Managing Remote Access Concentrators Using Command Line Interfaces

Marketing Release 5.1

Part No. 118357-A Rev. A
September 1997

Bay Networks

Bay Networks

4401 Great America Parkway
Santa Clara, CA 95054

8 Federal Street
Billerica, MA 01821

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.

Contents

About This Guide

Before You Begin . . . xxiii
Conventions . . . xxiii
Acronyms . . . xxiv
Ordering Bay Networks Publications . . . xxv
Bay Networks Customer Service . . . xxv
How to Get Help . . . xxvi

Chapter 1
Introduction to Remote Access Concentrators

RAC Overview . . . 1-1
Dual WAN Interfaces . . . 1-2
- PRI . . . 1-2
- CAS . . . 1-3
- Channel Allocation . . . 1-3

Multi-Protocol Support . . . 1-4
Typical Network Configurations . . . 1-5
- Typical PRI Configurations . . . 1-5
- Typical CAS Configuration . . . 1-8
- Mixed RAC/Remote Annex Environment . . . 1-9

Configuration Prerequisites . . . 1-10
Parameter Configuration . . . 1-10
- RAC Management Tools . . . 1-11
- Parameter Types . . . 1-12

Loading Configuration Files from Hosts . . . 1-12

Chapter 2
Using the CLIs

The Command Line Interface . . . 2-1
- Command Syntax . . . 2-2
- Squelch . . . 2-2
- CLI Commands . . . 2-3

The na and admin Utilities . . . 2-11
- Command Notation . . . 2-11
- Command Syntax . . . 2-12

Chapter 3
Using the Configuration File

The Local File System . . . 3-1
Configuring RAC Parameters . . . 3-2
- Using the na Utility . . . 3-3
- Using the CLI admin Command . . . 3-7

Customizing the RAC Environment . . . 3-10
- Setting the CLI Prompt . . . 3-10
- Setting a Limit on Virtual CLI Connections . . . 3-12
- Setting Up the Configuration File . . . 3-12
- Setting Up the motd File . . . 3-12
- Using RIP . . . 3-13

Setting the IP Encapsulation Type . . . 3-13
Configuring LAT Services . . . 3-14
Parsing the Configuration File . . . 3-14
File Sections . . . 3-15
Include Statement . . . 3-15
Setting Up the Configuration File . . . 3-15
Creating %gateway Entries in the Configuration File . . . 3-19
Creating %macro Entries in the Configuration File . . . 3-27
Creating %service Entries in the Configuration File . . . 3-37
Creating %rotary Entries in the Configuration File . . . 3-39
Creating %digital_modem Entries in the Configuration File . . . 3-39
Creating %dialout Entries in the Configuration File . . . 3-39
Creating %wan Entries in the Configuration File . . . 3-52
Automated Firmware Download (AFD) . . . 3-52
Creating %gateway Entries for AFD . . . 3-53
Console Port Status Messages . . . 3-54
CLI stats Command . . . 3-55
LEDs . . . 3-56
Syslog Messages . . . 3-56
Error Handling . . . 3-57

Chapter 4
Configuring Hosts and Servers

Accessing 4.2BSD Hosts . . . 4-1
RAC Internet Addressing . . . 4-2
The Internet (IP) Address . . . 4-2
The Broadcast Address . . . 4-2
The Subnet Mask . . . 4-2
Using Event Logging . . . 4-3
Setting Up the File Server . . . 4-5
Installing Software Using bfs . . . 4-5
Installing Software Using the tftp Protocol . . . 4-6
Multiple Server Hosts . . . 4-6
Booting and Dumping . . . 4-7
Setting the Preferred Load Host . . . 4-7
Dump Host Services . . . 4-8
Setting the Load-Dump Sequence . . . 4-10
Setting a RAC as a Load Server . . . 4-10
Disabling Broadcasting for Files During a Boot . . . 4-11
Self-Booting . . . 4-11
Using the Trivial File Transfer Protocol . . . 4-12
Using the RAC ftp Daemon . . . 4-13
Using a Time Server . . . 4-13
Installing a Time Server . . . 4-15
Using Name Servers . . . 4-15
Defining Name Servers . . . 4-16
Using the RWHO Protocol . . . 4-19
Managing the Size of the Host Table . . . 4-21
Minimum Uniqueness . . . 4-21

Configuring Name Servers 4-22
Using RAC Security 4-23
Installing the ACE/Server Software 4-23
Configuring LAT Services 4-23
Advertised Services 4-24
Learned Services 4-25
Group Codes 4-25
Accessing LAT Services 4-25
Reverse LAT 4-27
Reverse LAT vcli 4-27
Telnet-to-LAT Gateway 4-28
LAT-to-Telnet Gateway 4-30
Data-b Slot Support for LAT 4-31
Miscellaneous LAT Parameters 4-32

Chapter 5
Configuring the WAN Interfaces, Global Ports, and Sessions

Understanding Call Delivery 5-1
Using the Default Call Configuration 5-2
Setting WAN Interface Parameters 5-3
Setting Generic WAN Interface Parameters 5-3
Channel Parameters 5-14
Understanding Internal Port Handling 5-18
Internal Port Types 5-18
Using Global Port Parameters 5-19
Displaying Global Port Parameters 5-19
Changing Global Port Parameter Values 5-23
Overriding Global Port Parameter Values 5-24
Understanding WAN Sessions 5-24
Configuring Session Parameter Blocks 5-25
How SPBs Are Scanned 5-26
SPB Fields 5-26
SPB Configuration Procedure 5-32
Sample SPBs 5-32
Setting the Mode Parameter 5-33
Configuring CLI Sessions 5-35
Port Differences for RACs and Remote Annexes 5-37
The Port Server 5-38
TCP Port Numbers 5-38
Virtual CLI (VCLI) Connections 5-38
Configuring Security 5-39
Rotaries 5-41
Configuring Rotaries 5-41
Rotary Example 5-42
Dial-Up Networking 5-49
Dynamic Dialing 5-50
Network Inactivity 5-51
Enabling Dynamic Dialing 5-51

Sample Configurations for Dial-Out Routing 5-55
Displaying Dynamic Dialing Routes in the Routing Table 5-59

Chapter 6
Configuring Security

Enabling Security 6-1
Setting Security-Related Parameters 6-1
Configuring Local Security 6-2
Local Password Protection 6-2
Implementing Local Virtual CLI Password Protection 6-2
Administrative Password 6-4
Protecting the Superuser CLI 6-5
Protecting Resources from Unauthorized Access 6-6
Protecting the na Utility from Unauthorized Access 6-6
Enhancing Password Security 6-7
Overview of Password History and Aging 6-7
Enabling and Configuring Password Histories 6-8
Overview of Blacklisting 6-11
Configuring Blacklisting 6-12
Viewing and Managing the acp_dbm Database 6-14
Deleting the acp_dbm Database 6-15
Configuring RADIUS Security 6-17
Using erpcd As a Proxy RADIUS Client 6-18
Using the RAC's Native RADIUS Client 6-18
RADIUS Parameters 6-18
RADIUS Attributes 6-20
The RADIUS Dictionary File 6-23
Configuring RAC Functions Using RADIUS 6-25
CLI Scripting 6-29
Configuring ACP Security 6-48
Guidelines for Creating ACP-Related Files 6-48
Information for Users of Remote Annexes 6-49
Editing Existing ACP Files 6-49
About Host-Based Security 6-50
Overview of Host-Based Security 6-50
Basic ACP Configuration 6-52
Configuring the Security Server 6-57
Setting Up a Security Server 6-57
Specifying the Security Hosts 6-58
Disabling Broadcasting for Security Servers 6-58
Encrypting Security Messages 6-59
Creating the acp_keys File 6-59
Defining Security Profiles 6-61
Overview of Security Profile Criteria 6-61
Overview of Files Used to Define Security Profiles 6-68
Creating User Groups 6-69
Configuring the acp_regime File 6-70
Creating User Password Files 6-72
Creating the acp_userinfo File 6-74

Limiting Access to Hosts via acp_restrict 6-94
Using include Files in the acp_userinfo File 6-99
Modifying the Supplied Security Application 6-99
Disabling User Name and Password Validation 6-100
Linking NIS Password File Verification to ACP 6-100
Modifying Message Formats in the ACP Log File 6-101
Changing the Expected File Names Used by ACP 6-102
Locking the ACP Log File 6-106
Masking CLI Commands 6-107
Modifying the Code 6-109
Re-Compiling erpcd 6-110
Using the ch_passwd Utility 6-111
Configuring Third-Party Security Regimes 6-112
Using Kerberos Authentication 6-112
Configuring the RAC for Use with SecurID 6-114
Using SafeWord AS Security 6-119
SafeWord Backup Security 6-126
Configuring the IP Basic Security Option (IPSO) 6-126
Using the ACE/Server 6-127
Installation 6-131
Using AppleTalk Security 6-133
ARA Security 6-133
Zone Security 6-134
NVE Filtering 6-134
Logging 6-134
Using IPXCP Security 6-135
Using PPP Security 6-135
Password Authentication Protocol (PAP) 6-135
Challenge-Handshake Protocol (CHAP) 6-137
Using the PPP Security Parameters 6-140
Dynamic Allocation of Network Addresses 6-142
Introduction to DHCP 6-142
Unsupported Features of DHCP 6-143
Cautions 6-143
Creating the acp_dialup File 6-143
Determining Dial-up Addresses Using the acp_dialup File 6-145
Using Filters for Security 6-146
Include and Exclude 6-147
Accessing the Filter Subcommands 6-148
Filter Numbers 6-150
Filter Lists 6-150
Configuring Security for the RAC FTP Daemon 6-152
Logging Security Events 6-153

Chapter 7
Digital Modems

Digital Modem Support 7-1
Digital Modem Assignments 7-2
DNIS and ANI 7-2

Busying-Out Modems Manually . . . 7-3
Busying-Out DS0 Channels Automatically . . . 7-4
Customizing Modem Configuration . . . 7-6
Configuring Modem and Channel Error Detection . . . 7-7
Counting the Call Events . . . 7-8
Specifying the Consecutive Failure Threshold . . . 7-9
Displaying the Call Event Counts . . . 7-10
Resetting the Consecutive Failure Counts to Zero . . . 7-10
Displaying and Changing Modem Status . . . 7-11
Displaying Internal Modem Information . . . 7-11
Modem Statistics . . . 7-15

Chapter 8
Point-to-Point Protocol

Point-to-Point Protocol . . . 8-1
PPP Configuration Overview . . . 8-2
Step 1: Decide How to Assign IP Addressing . . . 8-2
Step 2: Edit the Configuration File . . . 8-5
Step 3: Review and Reset Global Port Parameters . . . 8-6
How to Assign IP Addresses to PRI B Channels . . . 8-9
Configuration Samples . . . 8-10
Sample Configuration for Addressing Using the acp_dialup File . . . 8-10
Sample Configuration for Addressing Using Asynchronous Port Parameters . . . 8-12
Sample Configuration for Connecting Two Subnets . . . 8-14
Routing across a PPP Link (Basic Passive RIP) . . . 8-17
Route Cache . . . 8-18
Protocol Stack . . . 8-18
Negotiating the LCP Options . . . 8-18
Negotiating the Network Control Protocol . . . 8-21
BOOTP Requests . . . 8-24
Using the CLI netstat -ip Command . . . 8-24
Displaying Data for Ports . . . 8-25
PPP Over Sync Port . . . 8-25

Chapter 9
Multilink Point-to-Point Protocol (MP)

Overview . . . 9-1
Supported MP Features . . . 9-3
Unsupported MP Features . . . 9-4
The MP Process . . . 9-4
Bundling Scenarios . . . 9-6
Closing Member Links . . . 9-6
Security Considerations . . . 9-7
MP Configuration . . . 9-7
MP Operational Characteristics . . . 9-9
MP Parameters . . . 9-9
mp_endpoint Class and Address Example . . . 9-10
The acp_userinfo File . . . 9-11
Using Multi-System Multilink PPP . . . 9-11

Establishing MMP Connections . . . 9-14
Configuring MMP . . . 9-15
Resetting Ports for MP Links . . . 9-16
Administration of Multilink PPP (MP Statistics) . . . 9-16

Chapter 10
Serial Line Internet Protocol

SLIP and Compressed SLIP . . . 10-1
SLIP Configuration Overview . . . 10-2
Step 1: Decide How to Handle IP Addressing . . . 10-2
Step 2: Edit the Configuration File . . . 10-4
Step 3: Review and Reset Global Port Parameters . . . 10-5
Sample Configuration for a Single Remote Node . . . 10-9
Sample Configuration for Connecting Two Subnets . . . 10-10
Routing Across a SLIP Link (Basic Passive RIP) . . . 10-12
Routing Between Two Networks . . . 10-13
Route Cache . . . 10-14
Extending a Single Host onto the Network . . . 10-14
BOOTP Requests . . . 10-15

Chapter 11
Routing Information Protocol (RIP)

Prerequisites . . . 11-2
Understanding IP Routing and RIP . . . 11-2
Definition of a Route . . . 11-3
Routing versus Forwarding . . . 11-4
Choosing Passive or Active RIP . . . 11-4
RIP Versions . . . 11-5
Route Cache and Routing Table . . . 11-5
How Hosts Learn Routing Information . . . 11-7
Routing Interfaces . . . 11-10
IP Addressing . . . 11-11
Proxy ARP for Interfaces on the Same Network . . . 11-18
Setting the Broadcast Address . . . 11-20
Overview of Configuration Parameters . . . 11-22
Setting Parameters for Routing . . . 11-24
Using SPBs to Set Parameters . . . 11-24
Activating RAC Parameter Settings . . . 11-25
Activating Interface Parameter Settings . . . 11-26
Enabling Passive RIP Alone . . . 11-26
Configuring Passive RIP . . . 11-27
Defining Routes . . . 11-27
Accepting RIP 1 and/or RIP 2 Packets . . . 11-42
Authenticating Incoming RIP 2 Updates and Requests . . . 11-44
Active RIP Prerequisites . . . 11-46
Configuring Active RIP . . . 11-47
Defining Routes . . . 11-47
Advertising RIP 1 and/or RIP 2 Updates . . . 11-48
Advertising Subnet Routes . . . 11-49

Using Split Horizon and Poison Reverse . . . 11-50
Authenticating Outgoing Updates and Requests . . . 11-51
Advertising the Default Route . . . 11-51
Advertising to a Subset of Routers . . . 11-52
RIP Configuration Parameters - Reference . . . 11-53
Displaying Routing Information . . . 11-60
Displaying RIP Statistics . . . 11-61
Displaying the RAC Routing Table . . . 11-62
Displaying the route cache . . . 11-66
Using the ping -t (traceroute) Option . . . 11-67
Troubleshooting . . . 11-72
CLI Commands Providing Routing Information . . . 11-72
Common Configuration Errors . . . 11-73
What To Do If the RAC Does Not Advertise Updates . . . 11-77
What to Do if the RAC Does not Receive Updates . . . 11-79
Other Documentation . . . 11-80

Chapter 12
Internetwork Packet Exchange Protocol

Novell Networks . . . 12-1
IPXCP Features . . . 12-2
Default IPX Parameter Settings . . . 12-3
IPX Configuration Overview . . . 12-3
Step 1: Decide How to Handle IPX Addressing . . . 12-4
Step 2: Edit the Configuration File . . . 12-5
Step 3: Review and Reset Global Port Parameters . . . 12-6
Default PPP-Related Global Port Parameters . . . 12-7
How to Change a Global Port Parameter Setting . . . 12-7
Assigning IPX Networks or Nodes to B Channels . . . 12-9
Configuration Samples . . . 12-10
Sample Configuration Using Dial-up Addresses . . . 12-11
Sample Configuration Using Fixed Addressing . . . 12-13
Obtaining IPX Information . . . 12-15
System Logs . . . 12-15
IPXCP Interface Statistics . . . 12-16
IPX Interfaces, Memory Buffers, Routes, and Servers . . . 12-18
IPX Frame Type and Network Number . . . 12-23
IPX State . . . 12-24
IPX Connections . . . 12-24
Statistics for All Interfaces and for 802.2 . . . 12-25

Chapter 13
AppleTalk

AppleTalk Remote Access Protocol (ARAP) . . . 13-1
Configuring the RAC for AppleTalk . . . 13-2
AppleTalk-Specific Configuration Parameters . . . 13-3
AppleTalk-Specific RAC Parameters . . . 13-3
AppleTalk-specific Global Port Parameters . . . 13-5
CLI AppleTalk Commands . . . 13-6

Command Syntax . . . 13-7
arap . . . 13-8
arp . . . 13-8
AppleTalk over ARA . . . 13-9
AppleTalk Configuration Overview . . . 13-9
Step 1: Edit the RAC Configuration File . . . 13-10
Step 3: Review and Reset Global Port Parameters . . . 13-12
Sample AppleTalk Configuration . . . 13-13
ARA Security . . . 13-15
Security Features . . . 13-16
Network-Visible Entity (NVE) Filtering . . . 13-17
AppleTalk over PPP . . . 13-17

