conditions that must occur on a WAN module within a 15-minute interval before an SNMP wanCvThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables wanCvThreshTrap.

data_bits

This asynchronous port parameter defines the number of data bits in a character. This value does not include the start, stop, or parity bits. Valid values are **5** through **8**. The default is **8**.

daylight_savings

This annex parameter defines the daylight savings time for your geographic location. The RAC uses this parameter to adjust the time display for daylight savings time. Valid options are **us**, **australian**, **british**, **canadian**, **east_european**, **mid_european**, **west_european**, and **none**. The default is **us**.

dedicated_arguments

This asynchronous port parameter defines the command-line arguments used for dedicated ports. The parameter accepts a string of up to 100 characters in length. Use this parameter in conjunction with the **mode** parameter (see *mode on page 3-55*). The default is a **null string** ("").

default_session_mode

This asynchronous port parameter defines the default session mode when the VMS interface is configured (that is, when **cli_interface** is set to **vci**). Valid options are **interactive**, **passthru**, **passall**, or **transparent**. The default is **interactive**.

default_zone_list

This annex parameter contains the zone list that is sent to AppleTalk clients in case of an ACP failure. The string can contain from 1 to 100 characters. You must use spaces to separate zone names (for example, *general engineering lab*). To use embedded spaces within a zone name, use the backslash (\) character. If you do not set this parameter, the RAC provides the network zone list. The default is a **null string** ("").

dhcp_broadcast

This annex parameter enables and disables the use of DHCP broadcast messages. The DHCP proxy client (the RAC) uses broadcast messages to try to discover a DHCP server when **pref_dhcp1_host** and **pref_dhcp2_host** are set to zero or do not respond (see page 3-72). The options for the parameter are **Y** or **N**. **N** is the default: it disables the broadcasting of DHCP messages. The **dhcp_broadcast** parameter functions consistently with all other broadcast parameters defined in the system.

diallink_trap_enable

This annex parameter enables and disables the generation of SNMP link-up and link-down traps for remote dial-in interfaces. Entering **Y** enables these traps; entering **N** disables them. The default is **N**.

digit_power_1

This WAN parameter (used for CAS interfaces only) specifies the power level, in dBm, of the first tone of each digit generated by the RAC. This parameter is used primarily for debugging. Valid values are **0** through **255**. The default is **0**.

digit_power_2

This WAN parameter (used for CAS interfaces only) specifies the power level, in dBm, of the second tone of each digit generated by the RAC. This parameter is used primarily for debugging. Valid values are **0** through **255**. The default is **0**.

digit_width

This WAN parameter (for CAS interfaces only) allows you to adjust the width of each digit, in milliseconds, generated by the RAC. This parameter is used primarily for debugging. Valid values are **0** through **255**. The default is **0**.

disabled_modules

This annex parameter 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**, **edit**, **fingerd**, **ftpd**, **httpd**, **ipx, lat**, **nameserver**, **ppp**, **slip**, **snmp**, **tn3270**, **tstty**, **udas**, **vci**, **all**, or **none**. The default is **vci** (disables the RAC VMS interface).

You should exercise extreme caution when disabling modules:

- Entering a null string ("") sets this parameter to its default value.
- 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 RAC interface for VMS environments along with the following commands: **backwards**, **change**, **clear**, **crash**, **define**, **disconnect**, **forward**, **list**, **logout**, **resume**, **set**, and **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 nonvolatile memory, and reconfigure the RAC.

dnis

This WAN parameter (for CAS interfaces only) specifies the number of digits in the called number that the telco provides via its Dialed Number Identification Service. This allows the called number on a given CAS line to be visible to the RAC, which, in turn, permits you to map the dialed number to a particular service. To do this, enter the actual called number in the **called_no** field of an SPB. The default for this parameter is **0**.

do_compression

This asynchronous port parameter starts TCP/IP header compression on a SLIP link. When this parameter is enabled, the RAC negotiates for TCP/IP compression for both sides of the connection. A **Y** enables this parameter; an **N** disables it. The default is **N**.

drop_first_req

This port parameter determines whether or not the RAC drops the first PPP LCP Configure Request it generates. This is for use with peers that have problems handling the first request. Entering **Y** enables the dropping of first requests; entering **N** disables it. The default is **N**.

ds0_error_threshold

This annex parameter specifies the number of errors that must occur within 15 minutes on a DS0 channel before an SNMP ds0ErrorThresholdTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

dsx1_line_length

This WAN parameter is displayed but does not affect the RAC.

echo

This port parameter directs a RAC to echo all characters as a user types. This echo occurs only at the CLI level. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

enable_radius_acct

This annex parameter determines whether or not RADIUS accounting is enabled for the RAC. A **Y** enables accounting; an **N** disables it. The default is **N**.

Once enabled, RADIUS accounting is performed on the RADIUS server specified by the **pref_secure1_host** parameter. If that parameter is undefined or the server is unavailable, the RAC enables accounting on the secondary host specified by **pref_secure2_host**. If neither host is defined or available, an error is syslogged; no broadcasting occurs.

For RADIUS accounting to work, the following parameter values must be set (in addition to **pref_secure1_host**):

- **enable_security** must be set to Y.
- **radius_auth_port** must be set to the UDP port on which the RADIUS server is configured to listen. The standard port is 1812 (or 1645 for older versions that predate RFC 2059).
- **radius_acct_port** must be set to a valid, Internet-assigned UDP port. The standard port is **1813** (or **1646** for older versions that predate RFC 2059).
- **radius_secret** must be set to the secret shared by the server and the RAC.

enable_security

This annex parameter activates security. To enable any security features, including RADIUS, set this parameter to **Y**. The default is **N**.

erase_char

This asynchronous port parameter defines a control character sequence for the CLI erase character. The default is the **Delete** key (displayed as **^?**).

erase_line

This asynchronous port parameter defines a control character sequence for the CLI line erase character. The default is **CTRL-U** (^U).

erase_word

This asynchronous port parameter defines a control character sequence for the CLI word erase character. The default is **CTRL-W** (^W).

es_threshold

This annex parameter specifies number of Errored Seconds conditions that must occur on a WAN module in a 15 minute interval before an SNMP wanEsThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

esf_threshold

This annex parameter specifies the number of ESF Error Event conditions that must occur on a WAN module in a 15 minute interval before and SNMP wanEsfThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

facility_num

This annex parameter identifies a LAT host by number. Valid values are from **0** through **32767**. The default value is **0**.

fdl_type

This WAN parameter specifies the type of Facilities Data Link supported by the telephone company for your PRI or CAS line. Valid values are **att** and **ansi**.

forced_call_inc

This annex parameter specifies the number of call-disconnect events can caused by inactivity timers that can occur before an SNMP forcedCallDisconnectTrap trap is sent. Valid range of values is **0** through **65535** The default is **0**, which disables forcedCallDisconnectTrap traps.

forward_key

This asynchronous port parameter specifies a character or string that reopens the next available, higher numbered session already established at your port. When defining this value, use a unique, unused character (such as Control F) or a string of characters. To clear an existing setting, enter a null string (""). The default is no control character sequence.

On virtual (*telnet*) ports, the **forward_key** value is limited to one printable or Control character. If you try to set this value to more than one character, the setting is ignored and the previous value is restored.

On physical (as opposed to virtual) ports, a **forward_key** string can contain from 1 through 16 characters.

forwarding_count

This asynchronous port parameter controls RAC port behavior for received characters. When set to a number other than **0**, the port does not forward characters until it receives the specified number of characters. When set to **0**, the port uses the value in the **forwarding_timer** parameter. Valid values are **0** through **255**. The default is **0**.

If you use both **forwarding_count** and **forwarding_timer**, the RAC uses the value that occurs first. Setting **forwarding_count** to **1** or **forwarding_timer** to **0** may have a severe effect on the network when heavy serial input occurs.

forwarding_timer

This asynchronous port parameter sets the amount of time in 10-millisecond (ms) intervals that can elapse before the RAC forwards received data. If new data arrives before the timer expires, the RAC resets the timer. Valid values are **0** through **255** or **off**. The default is **5** (**50 ms**); if you set the value to **0**, the RAC uses **5**.

If you use both **forwarding_count** and **forwarding_timer**, the RAC uses the value that occurs first. Setting **forwarding_count** to **1** or **forwarding_timer** to **0** may have a severe effect on the network when heavy serial input occurs.

framing

This WAN parameter applies to CAS interfaces only and defines the superframe format for which the interface is provisioned. Valid values for Channelized T1 are **esf** (extended superframe) and **d4** (superframe). Valid values for Channelized E1 are **ddf**, **mff_crc4**, and **mff_crc4_g706**. The default is **esf**.

group_value

This annex parameter specifies the LAT protocol remote group codes that can access local services offered by a RAC. To access these services, the RAC must have at least one enabled group code that matches the service’s group codes. Valid options are **all**, **none**, a series of numbers from **0** through **255** separated by commas, or a range of numbers from **0** through **255** separated by dashes, followed by **enabled** or **disabled**. The default is **all disabled**.

hardware_tabs

This asynchronous port parameter allows the RAC to convert ASCII tab characters to the correct number of spaces when a terminal does not support hardware tabs. This occurs only at the CLI level. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

host_table_size

This annex parameter defines the number of entries allowed in the host table. Valid values are **0** through **255**. Entering **255** allows an unlimited number of entries; entering **254** indicates that there is no host table. In this case, the RAC requires a name server to resolve every host name. The default is **64**.

image_name

This annex parameter is a string defining the name of the file that contains the operational code that loads by default when you boot a RAC. The string can contain from 0 through 100 characters. The default is a **null string** ("").

imask_7bits

This asynchronous port parameter enables a RAC to mask input to 7 bits. When this parameter is disabled, the RAC expects 8-bit ASCII input. This parameter has no effect on transmitted characters. A **Y** enables the parameter; an **N** disables it. The default is **N**.

SLIP and PPP do not work if this parameter is enabled.

inactivity_timer

This asynchronous port parameter specifies the number of minutes that a port can remain inactive. If the timer expires, the RAC terminates all sessions and resets the port.

You can use the **input_is_activity** and **output_is_activity** parameters to define activity as input to the port or output from the port. Setting these parameters to **N** causes the timer to run independent of activity. Valid values are **0** through **255**. The default is **0** (timer disabled).