Appendix A
Digital Modem Configuration Parameters

Custom Modem Configuration . . . A-1
Using the %digital_modem Section . . . A-1
Setting Parameter Values . . . A-2
Standard Digital Modem Configurations . . . A-3
U.S. Operation, V.42bis Primary/MNP5 Secondary . . . A-3
U.S. Operation, Disable V.42bis Compression (V.42 only) . . . A-4
U.S. Operation, MNP5 Primary: V.42/V.42bis Disabled . . . A-4
U.S. Operation, Disable V.34 Extended Speeds (33.6 Kbps) . . . A-5
International Operation . . . A-5
Parameter Maps and Option Settings . . . A-5
Parameter 0 . . . A-6
Parameter 1 . . . A-7
Parameter 2 . . . A-10
Parameter 3 . . . A-11
Parameter 4 . . . A-12
Parameter 5 . . . A-12
Parameter 6 . . . A-12
Parameter 7 . . . A-13
Parameter 8 . . . A-13
Parameter 9 . . . A-14
Parameter 10 . . . A-14
Parameter 11 . . . A-15
Parameter 12 . . . A-15
Parameter 13 . . . A-15
Parameter 14 . . . A-16
Parameter 15 . . . A-16
Parameter 16 . . . A-17
Parameter 17 . . . A-18
Parameter 18 . . . A-19
Parameter 19 . . . A-20
Parameter 20 . . . A-21
Parameter 21 . . . A-22

Parameter 22 . . . . . A-22
Parameter 23 . . . . . A-23
Parameter 24 . . . . . A-24
Parameter 25 . . . . . A-24
Parameter 26 . . . . . A-24
Parameter 27 . . . . . A-26
Parameter 28 . . . . . A-27
Parameter 29 . . . . . A-28
Parameter 30 . . . . . A-29
Parameter 31 . . . . . A-30
Parameter 32 . . . . . A-31
Parameter 33 . . . . . A-31
Parameter 34 . . . . . A-32
Parameter 35 . . . . . A-33
Parameter 36 . . . . . A-34
Parameter 37 . . . . . A-34
Parameter 38 . . . . . A-35
Parameter 39 . . . . . A-36
Parameter 40 . . . . . A-36
Parameter 41 . . . . . A-36
Parameter 42 . . . . . A-36
Parameter 43 . . . . . A-37
Parameter 44 . . . . . A-37
Parameter 45 . . . . . A-37
Parameter 46 . . . . . A-39
Parameter 47 . . . . . A-40
Parameter 48 . . . . . A-41
Parameter 49 . . . . . A-41
Parameter 50 . . . . . A-41
Parameter 51 . . . . . A-41
Parameter 52 . . . . . A-42
Parameter 53 . . . . . A-42
Parameter 54 . . . . . A-42
Parameter 55 . . . . . A-42
Parameter 56 . . . . . A-43
Parameter 57 . . . . . A-43
Parameter 58 . . . . . A-43
Parameter 59 . . . . . A-43
Parameter 60 . . . . . A-44
Parameter 61 . . . . . A-44
Parameter 62 . . . . . A-44
Parameter 63 . . . . . A-44

Index

Figures

Figure 1-1. Voice Call over Analog Line 1-5
Figure 1-2. TA Call over a BRI Line 1-7
Figure 1-3. Synchronous PPP Connection 1-7
Figure 1-4. Voice Call over Analog Line 1-8
Figure 1-5. Network with Mixed Remote Annex Types 1-9
Figure 4-1. RAC TCP/IP Gateway 4-30
Figure 5-1. RACs to Be Used for Dial-out 5-56
Figure 6-1. Sample Configuration for a route Entry in acp_userinfo 6-87
Figure 8-1. Connecting a Single Host Using PPP 8-11
Figure 8-2. Connecting a Single Host Using PPP with Fixed Addresses 8-13
Figure 8-3. PPP Link Connecting Two Ethernet Subnets 8-15
Figure 9-1. MP Functional Flow 9-2
Figure 9-2. Illustration of Single Hunt Group Configuration 9-13
Figure 10-1. Connecting a Single Host Using SLIP 10-9
Figure 10-2. SLIP Link with Two IP Addresses 10-11
Figure 11-1. Configuration Using Four Class C Node Addresses 11-13
Figure 11-2. Subnetting with Passive RIP 11-17
Figure 11-3. Proxy ARP versus Routing 11-19
Figure 11-4. Sample Network for Defining Default Routes with Passive RIP 11-29
Figure 11-5. Sample Network for Static and Default Routes (Passive RIP) 11-36
Figure 11-6. Advertising Subnet Routes 11-49
Figure 11-7. Overview of ping -t Actions 11-68
Figure 11-8. Topology for ping -t Examples 11-70
Figure 11-9. Configuration in Which Proxy ARP Can Fail 11-74
Figure 11-10. Overlapping Addresses 11-75
Figure 11-11. Non-contiguous Subnets 11-76
Figure 12-1. Connecting a Single Host Using PPP 12-11
Figure 12-2. Connecting a Single Host Using PPP with Fixed Addresses 12-13
Figure 13-1. Connecting a Macintosh Using ARA 13-14

Tables

Table 1-1. Channel Distribution on T1 and E1 Lines 1-4
Table 2-1. CLI Commands 2-3
Table 2-2. Non-Privileged RAC VMS Commands 2-6
Table 2-3. Privileged RAC VMS Commands 2-7
Table 2-4. Arguments for the na Commands 2-12
Table 2-5. The na Commands 2-13
Table 3-1. Formatting Codes for Annex Prompts 3-11
Table 3-2. Supported Keywords for %gateway Entries - Format 1 3-20
Table 3-3. Supported Keywords for %gateway Entries - Format 2 3-21
Table 3-4. Supported Keywords for %macro Entries 3-29
Table 3-5. Statements Permitted in an Alias Expansion 3-30
Table 3-6. Supported Arguments for %service Entries 3-38
Table 3-7. Field Definitions for %dialout Entries 3-40
Table 3-8. Parameters That Can Be Set within the set Field of the %dialout Entry 3-43
Table 3-9. Field Definitions for Chat Scripts 3-47
Table 3-10. Reserved Keywords Used in Place of a Script Name 3-49
Table 3-11. Reserved Keywords Used in Place of a String 3-49
Table 4-1. Priority Levels for the syslog_mask Parameter 4-5
Table 4-2. Dump File Naming Conventions 4-9
Table 4-3. Arguments for the server_capability Parameter 4-11
Table 5-1. Valid PRI switch_type Values 5-6
Table 5-2. Valid CAS switch_type Values 5-6
Table 5-3. Parameters For the set wan Command 5-9
Table 5-4. Channel Parameters 5-15
Table 5-5. Keywords for the show port Command 5-20
Table 5-6. SPB Field Definitions 5-27
Table 5-7. Valid Values for *ports* Arguments in RAC Rotaries 5-48
Table 5-8. Valid Port Types for ports Field in a Dial-out Entry 5-53
Table 6-1. The erpcd Options and acp_policy.h Variables 6-13
Table 6-2. Options for the acp_dbm Utility 6-15
Table 6-3. RAC Automatic Services 6-26
Table 6-4. RAC Port Mode/Service Restrictions 6-28
Table 6-5. Profile Criteria 6-61
Table 6-6. Authentication Regimes 6-71
Table 6-7. Entries for accesscode in the acp_userinfo File 6-77
Table 6-8. Arguments For the clicmd Entry in the acp_userinfo File 6-80
Table 6-9. Entry For climask in the acp_userinfo File 6-82
Table 6-10. Entry for deny in the acp_userinfo File 6-83
Table 6-11. Entry for filter in the acp_userinfo File 6-84
Table 6-12. Arguments forThe route Entry in the acp_userinfo File 6-86
Table 6-13. Entry for at_zone in the acp_userinfo File 6-88
Table 6-14. Entry for at_connect_time in the acp_userinfo File 6-89
Table 6-15. Entries for at_nve_filter in the acp_userinfo File 6-90
Table 6-16. Entry for at_passwd in the acp_userinfo File 6-92
Table 6-17. Entry for chap_secret in the acp_userinfo File 6-93
Table 6-18. Arguments in the acp_restrict File Entries 6-95
Table 6-19. Argument for the include File 6-99
Table 6-20. Supported Argument for ch_passwd 6-112

Table 6-21. Kerberos Parameter Settings 6-114
Table 6-22. PPP Security Parameters and Their Effect on RAC Activity 6-140
Table 6-23. Summary of filter Subcommands. 6-150
Table 6-24. Sample Commands Using the filter_list Arguments. 6-151
Table 7-1. Arguments for the modem Command 7-12
Table 8-1. Default PPP-related Global Parameter Settings 8-6
Table 8-2. Default wan b Parameter Settings. 8-7
Table 10-1. Default Serial Networking Parameter Settings 10-6
Table 11-1. Network Classes 11-12
Table 11-2. RIP-specific RAC Parameters 11-22
Table 11-3. RIP-specific Interface Parameters 11-23
Table 11-4. Values for Bits Field with Corresponding Subnet Masks 11-33
Table 11-5. Class A: Total Available Subnets and Hosts. 11-34
Table 11-6. Class B: Total Available Subnets and Hosts. 11-34
Table 11-7. Class C: Total Available Subnets and Hosts (with no supernetting). 11-35
Table 11-8. Arguments for the Superuser CLI route Command 11-40
Table 11-9. RAC RIP Version 2 Authentication 11-45
Table 11-10. Field Definitions for the netstat -g Command 11-61
Table 11-11. IP Fields in netstat -r Display 11-63
Table 11-12. Flag Descriptions for the netstat -C Command 11-66
Table 11-13. Fields Displayed by the ping -t Option 11-69
Table 12-1. Default Serial Networking Parameter Settings 12-3
Table 12-2. Wan b Networking Parameter Settings 12-3
Table 12-3. Default PPP-related Global Port Parameter Settings 12-7
Table 12-4. Fields in (NCP) IPXCP Status Display 12-17
Table 13-1. AppleTalk-specific RAC Parameters 13-3
Table 13-2. AppleTalk-specific Global Port Parameters 13-5
Table 13-3. CLI AppleTalk Commands 13-7
Table 13-4. Arguments for the arp Command. 13-9
Table A-1. Transmitting DCE Break Handling with Respect to Data A-8
Table A-2. Receiving DCE Break Handling with Respect to Data. A-9

If you are responsible for configuring and administering a Bay Networks® Remote Access Concentrator (RAC), you need to read this guide.

Before You Begin

Before using this guide, 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 *Line Provisioning 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. The WAN parameters you need to set for PRI and CAS are described in the *Remote Access Concentrator Software Reference Manual*.
- 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

This manual uses the following conventions:

Convention:	Represents:
`special type`	In examples, `special type` indicates system output.
`special type`	Bold **`special type`** indicates user input.

Convention:	Represents:
bold	Bold indicates commands, pathnames, or filenames that must be entered as displayed.
italics	In the context of commands and command syntax, lowercase italics indicate variables for which the user supplies a value.
[ ]	In command dialog, square brackets indicate default values. Pressing <CR> 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, this character separates the different options available for a parameter.
	Notes provide important information.
	Warnings inform you about conditions that can have adverse effects on processing.
	Cautions notify you about dangerous conditions.

Acronyms

BootP	Bootstrap Protocol
BRI	Basic Rate Interface
CCITT	International Telegraph and Telephone Consultative Committee (now ITU-T)
GUI	graphical user interface
HDLC	high-level data link control
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
OSI	Open Systems Interconnection
PPP	Point-to-Point Protocol
SNMP	Simple Network Management Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
Telnet	Telecommunication Network
TFTP	Trivial File Transfer 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

Chapter 1
Introduction to Remote Access Concentrators

This chapter provides information about:

- The Remote Access Concentrator (RAC)
- The dual WAN interfaces (the PRI interface and the CAS interface)
- A list of general capabilities supported by the RAC, including multiprotocol support and security systems
- Typical network configurations for the types of traffic the RAC recognizes
- Configuration prerequisites
- The different types of parameters used to configure the RAC
- Loading configuration files from hosts

RAC Overview

The RAC is a dual WAN server that supports analog calls and digital calls carried over ISDN.

The RAC handles the following types of traffic:

- Analog Pulse-Code Modulated (PCM) voice data and analog Dial Tone Multiple Frequency (DTMF) data, both originating from standard analog (for example, V.34) modems.
- Digital data conforming to the V.120, V.110, and X.75 rate adaptation protocols. These protocols are supported by various terminal adapters (TAs), and are referred to in this book as TA calls.

- Digital data conforming to the synchronous PPP specification.

Session configuration for analog, TA, and synchronous PPP dial access can be managed with a single phone number. You do this by adding a session parameter block (SPB) to the **%pri** section of the **config.annex** file. The SPB configures the RAC accordingly as the call is set up. Refer to *Configuring Session Parameter Blocks on page 5-25* for examples.

Dual WAN Interfaces

The RAC's dual WAN interfaces enable it to handle digital calls carried by PRI running over a T1/E1 line and analog calls carried over a channelized T1/E1 line (referred to in this book as channel associated signaling, or CAS). The RAC can configure the WAN interfaces in any combination: one of each type or two of the same type.

The RAC handles calls from both WAN interfaces using a common pool of digital modems; each incoming call is assigned to an available modem in a circular manner, regardless of which WAN interface it arrives on. For complete information, refer to *Digital Modems on page 7-1*.

Both WAN protocol families (ISDN PRI and CAS) are carried over the same type of framing medium, or line. In North America and Japan, this means a T1 line; in Europe, an E1 line is used.

PRI

The ISDN Primary Rate Interface (PRI) uses a number of B channels for the transmission of "payload" data, as well as a single D channel for the transmission of signaling data used for call setup, teardown, and connection management.

PRI returns a busy signal to the central office switch if no modem on the RAC is available to handle the call.

CAS

The channel associated signaling(CAS) protocol is used to carry analog (voice) data on channelized T1 or channelized E1 lines. These lines are also referred to as DS1 channels. Each DS1 channel is divided into DS0 channels for carrying data; each DS0 channel contains a subchannel that is used for call setup/teardown and framing management, also known as signaling information.

In this book, discussion of CAS is limited to channelized T1 and channelized E1, although switch types for other protocols are mentioned.

CAS supports only analog modem data, although TA and Sync PPP data may be handled if they are carried by DOSBS (Data Over Speech Bearer Service).

CAS does not support the transmission of a busy signal for all protocols. To avoid problems that may result from this, the RAC allows network administrators to busy-out DS0 channels when no modems are available to handle the calls they are carrying. Refer to *Busying-Out DS0 Channels Automatically on page 7-4* for complete details.

Channel Allocation

T1 lines are divided into 24 channels, E1 lines into 32 channels. These channels are allocated differently depending on which format is in use, PRI or CAS (see Table 1-1).

Table 1-1. Channel Distribution on T1 and E1 Lines

Framing Medium	Protocol	Channel Allocation
T1 (North America and Japan)	PRI	23 B channels and 1 D channel, each operating at 64kbps.
	CAS	24 DS0 channels, each operating at 56kbps. (8kbps are consumed by each channel's inband signaling information.)
E1 (Europe)	PRI	30 B channels and 1 D channel, each operating at 64kbps. (The 32nd channel is dedicated to framing overhead and is not used by PRI directly.)
	CAS	30 DS0 channels, each operating at 56kbps. (8kbps are consumed by each channel's inband signaling information. The 32nd channel is dedicated to framing overhead and is not used by CAS directly.)

Channel distribution may be different for CAS protocols other than channelized T1/E1 (such as R1, R2, etc.).

Multi-Protocol Support

The RAC supports all of the dial-in, multiprotocol, remote access features characteristic of Bay Networks Remote Annex products. Once a call has been established, the RAC allows the use of multiple protocols, including asynchronous IP, PPP, SLIP, IPX, LAT (with limitations), and AppleTalk, as well as synchronous IP over PPP, IPX over PPP (IPXCP), and AppleTalk over PPP.

In addition, the RAC provides ISDN calls originating from a router (such as the Bay Networks Clam or Marlin) with access to standard Remote Annex IP routing capabilities.

Typical Network Configurations

Typical PRI Configurations

Following are examples of the three PRI connection types supported by the RAC.

Voice (Analog) Calls

The example in Figure 1-1 shows a PC user with a standard modem (e.g., a V.34) dialing into the RAC over a PRI connection.

Other modem call configurations can be established instead of the one shown in Figure 1-1.

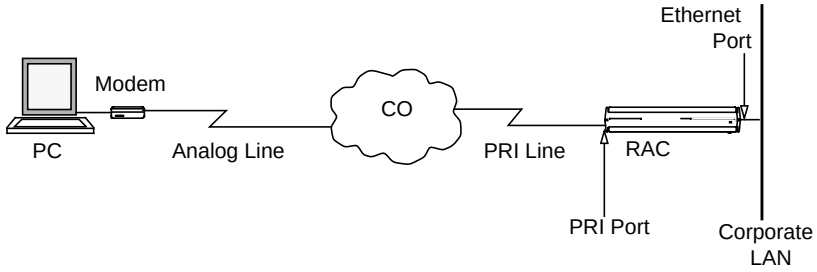

Figure 1-1. Voice Call over Analog Line

In Figure 1-1, the telco central office (CO) terminates the analog line from the modem, converts the voice call into digital data, and dynamically chooses a PRI B channel to carry the data through the telephone network to the RAC. The CO also converts the signals describing the characteristics of the call into ISDN out-of-band signaling messages, as defined by the Q.931 and Q.932 standards. These signaling messages are carried to the RAC on the PRI D channel.

The RAC does not automatically accept the modem call. First, the CO allocates a B channel and the RAC determines whether it has the resources, such as an available modem, to handle the call. RAC negotiations are based not only on physical resources but also on parameters that you configure. If the RAC's parameters are set properly, the call is accepted. Calls can be rejected explicitly through the use of session parameter blocks (SPBs).

Once a call is accepted (and the user is authenticated), the RAC places the user in the protocol environment you have configured. The supported protocols are asynchronous PPP, SLIP, ARAP, and CLI.

Terminal Adapter Calls

The RAC supports terminal adapter (TA) calls using the V.120, V.110, and X.75 rate adaptation protocols.