If you want a port to reset after a given number of minutes, regardless of any activity, you must set the following parameters:

inactivity_timer	**N**
input_is_activity	**N**
output_is_activity	**N**

These settings are required because the timer does not start until one of the following events occurs:

- Input occurs and **input_is_activity** is set to **Y**.
- Output occurs and **output_is_activity** is set to **Y**.

inet_addr

This annex parameter defines the RAC’s IP address. This 32-bit address contains four 8-bit fields separated by periods. Each field contains a decimal number from **0** through **255** or a hexadecimal number. The IP address always appears in dotted-decimal notation. This parameter has no default.

input_flow_control

This asynchronous port parameter specifies the method of flow control for input received from a device connected to an asynchronous port. Table 3-8 describes the valid options; the default is **eia**.

Table 3-8. Valid Options for the input_flow_control Parameter

Option	Description
bell	The RAC rings the terminal bell (sends ^G) when its input buffer is full.
eia	The RAC respects the flow control of the incoming modem.
start/stop	Designates flow control by recognizing XON and XOFF characters.
none	Disables flow control; characters are lost if the buffers overflow.

input_is_activity

This asynchronous port parameter defines activity as input. When this parameter is enabled, the RAC resets the inactivity timer when it receives input at the port. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

input_start_char

This asynchronous port parameter defines the control character sequence that restarts input if the **input_flow_control** parameter is set to **start/stop**. The default is **CTRL-Q** (^Q).

input_stop_char

This asynchronous port parameter defines the control character sequence that stops input if the **input_flow_control** parameter is set to **start/stop**. The default is CTRL-S (^S).

inter_digit

This WAN parameter (for CAS interfaces only) allows you to adjust the distance between each digit, in milliseconds, generated by the RAC. This parameter is used primarily for debugging. Valid values are **0** through **255**. The default is **0**.

ip_forward_broadcast

This annex parameter allows the RAC to broadcast a packet to the SLIP or PPP interfaces. When the RAC receives a packet sent to a broadcast address (except 0.0.0.0 and 255.255.255.255), it scans the list of installed interfaces and, using a subnet or network mask, tries to match the broadcast address to the interface's remote address. If the addresses match, the RAC copies the packet to that interface. When **ip_forward_broadcast** is disabled, the RAC does not scan the interface list and does not copy broadcast packets. A **Y** enables this parameter; an **N** disables it. The default is **N**.

ipcp_unnumbered

This port parameter enables or disables IP address negotiation between PPP peers. If you enable this parameter, you must then configure addresses at each end of a PPP link. A **Y** enables the parameter; an **N** disables it. Enabling this parameter is discouraged. The default is **N**.

ipencap_type

This annex parameter specifies whether the RAC LAN interface encapsulates IP packets in the Ethernet Version 2 format or the IEEE 802.3 data link layer format. The values for this parameter are **ethernet** or **ieee802**. The default is **ethernet**.

ipso_class

This asynchronous port parameter specifies the U.S. Department of Defense basic IP Security Option (IPSO) classification level included in TCP packets generated locally on RAC CLI, dedicated, or adaptive asynchronous ports.

The option is not added to locally generated ICMP messages, RIP updates, or other system packets. The RAC does not check incoming packets for the presence of IPSO. Valid options for the classification level are **topsecret**, **secret**, **confidential**, **unclassified**, and **none**. If you specify **none,** no IPSO classification is added. The default is **none**.

ipx_do_checksum

This annex parameter is displayed by the RAC but has been disabled.

ipx_dump_password

This annex parameter is a string that specifies the user password for logging on to the Novell file server before the RAC sends a dump file to the server. The string can contain from 0 through 16 characters. The default is a null string (""), which the RAC displays as **"<unset>."**

ipx_dump_path

This annex parameter specifies the full path name that stores the uploaded RAC dump image on the Novell file server. The string can contain from 0 through 100 characters. This parameter has no default value.

ipx_dump_username

This annex parameter provides a user name for logging on to the Novell file server before the RAC sends a dump file to the server. The name can contain from 0 through 48 characters. This parameter has no default value.

ipx_file_server

This annex parameter contains the name of the Novell file server from which the RAC boots. The string can contain from 1 through 48 characters. This parameter has no default value.

ipx_frame_type

This annex parameter defines the framing used for IPX packets on the Ethernet interface. Valid values are **ethernetII**, **raw802_3**, **802_2**, or **802_2snap**. The default is **raw802_3**.

ipx_network

This WAN B/DS0 channel parameter associates IPX network addresses with B or DS0 channels. This parameter works only when the **mode** parameter is set to **ppp**. The syntax is one of the following:

set wan b[=*channel_range* | **all**] **ipx_network** *net_number* [*increment*]

set wan ds0[=*channel_range* | **all**] **ipx_network** *net_number* [*increment*]

net_number is a 4-byte, Novell network number that the RAC suggests for the remote PC client on an IPXCP (IPX over PPP) link. Valid values are **00000001** to **FFFFFFFF**, or **0**. Leading zeroes, if any, should be included. The network number must be unique on the network and on the RAC itself.

When the IPXCP connection is established, the RAC and the client negotiate the network number, each suggesting a value. The peer suggesting the highest number wins the negotiation, and the network number is set to that value. If both ends of the link set the network number to 0, a unique, randomly generated number is used as the default.

This parameter is overridden by the value in the remote address field of the **acp_dialup** file, if that field is configured correctly.

Table 3-9 describes the arguments for **ipx_network**.

Table 3-9. Arguments for the ipx_network Parameter

Argument	Description
channel_range	An integer specifying a single B or DS0 channel number, a list of B or DS0 channel numbers separated by commas, a range of B or DS0 channel numbers separated by a hyphen, or the keyword **all**, which specifies all B or DS0 channels. Valid channel numbers for ISDN PRI are **1** through **23** (in the U.S.) and **1** through **30** (in Europe). Valid channel numbers for CAS are **1** through **24** (for Channelized T1) and **1** through **30** (for Channelized E1, R1, and R2). The default is **all**.
net_number	The IPX network number to be assigned to the B channel if only one channel is specified in *channel_range*, or the IPX network number to be assigned to the first channel in *channel_range*.
increment	An integer specifying how *net_number* is to be incremented to generate IPX network addresses for the B channels automatically specified in *channel_range*. For example, if you set *channel_range* to **1, 2, 3,** *net_number* to **00000001**, and *increment* to **2**, the RAC assigns IPX node numbers 00000003, 00000005, and 00000007 to the B channels. The default increment is **0**.

ipx_node

This WAN B/DS0 channel parameter is a string of 12 hexadecimal digits representing the 6-byte, nonzero node number the RAC suggests for the node number of the remote PC client on an IPXCP (IPX over PPP) link. Valid values are **00-00-00-00-00-00** through **FF-FF-FF-FF-FF-FE**, except for multicast addresses. Table 3-10 describes the arguments.

A multicast address is any address that has a 1 in the last bit of the first octet. For example, in the IP address 090007000000, of which the first octet (09) is 0000 1001 in binary, the rightmost 1 is the multicast indicator. The syntax for specifying this parameter is one of the following:

set wan b[=*channel_range* | **all**] **ipx_node** *node _number* [*increment*]

set wan ds0[=*channel_range* | **all**] **ipx_node** *node _number* [*increment*]

If the client suggests any valid value for the node number, that number is used instead of the **ipx_node** value.

If you are using ACP security, the **ipx_node** parameter is overridden by the network number in the remote address field of the **acp_dialup** file, if that field is configured correctly. If the node number is not set in **acp_dialup** or through the **ipx_node** parameter, and no value is suggested by the client, the RAC uses its own Ethernet address plus 1.

Table 3-10. Arguments for the ipx_node Parameter

Argument	Description
channel_range	An integer specifying a single B or DS0 channel number, a list of B or DS0 channel numbers separated by commas, a range of B or DS0 channel numbers separated by a hyphen, or the keyword **all**, which specifies all B or DS0 channels. Valid channel numbers for ISDN PRI are **1** through **23** (in the U.S.) and **1** through **30** (in Europe). Valid channel numbers for CAS are **1** through **24** (for Channelized T1) and **1** through **30** (for Channelized E1, R1, and R2). The default is **all**.
node_number	The IPX node number to be assigned to the B or DS0 channel if only one channel is specified in *channel_range*, or the IPX node number to be assigned to the first channel in *channel_range*.
increment	An integer specifying how *node _number* is to be incremented to generate IPX network addresses automatically for the B or DS0 channels specified in *channel_range*. For example, if you set *channel_range* to **1, 2, 3**, *node_number* to **00-00-00-00-00-01**, and *increment* to **4**, the RAC assigns IPX node numbers **00-00-00-00-00-05**, **00-00-00-00-00-09**, and **00-00-00-00-00-0D** to the B channels. The default increment is **0**.

ipx_security

This parameter is displayed by the RAC, but setting it has no effect.

ixany_flow_control

This asynchronous port parameter treats any input character as a start (XON) character if output has been suspended by a stop (XOFF) character. A **Y** enables this parameter; an **N** disables it. The default is **N**.

keep_alive_timer

This annex parameter defines the number of seconds between the transmission of identification packets during times of network inactivity. This parameter works only for the LAT protocol. The packets serve only as notices to remote nodes that the host's services are available. Valid values are **10** through **255** (seconds). The default is **20** (seconds).

lat_key

This annex parameter restricts access to LAT-related RAC commands, parameters, functions, and the LAT protocol within the RAC. Each RAC requires a unique key value (contact your supplier to obtain a LAT key). After setting the key, your system administrator must reboot the RAC.

lat_queue_max

This annex parameter limits the number of HIC requests that the RAC can queue. This parameter affects only the operation of HIC requests received after changing the parameter's value and setting LAT. Valid values are **1** through **255** or **none** (entering **none** sets the value to **255**). The default value is **4**.

latb_enable

This asynchronous port parameter enables the RAC to decode a LAT hosts's data-b packet. Data-b packets change certain asynchronous port parameters (see your LAT host's documentation for more details). A **Y** enables this parameter; an **N** disables it. The default is **N**.