The example in Figure 1-2 shows a TA using the V.120 rate adaptation protocol to convert asynchronous data generated by a PC into V.120 frames. These frames are transmitted over a BRI line to the appropriate switch at the CO and then sent to the RAC on a negotiated PRI B channel. If the RAC's parameters are configured correctly for this kind of call, the call is accepted. The RAC converts the V.120 frames into an asynchronous data stream, and the RAC software handles the data as if it originated at a V.120 asynchronous port.

Other V.120 call configurations can be established instead of the one shown in Figure 1-2. For example, the BRI line could be a PRI line.

Once a call is accepted (and the user is authenticated), the RAC places the user in the protocol environment you have configured. The supported protocols for V.120, V.110, and X.75 calls are asynchronous PPP, SLIP, ARAP, and CLI.

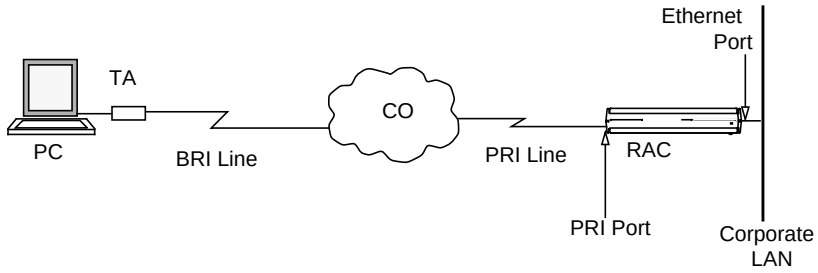

Figure 1-2. TA Call over a BRI Line

Synchronous PPP Calls

Figure 1-3 shows a sample synchronous PPP call from a user whose PC has an adapter card supporting synchronous PPP encapsulation over ISDN. The user connects to the CO via a BRI line and connects to the RAC over a negotiated PRI B channel. If the RAC's parameters are configured correctly for this kind of call, the call is accepted.

Once a call is accepted (and the user is authenticated), the RAC places the user in the protocol environment you have configured. The supported protocols for synchronous calls are IPCP (IP over PPP), IPXCP (IPX over PPP), and ATCP (AppleTalk over PPP).

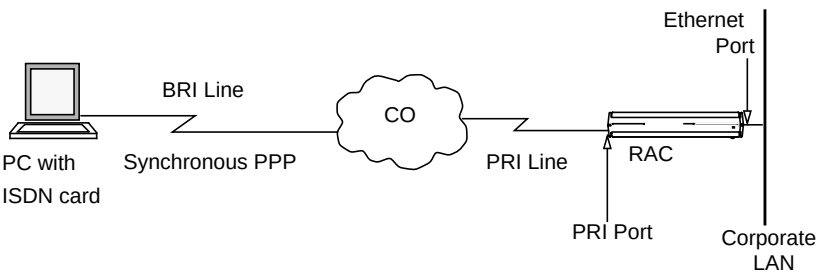

Figure 1-3. Synchronous PPP Connection

Multilink PPP

The RAC also supports synchronous Multilink PPP (MP). MP is a protocol standard that provides a means for data aggregation over multiple DS0 or B channels. This implementation of MP is based on the RFC1990 technical specification. For information on using and configuring MP, see *Multilink Point-to-Point Protocol (MP) on page 9-1*.

Typical CAS Configuration

Figure 1-4 shows a PC user with a standard modem (e.g., a V.34) dialing into the RAC over a CAS connection.

Other modem call configurations can be established instead of the one shown in Figure 1-4.

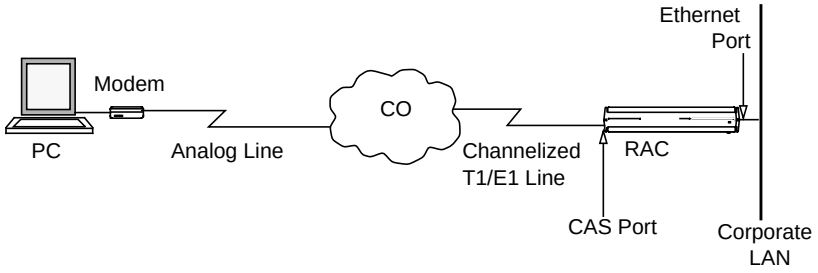

Figure 1-4. Voice Call over Analog Line

In Figure 1-4, the telco CO terminates the analog line from the modem and dynamically chooses a DS0 channel to carry the data through the telephone network to the RAC. The CO also converts the signals describing the characteristics of the call into signaling messages; these signaling messages are carried to the RAC in the signaling subchannel of the DS0.

The modem call is not automatically accepted by the RAC. First, the CO allocates a B channel and the RAC determines whether it has the resources, such as an available modem, to handle the call. RAC negotiations are based not only on physical resources but also on parameters that you configure. If the RAC's parameters are set properly, the call is accepted. (Calls can be explicitly rejected through the use of SPBs.)

Once a call is accepted and the user is authenticated, the RAC places the user in the protocol environment you have configured. The supported protocols are asynchronous PPP, SLIP, ARAP, and CLI.

Mixed RAC/Remote Annex Environment

Figure 1-5 shows how a RAC can fit into an environment with Bay Networks Remote Annexes to provide a complete enterprise solution. (Remote Annexes are not labeled in the figure.)

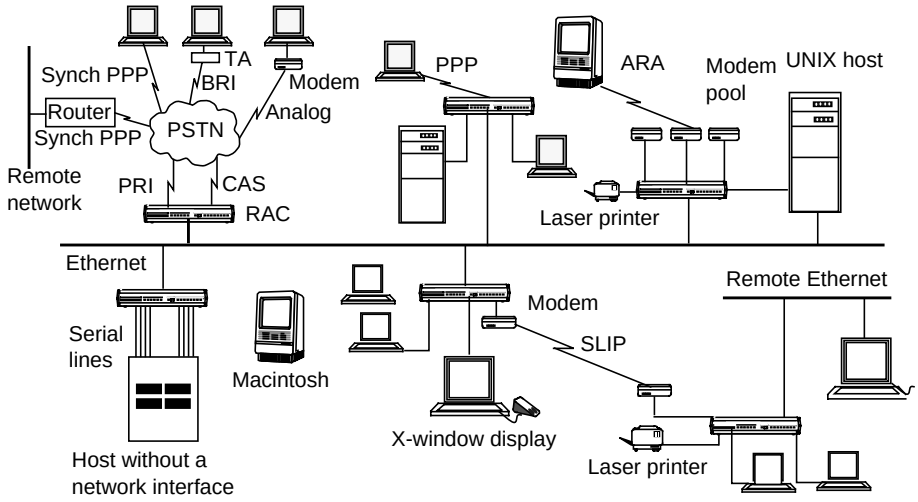

Figure 1-5. Network with Mixed Remote Annex Types

Configuration Prerequisites

Before you configure your RAC, you must:

- Order your WAN (PRI or CAS) lines from your telco. Order information is provided in the Bay Networks publication *Line Provisioning 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. The WAN parameters you need to set for PRI and CAS are described in the *Remote Access Concentrator Software Reference*.
- 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.

Parameter Configuration

The chapters that follow describe how to configure the RAC. As mentioned in the previous sections, the RAC does not accept a call unless you have set certain parameters properly. You can set parameters using various RAC management tools.

RAC Management Tools

The RAC software provides network management tools and files for setting the parameters that control the RAC environment.

Commands remain in effect until the next time the RAC is rebooted. Parameters remain in effect until they are explicitly set to a different value (even if the RAC is rebooted).

- The Network Administrator (**na**) program is a host-based UNIX utility. It provides commands for displaying and modifying operating characteristics of the RAC, its en0 or PRI port, and PRI call handling.
- The Command Line Interpreter (CLI) is the RAC's command interface. You can use CLI commands to connect to hosts and to display and change RAC characteristics. You can display known hosts, as well as statistics for the RAC and the network. The CLI also provides superuser commands for network administration and management.
- The CLI **admin** command, which you access as a superuser on a CLI connection, is a local (resides in the RAC) substitute for the host-resident **na** command. The **admin** command set provides a subset of the host-resident **na** commands. However, all parameters that you can set via **na** you can also set using **admin**.
- Certain host-based configuration files allow you to create entries that can control, among other things, PRI call handling and user security. The default configuration file is named **config.annex** and is located in the **/usr/annex/** directory on the UNIX load host.

Parameter Types

You can configure the RAC by setting the following types of parameters:

- Annex-wide parameters that apply to an entire RAC or set of RACs. This parameter type is described in *Configuring RAC Parameters on page 3-2*.
- Global port parameters that apply to all calls. You can also set these parameters for a subset of calls. To define port parameters globally, use **na** or **admin**. To define the same parameters for a subset of calls, enter the parameters in a Session Parameter Block (SPB) that handles the calls. SPBs are located in the RAC's configuration file.
- WAN parameters, which define characteristics of the WAN line and establish remote IP and IPX addresses.
- Interface parameters.
- Modem parameters that apply to the internal modem set.

For detailed information, see *Configuring the WAN Interfaces, Global Ports, and Sessions on page 5-1*.

Loading Configuration Files from Hosts

RAC files can be loaded from a host using either the Trivial File Transfer Protocol (**tftp**) or the expedited remote procedure call daemon (**erpcd**):

- The **erpcd** utility runs on a UNIX host; it listens for RAC file server host requests (to download the operational code and other files).
- The **tftp** program, supplied on most hosts, is supported as an alternative to **erpcd** and as a backup in case a UNIX host is not available to install **erpcd**.

If a software problem occurs, the RAC can send a memory dump to a host. These memory dumps assist Bay Networks customer support personnel in resolving problems.

Chapter 2
Using the CLIs

The RAC provides three command interfaces for performing tasks.

The first is referred to simply as the command line interface (CLI), a set of general commands available to non-administrative users for controlling jobs and sessions. In addition, the RAC provides **na** and **admin**, both of which are administrative utilities for configuring and managing the RAC. The **na** utility resides on and is accessed from a UNIX host; it is especially useful when managing one or more RACs across a network. The **admin** utility is a subset of the **na** utility (there are seven na commands which are not included in admin) which resides on the RAC itself and is accessed via a superuser CLI session on the RAC. The difference between **na** and **admin** is a set of seven commands which are relevant only to administering multiple RACs, or administering a RAC across a network.

The **na** and **admin** utilities provide commands that 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.

For a detailed description of each command in the CLI or **na** (and therefore **admin**), refer to the *Remote Access Concentrator Software Reference*.

The Command Line Interface

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

The CLI provides two groups of commands - user and superuser; the superuser commands are used for RAC administration.

Each 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. This is called *minimum uniqueness*. If you do not want the RAC to interpret a host name using minimum uniqueness, enclose the name in double quotes (""). 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.

Squelch

If six consecutive CLI errors occur within 6 seconds (e.g., invalid command, noise on the line), the RAC triggers a squelch, i.e., stops all I/O for approximately 4 seconds after receiving the sixth error. Pressing <Return> after this period of time returns you to the CLI prompt.

If the errors occur over a time period greater than 6 seconds, the RAC ignores them and restarts the timer.

If four consecutive squelches occur without an intervening known command, or if serial errors (framing/parity) occur while the line is squelched, the RAC shuts down the line for 20 seconds and syslogs: *excessive errors on port **n**; shutting down* (where ***n*** is the port number.)

CLI Commands

Table 2-1 through Table 2-3 list both the user and superuser CLI commands.

To access the superuser CLI commands, issue the **su** command at the user CLI prompt and enter the RAC's administrative (**su**) password (for more details on the superuser password, see the *Remote Access Concentrator Software Reference*). The default superuser prompt is a # symbol instead of a colon:

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

Table 2-1. CLI Commands

Command	Type	Description
actcall	superuser	Displays active calls.
admin	superuser	Enters administrative mode.
arap	user	Converts a CLI line into an ARAP connection.
arp	superuser	Displays the Internet-to-hardware address translation tables.
bg	user	Puts a job in the background.
boot	superuser	Reboots the RAC.
compact	superuser	Compresses non-volatile memory space.
connect	user	Uses LAT to connect to an advertised LAT service.
control	superuser	Changes the state of DTR and RTS or outputs a test message.
cp	superuser	Copies a file in the local file system.
dialout	superuser	Displays the current dial-out database.

(continued on next page)

Table 2-1. CLI Commands (continued)

Command	Type	Description
edit	superuser	Edits configuration files.
fg	user	Returns to an established job.
filter	superuser	Enters the filtering subsystem of the CLI user interface.
hangup	user	Disconnects all jobs and resets user CLI connections.
histcall	superuser	Displays the call history summary.
help	user	Displays help information for user CLI commands.
help -m	superuser	Displays help information on macros.
hosts	user	Displays the current host table.
hosts	superuser	Flushes the host table.
ipx	user	Converts a CLI port to an IPX port.
jobs	user	Displays a list of current jobs.
kill	user	Terminates a job.
lock	user	Locks a port.
ls	superuser	Displays the self-boot ROM's directory.
modem	superuser	Displays the modem types supported by the RAC.
more	superuser	A read-only mechanism for reviewing files on the local file system.
mv	superuser	Renames a file in the local file system.
netstat	user	Displays network statistics.
passwd	superuser	Changes the administrative password.
ping	superuser	Sends ICMP Echo Request packets to a host.

(continued on next page)

Table 2-1. CLI Commands (continued)

Command	Type	Description
ppp	user	Converts a CLI port to a PPP interface port.
procs	superuser	Displays processes at the RAC.
queue	user	Displays information about queued HIC requests or removes a particular HIC request from the queue.
rlogin	user	Connects to a host using the **rlogin** protocol.
rm	superuser	Deletes a file in the local file system.
route	superuser	Adds/deletes routes to/from the active routing table.
services	user	Displays the LAT services that have been advertised by LAT nodes.
slip	user	Converts a CLI port to a SLIP interface port.
stats	user	Displays RAC statistics.
stats -c	superuser	Clears all serial line statistics to zero.
stats -T	user	Displays T1 network interface statistics.
stty	user	Displays and modifies CLI port parameters.
su	superuser	Enters and exits superuser administrative mode.
t1_loopback	superuser	Places the T1 engine in loopback mode.
tap	superuser	Displays input and output for a serial port.
telnet	user	Connects to a host using the Telnet protocol.
tn3270	user	Connects to an IBM VM/CMS or MVS host using the **tn3270** variation of the Telnet protocol.
who	user	Displays RAC users.
write	user	Allows the user to enter a text file into the RAC file system.

Table 2-2. Non-Privileged RAC VMS Commands

Command	Description
backwards	Selects next available, lower numbered session to which your port is connected.
close	Closes sessions.
connect	Uses LAT to connect to an advertised LAT service.
disconnect	Disconnects sessions.
forwards	Selects next available, higher numbered session to which your port is connected.
list port	Displays information about communications server ports from the permanent database.
list server	Displays information from the permanent database about the communications server.
lock	Locks a port to prevent unauthorized access.
logout port	Logs out of the terminal server and disconnects all sessions.
resume session	Returns to a session you have suspended.
set port	Specifies or modifies port characteristics immediately.
set privileged	Enables your port to perform privileged operations.
set session	Specifies characteristics for your current LAT session.
show port	Displays information from the operational database about communication server ports.
show server	Displays information from the operational database about the communications server.
show service	Displays information about LAT services to which you can connect.
show sessions	Displays connected active sessions for your port.
show users	Displays information about interactive port users on the server.

Table 2-3. Privileged RAC VMS Commands

Command	Description
clear services	Deletes entries for one or all local LAT services from the operational database.
crash	Performs a dump before rebooting.
define port access	Specifies the type of access allowed for the device using the port.
define port authorized groups	Allows you to authorize that groups of LAT service nodes be available to the port.
define port autobaud	Sets automatic detection of the speed, parity, and character size of the port device on login and sets the RAC port characteristics to match.
define/set/change port break	Specifies handling of the Break key during a session.
define/set/change port character size	Specifies the number of bits in data characters exchanged between the port and the RAC.
define port CLI interface	Indicates CLI behavior as related to logging in, passwords, inactivity timers and the port default prompt.
define/set/change port default session mode	Specifies the initial setting of the LAT session mode.
define port dsrlogout	When enabled, logs out of a port when the attached device powers down.
define/set/change port flow control	Specifies flow control.
define/set/change port inactivity logout	Determines whether the communications sever automatically logs out of a port after a period of inactivity.
define/set/change port input flow control	Specifies input flow control.

(continued on next page)

Table 2-3. Privileged RAC VMS Commands (continued)

Command	Description
define/set/change port local switch	Specifies a switch character that you can use to reenter local mode from service mode.
define/set/change port multisessions	Allows two active windows over a single communication line.
define/set/change port username	Defines or changes a username assigned to a port.
define server circuit timer	Specifies the frequency with which the server communicates with service nodes when LAT sessions are active.
define server inactivity timer	Specifies the period of time after which the server will log out of an inactive port.
define server keepalive timer	Defines or changes the time interval at which the server will transmit a keepalive message over a LAT virtual circuit, when there is no other traffic originating at the server.
define server LAT key	Enables and disables the LAT protocol, and is used as a security mechanism to restrict access to LAT within the RAC.
define server lock	Specifies whether or not interactive port users can use the **lock** command.
define server login password	Defines or changes the port password that interactive users must type when logging in to a server port.
define server login prompt	Defines or changes the prompt that requests the user's login password.
define server maintenance password	Defines or changes the user password required for remote applications such as NCP and TSM.
define server login prompt	Defines or changes the prompt that requests the user's login password.