If **latb_enable** is set to **Y** and the LAT host sends a data-b slot message requesting that flow control (XON/XOFF) be turned off, the RAC turns off flow control and passes XON/XOFF characters to the host. This scenario can adversely affect both XON/XOFF and the cursor keys on the terminal.

line_erase

This asynchronous port parameter allows the RAC to echo line erase for a video terminal. When this parameter is enabled, the RAC erases all characters on the line and moves the cursor back to the beginning of the line. When this parameter is disabled, the RAC echoes the line erase character for hardcopy terminals, making the deleted line visible and positioning the print head at the beginning of the next line. The line erase occurs only at the CLI level. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

line_code

This WAN parameter applies to CAS interfaces only. It specifies the line code for which the CAS interface is provisioned. Valid values are **b8zs**, **ami** (Channelized T1 or E1), and **hdb3** (Channelized E1 only). The default is **b8zs**.

load_broadcast

This annex parameter defines, during a boot, whether or not the RAC requests the configuration or message-of-the-day files from other hosts on the network if any or all of the files are not available on the preferred load host. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

load_dump_gateway

This annex parameter specifies the gateway’s IP address. A gateway is required if the preferred load or dump host is on a different network or subnet than the RAC. The default is **0.0.0.0** (no gateway).

load_dump_sequence

This annex parameter specifies available network interfaces (Ethernet or self) and the order in which they are used for a downline load or an upline dump. You can list more than one interface by using commas to separate interface names. Table 3-11 describes the valid options.

Table 3-11. Valid Options for the load_dump_sequence Parameter

Option	Description
net	For use with a local area network. This is the default value.
self	Instructs the RAC to boot its image from the flash ROMs. Because the RAC cannot dump back to itself, when you boot via **self**, always have a secondary load/dump interface by setting **load_dump_sequence** to **self,net**.

local_address

This asynchronous/synchronous global port parameter defines the IP address for the **asy**, **ta**, or **syn** port on the RAC side of a link. This IP address is used only when the **mode** parameter is set to **slip** or **ppp**. The default is **0.0.0.0**.

local_phone_number

This WAN parameter specifies the telephone number of the RAC (as supplied by the telco). The RAC returns this number if queried for it by a telco switch (such as an R2 switch). This string can contain from 0 through 16 characters. The default is a null string (**""**).

location

This asynchronous port parameter defines an asynchronous port location or description that is displayed by the CLI **who** command. This string can contain from 0 through 16 characters. The default is a null string (**""**).

lock_enable

This annex parameter enables any port to use the RAC interface for the VMS Environment’s **lock** command. A **Y** enables this parameter; an **N** disables it. The default is **N**.

lofc_threshold

This annex parameter specifies the number of Loss of Frame Count errors that must occur on a WAN module in a 15 minute interval before an SNMP wanLofcThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

login_password

This annex parameter specifies the password for all ports using a VMS interface. This string can contain from 0 through 16 characters. For security reasons, the RAC displays this value as **“<set>”** or **“<unset>**.” The default is a null string (""), which the RAC displays as **“<unset>**.”

This parameter works only when **cli_interface** is set to **vci** and **login_port_password** is set to **Y**.

login_port_password

This asynchronous port parameter enables the port password when the VMS command interface is configured (that is, when **cli_interface** is set to **vci**). A **Y** enables this parameter; an **N** disables it. The default is **N**.

login_prompt

This annex parameter defines the prompt that appears for all ports using a VMS interface. The string can contain from 0 through 16 characters. The default is the # symbol.

This parameter works only when **cli_interface** is set to **vci.**

login_timer

This annex parameter specifies the number of minutes a port using a VMS interface can remain inactive. Valid values are **0** through **60** (minutes). Entering **0** sets the timer to 30 minutes. The default is **30**.

This parameter works only when **cli_interface** is set to **vci**.

login_timeout

This asynchronous port parameter enables a login timer when the VMS command interface is configured (that is, when **cli_interface** is set to **vci**). A **Y** enables this parameter; an **N** disables it. The default is **N**.

long_break

This asynchronous port parameter enables the RAC to return a user to the CLI prompt after receiving a break signal of more than two seconds. When this parameter is disabled, the RAC passes the break to the local application. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

loose_source_route

This annex parameter controls the Loose Source Routing protocol, which defines a sequence of IP addresses that a datagram must follow. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

When **loose_source_route** is enabled, the RAC forwards all IP packets that have the *Loose Source Routing and Record* option set. The RAC forwards all IP packets that have the *Strict Source Routing and Record* option set only if the next routing address is directly reachable by the RAC. Otherwise, the RAC drops these packets and sends an ICMP *Destination Unreachable* message with a code of *Source Route Failed*.

When **loose_source_route** is disabled, the RAC does not forward any IP packets that have the *Strict Source Routing and Record* or *Loose Source Routing and Record* options set. The RAC accepts these packets only if the RAC itself is the ultimate destination. If the packets are not addressed to the RAC, they are dropped and the RAC sends an ICMP type *Destination Unreachable* message with a code of *Source Route Failed* to the originator.

Loose Source Routing can pose a security risk if you use filters on your network router. If you are concerned about security, set this parameter to **N**.

map_to_lower

This asynchronous port parameter enables the RAC to convert uppercase characters sent from a terminal to lowercase characters. This conversion occurs only at the CLI level. Enable this parameter for older terminals that do not support lowercase characters. A **Y** enables this parameter; an **N** disables it. The default is **N**.

map_to_upper

This asynchronous port parameter enables the RAC to convert lowercase characters sent to a terminal to uppercase characters. This conversion occurs only at the CLI level. Enable this parameter for older terminals that do not support uppercase characters. A **Y** enables this parameter; an **N** disables it. The default is **N**.

max_chap_chall_int

This annex security parameter enables the RAC to reissue a CHAP challenge to a remote node at random times during the course of a PPP connection. The parameter itself specifies the maximum number of seconds in the interval from which the RAC randomly chooses the times to reissue the challenge. Valid values are **0** through **65535** (approximately 18.2 hours). For example, a value of **60** specifies an interval from 1 to 60 seconds. The default is **0**, which disables the reissuing of challenges.

max_session_count

This asynchronous port parameter specifies the number of active sessions (jobs) allowed per port. Valid values are **1** through **16**. The default is **3**.

max_vcli

This annex parameter determines the maximum number of virtual CLI connections the RAC can create at a time. Valid values are the string **unlimited** or a decimal number from **0** through **254**. A value of **0** prevents any virtual CLI connections from being made. The default is **unlimited**.

metric

This asynchronous/synchronous port parameter defines the hop count to the remote end of the asynchronous or synchronous line when the **mode** parameter is set to **slip** or **ppp**. Modify this parameter only if you want the RAC to use a route other than the SLIP or PPP interfaces to the remote end. Valid values are **1** through **15**. The default is **1**.

min_unique_hostnames

This annex parameter determines whether or not you can identify a host in the host table by entering a minimal string rather than the full host name. A **Y** enables minimum uniqueness; an **N** disables it. The default is **Y**.

mmp_enabled

This annex parameter enables Multisystem Multilink PPP (MMP) for incoming calls. MMP allows Multilink PPP (MP) links for a single PC or router to terminate on any RAC within an MMP group. RACs are combined into an MMP group via the **mp_endpoint_class** and **mp_endpoint_address** parameters.

The RACs in an MMP group must:

- Have the same Endpoint Discriminator, which you define via the **mp_endpoint_class** and **mp_endpoint_address** parameters
- Reside on the same Ethernet segment and, where applicable, the same IP subnet

A RAC can be part of only one MMP group.

You must reboot the RAC after changing the value of **mmp_enabled**.

Valid values for **mmp_enabled** are **Y** and **N**. The default is **N**.

For instructions on configuring MMP, see *Managing Remote Access Concentrators Using Command Line Interfaces*.

mode

This asynchronous/synchronous port parameter sets the mode for access to a port. Table 3-12 describes the valid options; the default is **auto_adapt**.

Table 3-12. Valid Options for the mode Parameter

Option	Description
arap	Allows a port to act as a network interface using ARAP.
auto_adapt	Allows a port to identify an incoming packet's protocol and to convert it to IPXCP, PPP, ARAP, or CLI.
auto_detect	Same as **auto_adapt**.
cli	Allows a port connected to a terminal or incoming modem to access the CLI. The CLI provides access to the network and connections to other hosts via the **telnet**, **connect**, **rlogin**, and **tn3270** commands.
connect	Allows a port to communicate with a LAT host via the **connect** command. This option works with the **dedicated_arguments** parameter.
ppp	Allows a port to perform as a network interface using PPP. IP packets are encapsulated by PPP.
rlogin	Allows a port to communicate via the **rlogin** command. Use this option in conjunction with the asynchronous port parameter **dedicated_arguments**.

(continued on next page)

Table 3-12. Valid Options for the mode Parameter (continued)

Option	Description
slip	Allows a port to perform as a network interface using SLIP. IP packets are encapsulated by SLIP.
telnet	Allows a port to communicate via the **telnet** command. Use this option in conjunction with the asynchronous port parameter **dedicated_arguments**.
tn3270	Allows a port to communicate via the **tn3270** command. Use this option in conjunction with the asynchronous port parameter **dedicated_arguments**.

modem_error_threshold

This annex parameter specifies the number of consecutive modem errors that must occur before wanMdmErrorThresTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

mop_password

This annex parameter is a string specifying the MOP maintenance password. In this 8-byte password, each byte consists of two hexadecimal digits. The string can contain from 0 through 16 characters. For security reasons, the RAC displays values as **“<set>”** or **“<unset>.”** The default is a null string (""), which the RAC displays as **“<unset>**.**”**

motd_file

This annex parameter is a string defining the file name for the message-of-the-day file maintained on the load host. The string can contain from 0 through 16 characters. The default file name is **motd**; the file resides in the directory chosen during the host installation process (typically **/usr/spool/erpcd/bfs**).

mp_endpoint_address

This asynchronous/synchronous port parameter is used with Multilink PPP (MP) and Multisystem Multilink PPP (MMP). For MMP, it defines the Endpoint Discriminator when the **mp_endpoint_class** parameter is set to **local** or **psndn**.

- When **mp_endpoint_class** is **local**, **mp_endpoint_address** can be an alphanumeric string containing from 1 through 16 characters. This string is a name to be associated with the RAC.
- When **mp_endpoint_class** is **psndn**, **mp_endpoint_address** can be an ISDN Directory Number (a phone number) of up to 15 characters.

All RACs in an MMP group must have the same **mp_endpoint_address** and the same **mp_endpoint_class**. The default a null string (""), which the RAC displays as **"unset."**

mp_endpoint_class

This asynchronous/synchronous port parameter is required to determine what the RAC uses as the Endpoint Discriminator for MP or MMP links. This information helps to determine whether a member link becomes part of an existing bundle or starts a new bundle. Table 3-13 describes the valid values for **mp_endpoint_class**. The default for MP is **mac**, which you can also specify as **default**.

When a RAC is configured for MMP (that is, **mmp_enabled** is set to **Y**), only three of the values are applicable: **null** (the default), **local**, and **psndn**. (The **mac** and **ip** values imply a unique binding to a single RAC; MMP operates across multiple RACs.)

All RACs in an MMP group must have the same **mp_endpoint_address** and the same **mp_endpoint_class**.

Table 3-13. Valid Values for the mp_endpoint_class Parameter

Value	Description
ip	(Does not apply to MMP.) The RAC uses the IP address of it's Ethernet interface as the Endpoint Discriminator.
mac	(Does not apply to MMP.) The RAC uses the MAC address of its Ethernet interface as the Endpoint Discriminator. This is the default for MP.
magic	(Does not apply to MMP.) The RAC generates the standard 16-octet Magic Number Block and uses it as the Endpoint Discriminator.
psndn	The RAC uses the ISDN phone number specified by the **mp_endpoint_address** parameter as the Endpoint Discriminator.

(continued on next page)

Table 3-13. Valid Values for the mp_endpoint_class Parameter (continued)

Value	Description
null	The RAC does not use an Endpoint Discriminator. This is the default for MMP.
local	The RAC uses the RAC name specified by **mp_endpoint_address** as the Endpoint Discriminator. If you are using MMP, this name must be the same for all RACs in the MMP group, and unique to that group. If you cannot guarantee the string’s uniqueness, use **psndn** and set **mp_endpoint_address** to a telephone number.

mp_mrru

This asynchronous/synchronous port parameter sets the Maximum Reconstructed Receive Unit (MRRU) that the RAC requests in LCP negotiations for Multilink PPP (MP) and Multisystem Multilink PPP (MMP). All LCP negotiations start with this value, but downward negotiation is allowed. You must set this parameter to a nonzero value to negotiate MP or MMP. Valid values are **64** through **1600**. The default is **1500**.

multicast_timer

This annex parameter defines the number of seconds that can elapse between service announcement transmissions for the LAT protocol. Valid values are **10** through **180** (seconds). The default is **30**.

multisessions_enable

This annex parameter allows multiple sessions to be managed on a terminal server basis. When this parameter is enabled, terminals that support DEC’s Terminal Device/Session Management Protocol (TD/SMP) can display two active windows simultaneously over one communication line. A **Y** enables this parameter; an **N** disables it. The default is **N**.

name_server_1

This annex parameter defines the type of name service used with the primary name server. The options are **dns, ien_116**, or **none**, which disables name service.

If you use this parameter, you must also specify the address of the primary name server by using the **pref_name1_addr** parameter, and that host must be running the appropriate daemon (**dns** or **ien_116**).

name_server_2

This annex parameter defines the type of name service used with the secondary name server. The service type specified with this parameter is queried if the type specified by **name_server_1** is not available. The options are **dns**, **ien_116**, or **none**, which disables name service. The default is **none**.

If you use this parameter, you must also specify the address of the secondary name server by using the **pref_name2_addr** parameter.

nameserver_broadcast

This annex parameter defines whether or not the RAC broadcasts a name server request if the preferred name servers do not respond. A **Y** enables this parameter; an **N** disables it. The default is **N**.

need_dsr

This asynchronous port parameter allows the RAC to use the DSR (Data Set Ready) signal to determine whether a device is attached to the corresponding asynchronous line. The RAC will not allow connection to a slave port and will not activate the CLI until the DSR signal is active. If DSR is deactivated, the connection to a slave line is terminated and the CLI is deactivated.

When **need_dsr** is disabled, the DSR signal is not required to connect to a slave line, and DSR is not required to activate a CLI line. DSR is always considered active on ports that do not have a DSR signal (ports with partial modem control lines). A **Y** enables this parameter; an **N** disables it. The default is **N**.

net_inactivity

This asynchronous/synchronous port parameter defines the amount of time that network protocols (for example, SLIP, PPP, ARAP) running on the serial line can remain inactive before the port is reset. Valid values are **0** through **255**; a value of **0** indicates *off*. The default is **0** (*off)*.

Use this parameter in conjunction with the **net_inactivity_units** parameter.

The accuracy of the inactivity timer is within 5 seconds. For this reason, if you set **net_inactivity_units** to seconds, use a value for **net_inactivity** that is a multiple of 5.

net_inactivity_units

This asynchronous/synchronous port parameter defines the unit of time used for the port's inactivity timer. Valid options are **minutes** and **seconds**. The default is **minutes**.

Use this parameter in conjunction with the **net_inactivity** timer.

An inactivity timer can be set to the following combinations:

- 0 through 4 minutes 15 seconds (255 seconds) in 1-second intervals
- 0 through 4 hours 15 minutes (255 minutes) in 1-minute intervals

If you are using an ISDN line, set **net_inactivity** to a small value, such as **30,** and **net_inactivity_units** to **seconds**; otherwise, you may incur high costs from your ISDN service provider.

To set an inactivity timer of 2 minutes, set the parameters as follows:

```
set net_inactivity_units minutes
set net_inactivity 2
```

To set an inactivity timer of 30 seconds, set the parameters as follows:

```
set net_inactivity_units seconds
set net_inactivity 30
```

network_turnaround

This annex parameter defines the approximate number of seconds that the RAC waits for a response from a security server (an algorithm defines the actual time, which typically is longer than the defined value). This parameter works only when the **enable_security** parameter is set to **Y**. Valid values are **1** through **255**. The default is **2**.

Do not set this parameter to a high number unless a large timeout value is required for contacting (for security) slow hosts or waiting for a slow host's response to a security request.

newline_terminal

This asynchronous port parameter interprets carriage returns and line feeds at the CLI level. When this parameter is enabled, a line feed terminates both the input and the output lines. When the parameter is disabled, a carriage return or a line feed terminates the input line and a carriage return followed by a line feed terminates output lines. A **Y** enables the parameter; an **N** disables it. The default is **N**.

node_id

This annex parameter specifies the AppleTalk address the RAC tries to acquire at startup. If this address is in use, the RAC must acquire a new node ID. The **node_id** is an AppleTalk address in the form *net.node*. Valid *net* values are **0** through **65534**; valid *node* values are **0** through **254**. The default is **0.0**.

num_b_channels

This WAN parameter specifies the maximum number of calls that the RAC handles via B channels. The default is **0**, which the RAC interprets as 23 for T1/PRI connections and 30 for E1/PRI connections. Valid values are **1** through **23** for T1/PRI, **1** through **30** for E1/PRI.

oof_threshold

The number of Out of Frame errors that must occur on a WAN module in a 15 minute interval before wanOofThreshTrap is sent. Valid values are **0** through **65535**. The default value is **0**, which disables this trap.

option_key

This annex parameter is displayed but has been disabled.

output_flow_control

This asynchronous port parameter defines the method that a device uses to stop output from the RAC. Table 3-14 describes the valid options; the default is **start/stop**.

Table 3-14. Valid Options for the output_flow_control Parameter

Option	Description
bell	Comparable to setting the parameter to **none**.
eia	Selects hardware flow control.
start/ stop	Specifies XON/XOFF flow control. Upon receiving XOFF (**output_stop_char**), the RAC stops sending output to the device. Upon receiving XON (**output_start_char**), the RAC starts sending output to the device. The RAC removes these characters from the data stream.
both	Specifies both in-band (XON/XOFF) and out-of-band (CTS/RTS) flow control. Both flow controls are independent; data flows out of the port only if CTS is high and the last received character was XON. Receiving XOFF or dropping CTS stops output (RAC to device) flow.
none	Specifies no flow control; characters are lost if the buffers overflow.

output_is_activity

This asynchronous port parameter defines activity as output. When this parameter is enabled, the RAC resets the inactivity timer when it sends output from the port. (For more information, see *inactivity_timer on page -37*.) A **Y** enables the parameter; an **N** disables it. The default is **N**.

output_start_char

This asynchronous port parameter defines the control character sequence that restarts output if **output_flow_control** is set to **start/stop**. The default is **CTRL-Q** (^Q).

output_stop_char

This asynchronous port parameter defines the control character sequence that stops output if the **output_flow_control** parameter is set to **start/stop**. The default is **CTRL-S** (^S).

output_ttl

This annex parameter sets the time-to-live (TTL) value for packets the RAC generates for RIP updates. TTL is a field in IP packets that limits their lifetime on the network. Each time a packet crosses a router, the router decrements the packet's TTL by 1. When the value reaches 0, the packet is discarded. Valid values are from **1** through **255**. The default is **64**.

parity

This asynchronous port parameter defines the type of parity that the asynchronous port uses. The options are **even**, **odd**, or **none**. The default is **none**.

password

This annex parameter modifies the RAC's administrative password. This password is used for access to the superuser CLI commands and for administrative access to the RAC. It overrides the CLI **lock** command and virtual CLI passwords. The string can contain from 0 through 15 characters.

If the RAC is configured with an IP address, the default administrative password is the RAC's IP address in dotted-decimal notation.

If the RAC is not yet configured with an IP address and the administrative password has not been modified (using either this parameter or the CLI **passwd** command), the default password is a null string ("").

If the RAC is not configured with an IP address and boots via MOP, IPX, or from flash ROM, the default password is a null string (""), and entering a carriage return at the *Password* prompt places you in superuser mode.

Even if **password** is not set, the default administrative password is required to access the superuser CLI commands.

passwd_limit

This annex parameter defines the maximum number of times a user can try to enter a password before the RAC resets the port. Entering **0** sets the limit to 3. Valid values are **0** through **10**. The default is **3**.

Changes to this parameter take effect immediately.

port_password

This asynchronous/synchronous port parameter defines an **asy**, **ta**, or **syn** port password for local password protection. You can use this password as a backup for host-based security if the security servers do not respond, or as an additional line of security after entering a user name password.

When using SecurID, set **port_password** to a null string ("") and do not set a port password in the **acp_passwd** file.

port_server_security

This asynchronous port parameter enables a host-based security policy for access to the port through the port server. When this parameter is enabled, only authorized users can access the port. A **Y** enables the parameter; an **N** disables it. The default is **N**.

ppp_acm

This asynchronous port parameter (async control mask) specifies which of the first 32 bytes (0x0 through 0x1F) of the ASCII character can be sent as clear text and which should be protocol-escaped.