(continued on next page)

Table 2-3. Privileged RAC VMS Commands (continued)

Command	Description
define server MOP host	Requires the physical Ethernet address for the preferred VMS host.
define/set/change port modem control	Specifies whether or not the RAC manipulates modem signals.
define port name	Specifies the name of the port.
define/set/change port noloss	Specifies whether or not the port will store data in its type ahead buffer while waiting for a session connection to be made. If so, it will then pass the data to the connection partner.
define/set/change port output flow control	Specifies output flow control.
define/set/change port parity	Specifies whether or not the port will provide a parity bit with each character for error checking.
define/set/change port password	Specifies whether or not a user requires a password to log in to the RAC.
define port session limit	Limits the number of connected sessions on the port.
define/set/change port speed	Specifies the port speed in bits per second.
define/set/change port stop bits	Tells the RAC to use 1, 1.5, or 2 stop bits when outputting a character.
define/set/change port type	Defines or changes the type of terminal connected to your port, the ports specified in the port-list, or all ports.
define server multicast timer	Specifies time to elapse between service announcement transmissions.
define server name	Specifies a 1 to 16 character name for the RAC.
define server number	Specifies a facility number for the RAC.

(continued on next page)

Table 2-3. Privileged RAC VMS Commands (continued)

Command	Description
define server password limit	Specifies the number of times a user can try to enter the correct password for any password-protected RAC operation.
define server privileged password	Specifies the password a user must enter following a **set privileged** command in order to use privileged commands at the port.
define server queue limit	Specifies the maximum number of queued connection requests for remote access to RAC ports.
define server retransmit limit	Specifies the number of times a LAT message is retransmitted to a service node when the RAC receives no messages of acknowledgment.
define server security	Determines whether or not the RAC performs any security checking.
define server session limit	Specifies the maximum number of active sessions that the RAC allows at one time.
define server software	Specifies the file name of the RAC software image.
initialize server	Reboots the RAC.
set noprivileged set privileged	Returns the port to non-privileged status to inhibit unauthorized use. Enables your port to perform privileged operations.
set server all	Resets the RAC subsystems (security, LAT, etc.) from the permanent database, re-reads the **macro** and **service** sections of the configuration file, and re-reads the **message-of-the-day** file.

The na and admin Utilities

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

All **na** commands are taken from the **na** standard input: you can run **na** interactively or provide it with input through a file or pipeline. You can create a script file containing **na** commands to configure a RAC. This script file can save the configuration information for a specific RAC and, when required, restore the configuration.

This chapter illustrates command names, parameter names, and keywords in their long forms. Examples of **na** commands sometimes appear without the interactive command 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 command prompt appears in bold type.

Command Notation

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

- You can abbreviate commands and parameter names to the minimum number of characters that uniquely distinguish the name from any other name that may appear in the same context.
- Type a new-line character to end a command 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 double quotes (" "). Otherwise, the space is assumed to be a delimiter.
- The UNIX interrupt character (usually CTRL-C) returns you to the command prompt.

Additionally, **na** permits comments when the # character is present at the beginning of a comment line. All characters between the # and the next new line are ignored. Table 2-4 describes the supported arguments for **na**.

Command Syntax

Table 2-4. Arguments for the na Commands

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 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`
port_parameters	A list of one or more global port parameters, with or without values, separated by white space: `input_flow_control eia`

na and admin Commands

Table 2-5 lists the **na** commands. Refer to the *Remote Access Concentrator Software Reference* for a detailed description of each.

Seven of the **na** commands use standard UNIX superuser protection - only a superuser at the host can execute these commands: **boot**, **broadcast**, **copy**, **dumpboot**, **read**, **reset**, and **set**. These seven commands are the **na** commands which are not included in the subset of **na** commands which make up the **admin** utility.

Table 2-5. The na Commands

Command	Description
annex	Defines a default *annex_list* used with subsequent commands.
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 commands and parameters.
interface	Defines a default interface used with subsequent commands.
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 2-5. The na Commands (continued)

Command	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.
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:
```

Chapter 3
Using the Configuration File

This chapter describes configuring **annex** parameters, which apply to an entire RAC or to multiple RACs (as opposed, for example, to **wan** parameters that apply only to the WAN interfaces). The following topics are covered:

- Configuring RAC parameters with na and admin
- Customizing the RAC environment
- Parsing and setting up the configuration file, including special sections
- Using automated firmware download (AFD)

The Local File System

The stand-alone file system allows the RAC to store its configuration and message-of-the-day files in local non-volatile memory. The configuration files must have the appropriate file names for the operational image to locate and load them. These files exist in the **root** directory rather than the **/usr/spool/erpcd/bfs** directory. You can edit the files using the CLI local file system commands.

Configuring RAC Parameters

You can configure RAC parameters using the following:

- The host-based **na** utility
- The CLI superuser **admin** command
- Annex Manager (GUI)
- An SNMP-based manager such as Sun NetManager

To determine the current settings of RAC parameters, use the **show annex all** command. The **set annex** command allows you to change any setting. All parameters have default settings. Some of these parameters must be set using the ROM Monitor before booting the RAC with its operational code (see the hardware installation manual for your RAC for more details).

By default, the **show annex** command scrolls the selected parameters line by line in two-column format.

You can set up a pager as follows:

`setenv PAGER more` (BSD)

or

`set PAGER=more; export pager` (System V)

The **./src/na/README** file describes how to use a pager along with the **show** command.

Using the na Utility

To use the **na** utility:

1. **Log into a UNIX host and enter** na**:**

```
% na
Annex network administrator Rx.x January 1997
command:
```

2. **Specify one or more RACs:**

```
command: annex 192.9.200.95
         or annex 192.9.200.95,frontlobby
         or annex
enter default annex list: 192.9.200.95,frontlobby
```

3. **Execute the** set annex **command to change parameters. The following sample command lines:**

   - Enable the DNS name server.
   - Define two name server hosts.
   - Enable security on the RAC.
   - Define a security server host.
   - Enable security for virtual CLI connections.
   - Define an administrative password.
   - Enable event logging.
   - Define a CLI prompt.

```
command:  set annex name_server_1 dns
command:  set annex pref_name1_addr 192.9.200.95
command:  set annex name_server_2 dns
command:  set annex pref_name2_addr 192.9.200.85
command:  set annex enable_security Y
command:  set annex vcli_security Y
command:  set annex pref_secure1_host 192.9.200.95
command:  set annex password piano
command:  set annex syslog_mask all
command:  set annex syslog_host 192.9.200.95
command:  set annex cli_prompt "%a%c"
```

4. Execute the show annex all **command to review your changes. Using the example in step 3, the terminal displays:**

```
command: show annex all

Annex Generic Parameters

inet_addr: 132.245.44.187          subnet_mask:255.255.255.0
pref_load_addr: 132.245.44.80      pref_dump_addr:132.245.33.8
load_broadcast: Y                  broadcast_addr:132.245.44.255
load_dump_gateway:132.245.44.22    load_dump_sequence: net
image_name: ""                     motd_file: "motd"
config_file: "config.annex"        authoritative_agent: Y
routed: Y                          server_capability: none
disabled_modules: vci              tftp_load_dir: ""
tftp_dump_name: ""                 ipencap_type: ethernet
ip_forward_broadcast: N            tcp_keepalive: 120
option_key: ""                     session_limit: 1152
output_ttl: 64


VCLI Parameters

max_vcli: unlimited                cli_prompt: "%a%c"
vcli_security: Y                   vcli_password: "<unset>"
vcli_inactivity off


Nameserver Parameters

nameserver_broadcast: N            rwhod: Y
pref_name1_addr: 192.9.200.95      name_server_1: dns
pref_name2_addr: 192.9.200.85      name_server_2: dns
host_table_size: 64                min_unique_hostnames: Y


Security Parameters

enable_security: Y                 security_broadcast: Y
pref_secure1_host:192.9.200.95     pref_secure2_host: 0.0.0.0
network_turnaround: 2              loose_source_route: Y
acp_key: "<unset>"                 password: "<set>"
allow_snmp_sets: N                 lock_enable: Y
passwd_limit: 3                    chap_auth_name: "chap"
max_chap_chall_int: 0
```

(continued on next page)

```
Time Parameters

time_broadcast: N                    daylight_savings: us
timezone_minuteswest: 300            time_server: 0.0.0.0

SysLog Parameters

syslog_mask: all                     syslog_facility: log_local7
syslog_host: 192.9.200.95            syslog_port: 0

MOP and "Login" User Parameters

pref_mop_host: 00-00-00-00-00-00
mop_password: “<unset>”              login_password: “<set>”
login_prompt: “#”                    login_timer: 30

LAT Parameters

lat_key: ""                          facility_num: 0
server_name: ""                      sys_location: ""
lat_queue_max: 4                     service_limit: 256
keep_alive_timer: 20                 circuit_timer: 8
retrans_limit: 8                     group_value: none
vcli_groups: none                    multicast_timer: 30
multisessions_enable: N

AppleTalk Parameters

a_router: 00-00-00-00-00-00
default_zone_list: ""                node_id: 0.0
zone: ""

Router Parameters

rip_auth: "<unset>"                  rip_routers: all

IPX Parameters

ipx_file_server: ""                  ipx_frame_type: raw802_3
ipx_dump_username: ""                ipx_dump_password: "<unset>"
ipx_dump_path: ""                    ipx_do_checksum: N
```

(continued on next page)

```
TMux Parameters
tmux_enable: N                      tmux_max_host: 64
tmux_delay: 20                      tmux_max_mpx: 700

DHCP Parameters
pref_dhcp1_host: 0.0.0.0            pref_dhcp2_host: 0.0.0.0
dhcp_bcast: N
```

5. **Execute either** boot **or** reset annex all **to effect these changes on the RAC.**

You can configure more than one RAC simultaneously using this sequence:

- Define the RAC using the **annex** command. Next, use the **set annex** command to change the parameters.
- Define the parameters for one RAC and use the **write** command to copy the parameters to a script file. The script file will contain all copied parameter settings with a comment character (#) at the beginning of lines defining the settings for the IP address, administrative password, virtual CLI password, LAT key, option key, and ACP key. Remove the comment character for any of the settings you want to use.
- Next, execute the **read** command for all RACs you want to configure.

Using the CLI admin Command

Entering the RAC superuser CLI **admin** command puts you in administrative mode. The *admin* prompt replaces the CLI prompt. Pressing the attention key or typing **quit** at the *admin* prompt terminates the **admin** session and returns you to the superuser CLI prompt.

The admin command functions only on the local RAC.

To modify RAC parameters using the **admin** command:

1. **At the CLI prompt, execute the** su **command and enter the superuser password:**

   ```
   annex: su
   password:
   ```

2. **At the superuser CLI prompt, execute the** admin **command:**

   ```
   annex# admin
   ANNEX-PRI Rx.x, 32 async, 32 sync, 32 ta, 32 modem ports
   admin:
   ```

3. **Execute the** set annex **command to change parameters. The following sample command lines:**

   - Enable the DNS name server
   - Define two name server hosts
   - Enable security on the RAC

   ```
   admin: set annex name_server_1 dns
   admin: set annex pref_name1_addr 192.9.200.95
   admin: set annex name_server_2 dns
   admin: set annex pref_name2_addr 192.9.200.85
   admin: set annex enable_security Y
   ```

4. Execute the show annex all **command to review your changes. Using the example in step 3, the terminal displays:**

```
admin: show annex all

Annex Generic Parameters

inet_addr:132.245.44.187            subnet_mask:255.255.255.0
pref_load_addr:132.245.44.80        pref_dump_addr:132.245.33.8
load_broadcast:Y                    broadcast_addr:132.245.44.255
load_dump_gateway:132.245.44.22     load_dump_sequence: net
image_name: ""                      motd_file: "motd"
config_file: "config.annex"         authoritative_agent: Y
routed: Y                           server_capability: none
disabled_modules: vci               tftp_load_dir: ""
tftp_dump_name: ""                  ipencap_type: ethernet
ip_forward_broadcast: N             tcp_keepalive: 120
option_key: ""                      session_limit: 1152
output_ttl: 64


VCLI Parameters

max_vcli: unlimited                 cli_prompt: "%a%c"
vcli_security: Y                    vcli_password: "<unset>"
vcli_inactivity off


Nameserver Parameters

nameserver_broadcast: N             rwhod: Y
pref_name1_addr: 192.9.200.95       name_server_1: dns
pref_name2_addr: 192.9.200.85       name_server_2: dns
host_table_size: 64                 min_unique_hostnames: Y


Security Parameters

enable_security: Y                  security_broadcast: Y
pref_secure1_host:192.9.200.95      pref_secure2_host: 0.0.0.0
network_turnaround: 2               loose_source_route: Y
acp_key: "<unset>"                  password: "<set>"
allow_snmp_sets: N                  lock_enable: Y
passwd_limit: 3                     chap_auth_name: "chap"
max_chap_chall_int: 0
```

(continued on next page)

```
Time Parameters

time_broadcast: N                   daylight_savings: us
timezone_minuteswest: 300           time_server: 0.0.0.0


SysLog Parameters

syslog_mask: all                    syslog_facility: log_local7
syslog_host: 192.9.200.95           syslog_port: 0

MOP and "Login" User Parameters

pref_mop_host: 00-00-00-00-00-00
mop_password: “<unset>”             login_password: “<set>”
login_prompt: “#”                   login_timer: 30


LAT Parameters

lat_key: ""                         facility_num: 0
server_name: ""                     sys_location: ""
lat_queue_max: 4                    service_limit: 256
keep_alive_timer: 20                circuit_timer: 8
retrans_limit: 8                    group_value: none
vcli_groups: none                   multicast_timer: 30
multisessions_enable: N


AppleTalk Parameters

a_router: 00-00-00-00-00-00
default_zone_list: ""               node_id: 0.0
zone: ""


Router Parameters

rip_auth: "<unset>"                 rip_routers: all


IPX Parameters

ipx_file_server: ""                 ipx_frame_type: raw802_3
ipx_dump_username: ""               ipx_dump_password: "<unset>"
ipx_dump_path: ""                   ipx_do_checksum: N


TMux Parameters

tmux_enable: N                      tmux_max_host: 64
tmux_delay: 20                      tmux_max_mpx: 700
```

(continued on next page)

```
DHCP Parameters
pref_dhcp1_host: 0.0.0.0          pref_dhcp2_host: 0.0.0.0
dhcp_bcast: N
```

5. **Execute either** boot **or** reset annex all **to effect these changes on the RAC.**

Customizing the RAC Environment

You can customize the following RAC attributes:

- The prompt that displays when a user accesses the CLI
- The number of simultaneous virtual CLI connections
- The name of the configuration file
- The name of the message-of-the-day file
- RIP
- The IP encapsulation type used by the LAN
- TMux
- LAT
- AppleTalk
- IPX

Setting the CLI Prompt

The RAC displays a prompt when a user accesses the CLI. The **cli_prompt** parameter allows you to customize the RAC prompt. You can also customize the prompt for each serial port using the **prompt** port parameter (for more details, see the *Remote Access Concentrator Software Reference Guide*).

The values for this parameter are called prompt strings. A prompt string consists of characters and embedded formatting codes that are expanded when the prompt is displayed. The formatting codes consist of a percent character (%) followed by a single lowercase character. Each formatting code occupies one character in storage. You can also specify a string for the prompt using these codes. The default is **%a%c** (*annex:*). Table 3-1 describes the codes for the prompt string.

Table 3-1. Formatting Codes for Annex Prompts

Code	Expansion
%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 1989.
%i	The RAC's Internet address, such as 132.245.6.40.
%j	A new line character, skip to the beginning of the next line.
%n	The RAC's name or Internet address, such as 132.245.6.40.
%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; if none, a null string.

If you want a prompt to appear as *date and time* (new line) *annex-name*, use the following code:

```
%d%j%n%s%c
```

For the port on the RAC named *thirdfloor*, the prompt is:

```
Mon Jan 6 11:10:25 1997
thirdfloor
```

For the superuser CLI prompt, a pound sign (#) and a space replace the code **%c**; otherwise a # is appended at the end.

Setting a Limit on Virtual CLI Connections

The number of virtual CLI connections on a RAC can affect the use of memory, because each virtual CLI connection uses memory. The **max_vcli** parameter determines the maximum number of virtual CLI connections the RAC can create at any one time. You can set the number of virtual CLI connections from an unlimited number to none. The range of values that you can enter are from **0** to **254** or **unlimited**. The default is **"unlimited"**. If you define this parameter as zero, users cannot create a virtual CLI connection on the RAC.

Setting Up the Configuration File

The configuration file contains all RAC configuration information. It resides on the preferred boot host or the local media and is loaded during the RAC booting process (see *Configuring Hosts and Servers on page 4-1* for more details on creating and using the configuration file).

You can define a name for the configuration file using the configuration parameter **config_file**. The default file name is **config.annex**.

You can create these files on the local media using the superuser CLI **edit** command.

Setting Up the motd File

The RAC can optionally display a message of the day at the CLI prompt. The RAC reads the **motd** file from the file server host each time it boots, and when the **reset annex motd** command is issued. Initially, the RAC requests the file from the preferred load host. If that host is not defined or available, the RAC broadcasts a request for this file unless you disable broadcasting by setting the **load_broadcast** parameter to **N**.

To use this option, create an ASCII file on a file server host with the desired message. This file must be located in the same directory that holds the operational images (usually **/usr/spool/erpcd/bfs**). The default file name is **motd**. If you use another name for this file, you must specify this name using the RAC parameter **motd_file**.

Remote Access Concentrator Service Tools for Windows NT stores the **motd** file in: <*product-installation-root-drive*>\bfs sub-directory.

Using RIP

The RAC uses a routing daemon (**routed**) for its routing services. This daemon implements Versions 1 and 2 of the Routing Information Protocol (RIP).

The **routed** parameter enables or disables RIP; the default is **enabled**. If RIP is disabled, the RAC does not listen for or transmit RIP routing updates. Instead, it depends on the routing information in the **gateway** section of the configuration file. If you disable RIP, define a default route in the configuration file.

Setting the IP Encapsulation Type

The RAC supports two types of LAN encapsulation for IP packets: Ethernet Version 2 format or IEEE 802.3 Data Link Layer format. The **ipencap_type** parameter specifies which type of encapsulation to use; the default is **ethernet**.

This parameter should be changed only at installation time using the ROM monitor. Do not change this parameter using **na** or **admin** because the RAC cannot boot with the wrong IP encapsulation.

Configuring LAT Services

The RAC can display, and connect to, currently available LAT services. Initially, all LAT functions in the RAC are disabled since this feature is optional. To enable the LAT functions, you must enter the correct **lat_key** parameter value and reboot the RAC (see *Configuring Hosts and Servers on page 4-1* for more details).

The **lat_key** parameter value is unique for each RAC. If you purchased LAT, contact Bay Networks to obtain your key.

Parsing the Configuration File

The configuration file contains RAC configuration information. It resides on the preferred boot host and is loaded during the RAC booting process.

The configuration file is parsed one line at a time during the boot process. File entries are grouped into sections. Each section begins with a percent symbol (%) followed by a defining keyword.

The configuration file can also contain **include** statements. The **include** statement incorporates separate file entries for backward compatibility.

In earlier software releases, **gateways**, **rotaries**, and **macros** are separate files. Beginning with Release 7.0, the keywords **%gateway**, **%macro**, and **%rotary** are entries in the configuration file. The file syntax has not changed. You can use **include** statements in the configuration file to incorporate files from earlier releases.

You can define a file name for the configuration file maintained on the load host using the RAC parameter **config_file**. The default file name is **config.annex**.

If the RAC is configured for self-booting, the configuration file must reside in the RAC's root directory.

File Sections

Each section of the configuration file begins with a keyword preceded by the percent symbol (%). The keywords are: **gateway**, **macro**, **modem**, **digital_modem**, **rotary**, **dialout**, **service**, and **wan**. The syntax is:

%*keyword*

You must specify the keyword before defining each section entry; otherwise, a format error occurs. The keyword distinguishes one section of the configuration file from another for the parser. All the statements under the **%**keyword should belong to the same section, including the statements in the **include** file.

Include Statement

The **include** statement tells the parser that entries in the file specified in the *filename* field are part of the configuration file. The syntax is:

%*keyword*
%include *filename*

Setting Up the Configuration File

The following sample configuration file defines the **%gateway** entry first. This entry includes a separate file named **test.route** followed by %**macro**, %**rotary**, %**dialout**, and %**service** entries. The **%rotary** entry includes another file named **test.rotary**.

```
#
#The following are definitions of the gateway entries
#
```

```
%gateway

route add -h 129.91.0.0/24 132.245.1.1 1
route add -h 129.122.0.0/24 132.245.1.1 1

annex 192.9.200.228
route add -h 129.123.0.0/24 132.245.2.1 1
snmp contact crow@xenna
snmp location Room without a view
end

# the file test.route is also part of configuration
%include test.route

# Configure SNMP
snmp community public
snmp traphost 192.9.200.95

#
# The following are definitions of the macro entries
#
%macro

alias  /Show users on the network/
     keyin “3” asy,pts@opus
     keyin “4” asy,pts@opus
{
<who
<pause
}

menu  |ANNEX MENU ONE|
     keyin “menu1” asy,pts@opus
     init_cli asy,pts@opus
     cmd_list bg,db_mgr,fg,hosts,jobs,rlogin

{
    MY MENU SCREEN
Choices are:

bg db_mgr fg hosts jobs and rlogin

command:
}
```

(continued on next page)

```
# service entries
%service

service adm_modem\
identification ‘system administrator modem’\
password anypasswd3453 ports asy\
connections enabled queue disabled

end

# rotary entries

%rotary
%include test.rotary

# All consoles from rack annexes

titanic_co:   asy@192.9.200.232
brazil_co:    asy@192.9.200.232
botswana_co:   asy@192.9.200.232

# Table top annex consoles

zinc_co:         25@192.9.200.230
total_recall_co: 26@192.9.200.230
conan_co:        27@192.9.200.230

# Remote Annex (192.9.200.247) - titanic

titanic_1:            asy@192.9.200.232
titanic_2:            asy@192.9.200.232

# Remote Annex (jdc)(192.9.200.249) - rlogin

rlogin_1:            protocol=rlogin asy@jdc
rlogin_2:            asy@jdc+132.245.33.229/513
rlogin_3:            protocol=rlogin
asy@jdc+132.245.33.228
```

(continued on next page)

```
# dial-out route for jupiter

%dialout

begin_route 5
local                192.9.200.233
remote               192.9.200.234
mode                 slip
ports                asy@192.9.200.230
phone                2522555
chat                 chat3
filter               out incl proto icmp disc
disabled             8:00am-6:00pm Mon-Fri
advertise            y
set                  slip_mtu small slip_tos Y
set                  rip_horizon split
end_route

begin_script chat3
send "Slip\r"
end_script
```

Another sample configuration file containing four include statements follows:

```
# The following are definitions of the gateway entries
#
%gateway
%include gateways
#
# The following are definitions of the macro entries
#
%macro
%include macros
#
# The following are definitions of the rotary entries
#
%rotary
%include rotaries
#
# The following are definitions of the LAT service entries
#
%service
%include service.lat
```

Creating %gateway Entries in the Configuration File

You can create **%gateway** entries using any text editor. After a RAC boots, it downloads the information from the preferred load host. If the RAC does not locate the configuration file, it assumes the file does not exist on the network. Table 3-2 and Table 3-3 describe the supported keywords for **%gateway** entries.

The **%gateway** entries configure routes for SLIP and PPP links and enable the Simple Network Management Protocol (SNMP) agent. The **%gateway** entries must conform to the following conventions:

- A number sign (#) in the first column starts a comment. The comment is terminated by the end-of-line.
- Non-delimited white space (i.e., spaces, tabs, etc.) is treated as a single space.
- All keywords and port information are case-sensitive.

A **%gateway** entry for a route in the configuration file can have one of the two equivalent formats shown below. The first format is preferred. The second is allowed for backward compatibility with releases prior to R9.3.

Format 1:

route add [**-h**] {*addr1 subnet_mask* | *addr1/bits* |**default**} *addr2* [*metric*]

Format 2:

{**net**|**host**} *addr1/bits* **gateway** *addr2* **metric** *hops* {**active**|**passive**|**hardwired**}

domain search *pathname pathname...*

domain default *pathname*

Table 3-2. Supported Keywords for %gateway Entries - Format 1

Keyword	Definition
route	The %gateway entry is for a host, network, or default route.
add	The route is to be added to the configuration file.
-h	This route is hardwired; it cannot be changed or deleted, even if a routing update is received.
addr1	The route's destination address.
subnet_mask	The subnet mask for *addr1*. If the route is not a default route, you must specify a subnet mask, either in this field or in the /*bits* field.
/*bits*	The subnet mask for *addr1*. This number represents the combined number of 1 bits in the network and subnet mask, from left to right. If the route is not the default, you must specify a subnet mask either in this field or in the *subnet_mask* field. You cannot specify /*bits* for default routes.
default	This is the default route for the RAC.
addr2	The IP address of the next gateway the RAC uses to get to the destination address.
metric	The cost of using this route; typically, this is the number of hops from the RAC to *addr2*.

Table 3-3. Supported Keywords for %gateway Entries - Format 2

Keyword	Definition
net|host	The destination specified by *addr1* is a network or a host.
addr1	The IP address of the destination; use either 0 or default to add a default route.
/*bits*	This field specifies a subnet mask for *addr1*. The value of *bits* represents the combined number of 1 bits in the network and subnet mask. You cannot specify this field for default routes.
addr2	The IP address of the gateway that reaches the destination address.
hops	The number of hops needed to reach the destination.
active	The gateway can generate RIP update messages. If the gateway does not transmit these broadcasts on a regular basis, it eventually is deleted from the routing table. If another route to the destination associated with the gateway is received, the new route replaces the active gateway entry.
passive	The gateway cannot generate RIP messages, and, therefore, is never removed from the routing table. If a better route is received for the destination, the entry in the routing table is updated.
hardwired	The gateway has a fixed route to the destination. This route cannot be changed or deleted, even if a routing update is received.
domain search	Adds a network domain to the DNS search path. The given paths are decomposed into higher-level names, and all of the forms are added.
domain default	Sets the default search path. This path should be set after all other search paths are added. The RAC propagates this path to the top of the search list and removes it from all of the *hosts* entries.

The CLI **hosts -n** command displays the domain search and the domain default list contents along with the name server data.

%gateway Extensions

The **%gateway** extensions allow you to define lines in the configuration file that refer only to a specific RAC, or to all RACs, or to a specific subnet.

The syntax for an extension that includes a specific IP address is:

annex *ipaddr*
...
end

The syntax for an extension that matches all RACs (useful for local files and to force route caching) is:

annex *
...
end

The lines enclosed by the **annex...end** block are to be used only by the RAC with the IP address *ipaddr*. Any routes enclosed by the **annex...end** block are cached. An **else** keyword can also be used (alone on a line) to list configuration information for all RACs, except the one identified on the **annex** line. You cannot nest **annex...end** blocks.

The following sample extension to a **%gateway** entry in the configuration file shows how to configure a SLIP interface (see *. SLIP Link with Two IP Addresses on page 10-11*).

```
%gateway
# SLIP link to the 132.245.5 net
annex 132.245.5.9

   # 132.245.10.7 is a gateway to the entire 132.245.5 net
  net 132.245.5.0/24 gateway 132.245.10.7 metric 1 hardwired

else

   # other Annexes will route to 132.245.5 via 132.245.5.9
  net 132.245.5.0/24 gateway 132.245.5.9 metric 2 hardwired

end
```

An extension that matches all RACs on a given network or subnet is particularly useful for defining a default route shared by these RACs. The syntax is as follows:

subnet *ipaddr*
...
end

The lines enclosed by the **subnet...end** block are to be used only by RACs on the subnet or network with the IP address *ipaddr*. Any routes enclosed by the **subnet...end** block are cached. An **else** keyword can also be used (alone on a line) to list configuration information for all subnets/networks except the one identified on the **subnet** line. You cannot nest **subnet...end** blocks.

The following are sample **subnet...end** blocks:

```
subnet 132.245.33.0
route add default 132.245.33.22 1
end
subnet 132.245.66.0
route add -h default 132.245.66.22.1
end
```

In the sample above, the first default route applies to all RACs on subnet 132.245.33.0. The route is not hardwired, so it can be replaced by a default route learned by RIP or defined elsewhere. The second default route will never be replaced (unless you change it) because it is specified as hardwired (-h).

The RAC logs errors in **%gateway** entries if *syslogging* is enabled. For more details on event logging, see *Encrypting Security Messages on page 6-59*. The **syslog_mask** parameter determines the priority levels for logging these event messages.

For information on SLIP links, see *Serial Line Internet Protocol on page 10-1*.

For information on PPP links, see *Point-to-Point Protocol on page 8-1*.

For information regarding keywords for configuring the SNMP agent, see the *Remote Access Concentrator SNMP MIB Reference*.

Loading the Host Table from the Configuration File

When the RAC boots, it adds the host name entries in the **%gateway** section of the configuration file to the host table. These entries reside on the host table until a name server overrides the entries' information or until you reset the RAC name server using the **na** or **admin** commands. These entries are similar to the **/etc/hosts** file entries, except that aliasing is not supported.

A host name entry has an IP address followed by white space (blanks and/or tabs) followed by a host name (the host name may not contain blanks, tabs, or newlines). Some sample host name entries are:

```
192.9.200.1 cbrown
192.9.200.2 snoopy
192.9.200.3 linus
192.9.200.4 lucy
192.9.200.5 sally
```

Host name entries may be conditional with the use of the **annex...end** blocks (see *Creating %gateway Entries in the Configuration File on page 3-19*). The address on the **annex** statement must be an IP address. Each IP address has only one associated host name.

Routing Services and %gateway Entries

The RAC parameter **routed** determines whether or not the Routing Information Protocol (RIP) routing daemon is enabled (see the *Remote Access Concentrator Software Reference Guide* for more details). When the daemon is enabled, the RAC performs both active and passive RIP routing. If the daemon is disabled, no RIP routing occurs (for more information on RIP, see *Routing Information Protocol (RIP) on page 11-1*).

A routing table maintains information about gateways that provide routes to hosts on different networks. If the **routed** parameter is set to **Y**, the RAC builds this table dynamically by monitoring RIP broadcast messages; both Version 1 and Version 2 of RIP messages are accepted. If **routed** is set to **N**, the RAC builds the table by monitoring only ICMP redirects.

The CLI **netstat -r** command displays the routing table.

Another option for maintaining the routing table is to create **%gateway** entries in the configuration file for the RAC in which you define fixed routes to a destination. If you disable RIP, the RAC relies only on the **%gateway** entries and ICMP redirects.

Route Cache

The route cache contains user-configured routing information (static routes, and sometimes a default route). The RAC copies these routes to the route cache from **annex...end** blocks in the **%gateway** section of the RAC configuration file (see *Creating %gateway Entries in the Configuration File on page 3-19*) and/or from routes defined via the CLI superuser **route** command (see *Defining Routes on page 11-27*).

Routes in the cache include those whose next hops are directly reachable, that is, up and running on a network directly connected to the RAC, and those that are not yet reachable. Routes whose next hops are reachable are immediately copied to the RIP and kernel routing tables. Routes whose next hops are not yet directly reachable are copied to the RIP routing table as soon as their next hops become reachable. The latter technique saves the RAC the trouble of consulting the configuration file, which is typically not stored on the RAC, each time a route's status changes. A copy of the route remains in the routing cache, in case its next hop again becomes unreachable.

The **netstat -C** command displays the route cache.

Disabling RIP

Another option for maintaining routing tables is to disable RIP on the RAC by setting the RAC parameter **routed** to **N**. This prevents the RAC from reacting to RIP broadcasts and using alternate routes. The RAC constructs its routing table from the information contained in the **%gateway** section of the configuration file and ICMP redirects.

If you disable RIP, define a default route to a smart gateway in the **%gateway** section of the configuration file. This gateway sends ICMP redirects to the RAC, allowing it to learn only the required routes.

Enabling and Disabling AFD

Refer to *Creating %gateway Entries for AFD on page 3-53* for complete information on using **%gateway** entries to enable or disable automated firmware download (AFD) for the RAC.

Creating %macro Entries in the Configuration File

You can customize the CLI user interface by adding **%macro** entries to the RAC configuration file. Using **%macro** entries, you can set up site-specific prompts, commands, and menus. Aliases can make the RAC CLI invisible to the user. Menus hide the RAC's command interface and at the same time provide user options. Aliases and menus can be bound to specific ports on a RAC. Also, aliases can be created for slave ports that are accessed through the port server. Individual aliases and menus can be configured to appear each time a port is accessed. Table 3-4 describes the supported keywords for **%macro** entries.

The maximum number of macros per RAC is 128.

The RAC loads the **%macro** entries when it boots or when the **reset annex macros** command is issued using either **na** or **admin**. If the preferred load host is not available, and the RAC **load_broadcast** parameter is set to **Y**, the RAC broadcasts for the configuration file.

The **%macro** entries in the configuration file must conform to the following conventions:

- A pound sign (**#**) in the first column starts a comment. The comment is terminated by the end-of-line.
- The sequence of backslash-closing brace (**\}**) allows a closing brace character (**}**) to appear in the body of a macro.
- Non-delimited white space (i.e., space, tab, etc.) is treated as a single space.
- All keywords and port information are case-sensitive.
- Many strings, including description strings, must be enclosed in delimiters. The delimiters can be any printable character not contained within the string.
- The maximum number of characters allowed between the opening and closing braces is 1024 (a *syslog* message is sent if this length is exceeded).

- Start a macro with a left brace (**{**) character on a new line.
- End a macro with a right brace (**}**) character on a new line.

The entries in the **%macro** section of the configuration file follow one of these forms:

alias |*description*|		**menu** |*description*|
keyword arguments	or	*keyword arguments*
{		***{***
alias expansion		*menu expansion*
}		***}***

The **alias** command begins an alias definition; the **menu** command begins a menu definition. The *description* is a string that can contain spaces. The string is the information displayed by the CLI **help** command for the alias or menu. The bar character (|) around the *description* is the string delimiter; the delimiter can be any character.

The RAC does not support subsets of virtual connections.

Following the keyword is the expansion text of the macro. This text consists of a series of lines with one entry per line. The expansion text begins after a line with a single left brace (**{**) and ends before a line with a single right brace (**}**). If the expansion text is for an alias, it contains an *alias expansion*; for a menu, the expansion text contains a *menu expansion*. Table 3-4 describes the supported keywords for %macro entries.

Table 3-4. Supported Keywords for %macro Entries

Keyword	Description
keyin |*name*| *port_type*	Binds *name* to current macro on the port types defined in *port_type*. The *name* is used to execute the **menu** or **alias**. The *name* must be delimited.
init_cli *port_type*	Binds the macro to be executed on the initialization of a CLI at any port in the *port_type*. Applicable only with CLI, adaptive, or virtual CLI ports.
init_psrv *port_type*	Binds the macro to be executed when the port server connects to any port in the *port_type*. Applicable only with ports whose **mode** parameter is set to **slave** or **adaptive**. Use only with the **alias** macro form.
cmd_list *cmd1*,*cmd2*,...	Specifies the commands that are available with the menu being defined. Each command is separated by a comma. A command can be a macro (**menu** or **alias**) or a CLI command. Spaces are not permitted in the command list. Applicable only with menus.

Aliases listed in a **cmd_list** must be valid for the same ports defined with the keyword **keyin** that defined the alias.

The whole *port_type* can be followed by an @ and the name or IP address of a RAC. If you do not specify a RAC for a *port_set*, the macro applies to the specified ports for any RAC that reads **%macro**. The *port_type* cannot contain spaces. If a *port_type* is not defined for a menu or alias, the entry applies to all ports on the RAC reading the file and all virtual CLI connections.

Sample *port_type* entries look like this:

```
asy@132.245.6.42
asy,pts@132.245.6.78
asy@thirdfloor
asy,pts
asy
```

Table 3-5 describes the statements permitted in an *alias expansion* on a CLI line.

Table 3-5. Statements Permitted in an Alias Expansion

Statement	Description
> *string*	Indicates that the *string* should be transmitted to a port; for a virtual CLI, it directs the *string* to the connected device. This can be used to display messages to the CLI, but not jobs.
< *string*	Simulates the *string* being input from a port; for a virtual CLI, it simulates input to the CLI, e.g., CLI commands.
< pause	Indicates a pseudo-CLI command that causes the macro to display *Hit any key to continue* on the port and to wait for a keystroke before continuing. This is applicable only with alias definitions used on a CLI. To define the wait time, use the syntax < *pause xx.x* where *xx.x* is the time in seconds and tenths of seconds.

The *menu expansion* defines the menu that is displayed. Each line of the *menu expansion* is a separate line of the menu. If no lines are defined between the open and close braces, the RAC creates a generic menu containing a list of the commands that were defined with the **cmd_list** keyword and their descriptions. In this case, the RAC displays the default RAC prompt. If you include any superuser CLI commands in a menu, they appear only when a superuser accesses the menu.

Examples of Aliases and Menus

The following sample **%macro** entries define a series of aliases combined in a menu. The menu is accessible from port 3 and all virtual CLI connections for a RAC at a given address.