The RAC uses the value of the **ppp_acm** parameter as its local mask. If the peer rejects **ppp_acm**, the RAC accepts the hint if it is a superset of the RAC's mask; otherwise, it uses the PPP default of 0xFFFFFFFF. The RAC accepts any mask from the peer. Values range from **0x00000000** through **0xffffffff**. The RAC default is **0x00000000**.

The **ppp_acm** parameter is a bit mask that is set as follows:

- **ppp_acm** for ASCII NUL (decimal 0) is 2 to the power of 0 = 0x00000001
- **ppp_acm** for ASCII SOH (decimal 1) is 2 to the power of 1 = 0x00000002
- **ppp_acm** for ASCII DC1 (decimal 17) is 2 to the power of 17 = 0x00020000
- **ppp_acm** for ASCII DC3 (decimal 19) is 2 to the power of 19 = 0x00080000

Thus, the mask for XON/XOFF (DC1 and DC3) equals the OR function of 0x00020000 and 0x00080000 (that is, 0x000a0000).

When the RAC sends an ACCM to the host, it follows this calculation to determine the initial value requested:

- The value set for **ppp_acm** (a 32-bit integer) is read in as the ACCM.
- If **input_flow_control** is set to **start/stop**, the following two additions are made:

  If **input_start_char** is 0 through 31 decimal, the bit indexed by this parameter is set in the ACCM.

  If **input_stop_char** is 0 through 31 decimal, the bit indexed by this parameter is set in the ACCM.
- If **output_flow_control** is set to **start/stop**, the following two additions are made:

  If **output_start_char** is 0 through 31 decimal, the bit indexed by this parameter is set in the ACCM.

  If **output_stop_char** is 0 through 31 decimal, the bit indexed by this parameter is set in the ACCM.

For example, the initial ACCM sent to the peer is 0x000A0001 if **ppp_acm** is set to 0x00000001 (that is, the ASCII NUL character will not be sent) and the following parameters are set as indicated:

input_flow_control	**start/stop**
input_start_char	**^S**
input_stop_char	**^Q**
output_flow_control	**start/stop**
output_start_char	**f**
output_stop_char	**h**

Because the output flow control parameters are outside the range of 0 through 31 decimal, they do not affect the ACCM.

The **na/admin** command **show port ppp_acm** still displays the **ppp_acm** setting. The CLI command **netstat -ip***nn*, where *nn* is the port number, displays the true mask (ACCM) value, that is, the value negotiated between the two PPP processes.

ppp_mru

This asynchronous/synchronous port parameter defines the maximum receive unit (MRU) that the RAC requests as its local MRU. If the MRU requested by the RAC is rejected (that is, a NAK packet is returned) and the remote hint is less than this value, the RAC accepts the hint; otherwise, the RAC requests the PPP default (**1500**). Valid values range are **64** through **1600**. The default is **1500**.

ppp_ncp

This asynchronous/synchronous port parameter specifies the network control protocols that run on the interface. The RAC negotiates for these protocols only. Valid settings are one or more of the following: **ipcp** (Internet Protocol Control Protocol), **atcp** (AppleTalk Control Protocol), **ipxcp** (Internet Packet Exchange Control Protocol), **mp** (Multilink PPP), **ccp** (Compression Control Protocol), and **all (**all of the protocols). Separate multiple protocols with commas. The default is **all**.

ppp_password_remote

This asynchronous/synchronous port parameter is a string defining a PPP port user's password. The string can contain from 0 through 16 characters. For security reasons, the RAC displays this parameter's value as **"<set>"** or **"<unset>."** The default is a null string (""), which the RAC displays as **"<unset>."**

ppp_sec_auto

This asynchronous/synchronous parameter, used in conjunction with the **ppp_security_protocol** parameter, allows you to use **auto_detect** mode for PPP clients whether or not the clients support PAP/CHAP. Valid values are **Y** and **N**.

If **ppp_sec_auto** is set to **Y** and **enable_security** is set to **Y**, and if the user accesses the RAC in CLI mode (for example, by entering a carriage return after dialing in) and then switches to **ppp** mode (by issuing the **ppp** command), the RAC treats the **ppp_security_protocol** as if it were set to **none**.

If **ppp_sec_auto** is set to **Y** and **enable_security** is set to **Y**, and if the RAC determines (via a **mode** parameter of **auto_detect**) that a dial-in user is using PPP, the RAC uses the current value of the **ppp_security_protocol** parameter.

The default for **ppp_sec_auto** is **N**, which specifies that no matter how the user is placed in **ppp** mode, the RAC uses the value of the **ppp_security_protocol** parameter.

ppp_security_protocol

This asynchronous/synchronous port parameter defines the security check for the client that the RAC requires before starting the network control protocol. If the RAC wants to use security and the client refuses, the RAC closes the link. Valid options are **chap** (Challenge-Handshake Protocol), **pap** (Password Authentication Protocol), **chap-pap**, and **none**.The default is **none**.

Specifying **chap-pap** allows the RAC to use **chap** or **pap**, depending on the capability of the client. The RAC tries to use **chap** first. If the client supports **chap** and fails authentication, the RAC does not attempt to use **pap**.

LCP requests for CHAP received by the RAC are always acknowledged, regardless of this parameter's setting.

ppp_username_remote

This asynchronous/synchronous port parameter is a string defining the user name by which the RAC identifies itself when the remote PPP peer asks for authentication. The string can contain from 0 through 15 characters. The default is a null string (""), which the RAC displays as **"<unset>."**

pref_dhcp1_host

This annex parameter specifies the IP address of the Dynamic Host Configuration Protocol (DHCP) server that the RAC tries to use as the primary source for DHCP services. DHCP enables dynamic IP addressing for remote access PPP clients. This eliminates the need to assign an IP address manually (and the subsequent need to reconfigure and reboot) each time a host is added or moved to a new subnet location. The default is **0.0.0.0**.

If **pref_dhcp1_addr** does not respond, the RAC tries to use the server specified by **pref_dhcp2_addr**. If **pref_dhcp2_addr** does not respond, and if the **dhcp_broadcast** parameter is set to **Y**, the RAC broadcasts for a DHCP server.

pref_dhcp2_host

This annex parameter specifies the IP address of the DHCP server that the RAC tries to use as a backup source for DHCP services when **pref_dhcp1_addr** is non-zero and does not respond. The default is **0.0.0.0**.

If **pref_dhcp2_addr** does not respond, and if the **dhcp_broadcast** parameter is set to **Y**, the RAC broadcasts for a DHCP server.

pref_dump_addr

This annex parameter specifies the IP address for the preferred dump host. This is the host to which the RAC first tries to dump. The default is **0.0.0.0**.

A dump is not sent if the address is set to the default value.

pref_load_addr

This annex parameter specifies the IP address for the preferred load host. This is the host from which the RAC first requests a load of its operational code. The default is **0.0.0.0**.

Set this address to the boot host’s IP address.

pref_mop_host

This annex parameter specifies the Ethernet address of the preferred MOP load or dump host. This address consists of six parts separated by dashes. Each part contains a hexadecimal value. The default value is **00-00-00-00-00-00**.

pref_name1_addr

This annex parameter defines the IP address of the primary name server. The default is **0.0.0.0**.

When a PC requests a primary name server address during IPCP negotiation, the RAC returns the value specified in **pref_name1_addr**.

If the value of **pref_name1_addr** is 0, the RAC rejects any request from a PC for a primary name server address.

To define the type of primary name server in use (**dns** or **ien_116**), set the **name_server_1** parameter.

pref_name2_addr

This annex parameter specifies the IP address of the secondary name server. The default is **0.0.0.0**.

When a PC requests a secondary name server address during IPCP negotiation, the RAC returns the value specified in **pref_name2_addr**.

If the value of **pref_name2_addr** is **0**, the RAC rejects any request from a PC for a secondary name server address.

To define the type of secondary name server in use (**dns** or **ien_116**), set the **name_server_2** parameter.

pref_secure1_host

This annex parameter specifies the IP address of the ACP security server or the RADIUS security server to which the RAC first sends requests. This parameter works only if the **enable_security** parameter is set to **Y**. The default is **0.0.0.0**.

pref_secure2_host

This annex parameter specifies the IP address of the host that is the backup ACP or RADIUS server if the host specified in **pref_secure1_host** is not available. This parameter works only if the **enable_security** parameter is set to **Y**. The default is **0.0.0.0**.

printer_host

This asynchronous port parameter specifies the IP address or fully qualified domain name of a host running a Berkeley-style *lpd* server. The **tn3270** command uses this server for the print-screen function.

printer_name

This asynchronous port parameter specifies the printer used by the **tn3270** command's print-screen function. You must enter a name listed in the **/etc/printcap** file on the remote host by setting the **printer_host** parameter.

prompt

This asynchronous port parameter defines a port-specific prompt string. The prompt string consists of displayable characters and embedded formatting codes. Each formatting code, which consists of a percent character (%) followed by a single character, is compressed and stored as a single character in nonvolatile memory. The maximum number of characters stored for the prompt string is 16.

Because each formatting code consists of two characters, the maximum string size is 32 characters. String sizes smaller than 32 characters are rejected as bad values if they cannot be stored as 16 characters in nonvolatile memory after the formatting codes are compressed into single characters. Table 3-7 on page 3-24 lists and describes these codes. The RAC parameter **cli_prompt** defines the default prompt.

radius_acct_level

This annex parameter specifies the level of RADIUS accounting to be used. Valid values are **basic** and **advanced**. If **basic** is specified, the only events, or Accounting Status Types (40), that are logged are Start, Stop, Accounting-On, and Accounting-Off. If **advanced** is specified, all events are logged. The default is **basic.**

radius_acct_port

This annex parameter specifies the UDP port on which the RADIUS accounting server is listening. Permissible values are **1813**, **1646**, or any Internet assigned UDP port. Port 1646 is used with older versions of RADIUS. The default is **1813**, as specified in RFC 2058.

radius_auth_port

This annex parameter specifies the UDP port on which the RADIUS authentication server is listening. Permissible values are **1812, 1645**, or any Internet assigned UDP port. Port 1645 is used with older versions of RADIUS. The default is **1812**, as specified in IETF RFC 2058.

radius_port_encoding

This annex parameter specifies the format in which the NAS-Port (5) attribute is communicated and logged to the RADIUS server. Valid values are **device** and **channel**. The default is **device**.

When set to **device**, this parameter displays the number of the port. For physical ports, this is a number from 1 through the total number of possible RAC ports of a given type. For virtual ports, the port number is represented as shown in Table 3-15.

Table 3-15. Port Numbers and Virtual Device Types

Port Number	Virtual Device Type
2000+*port_index*	VCLI and FTP
3000+*port_index*	Dialout
4000+*port_index*	Ethernet (en0)
5000+*port_index*	VPN (for MMP links)
6000+*port_index*	MP bundles

For example, if two users are connected via FTP at the same time, the port number of the first user to connect is 2001, and the port number of the second user is 2002.

When set to *channel*, the port number for physical ports is a five-digit decimal value of the form *twwcc*, where:

- *t* is the type of device: 1 for digital, 2 for analog
- *ww* is the number of the WAN interface: 01 or 02
- *cc* is the channel

For example, the first ISDN channel used on WAN 2 would be reported as 10201.

When set to **channel**, the port number for virtual ports is represented as shown in Table 3-16.

Table 3-16. Port Numbers and Virtual Channel Types

Port Number	Virtual Channel Type
200+*port_index*	VCLI and FTP
300+*port_index*	Dialout
400+*port_index*	Ethernet (en0)
500+*port_index*	VPN (for MMP links)
600+*port_index*	MP bundle

radius_retries

This annex parameter specifies the number of times the RAC sends access requests to the primary RADIUS server before trying the secondary server. If the attempts to send requests to the secondary server exceed the specified number of retries, the RAC rejects the user. Valid values for this parameter are **0** through **65535**. If the parameter is set to **0**, the RAC sends a request only once before switching to the secondary server. The default is **10**.

radius_secret

This annex parameter defines the key that the RAC and the RADIUS server use to verify the integrity of the messages they exchange. This key is required in order for RADIUS security to work. Set this parameter to a character string containing from 1 through 99 characters. It is strongly recommended that the string:

- Contain at least 16 characters
- Contain at least two non-alphanumeric characters
- Be difficult to guess

If **radius_secret** is preceded by **0x**, it is interpreted as a hexadecimal number. (In this case, the minimum number of characters is three.) For example, 0x123456 would be interpreted as three bytes: 0x12, 0x34, and 0x56.

The clear-text form of **radius_secret** should not be exposed on the network, at least not outside the firewall, if any. Methods for setting this parameter that protect it from exposure are:

- Using the host-based **na** utility (see *na Utility on page 2-1*)
- Using the RAC-based CLI **admin** command (see *admin on page 1-2*) from a console attached either to a System 5000™ hub containing a 5399 RAC or to a terminal connected directly to an 8000 RAC.

Methods for setting **radius_secret** that are not protected include:

- Using **admin** via a Telnet session
- Using Annex Manager
- Using an SNMP client

The RAC displays this parameter's value as **"<set>"** if a secret is entered or **"<unset>"** if a null string ("") is entered. If this parameter is not set, RADIUS authentication or accounting cannot take place. The default is a null string.

radius_timeout

This annex parameter specifies the number of seconds the RAC waits for a reply from a server to which it has sent an access request. Valid values are from **1** through **65535**. The default is **4**.

This parameter operates in conjunction with the **radius_retries** parameter. Each time the RAC tries to send an access request, the RAC waits the number of seconds specified by **radius_timeout**. For example, if **radius_timeout** is set to 4 and **radius_retries** is set to 3, the RAC waits 4 seconds after the first request, 4 seconds after the second request, and 4 seconds after the third request. If the third timeout period is exceeded, the user is not authenticated.

The **radius_timeout** parameter can also be used to time out RAC accounting requests. However, in the case of accounting requests, the RAC waits longer and longer with each request, to adjust for network traffic. Specifically, the RAC determines the actual timeout by multiplying the value of **radius_timeout** by the number of the request attempt. For example, if **radius_timeout** is set to 4 and **radius_retries** is set to 3, the RAC waits 4 seconds after it sends the first request, 8 seconds after it sends the second request, and 12 seconds after it sends the third (and final) request.

redisplay_line

This asynchronous port parameter defines the reprint line character for CLI users. The value must be a control character sequence. The default is **CTRL-R** (^R).

remote_address

This WAN B/DS0 channel parameter associates IP addresses with B or DS0 channels. This parameter works only when the **mode** parameter is set to **slip** or **ppp**, or when it is set to **auto_detect** and PPP is detected. Table 3-17 describes the arguments. The syntax is one of the following:

set wan b[=*channel_range***|all**] **remote_address** *ip_addr* [*increment*]

set wan ds0[=*channel_range***|all**] **remote_address** *ip_addr* [*increment*]

If the **address_origin** parameter is set to **acp**, the remote address specified in the **acp_dialup** file overrides the remote address described here.

Table 3-17. Arguments for the remote_address Parameter

Argument	Description
channel_range	An integer specifying a single B or DS0 channel number, a list of B or DS0 channel numbers separated by commas, a range of B or DS0 channel numbers separated by a hyphen, or the keyword **all**, which specifies all B or DS0 channels. Valid ISDN PRI channel numbers are **1** through **23** (in the U.S.) and **1** through **30** (in Europe). Valid CAS channel numbers are **1** through **24** (for Channelized T1) and **1** through **30** (for Channelized E1, R1, and R2). The default is **all**.
ip_addr	The IP address to be assigned to the B or DS0 channel if only one channel is specified in *channel_range*, or the IP address to be assigned to the first channel in *channel_range*.
increment	An integer specifying how *ip_addr* is to be incremented to generate IP addresses automatically for the channels in *channel_range*. For example, if *ip_addr* is set to **132.245.22.2**, *increment* is set to **2**, and **3** B channels are specified, the RAC assigns IP addresses 132.245.22.2, 132.245.22.4, and 132.245.22.6 to the B channels. The default is **0**.

reset_idle_time_on

This asynchronous port parameter defines whether **input** or **output** resets the idle timer. The idle time is the time interval between activity and inactivity on the device. This parameter is used with the **who** command. Valid options are **input** and **output**. The default is **input**.

retrans_limit

This annex parameter defines the number of times the RAC retransmits a packet before notifying the LAT user about a network failure. Valid values are from **4** through **120**. The default value is **8**.

ringback

This WAN B/DS0 channel parameter specifies whether or not an audible ring is sent to the telco's Central Office for incoming calls. Valid values are **Y** and **N**. The default is **Y.**

rip_accept

This interface parameter defines the networks for which the RAC accepts advertised routes. Table 3-18 lists the valid options; the default is **all**.

Table 3-18. Valid Options for the rip_accept Parameter

Option	Description
access_spec	Uses the form [**include** | **exclude**] *network_list* where **include** means accept RIP updates only for the networks in *network_list*, and **exclude** means accept all RIP updates except for those in *network_list*. You can list up to eight network IP addresses in *network_list*.
none	No RIP updates are accepted over the interface.
all	RIP updates for all networks are accepted.

rip_advertise

This interface parameter defines the networks for which the RAC will advertise routes. Table 3-19 lists the valid options; the default is **all**.

Table 3-19. Valid Options for the rip_advertise Parameter

Option	Description
access_spec	Uses the form [**include** | **exclude**] *network_list* where **include** means advertise only the networks in *network_list*, and **exclude** means advertise all networks except those in *network_list*. The list can contain up to eight network addresses.
none	Turns off advertising for the interface.
all	Advertises all networks over the interface.

rip_auth

This annex parameter is a character string specifying the password that controls authentication for RIP 2 packets. The string can contain from 1 through 16 characters. The RAC displays this parameter's value as "<set>" if a password is entered or "<unset>" if a null string is entered. When "<unset>," authentication is turned off and all RIP packets are accepted. The default is a null string (""), which the RAC displays as **"<unset>."**

rip_default_route

This interface parameter allows the RAC to advertise that it is the default router. Valid values are **0** through **15**, or **off**. A value of **1** though **15** indicates the hop count that will be advertised. A value of **0** or **off** turns off the advertisement. The default is **off**.

rip_horizon

This interface parameter controls the split horizon algorithm for RIP. Table 3-20 describes the valid options. The default is **poison**.

Table 3-20. Valid Options for the rip_horizon Parameter

Option	Description
off	Disables split horizon.
split	Enables split horizon without poison reverse.
poison	Enables split horizon with poison reverse.

rip_next_hop

This interface parameter specifies whether or not the next hop value is included in RIP Version 2 advertisements. Valid options are **never**, **needed**, or **always**. The default is **needed**.

rip_recv_version

This interface parameter controls the RIP version(s) that the RAC accepts. Table 3-21 describes the valid options. The default is **both**.

Table 3-21. Valid Options for the rip_recv_version Parameter

Option	Description
1	Only Version 1 packets are accepted.
2	Only Version 2 packets are accepted.
both	Both versions are accepted.

rip_routers

This annex parameter lets you force RIP to direct periodic RIP updates to a router list rather than broadcasting updates. Valid values are the IP addresses of up to eight directly reachable routers. The RAC ignores any address that is not on an attached subnet. Specifying the default, **all**, restores broadcasting.

rip_send_version

This interface parameter controls the RIP version(s) that the RAC sends over the IP interfaces. Table 3-22 describes the valid options. The default is **compatibility**.

Table 3-22. Valid Options for the rip_send_version Parameter

Option	Description
1	Version 1 packets are sent to the broadcast address.
2	Version 2 packets are sent to the RIP multicast address.
compatibility	Version 2 packets are sent to the broadcast address.

rip_sub_accept

This interface parameter controls whether or not subnet routes are accepted over the SLIP, PPP, and Ethernet interfaces. When this parameter is enabled, subnet routes are accepted; when it is disabled, subnet routes are rejected. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

rip_sub_advertise

This interface parameter controls whether or not the RAC advertises subnet routes over the SLIP, PPP, and Ethernet interfaces. When this parameter is enabled, subnet routes are advertised; when it is disabled, subnet routes are not advertised. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

routed

This annex parameter determines whether or not the RIP routing daemon is enabled. When this parameter is enabled, the RAC performs both active and passive RIP routing. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

rwhod

This annex parameter determines whether or not the RAC listens for RWHO broadcasts when it builds the host table. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

security_broadcast

This annex parameter determines whether or not the RAC broadcasts for security validation if the preferred security servers are not available. When this parameter is enabled, the RAC broadcasts for security; when it is disabled, the RAC does not broadcast for security. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

server_capability

This annex parameter defines the RAC as a file server host. A RAC can provide operational code only for another RAC. For example, a Model 8000 RAC can provide operational code for a Model 5399 RAC, but neither can provide code for a Remote Annex 6300. Table 3-23 describes the valid options; the default is **none**.