```
%macro
#
# Set up a macro for annex at annex-address
#
# This macro sets up a menu for port 3 and all virtual ports
# on the specified annex.
#
# Other examples of setups are:
#
#  keyin "1" asy@annex-address
#  ( asy@address = asy ports for annex @ address)
#
#  keyin "2" pts@annex-address
#  ( pts@address   = all virtual ports for annex @ address)
#
#  keyin "3" pts
#  ( pts           = all virtual ports for any annex that
#                    boots from this host)
#
#  keyin "4" asy
#  (asy = ports 1 through 72 for any annex that
#                    boots from this host)
#

# The menu displays after clearing the screen, five lines down
# from the top of the screen and 27 spaces over.
#
# This macro limits the possible commands at the Annex prompt
# to only those listed on the menu.
#
# This macro file does not affect any other Annex that boots
# from this Unix host.
#
# The menu displays after clearing the screen, five lines down
# from the top of the screen and 27 spaces over.
#
# This macro limits the possible commands at the Annex prompt
# to only those listed on the menu.
#


# This macro file does not affect any other Annex that boots
# from this Unix host.
#
# All other ports on the Annex have full access to cli
# commands.
#
```

(continued on next page)

```
# Note:Replace annex-address with your Annex's IP address
#      (e.g., 192.9.200.1).
#
#      Replace system-address with your system's name or IP
#      address (e.g., fred or 192.9.200.2).
#
# Rlogin to system1
#
alias "Connect to System1"
     keyin "1" asy,pts@annex-address
{
<rlogin system1-address
}

#
# Rlogin to system2
#
alias "Connect to System2"
     keyin "2" asy,pts@annex-address
{
<rlogin system2-address
}

#
# Issue a "who" command to determine who is running on the
# Annex; wait for a <Return> before returning to the menu.
#
alias "Who?"
     keyin "3" asy,pts@annex-address
{
<who
>
<pause
}
#
# Do hangup from Annex port. This disconnects the Annex port.
#
alias "Exit"
     keyin "4" asy,pts@annex-address
{
<hangup
}
```

(continued on next page)

```
#
# This section defines the actual menu.
#
menu |Generic Menu Header|
     init_cli asy,pts@annex-address
     keyin “menu” asy,pts@annex-address
     cmd_list 1,2,3,4
{

#
# The “^[[2J” is a control sequence of VT100-type term commands
# CTRL-[ followed by “[2J”    (clear the terminal screen)
# CTRL-[ followed by “[5;27H” (go to line 5, space over 27
# spaces)
# CTRL-[ followed by “[9;15H” (go to line 9, space over 15
# spaces)
#
^[[2J
^[[5;27HGeneric Menu Header
^[[9;15H1)    Connect to System1
^[[11;15H2)   Connect to System2
^[[13;15H3)   Who?
^[[15;15H4)   Exit
^[[17;15H^[[1mEnter Number: ^[[m
}
```

The last entry creates the display for the menu and includes the above aliases in the menu’s **cmd_list**. This entry also includes an **init_cli** keyword, which causes the menu to be initialized each time the port or the virtual CLI is reset. This displays the menu without communicating with the CLI. The menu includes ANSI terminal control codes to set up the menu display:

```
          Generic Menu Header

     1) Connect to System1
     2) Connect to System2
     3) Who?
     4) Exit

     Enter Number:
```

The following example uses the same aliases as in the previous example, but does not provide a *menu expansion* to define the menu display:

```
menu |Annex menu|
     init_cli asy,pts@annex-address
     keyin “menu” asy,pts@annex-address
     cmd_list 1,2,3,4
{
}

This entry creates the following menu:

     Annex Menu

1    : Connect to System1
2    : Connect to System2
3    : Who?
4    : Exit
```

The following sample **%macro** entries automatically connect any user logging in on the asy ports of the defined RAC to the given *system-address*, without requiring a keystroke; the virtual ports have normal connection options. This macro is both RAC- and host-specific.

```
alias “Connecting to host”

init_cli asy@annex-address
{
>
> Please wait while you are connected......
>
<rlogin system-address
}
```

Set the **cli_inactivity** parameter to **immediate**: when a user logs off the last job, the macro is re-initiated; otherwise, the CLI prompt returns at logout.

Adding Control Characters

Adding control characters in macros for cursor movement is editor- and system-dependent:

- In the built-in RAC editors, *vi*, *ex*, or *ed*, press CTRL-V followed by the control character you want to insert (e.g., CTRL-[for escape or CTRL-M for a carriage return).
- In *emacs*, press CTRL-Q followed by the control character you want to insert.

For more details or information on other editors, see that editor's documentation.

The following sample macro inserts carriage returns along with the line feeds:

```
alias “Connecting to host”
#
# Where the ^M is a carriage return 0xb character.
#
# Menus for logins to various locations
#
#
alias "Aqua"
  keyin "1" asy@annex-address
{
<rlogin aqua
}

alias "Peach"
  keyin "2" asy@annex-address
{
<rlogin peach
}
alias "Node 7"
  keyin "3" asy@annex-address
```

(continued on next page)

```
{

<telnet node7
}

alias "Exit"
  keyin "4" asy@annex-address
{
<kill
<hangup
}

menu | Administrative Terminal Service |
  init_cli asy@annex-address
  keyin "menu" asy@annex-address
  cmd_list 1,2,3,4
{^M
^M
^M
^M

Administrative Terminal Service^M
================================^M
^M
^M

        1) Aqua
        2) Peach^M
        3) Node  7^M
        4) EXIT^M
^M
Enter selection for desired application (1 - 4) : ^M
}
```

Managing %macro Entries in the Configuration File

The CLI **help** command displays an alias name and description with other valid CLI commands. The superuser CLI **help -m** command displays a list of all macros and their assigned *port_type* defined for that RAC. For example:

```
annex01# help -m

Name    Assigned Ports            Description
=========================================================
==
1       asy:Menu 1
2       asy,pts                  :Read EMAIL
3       asy,pts                  :Command disabled
4       asy,pts                  :Another who command
=========================================================
==
init_cli asy:Set stty commands
init_cli asy,pts    :Another who command
init_cli asy        :Dedicated port macro
init_psr asy        :Port 10 information
annex01#
```

If you issue the name defined with the keyword **keyin** along with **help -m**, the command displays the definition defined with that entry:

```
annex01# help -m 2

Macro Name: 2   Description: Read EMAIL
Assigned Ports: asy,pts
Functional Text:

<rlogin maildrop
<mail

<<
annex01#
```

The **reset annex macros** command instructs the RAC to reload **%macro**. Thus, you can modify the **%macro** section of the configuration file and load it onto a RAC without having to reboot.

Creating %service Entries in the Configuration File

The **%service** entries in the configuration file define the LAT services that the RAC advertises. Table 3-6 defines the available arguments. The syntax for each entry is:

service *name* [**identification** *id_string*] [**password** *password*] \
ports *port_type* [**connections** ***enabled***|***disabled***]\
[**queue** ***enabled***|***disabled***]

Table 3-6. Supported Arguments for %service Entries

Argument	Description
name	A 16-byte ASCII string defining the service name.
id_string	An optional 32-byte ASCII string containing additional information about the service. The default is an empty string (no identification).
password	An optional 16-byte ASCII string defining the service password. The default is an empty string (no password).
port_type	Ports offering a particular service.
connections	An optional parameter that defaults to ***enabled***.
queue	An optional parameter that defaults to ***disabled***.

In the following sample **%service** entry in the configuration file, the first sample **%service** entry defines a service called *adm_modem*. The identification field provides additional information about the service. This service is attached to RAC asy ports; it is password protected and it is enabled for connection requests. Request queueing is disabled.

The second sample **%service** entry prevents the RAC from advertising VCLI service.

```
% service
annex 132.0.0.50

service adm_modem\
identification ’system administrator modem’\
password anypasswd3453 ports asy\
connections enabled queue disabled

service vcli no

end
```

When queueing is enabled, the service is not advertised and is not visible through the keyword **%service**. Also, these services require a special set-up on the LAT host.

Creating %rotary Entries in the Configuration File

Refer to *The Port Server on page 5-38* for complete information on creating rotaries.

Creating %digital_modem Entries in the Configuration File

Refer to *Customizing Modem Configuration on page 7-6* for complete information on configuring digital modems for the RAC.

Creating %dialout Entries in the Configuration File

Each entry in the **%dialout** section of the configuration file defines a dial-out route. Table 3-7 lists the field definitions for **%dialout** entries. The format for a **%dialout** entry looks like this:

```
%dialout

global_timeout <time-out value>

# this is a comment line

begin_route          <route id>
local                <local address>
remote               <remote address>
mode                 <SLIP or PPP>
ports                <port type/rotary>
phone                <phone number>
chat                 <chat script list>
filter               <filter command>
disabled             <time interval>
advertise            <Y or N>
set                  <parameter_name setting>
end_route
```

Table 3-7. Field Definitions for %dialout Entries

Field	Definition
begin_route	Marks the beginning of a dial-route entry and defines the route ID (an integer).
local	The IP address or machine name of the interface's local endpoint. If this optional field is omitted, the RAC's address is used.
remote	The IP address or machine name of the remote endpoint of the interface. This mandatory field must appear once.
mode	The serial protocol to be used, e.g., SLIP or PPP. This mandatory field must appear once.
ports	Specifies the port type to which this route can be assigned. This field can have an associated port type (e.g., asy), rotary name (e.g., rotary_1), or a combination of rotary names and port types (e.g., rotary_1,asy). If the entry is a rotary name, you must also enter this name in the **%rotary** section of the configuration file. This field must appear at least once (i.e., you can enter multiple *ports* fields). The order in which the rotary name and port types are entered determines the order in which ports are selected when a port must be chosen. The first port entry has the highest selection level and the last entry has the lowest level. If a port re-appears on another *ports* line, the selection level is determined by the port's initial appearance.
phone	A string of up to 32 characters that defines the phone number for the modem to dial.
chat	This string of up to 32 characters defines the name of the script that coordinates communications with the remote side immediately after the phone connection is established. You can enter multiple chat scripts on a single line separated by commas or use multiple chat lines; the chat scripts are executed in the order in which they appear in the **%dialout** entry.

(continued on next page)

Table 3-7 . Field Definitions for %dialout Entries (continued)

Field	Definition
filter	Defines a filter to apply to the dial-out route. You can enter only one filter per line. The filters are executed in the order in which they appear in the **%dialout** entry. Using this field is the only way you can add a dial-out filter; you cannot add one via the CLI **filter** command. Moreover, you omit the word **add** and the interface name from the filters you define in the **%dialout** entry.
disabled	Specifies when the dial-out route will be disabled automatically. The syntax is: [*time*] [*day*][- *time day*]. *Time* is specified as: *hh:mm*[*am*,*pm*]. If *am* or *pm* is not specified, 24-hour notation is assumed. If *time* is not specified, the default is all day. *Day* can be a weekday, i.e., Sunday, Monday, etc., or a month and a day. Weekday and month specifications observe minimum uniqueness and are not case sensitive. If *day* is not specified, the default is every day. During the disabled period, the route will not appear in the RAC's routing tables and cannot be activated. An active route that enters its disabled time interval will have its physical port reset and is removed from the routing tables. A disabled dial-out route will appear in the output of the CLI **dialout** command. Dialing into a RAC with a disabled route can activate the route if the remote address is within the dial-out's subnet. For this reason, disabling the route is effective for saving telephone costs but not for providing security. (continued)

(continued on next page)

Table 3-7 Field Definitions for %dialout Entries (continued)

Field	Definition

disabled (continued)	Sample entries for the *disabled* field are:
	`8:00am Friday - 6:35pm Friday`
	`Wednesday`
	`10:30 Nov 30 - 21:30 Nov 31`
	`Friday - Sunday`
	In the first example, the dial-out route is disabled at 8:00 a.m. on Friday and re-enabled at 6:35 p.m. on Friday. In the second example, the dial-out route is disabled all day on Wednesdays. In the third example, the dial-out route is disabled at 10:30 a.m. on November 30 and enabled at 9:30 p.m. on November 31. In the last example, the dial-out route is disabled on Fridays, Saturdays, and Sundays.
	Setting the *disabled* field does not prevent another RAC from dialing into this RAC via a complementary dialout route.

advertise	Specifies whether or not a dial-out route will be advertised via RIP even if there are no available ports in its rotary. A **Y** enables this field, an **N** disables it. If *advertise* is not specified, the default is **Y**.
set	Specifies the parameter settings that will be applied to the dial-out connection. These settings override the values in non-volatile memory while the process is alive, but they will not change the actual values in non-volatile memory. The syntax is: **set** [*parameter parameter_value*] Any parameters not specified in the *set* field are determined by the actual non-volatile memory settings; the RAC disregards any duplicate valid parameter settings. Table 3-8 on page -43 1ists the configuration parameters that can be set within this field.
end_route	Marks the end of a dial-out entry.

Table 3-8. Parameters That Can Be Set within the set Field of the %dialout Entry

Port Generic Parameters	
autobaud	parity
data_bits	speed
location	stop_bits
mode	
Flow Control and Signal Parameters	
control_lines	input_flow_control
input_start_char	input_stop_char
output_flow_control	output_start_char
output_start_char	need_dsr
ixany_flow_control	backward_key
forward_key	
Port Security Parameters	
port_password	user_name
Serial Networking Protocol Parameters	
allow_compression	net_inactivity
dialup_addresses	net_inactivity_units
do_compression	phone_number
local_address	remote_address
metric	slip_ppp_security
SLIP Parameters	
slip_allow_dump	slip_no_icmp
slip_load_dump_host	slip_tos
slip_mtu_size	subnet_mask

(continued on next page)

Table 3-8 Parameters that can be Set within the set Field (continued)

PPP Parameters	
ppp_acm	ppp_security_protocol
ppp_mru	ppp_username_remote
ppp_password_remote	
Interface Routing Parameters	
rip_accept	rip_recv_version
rip_advertise	rip_send_version
rip_default_route	rip_sub_accept
rip_horizon	rip_sub_advertise
rip_next_hop	

You do not need to provide explicit values for the port security parameters **port_password** and **ppp_password_remote**. If an asterisk (*) appears as the value for either of these parameters, and the RAC parameter **enable_security** is set to **Y**, the ACP security server provides the password by searching the **acp_userinfo** file (see *Dial-Out Passwords* in the following section).

Dial-Out Passwords

A user entry in the **acp_userinfo** file can specify a dial-out password. The keyword *dyndial_passwd* marks these entries, which can appear anywhere in the user record. ACP uses the *dyndial_passwd* if the configuration file's *dialout set* field for either *port_password* or *ppp_password_remote* contains an asterisk (*) and the **enable_security** parameter is set to Y. (For more details on using the **acp_userinfo** file, see *Creating the acp_userinfo File on page 6-74*):

```
user smith
     dyndial_passwd   jupiter
end
```

If the configuration file's *dialout set* field for either *port_password* or *ppp_password_remote* contains a valid value other than *, the RAC uses that value.

Dial-out Routes

After booting or executing the **na**/**admin** command **reset annex %dialout**, the RAC loads each dial-out route belonging to it and assigns each route an interface name. Each dial-out interface name is in the form **do** *route_id* (the *route_id* is specified in the *begin_route* entry in the **%dialout** section of the configuration file). You can configure the RAC for any number of dial-out routes and the *route_id*s do not have to be contiguous. The dial-out route interface names can change only after a system reboot or a **reset annex dialout** command.

The RAC ignores any dial-out route that has a mode (SLIP, PPP) that is disabled via the **disabled_modules** parameter.

Chat Scripts

Chat scripts coordinate the communication between each side of a dialed connection. They consist of a sequence of exchanges, usually following the pattern: *send x, expect y, receive z, compare y and z. If OK, continue with the next exchange*.

A common application for a chat script is the login sequence. When dialing into a host that is running a login process on that port, a sequence of username/password prompting occurs. A chat script can send the username and password automatically, thereby logging the user into the port. Chat scripts are also helpful when dialing into the RAC. For example, a chat script can start a SLIP process on the dialed port by including the **slip** command in the script.

A chat script is not applicable to all dial-out routes. It is not applicable when the port receiving the call is running SLIP or PPP and waiting to be dialed. It is applicable (and mandatory) for a dial-out route when the dialed port is a dial-in CLI or is running some shell native to the dialed machine.

Chat scripts, like dial-out routes, are configured in the **%dialout** section of the configuration file. Each chat entry begins with the field called *begin_script*. Table 3-9 defines the field definitions for chat scripts.

The format of a chat script looks like this:

```
% dialout

begin_script          <script name>, <time-out value
(optional)>
call                  <script name>
sleep                 <sleep length>
send                  <string>[,<string>]...
expect                <string>[,<script name>]
expect_case           <string>[,<script name>]
timeout               [<time-out value>][,<script name>]
end_script
```

The *call*, *sleep*, *send*, *expect*, and *expect_case* statements can appear in any order and in any combination.

Table 3-9. Field Definitions for Chat Scripts

Field	Definition
begin_script	Marks the beginning of a chat script; specifies the name of the script along with an optional time-out value. Contains a maximum of 32 characters.
call	Executes another chat script.
sleep	Causes a pause in the script for <*sleep length*> in seconds.
send	Sends a string. The string must be enclosed in double quotes. Multiple strings can be entered on one line or on multiple lines.

expect
expect_case
	Sends control to another script upon reception of a string; the first match is used. When <*string*> is received, <*script name*> is called; the default is *continue*. The *expect* field is not case-sensitive. The *expect_case* field is for use with case-sensitive matches. You can block together multiple *expect* statements:
	`expect <string 1>, <script 1>`
	`expect <string 2>, <script 1>`
	`expect <string 3>, <script 3>`

timeout	Specifies the total amount of time to wait (in seconds) for strings defined in a block of *expect* statements. The *timeout* statement terminates a block of *expect* statements. The <*script name*> is the chat script name to execute upon timeout; the default is *error.*
end_script	Marks the end of a chat script.

String Formatting Extensions

Any character can be inserted into the *send*, *expect*, and *expect_case* strings using the backslash (\) character followed by an octal number of no more than three characters. For example, to send a newline character (octal: 12), insert a *\12* in the *send* string. The sequence of octal numbers following the backslash character cannot exceed 377 (255 decimal). Some control characters can also be represented as follows:

Name	Decimal Value	Representation
BEL	7	\a
BS	8	\b
TAB	9	\t
LF	10	\n
FF	12	\f
CR	13	\r

To send the backslash character (\), insert it into the string using "\\". Send a break using "\k".

Reserved Keywords

Table 3-10 defines the four reserved keywords that can be used in place of a script name in the *expect*, *expect_case*, and *timeout* chat script statements.

Table 3-10. Reserved Keywords Used in Place of a Script Name

Keyword	Definition
success	Stop all chat activity and return successfully. Go to next chat in dialout, if any.
error	Stop all chat activity and return unsuccessfully. Abort dialout. In the *timeout* statement, if <*script name*> is omitted, *error* is assumed.
continue	Continue executing current script. In the *expect* statement, if <*script name*> is omitted, *continue* is assumed.
return	Return successfully from currently executing script.

Table 3-11 defines the two reserved keywords that can be used in place of a string in the *send*, *expect*, and *expect_case* statements (see *Chat Script Examples on page 3-50* for a sample chat script that uses these keywords).

Table 3-11. Reserved Keywords Used in Place of a String

Keyword	Definition
user_name	Use the user name associated with the port. Do not put quotes around this entry.
port_password	Use the password associated with the port type. Do not put quotes around this entry.

Default Global Timeout Values

The overall time for a dial-out route to complete its chat script activity is two minutes (120 seconds). You can override this value using the *global_timeout* statement. The syntax is:

global_timeout <global time-out value>

The *global_timeout* statement must appear within the **%dialout** section of the configuration file but outside any route.

Default Timeout Values

If the <*timeout value*> is omitted from the *timeout* statement, the default of 5 seconds is used. Each chat script can have its own default *timeout* value by specifying a value in the *begin_script* statement using the format:

```
begin_script <script name>, <default time-out value>
```

See *Chat Script Examples on page 3-50* for a sample chat script that uses this format.

Typically, the *timeout* statement terminates a block of *expect* statements. If a block of *expect* statements is terminated by a statement other than *timeout*, the <*time-out value*> used is the <*default time-out value*> for the script, and the <*script name*> used is *error*.

Chat Script Examples

The following sample chat script illustrates the RAC's chat script language. This script first calls the chat script called *chat2*. If *chat2* is successful, *chat1* continues (i.e., sleeps for 5 seconds). If *chat2* is unsuccessful, *chat1* returns as unsuccessful. The chat script then sends the string called *String1*. If *String2* (case-sensitive) does not arrive within 5 seconds, an error is returned. If *String2* does arrive within 5 seconds, the chat script called *chat3* is called. Since *chat3* always returns successfully, *chat1* also returns successfully.

```
begin_script          chat1
call                  chat2
sleep                 5
send                  "String1\r"
expect_case           "String2", chat3
timeout               5, error
end_script

begin_script          chat3
send                  "String3\r"
end_script

begin_script          chat2
sleep                 3
send                  "Message #1\n"
expect                "OK"
expect                "NOT OK", error
timeout               10
end_script
```

The following sample chat script can start a SLIP line on the called RAC CLI port. This script sends the string *slip* with a carriage return. Then it waits 5 seconds for a case-sensitive match on the string *Switching to SLIP*. If the match times out, the script will return as unsuccessful. If the *expect_case* field receives the expected string, it returns successfully.

```
begin_script          chat_slip
send                  "slip\r"
expect_case           "Switching to SLIP", success
timeout               5, error
end_script
```

The following sample script waits 10 seconds for the string *username*: because this <*default time-out value*> is specified in the *begin_script* statement. The <*default time-out value*> is used only if a <*time-out value*> is omitted from the *timeout* statement. If a <*default time-out value*> is not specified on this line, the default remains 5 seconds.

```
begin_script          script1, 10
send                  “login”
expect                “username:”
timeout               success
end_script
```

The following sample chat script illustrates the use of the reserved keywords *user_name* and *port_password*:

```
begin_script          script1
send                  “\r\r”, user_name, “\r”
expect                “password:”, continue
timeout               error
send                  port_password, “\r”
expect                “successful”, success
timeout               error
end_script
```

Creating %wan Entries in the Configuration File

Refer to *Configuring the WAN Interfaces, Global Ports, and Sessions on page 5-1* for complete information on creating **%wan** entries.

Automated Firmware Download (AFD)

The RAC uses automated firmware download (AFD) to obtain the correct firmware it requires for operation. The RAC downloads the version of the firmware that is appropriate to the switch type and hardware platform in use.

AFD is enabled or disabled for up to two WAN modules by entries in the **%gateway** section of the **config.annex** file. AFD is enabled by default; therefore, no entry in the **%gateway** section is necessary. AFD executes in normal download mode by default.

AFD has two modes of operation: normal download and never download. When normal download is enabled, AFD will attempt to download if the current revision of the firmware is not up to date or if it is inappropriate for the switch type in use. Also, AFD will attempt to download firmware if a WAN module is marked as failed by diagnostics, regardless of what mode is specified. AFD will not attempt any download in never download mode.

AFD does not download firmware for the RAC's digital modems. Digital modems are configured according to the settings in the **%digital_modem** section of the RAC's configuration file. (Refer to *Customizing Modem Configuration on page 7-6* for detailed instructions.)

Creating %gateway Entries for AFD

Settings for AFD are included in the **%gateway** section of the configuration file. If no entries are present in the **%gateway** section, the default AFD behavior is normal download mode.

In the event that two or more entries in the **%gateway** section indicate conflicting or overlapping instructions, the latest entry has precedence. For example, the following example results in WAN 1 enabled in normal mode and WAN 2 disabled:

```
%gateway
download pri never /*prevent download of both wan modules */
download wan 1 /* enable download of wan module 1 */
```

Normal Download Mode

Normal download mode downloads firmware if it determines that the current firmware revision is outdated, or the current firmware does not support the switch type in use. Any of the entries that follow can be used to enable AFD in normal mode. Note that **download pri** and **download wan** perform the same function; **download pri** has been retained to ensure backward compatibility with earlier releases of the RAC software.

```
%gateway
download pri /* download both wan modules */
download wan /* download both wan modules */
download wan1 /* download wan module 1 */
download wan2 /* download wan module 2 */
```

Never Download Mode

Never download mode disables AFD, preventing it from downloading firmware. Any of the entries that follow can be used to disable AFD. Note that **download pri never** and **download wan never** perform the same function; **download pri never** has been retained to ensure backward compatibility with earlier releases of the software.

```
%gateway
download pri never /* prevent download of both wan modules */
download wan never /*prevent download of both wan modules */
download wan1 never /*prevent download of wan module #1 */
download wan2 never /* prevent download of wan module #2 */
```

Console Port Status Messages

Various status messages may be displayed in the console window during AFD.

Unsolicited Status

The following messages are displayed in the console if AFD downloads firmware to any valid device:

```
Downloading firmware, may take up to 5 mins, type afd for
status
Downloading firmware completed
```

Solicited Status

Typing the console port command **afd** displays the status of AFD. Following is the list of messages which may be displayed, depending on the status of AFD:

- afd not started yet:
  ```
  AFD not invoked:
  ```
- afd started, no download in progress:
  ```
  AFD executing:
  view syslog for status, version strings or
  error(s)
  ```
- download in progress:
  ```
  AFD executing:
  LOADING internal wan module(s) firmware
  view syslog for status, version strings or
  error(s)
  ```
- afd done with no download attempted:
  ```
  AFD completed:
  view syslog for status, version strings or
  error(s)
  ```
- afd done with download completed:
  ```
  AFD completed:
  ABORTED loading internal wan module(s) firmware
  ```
  or
  ```
  SUCCESS loading internal wan module(s) firmware
  view syslog for status, version strings or
  error(s)
  ```

CLI stats Command

Entering the CLI command **stats** displays the AFD status along with other informational status messages. The following message is displayed if AFD is in progress:

```
*****DOWNLOADING WAN MODULE(S)*****
```

Display of this AFD message ends upon completion of AFD.

LEDs

Each WAN module has an LED associated with it; the LEDs indicate when the RAC is downloading firmware to a WAN module.

The WAN Port Usage LEDs are active for the associated WAN module. The pattern that is generated is based on the size of the firmware. Each LED represents 25% of the firmware that is being downloaded (the LED blinks wile the firware is downloading). When that percentage of the firmware has been downloaded, the LED remains on and the next LED begins blinking.

Syslog Messages

Syslog messages are generated when AFD executes. If AFD detects the correct firmware version, that version is reported. If AFD downloads firmware, a downloading message is reported. Once AFD completes downloading, either the version or an error is reported. Messages are displayed with the “INFO” priority level, except for error messages, which have the “ERROR” priority level. Following are some of the valid syslog messages:

```
Sep  5 10:13:59 cyclone loader[1311]:
AFD downloading current WAN MODULE firmware on ctl2

Sep  5 10:13:59 cyclone loader[1312]:
AFD downloading current WAN MODULE firmware on ctl3

Sep  5 10:13:59 cyclone loader[1311]:
AFD force downloading WAN MODULE firmware on ctl2

Sep  5 10:13:59 cyclone loader[1312]:
AFD force downloading WAN MODULE firmware on ctl3

Sep  5 10:12:55 cyclone loader[1311]:
AFD WAN MODULE queried #00:VER=VERSION A MGR=1.120 on ctl2

Sep  5 10:12:55 cyclone loader[1312]:
AFD WAN MODULE queried #00:VER=VERSION A MGR=1.120 on ctl3
```

Error Handling

If AFD aborts because of an error, an error message is generated for syslog, snmp trap, and console port via AFD commands. If the error occurred during the actual download, all WAN Port Usage LEDs will blink for the associated WAN module and the WAN module will be marked as failed, rendering it inoperable. The box will not panic if both modules fail during the download. If AFD aborts before downloading firmware, the WAN module will be left in its present state. Following is a list of all possible syslog error messages (the same messages can apply to the ctl3 port):

```
Sep 5 10:13:59 cyclone loader[1311]:
AFD WAN MODULE on ctl2 could not create process, Device in use

Sep  5 10:13:59 cyclone loader[1311]:
AFD WAN MODULE on ctl2 could not open file <file_name>

Sep  5 10:13:59 cyclone loader[1311]:
AFD WAN MODULE on ctl2 has invalid combination of ATTYPE &
switch

Sep  5 10:13:59 cyclone loader[1311]:
AFD WAN MODULE on ctl2 has invalid switch type

Sep  5 10:13:59 cyclone loader[1311]:
AFD WAN MODULE on ctl2 can't load cas image on pri module

Sep  5 10:13:59 cyclone loader[1311]:
AFD WAN MODULE on ctl2 failed wan port not in loader mode

Sep  5 10:13:59 cyclone loader[1311]: /* fatal snmp trap sent
*/
AFD WAN MODULE on ctl2 responded with unrecognized ATTYPE