Table 3-23. Valid Options for the server_capability Parameter

Option	Description
all	The RAC is a file server for the configuration, operational image, and message-of-the-day files.
config	The RAC is a file server for configuration files.
image	The RAC is a file server for operational code.
motd	The RAC is a file server for the message-of-the-day file.
none	The RAC is not a file server.

server_name

This annex parameter names the RAC in the LAT protocol. The name should match the NMS host’s node name used in the HIC configuration file. The name can contain from 1 to 16 characters. The default value is the physical Ethernet address, represented as a hexadecimal value, appended to the string **LAT_** (for example, *LAT_080002BF0020*).

service_limit

This annex parameter defines the maximum number of LAT services that the RAC can maintain in its local service table. When the table is full, the RAC removes the service that has been idle longest. If all services are busy and the table is full, the RAC discards a new service. Valid values are from **16** through **2048**. The default is **256**.

ses_threshold

This annex parameter specifies the number of Severely Errored Seconds that must occur within a 15-minute interval on a WAN module before an SNMP wanEsfThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

session_limit

This annex parameter specifies the maximum number of active sessions the RAC allows at one time. Valid values are **1** through **1152** or **none** (entering **none** sets the value to **1152**). The default is **1152**.

short_break

This asynchronous port parameter allows a RAC to return a user to the CLI prompt after receiving a break of less than two seconds. This occurs only at the CLI level. A **Y** enables this parameter; an **N** disables it. The default is **Y**.

sigproto

This WAN B/DS0 parameter is used with CAS channels that have switch type UST1. The parameter defines the inbound and outbound signaling protocols supported by each DS0 channel. Valid values are any one of the following:

loop_in (loop start inbound)
loop_out (loop start outbound)
loop_bi (loop start bidirectional)

wink_in (wink start inbound)
wink_out (wink start outbound)
wink_bi (wink start bidirectional)

gnd_in (ground start inbound)
gnd_out (ground start outbound)
gnd_bi (ground start bidirectional)

imm_in (immediate start inbound)
imm_out (immediate start outbound)

none (no signalling protocols; this busies out the specified DS0s until you set **sigproto** again)

When you display the **sigproto** values, you see only the single value you specified. For example, **wink_in** displays as *wink_in*, not *wink_in, none*.

The default is **none**.

slip_mtu_size

This asynchronous port parameter sets the maximum transmission unit (MTU) size on a SLIP/CSLIP port. This parameter forces the SLIP interface to use **large** (1006) or **small** (256) MTUs. The default is **small**.

slip_no_icmp

This asynchronous port parameter controls whether or not the RAC discards any ICMP packets directed to the SLIP link. When this parameter is enabled, the RAC reduces unnecessary traffic and messages over the SLIP link. A **Y** enables this parameter; an **N** disables it. The default is **N**.

slip_ppp_security

This asynchronous port parameter controls dial-in SLIP or PPP access. When this parameter and **enable_security** are set to **Y**, the RAC prompts for a user name and password when a user attempts to execute a **slip** or **ppp** command at the CLI prompt.

This parameter also determines whether the RAC uses local or host-based (server) security to authenticate PPP users. If **slip_ppp_security** is set to **Y**, the RAC consults the ACP or RADIUS server for authentication. A **Y** enables this parameter; an **N** disables it. The default is **N**, which causes the RAC to use the global port parameters **user_name** and **port_password** to authenticate PPP.

The type of server (ACP or RADIUS) is determined by the annex **auth_protocol** parameter, which defaults to ACP.

The type of PPP authentication (CHAP, PAP, or CHAP-PAP) used is determined by the value of the **ppp_security_protocol** parameter.

slip_tos

This asynchronous port parameter allows a RAC to send interactive traffic (**telnet**, **rlogin**, and **ftp** control sessions) before sending any other traffic. This parameter provides a type-of-service based SLIP queuing. A **Y** enables this parameter; an **N** disables it. The default is **N**.

stop_bits

This asynchronous port parameter specifies the number of stop bits for a port. Valid values are **1**, **1.5**, or **2**. The default is **1**.

subnet_mask

This annex parameter defines the RAC’s IP subnet mask. It is used to divide a network into subnets. The parameter’s default is based on the network portion of the RAC’s IP address.

Setting this parameter incorrectly can cause routing problems.

subnet_mask (port)

This port parameter defines the subnet mask for an asynchronous SLIP port or a synchronous PPP interface. Typically, you use this parameter to divide a network into subnets. Specifying **0.0.0.0**, which is the default, sets the subnet mask to **255.255.255.255**, which denotes a host address that is not subnetted.

Setting this parameter incorrectly can cause routing problems.

sys_location

This annex parameter supplies LAT host location or identification information. The string can contain 0 through 32 characters. The default is a **null string** ("").

switch_type

This required WAN parameter is a string specifying the physical device, located at the telco's Central Office, that provides the RAC with PRI or CAS (Channelized T1, E1, etc.) data transmission service. Valid PRI switch types are shown in Table 3-24. Valid CAS switch types are shown in Table 3-25.The default switch type is a **null string** ("").

The switch type parameter is not case-sensitive.

Table 3-24. Valid PRI switch_type Values

Protocol	switch_type	Used In
T1/PRI	AT9	North America; AT&T 5ESS#9 switch
	AT4	North America; AT&T 4ESS support
	DMS	North America; Nortel's DMS100 switch
	NI2	North America; a switch supporting National ISDN2
E1/PRI	ETS	Europe; ETSI
	ETS-NCRC4	Europe; ETSI without CRC
	AU1	Australia

Table 3-25. Valid CAS switch_type Values

Protocol	switch_type	Used In
Channelized T1	UST1 HKT1	North America Hong Kong
Channelized T1 - R1	TWT1R1	Taiwan
Channelized E1 - P7	SWE1P7	Used in Sweden
Channelized E1 - R2	ANE1R2 ARE1R2 BBE1R2 BRE1R2 CNE1R2 KRE1R2 IDE1R2 ILE1R2 NZE1R2 PHE1R2 MYE1R2 MXE1R2 THE1R2 TRE1R2	St. Martin Argentina CCITT BlueBookR2 (used in most of Europe); also used in Croatia, Chile, and South Africa Brazil China Korea Indonesia Israel New Zealand The Philippines Malaysia Mexico Thailand Turkey

For the most up-to-date information on switch types, consult the Release Notes shipped with this product.

syslog_facility

This annex parameter defines the local facility to which the UNIX *syslogd* daemon sends RAC syslog messages. Valid options are **log_local0** through **log_local7**. The default is **log_local7**.

If the host to which messages are logged does not support 4.3BSD logging, this parameter is ignored and messages are logged by priority level (defined by **syslog_mask**).

syslog_host

This annex parameter defines the IP address of the host that logs RAC messages. The default, **0.0.0.0**, causes the RAC to log messages to the console port that is attached either to the 8000 RAC or to the System 5000 hub containing the 5399 RAC.

syslog_mask

This annex parameter defines the priority levels that the RAC logs. The options are **all**, **none**, or a combination of levels separated by commas. The default, **none**, disables logging. Table 3-26 lists the levels in priority order.

Table 3-26. Priority Levels for the syslog_mask Parameter

Priority Level	Description
emergency	Hardware failures.
alert	All RAC reboots.
critical	Configuration and initialization problems, such as format errors in the configuration file or lack of memory.
error	All line initialization errors, including CLI.
warning	Indications of minor problems.
notice	Time server queries and information about responses.
info	Start and end CLI sessions and RAC jobs created by the **rlogin**, **telnet**, **connect**, **ping**, and **tap** commands.
debug	Activate and exit all RAC processes.

syslog_port

This parameter is displayed but does not apply to the RAC.

tcp_keepalive (annex)

This annex parameter specifies the length of time a TCP connection must be idle before a RAC sends keep-alive messages. A keep-alive message contains no data but solicits an acknowledgment from the other end of a connection to determine whether the connection is still active. If the recipient does not acknowledge the message after eight retries, the RAC drops the connection. Valid values are **0** through **255** (minutes). A value of **0** sets the keep-alive time to **120** minutes, which is the default; a value of **255** disables the keep-alive mechanism. The **tcp_keepalive** parameter for asynchronous ports overrides this parameter for those individual ports.

tcp_keepalive (port)

This asynchronous port parameter specifies the length of time a TCP connection must be idle on the global port. This parameter overrides the RAC **tcp_keepalive** parameter for connections to the host from adaptive, CLI, and dedicated ports, and for connections from the host to slave and adaptive ports. Valid values are **0** through **255** (minutes). The default is **0**. Entering **0** specifies that the keep-alive time is the value set in the annex **tcp_keepalive** parameter; entering **255** disables the keep-alive mechanism for the port.

telnet_crlf

This asynchronous port parameter converts a carriage return in a Telnet session to a carriage return followed by a line feed. When **telnet_crlf** is disabled, a carriage return translates to a carriage return followed by a null string. A **Y** enables this parameter; an **N** disables it. The default is **N**.

telnet_escape

This asynchronous port parameter defines the character that returns a CLI user to the *telnet* prompt. Setting this parameter to **U** disables the Telnet escape character. The default is **CTRL-**] (^]).

term_var

This asynchronous port parameter identifies the type of terminal using the CLI connection. You must enter a valid terminal type for the host. The RAC passes the terminal type setting to the host. The string can contain from 0 through 16 characters. The default is a **null string** ("").

tftp_dump_name

This annex parameter provides the file name used to dump a RAC's core image via **tftp** if the RAC operational image and **erpcd** fail. The parameter must include the entire path of the dump file, including parent directories. The file you enter must have read and write permissions.

tftp_load_dir

This annex parameter defines the string that precedes all files (for example, image name, configuration, and **motd** files) when you boot a RAC via **tftp**. This string's value is determined by the system serving the **tftp** requests. This string does not precede the **tftp_dump_name**.

time_broadcast

This annex parameter defines whether the RAC broadcasts for the time if the preferred load host is not available or does not provide a time server. A **Y** enables this parameter; an **N** disables it. The default is **N**.

time_server

This annex parameter determines whether or not the RAC queries for time service. Table 3-27 lists the IP addresses to which the network administrator can set this parameter. The default is **0.0.0.0**.

Table 3-27. IP Addresses for the time_server Parameter

Address	Description
loopback address	Do not query for time service.
0.0.0.0	Query the boot host for time service.
127.0.0.1	Do not send out direct time queries.
host address	Send queries to the given host address.
broadcast address	Send queries to given broadcast address; may require *direct broadcast* service on the network routers. This setting ignores the **time_broadcast** parameter setting.

timezone_minuteswest

This annex parameter defines the time zone in which the RAC resides. Enter a positive number of minutes for time zones west of GMT, or a negative number for time zones east of GMT. For example, enter *300* for U.S. Eastern Standard Time, which is five hours west of GMT, or *-60* for Paris, which is one hour east of GMT. The default is **300**.

tmux_delay

This annex parameter defines the maximum number of milliseconds during which small packets can accumulate to form larger packets. When the time expires, the RAC sends the multiplexed packet. Valid values are **0** through **255** (milliseconds). Entering **0** sets this parameter to 20. The default is **20**.

tmux_enable

This annex parameter controls whether or not a RAC uses TMux to multiplex small TCP packets into a single IP packet. This parameter works only if the host supports TMux. When this parameter is enabled, and the host does not support TMux, the RAC will not support multiplexing. A **Y** enables this parameter; an **N** disables it. The default is **N**.

tmux_max_host

This annex parameter specifies the maximum number of host addresses allowed in the TMux address table. If the number of host addresses exceeds the value entered here, the RAC discards the oldest entry. Valid values are **10** through **255**; the default is **64**.

tmux_max_mpx

This annex parameter specifies the largest user packet that can be placed in a TMux packet. The RAC does not multiplex larger packets, but passes them directly to the IP layer. Valid values are **5** through **65535**; the default is **700**.

toggle_output

This asynchronous port parameter defines the character that flushes the output buffer for CLI users. The flush character must be CTRL-*X* (^*X*), where *X* represents an alphanumeric value (not case-sensitive). Pressing this character flushes the output buffer. The default is **CTRL-O** (^O).

type_of_modem

This parameter is used to create a user-defined modem type for use in SPBs. This user-defined modem type can be used to handle particular call types. The **type_of_modem** parameter corresponds to a **type_of_modem** entry in the **digital_modem** section of the RAC configuration file (the default for which is **config.annex**). The **type_of_modem** entry in the configuration file is used to set modem configuration parameters to customize the digital modems.

uas_threshold

This annex parameter specifies the number of Unavailable Seconds that must occur within a 15-minute period on a WAN module before an SNMP wanUasThreshTrap is sent. Valid values are **0** through **65535**. The default is **0**, which disables the trap.

user_name

This asynchronous/synchronous port parameter defines an asynchronous or synchronous port's user name as a string. The CLI **who** command displays this value; the CLI **rlogin** command passes this value to a host. The default is a null string (**""**).

unexpected_trap_inc

This annex parameter specifies the number of unexpected call-disconnect events that must occur before an unexpected-call-disconnect SNMP trap is sent. Valid values are **0** through **65535**. The default is **0**, which disables unexpected-call-disconnect traps.

v120_mru

This asynchronous parameter applies exclusively to V.120 (TA) connections. It allows you to change the number of bytes allowed in an incoming V.120 frame if your TA cannot handle the default of **256** bytes. Valid values are **30** through **260**.

vcli_groups

This annex parameter specifies which LAT remote group code is assigned to virtual CLI users. All virtual CLI users have the same group code. Values are specified as a series of numbers separated by commas (for example, **1,5,7**) or a range of numbers separated by a dash (for example, **200-255**). Following the series or range, specify the keyword **enable** or **disable**. Valid values are **all**, **none**, or a number from **0** through **255**. The default is **none enable**.

vcli_inactivity

This annex parameter specifies the number of minutes that a virtual CLI connection can remain inactive. If the number of minutes is exceeded, the RAC terminates the virtual CLI connection. Valid values are **1** through **255** or **off**. The default is **off**, which specifies that no time limit is imposed on VCLI connections.

vcli_password

This annex parameter is a string defining a password required for virtual CLI connections to the RAC. The string can contain from 0 through 15 characters. This parameter is useful for local password protection and as a backup to host-based security. For local password protection, set the **enable_security** parameter to **Y**, set the **vcli_security** parameter to **N**, and define a password for this parameter. As a backup for host-based security, setting this parameter causes the RAC to request a password on a virtual CLI connection whenever the security server does not respond. The default is a null string (""), which the RAC displays as **"<unset>."**

Changes to this parameter take effect immediately.

vcli_security

This annex parameter enables user validation on virtual CLI connections to and from the RAC for the duration of the connection. When this parameter is enabled, the RAC enables connection security for all virtual CLI connections and executes the same user validation, including user name and password, that it uses with CLI security on asynchronous ports. This parameter works with host-based security only when the **enable_security** parameter is set to **Y**. A **Y** enables this parameter; an **N** disables it. The default is **N**.

zone

This annex parameter defines the AppleTalk zone name that the RAC uses at startup. The string can contain from **0** through **32** characters. You must separate zone names with spaces (for example, *general pubs lab*). To escape embedded spaces, use the backslash (\) character. The default is a **null string** ("").

Chapter 4
erpcd

The **erpcd**, or expedited remote procedure call daemon, responds to boot, dump, and ACP security requests. This daemon contains two programs:

- **bfs**, the block file server used to access host files and dump RAC images.
- **acp**, the Access Control Protocol program for host-based security requests.

erpcd Arguments

Table 4-1 lists the arguments for **erpcd**. The syntax is:

```
/etc/erpcd [ [-D[level]] [-c [maxnumber] [-d [udpport][-f [directory]\
     [-p][-s [directory] [-u [filename] [-b [max_con] [-x [max_total]\
     [-g [period]] [- [l, L]] [- [a, A]] [-n] [- [t, T]] [-v [version]]
```

When operational code is downloaded to RACs, a minimum of one host, accessible to a RAC, must be running **erpcd** with the **bfs** program enabled. A UDP port (121) for **erpcd** must be defined in the services database and the **eservices** file must be configured properly.

For expert *C* coders only: the host **erpcd** daemon can implement macros, extended commands, command logging, and arbitrary security restrictions through a set of interface routines to the RAC CLI. These are documented in the **acp_policy.doc** file; the **acp_policy.c** file contains examples. For details on implementing code changes, see the document *Managing Remote Access Concentrators Using Command Line Interfaces.*

Table 4-1. Arguments for erpcd

Argument	Description
-D*level*	Restarts **erpcd** in test mode on the load server host; does not detach from the *tty* and it prints out extensive debugging information. Entering a debugging *level* increases the amount of debugging information. For each *level*, specify a **D**. For example **-DD** specifies 2 levels. Note that there is no space between the **D**s.
-c *maxnumber*	Specifies the maximum number of child processes that **erpcd** can create (for handling simultaneous requests).
-d *udpport*	Specifies a UDP port number; the default is 121.
-f *directory*	Specifies the location of the **bfs** files (load/dump); defaults to the defined **bfs** directory (usually **/usr/spool/erpcd/bfs**).
-p	Prints the daemon's process ID to standard output so that automatic mechanisms can start or stop this process.
-s *directory*	Specifies the location of the security files (**acp_*** and **eservices**); defaults to the RAC installation directory (usually **/etc**).
-u *filename*	Invokes the **acp_userinfo** file syntax checker on the file designated by *filename*. If *filename* is omitted, **stdin** is used. Running this option does not interfere with any other **erpcd** running on the system. Parsing errors are printed on **stderr**. Error messages are in the form: <*filename*>: line <*number*>: <*severity*>: <*description*> where <*filename*> is the name of the file, <*number*> designates the line on which the error occurs, <*severity*> is either an *error* or a *warning* (*error* indicates there is a serious parsing error; *warning* indicates the parser remedied the situation by conversion), and <*description*> describes the error.

(continued on next page)

Table 4-1.Arguments for erpcd (continued)

Argument	Description
-b *max_con*	Specifies the number of consecutive login failures a user is permitted before being blacklisted. Valid values are **0** through **8**. A value of **0** enables blacklisting upon any login failure (not recommended). This value can also be set via the MAX_BL_CON variable in **acp_policy.h**. The default, as pre-set via MAX_BL_CON, is **5**. If MAX_BL_CON is undefined and you do not specify **-b max_con**, ACP never blacklists based on consecutive login failures.
-x *max_total*	Specifies the number of non-consecutive login failures a user is permitted before being blacklisted. Valid values are **0** through **20**. A value of **0** enables blacklisting upon any login failure (not recommended). This value can also be set via the MAX_BL_NONCON variable in **acp_policy.h**. The default, as pre-set by MAX_BL_NONCON, is **10**. If MAX_BL_NONCON is undefined and you do not specify **-x max_total**, ACP never blacklists based on consecutive login failures.
-g *period*	Specifies the time period, in weeks, over which *max_total* is applied. Login failures that occurred more than this number of weeks ago do not count toward blacklisting. Valid values are **0** through **52**. This value can also be set via the MAX_BL_NONCON variable in **acp_policy.h**. The default, as pre-set via MAX_BL_PERIOD, is **26**. If MAX_BL_PERIOD is undefined or is set to **0**, MAX_BL_NONCON is effectively disabled.
-lL	Directs ACP logfile information to syslog. **-l** turns it off; **-L** turns it on.
-aA	ACP logfile. **-a** turns it off; **-A** turns it on.

(continued on next page)

Table 4-1. Arguments for erpcd (continued)

Argument	Description
-n	Uses a host name instead of an IP address in logfile.
-tT	Stamps ACP log information with the date/seconds; **-t** uses seconds;**-T** uses standard time format.
-v	Displays the software version number.

The spy Command

You can use the UNIX **spy** command to determine which version of **erpcd** is running on the host machine. The syntax is:

spy [*hostname* | *ip address*]

Enter either a host's name or its IP address as an argument. When issued without arguments, **spy** queries the current host and returns the version of **erpcd** running there:

```
Trying host 132.245.33.7...
Version: I14.0.16 (April 12th, 1997), Pid 20524, Children:
     0 out of 25
```

When a hostname or IP address is given as an argument, the same message is returned, substituting the corresponding IP address for that of the current host. The message always returns the IP address of the specified host, even if the hostname is given as the argument.

If the specified host is unreachable, this message is returned:

```
Unable to get IP address from host hostname
```

If **erpcd** is not running on the specified host, this message is returned:

```
No reply from host
```

If an earlier version of **erpcd** is running on the specified host, this message is returned:

```
ERPCD version too old to support spy request
```

Index