Sep  5 10:13:59 cyclone loader[1311]: /* fatal snmp trap sent
*/
AFD WAN MODULE on ctl2 failed download, unknown state
```

Chapter 4
Configuring Hosts and Servers

This chapter describes host-related configuration requirements, including the following topics:

- Accessing 4.2BSD hosts
- The IP addresses for the RAC
- Event logging
- Setting up the file server
- Setting up an RAC as a boot server
- Self-booting without a local Ethernet interface
- Dump host services
- Installing a time server
- Setting up name servers
- RAC security
- Configuring LAT services

Accessing 4.2BSD Hosts

The 4.2BSD version of the **rlogin** protocol allows logins only from hosts whose names and IP addresses are listed in the host's **/etc/hosts** file. The 4.3BSD version of the protocol does not impose this restriction.

Add the RAC to the **/etc/hosts** file on each 4.2BSD host. Add the new entry near the beginning of the file because UNIX software searches this file sequentially.

RAC Internet Addressing

The RAC uses IP addressing to communicate with hosts on the network. Internet support requires an IP address, a broadcast address, and a subnet mask.

The Internet (IP) Address

The RAC's IP address is defined in the **inet_addr** parameter. This address must be set prior to downloading the operational code to the RAC. To set the IP address, use the ROM monitor **addr** command during the RAC's initial installation. You can reset the address at any time thereafter by editing the **inet_addr** parameter.

The CLI, **na**, and ROM Monitor commands always display the IP address in dotted decimal notation.

The Broadcast Address

The broadcast address defines the IP address the RAC uses to broadcast. The RAC broadcasts requests when it has not received a response from a server, such as file server or security server. The **broadcast_addr** parameter defines this address.

The Subnet Mask

If the network is divided into subnets, you must specify the RAC's IP subnet mask using the **subnet_mask** parameter. If you do not define the subnet mask, the RAC assigns one based on the network part of its IP address. Set this parameter using the ROM monitor **addr** command during the RAC's initial installation. You can reset the address at any time by editing the RAC **subnet_mask** parameter.

Certain combinations of the RAC subnet mask and IP address have special meaning:

- Setting the RAC IP address to 0.0.0.0 or 255.255.255.255 turns off all IP services, including SLIP, PPP, and IP routing. The RAC continues to support non-IP services, such as ARAP and LAT, provided that they are configured properly.
- Setting the RAC IP address to a valid value and RAC **subnet_mask** to 255.255.255.255 installs IP but specifies that the RAC does not have an Ethernet connection. IP services, including SLIP, PPP, and IP routing, are still available.
- By default, the RAC acts as an authoritative agent for ICMP Address Mask Requests. If another host broadcasts this message querying for the subnet mask, the RAC replies with the subnet mask. Optionally, you can prevent the RAC from responding by setting the **authoritative_agent** parameter to **N**.

Using Event Logging

The RAC can log events to a 4.3BSD system log daemon (**syslogd**) on the system you specify via the **syslog_host** parameter. The RAC can log events to a 4.2BSD system using the **syslog** daemon or to a System V if it has system logging similar to 4.3BSD syslogging.

Refer to the *Remote Access Concentrator Server Tools for Windows NT® User Guide* for information about syslogging in a Windows NT environment.

The 4.3BSD system logging daemon provides a *facility* as an addition to the *selector* field. The selector field is a list of priorities for a message and includes a level, which indicates the severity of a message. The facility defines the part of the system that generates the message. Certain facilities are reserved, such as kernel, mail, and daemons; other facilities can be defined in the configuration file **/etc/syslog.conf**. Facilities allow you to selectively log messages by priority.

If the **syslog_host** does not have a **syslog** daemon, or if you do not specify a **syslog_host**, the RAC logs events to the RAC console.

When configuring the host and the RAC for system logging, consider the following parameters: **syslog_host**, **syslog_facility**, and **syslog_mask**. (Reboot the RAC after configuring any parameters related to system logging.)

- The **syslog_host** parameter defines the IP address of the host configured to log RAC messages. The default, **0.0.0.0**, causes the RAC to broadcast its log messages.
- The **syslog_facility** parameter defines the facility used in the syslog messages (specified as **log_local***n* where *n* is a number from 0 through 7). The default is **log_local7**.

  f the host to which messages are logged does not support 4.3BSD syslogging, this parameter is ignored and messages are logged only by priority level as defined by the **syslog_mask** parameter.
- The **syslog_mask** parameter defines the priority levels for logging messages. The options are **all**, **none**, or a combination of levels. The default, **none**, disables logging. Table 4-1 describes the levels in order of priority.

  

  When defining a priority level, all messages of that level or greater (i.e., of greater severity) are forwarded to **syslogd**. For example, selecting **error** logs all **error**, **critical**, **alert**, and **emergency** messages.

Table 4-1. Priority Levels for the syslog_mask Parameter

Level	Description
emergency	Hardware failures.
alert	All RAC reboots.
critical	Configuration and initialization problems, such as format errors in the **gateway** section of 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	Starting and ending of CLIs and of RAC jobs created by the **rlogin** and **telnet** commands and the **ping** and **tap** superuser CLI commands.
debug	Activation and exit of all RAC processes.

Setting Up the File Server

The RAC can boot via the block file server (**bfs**) program using **erpcd** or via **tftp**.

Installing Software Using bfs

Setting up a file server for a **bfs** installation involves loading, compiling, and installing RAC source code on the host. This process has four stages:

- Loading the software from the media into a directory
- Running the **install-annex** script
- Editing the **/etc/services** file
- Starting **erpcd**

bfs booting is automatically enabled when you install Remote Annex Server Tools for Windows NT®. The script and file editing steps are not required.

When the installation is complete, by default, the image and configuration files are located in the directory **/usr/spool/erpcd/bfs**; the utilities are located in the directory specified during the **install-annex** sequence (the default is **/usr/annex**).

Installing Software Using the tftp Protocol

Setting up a file server for a **tftp** installation involves loading the RAC software onto the host. This process has three stages:

- Loading the software from CD-ROM into a directory
- Copying **bfs/oper.*xx*.enet** (where ***xx*** represents the software variable for your RAC) to your **tftp** directory
- Creating the configuration file in your **tftp** directory

Multiple Server Hosts

To install file server software on multiple hosts, repeat the installation procedure on each host that will be a file and/or a security server. If you are defining multiple security servers, the contents of the **acp_passwd**, **acp_keys**, and **acp_restrict** files must be identical on all security servers.

Booting and Dumping

The RAC obtains its operational code by downloading it over the network from a UNIX host running RAC file server software, a non-UNIX host running **tftp**, another RAC configured as a boot server (running the same operational code), or the local media (self-boot). The RAC boots each time it is powered up and upon receipt of a **boot** command.

The RAC can dump to a file server or a host running **tftp**. The RAC performs a dump upon receipt of either the **na** command **dumpboot** or the superuser CLI **boot -d** command, or automatically when it detects fatal internal errors or failures.

Setting the Preferred Load Host

The **pref_load_addr** parameter specifies the preferred load (or file server) host. This is the host from which the RAC first requests a download of its operational code. If this parameter is not defined or the specified host is not available, the RAC broadcasts its boot request and loads operational code from the first host that responds. You can modify the **pref_load_addr** parameter using **na** or the **admin** command; specify the host by its IP address or its name.

The **image_name** parameter specifies the name of the image file that contains the RAC's operational code. This file resides in different host directories, depending on which transfer protocol (**tftp** or **erpcd**) is used.

If the load host has a different network or subnet address, you must define a gateway through which the RAC can reach the host. The **load_dump_gateway** parameter specifies the IP address for the gateway.

During the initial boot of the operational code, the ROM monitor requires the address of a gateway if the specified load host is on another network or has a different subnet address. In this case, enter the gateway's address using the ROM monitor **addr** command. The RAC automatically adds this gateway to its routing table.

Dump Host Services

The RAC can dump its memory image to a dump host on demand through either the superuser CLI command **boot -d** or the **na** command **dumpboot** or on certain software or hardware events. A non-recoverable hardware or software error triggers RAC dumps. Dump files are intended for use by technical support personnel only.

The host to which a RAC sends a dump must have **bfs** or **tftp** capability. You can define a preferred dump host to which the RAC first tries to upload a dump file. If this address is not specified or the host is not available, the RAC broadcasts a request and dumps to the first host that responds.

Self-boot units without a network host cannot perform a dump.

At the dump host, the dump creates a file, between one and four megabytes in size, in the directory **/usr/spool/erpcd/bfs** for **bfs** dumping (in the **tftp** directory for **tftp** dumping), and assigns a unique dump file name to each RAC. The assigned name depends on whether the dump host can support file names longer than 14 characters.

On hosts that support file names longer than 14 characters (for example, BSD UNIX hosts), dump files are named **dump.***xxx.xxx.xxx.xxx*. The file extensions *xxx.xxx.xxx.xxx* are the RAC's IP address.

On hosts that limit file names to 14 characters (for example, System V hosts), the dump creates two additional directories under **/usr/spool/erpcd/bfs**. The name of the first directory is **dump**; the second is the RAC's IP network address. Subnet addresses are not used in naming the dump file. The name of the dump file is the RAC's IP host address.

If the **pref_dump_addr** parameter is not set and your IP address is corrupted (zero), leave the **Test** button on the RAC's front panel *on* and the unit will prompt for an IP address when it tries to dump.

Table 4-2 provides sample dump file names; all pathnames are relative to **/usr/spool/erpcd/bfs**.

Table 4-2. Dump File Naming Conventions

RAC Address	Network Address	BSD Filename	System V Pathname
63.0.0.75	63	dump.63.0.0.75	dump/63/0.0.75
131.14.23.1	131.14	dump.131.14.23.1	dump/131.14/23.1
195.46.2.15	195.46.2	dump.195.46.2.15	dump/195.46.2/15

Setting the Preferred Dump Host

The **pref_dump_addr** allows you to specify the preferred host to which the RAC performs a dump. If this parameter is not defined or the specified host is not available, the RAC broadcasts its dump request and dumps to the first host that responds.

The dump creates a file that is between one and three megabytes in size. If using **erpcd**, the RAC assigns the dump file a unique name and places it in a directory named **/usr/spool/erpcd/bfs**. If using **tftp**, the file name is defined by the **tftp_dump_name** parameter and file placement is user-defined. If the dump host has a different network or subnet address, you must define a gateway through which the RAC can reach the host. The **load_dump_gateway** parameter specifies the IP address for the gateway.

Setting the Load-Dump Sequence

The configuration parameter **load_dump_sequence** specifies the network interface and the order to be used for a download or an up-line dump. The arguments are **net** (for use with a LAN), and **self** (to boot from the local media). For more details, see the *Remote Access Concentrator Software Reference*.

Setting a RAC as a Load Server

The **server_capability** parameter defines the RAC as a file server host. A RAC can provide operational code only for another RAC of the same type. When a RAC boots, it uses the **image** file to load the operational code, and the configuration file to initialize the routing table, rotaries, and macros. The RAC normally does not store these files because they use memory. As a file server host, the RAC uses approximately 120 KB for the operational code; for the message-of-the-day (**motd**) and configuration files, it uses the amount of space relative to the size of the files.

The **server_capability** parameter defines the files that the server supplies during a boot. Table 4-3 describes the arguments for **server_capability**; the default is **none**.

If you configure a RAC to supply only a copy of the operational code, the default is for the RACs booting from it to broadcast for the configuration and **motd** files. The file server RAC uses **erpcd** to serve other RACs.

Table 4-3. Arguments for the server_capability Parameter

Argument	Description
all	The RAC is a file server for the configuration, operational image, and message-of-the-day files.
config	The configuration files.
image	The operational code.
motd	The message-of-the-day file.
none	The RAC is not a file server.

Disabling Broadcasting for Files During a Boot

During a boot, the RAC broadcasts for the configuration, **image**, and **motd** files if they are not available on the preferred load host. You can disable broadcasting for these files by setting the **load_broadcast** parameter to **N**.

Self-Booting

The self-boot option loads and boots the operational image from local non-volatile memory. To store the image on the local media, issue the **boot -l** command from **na**, the superuser CLI, or the ROM monitor.

Only ROM revisions 0601 and greater with the self-boot option installed support the **boot -l** command.

After executing a **boot -l** command, the **ls** command may not show the newly loaded image.

To boot the stored (local) image, set the configuration parameter **load_dump_sequence** (or the ROM monitor parameter **sequence**) to **self** and reboot the RAC. This sequence instructs the RAC to load the operational image and the configuration file from the local media.

To boot from both the local media and the network, set **load_dump_sequence** to either **self**,**net**, or **self**. The RAC will first load the files from the local media; whatever files it cannot find there it will seek from the network.

Self-Booting Without a Local Ethernet Interface

When booting a RAC from the Flash ROM without a local Ethernet interface:

1. **Enter the ROM Monitor prompt from the console.**
2. **Set the IP address to a valid IP address and the subnet mask to a valid mask using the** addr **command.**
3. **Set the interface sequence to self using the** sequence **command.**
4. **Boot the RAC.**

Using the Trivial File Transfer Protocol

The Trivial File Transfer Protocol (**tftp**) is a standard network interface loading program. The RAC operational code opens and reads the operational image, configuration, and **motd** files. The RAC accesses one file at a time.

The RAC initially tries to open a file using **erpcd** (except when using the self-boot option). If **erpcd** fails or times out, the RAC tries to open a file using **tftp**. If the **tftp** request fails or times out, the RAC retries opening the file using **erpcd**. This cycle continues until the RAC succeeds in opening the file or until it reaches the maximum try count (currently eight cycles). If the **load_broadcast** parameter is enabled and the RAC cannot open a file from the **pref_load_host**, it broadcasts the open request (this is true for both **erpcd** and **tftp**). Once a file is successfully opened, the RAC continues to read it using the protocol with which it was opened.

The protocol used to transfer one file is independent of the protocol used to transfer another file. For environments that support both **erpcd** and **tftp**, the RAC may use **tftp** to transfer one file and **erpcd** to transfer another file.

Using the RAC ftp Daemon

Using the RAC ftp daemon, you can upload or download files (those visible through the superuser CLI **ls** and **edit** commands) in the RAC's non-volatile memory (EEPROM) from a remote host.

The RAC ftp daemon is primarily useful for saving RAC configuration files to a host on the network for the purpose of "swapping" RACs.

You cannot "get" or "put" boot images using the RAC ftp daemon..

Using a Time Server

The RAC maintains a UNIX-style time-of-day clock, which is based on the Internet date and time server. The RAC distribution includes source code for a time server in case one is not available on the preferred load host. The RAC synchronizes its clock by requesting the time from a time server.

The time server expresses time in the number of seconds since midnight (00:00:00), January 1, 1970, Greenwich Mean Time (GMT). The RAC converts time server time to local time and uses it to log events to **syslog** and to calculate the time of a boot and/or dump. The CLI **stats** and **who** commands display this time; the local file system **ls** command displays the time the files were last modified.

The RAC requests the time when it boots and synchronizes its clock with a server every 30 minutes. It always queries the preferred load host first if one is defined. If a time server does not respond, the RAC displays *unknown* in place of the time.

By default, if a time server is not available on the preferred load host, the RAC does not broadcast for the time. However, you can enable broadcasting for a time server by setting the **time_broadcast** parameter to **Y**. Most UNIX systems provide a time server with the **inetd** daemon.

Remote Access Concentrator Server Tools for Windows NT ships with its own time server.

DNS does not require the **timserver** program.

Every host on the network that has a timer server will respond to a broadcast for the time.

The RAC does not reset its time by more than 10 minutes based on an answer to a broadcast request. If the time returned to the broadcast query is greater than 10 minutes from the RAC's current time, the RAC only resets its time by a maximum of 10 minutes. If the timer server is on the preferred load host, the RAC adjusts to the time reported by the time server, regardless of the time interval.

The **timezone_minuteswest** parameter defines the time zone in which the RAC resides. Enter a positive number of minutes for time zones west of GMT and a negative number for time zones east of GMT. For example, since U.S. Eastern Standard Time is five hours west of GMT, its value is 300 minutes; since Paris is one hour east of GMT, its value is -60 minutes.

The **daylight_savings** parameter defines the daylight savings time to which your geographic area adheres. The RAC uses this parameter to adjust the time display for daylight savings time. Valid arguments include: **us, australian**, **british**, **canadian**, **east_european**, **mid_european**, **west_european**, and **none**.

Installing a Time Server

If a host time server is not present, use the supplied **timserver** program:

1. **Add the following line to** /etc/services **if not already included:**

   ```
   time 37/udp timserver
   ```

2. **Start the server:**

   ```
   # /etc/timserver
   ```

3. **Edit the appropriate** rc **file so that the time server starts automatically when the system is booted.**

Using Name Servers

Name servers allow users to enter names in place of addresses in order to access a host or other entity on the network. The RAC supports two standard types of name servers: a Domain Name System (DNS) server and an IEN-116 server. In addition, the RAC can use RWHO broadcast messages to provide name-to-IP address translation. You can configure the RAC to use one of these, a combination, or none.

The RAC supports the minimum uniqueness feature when entering host names. This feature allows users to enter the host name with a minimal string that is unique enough to identify that host from any other in the host table. If this feature is not enabled, the user must enter the complete name to access a host. Host name to Internet address translation entries can be downloaded to the RAC from the **%gateway** section of the configuration file. The format is the same as in the **/etc/hosts** file, but aliasing is not permitted. To set up a RAC for use with a name server:

- Specify the name server type
- Specify the host(s) using the name server
- Enable or disable the **rwhod** parameter
- Specify the host table size
- Enable or disable the **min_unique_hostnames** parameter

Defining Name Servers

The RAC supports two standard name server protocols: the Domain Name System (DNS) server and the IEN-116 server. Both name server protocols are available in the UNIX environment. You can use one or both on the network, and the RAC allows you to specify the preferred protocol. If you choose not to use either protocol, you can configure the RAC to build the host table by listening to RWHO broadcasts.

Domain Name System

Domain Name System (DNS) servers use a distributed database to maintain host names and IP addresses for network hosts. DNS provides a full range of capabilities that enable its use in very large networks, such as the Internet.

Each DNS server is responsible for maintaining information on all hosts in its domain. If the server receives a request for a host that is not in its domain, the server retrieves the information from another domain server for the requesting host.

A number of DNS servers are available and the RAC can support them all. One typical DNS server is the Berkeley Internet Name Domain (BIND) server. The BIND server is a standard part of 4.3BSD (see 4.3BSD documentation for more details). DNS provides:

- Address-to-name translation
- Multiple aliases for a host
- Multiple addresses for the same host

Address-to-name translation allows a host to obtain a name for a specific Internet address, allowing a RAC to learn its name from a DNS server. The DNS' capabilities for assigning multiple aliases or multiple IP addresses to a single host allow you to assign multiple names to a rotary or multiple RACs to the same rotary (for more details, see *The Port Server on page 5-38*).

Adding a Resource Record For a RAC

The following discussion on adding a resource record for a RAC on a domain name server is specific to the Berkeley Internet Name Domain (BIND) server. If your network uses an alternate domain name server, see the documentation for that server.

For a RAC to obtain its name, you must include a PTR resource record for the RAC in the server's domain data files, specifically the IN-ADDR.ARPA domain. This record must contain the RAC's fully qualified domain name (FQDN), so the RAC can use part of the full domain name to expand host names to full domain names. The FQDN must always be supplied with a query to a DNS server; otherwise, the RAC adds one.

The following example shows a PTR resource record in a BIND name server for the RAC *annex01* with an IP address of 132.245.6.34 and a full domain name of *annex01.eng.Widget.COM*:

```
34.6.245.132.IN-ADDR.ARPA. IN  PTR annex01.eng.Widget.COM.
```

After the RAC boots, it queries the name server for 34.6.245.132.IN-ADDR.ARPA. The name server returns the RAC's full domain name. The RAC uses the part of the name up to the first period as its name, and stores the rest as the default domain name. The RAC uses the default domain name to expand other host names when it queries the server for their IP addresses.

For example, to obtain an IP address for the name *wayland*, the RAC sends a query to the name server for *wayland.eng.Widget.COM*. If the name server does not have this name, the RAC then queries for *wayland.Widget.COM*. If this query also fails, the RAC queries for *wayland*. If you do not want the host name to be expanded with the default domain name, append a period to the host name. For example:

```
annex: rlogin wayland.
```

IEN-116 Name Server

The IEN-116 name server is a simple host-resident name server that uses the local **/etc/hosts** file as a database. One host is designated as the name server host, and other hosts query that host for an address. Using this method, not every host on the network needs its own up-to-date **/etc/hosts** file, and not every host needs to run **rwhod**. The RAC distribution medium supplies the source for IEN-116 (see *Configuring Hosts and Servers on page 4-1* for installation instructions).

IEN-116 name servers cannot do reverse address queries.

If an IEN-116 name server is not available on the network, the source code for a server is provided in the file */annex_root*/**src/ien-116**.

To install the server:

1. **Compile the source if necessary.**
2. **Examine** /etc/services **and add the following line (if necessary):**

   ```
   name    42/udp   nameserver   #IEN-116
   ```

3. **Start the name server by entering:**

   ```
   # /etc/ien116d
   ```

4. **Edit the appropriate** rc **file so that the name server starts automatically when the system is booted.**

Remote Access Concentrator Server Tools for Windows NT® does not support the IEN-116 name server.

To verify that the server responds to queries from the RAC, configure the RAC to use an IEN-116 server (see *Using Name Servers on page 4-15*). Look at the RAC host table. Then look at the name server host's **/etc/hosts** file and select a host that does not appear in the RAC host table. Using the CLI **hosts** command, force the RAC to query the name server (see the *Remote Access Concentrator Software Reference Guide*).

Setting Configuration Parameters For Name Servers

The **name_server_1** parameter defines the type of name service that the primary name server will supply when queried by the RAC. Valid service types are **dns**, **ien_116**, or **none**; the default is **none.** You specify the IP address of the primary name server by setting the **pref_name1_addr** parameter.

The **name_server_2** parameter defines the type of name service that the secondary name server will supply when queried by the RAC. Valid service types are **dns**, **ien_116**, or **none**; the default is **none**. You specify the IP address of the secondary name server by setting the **pref_name2_addr**. This server is queried only if **pref_name1** server does not respond.

Broadcasting for a Name Server

By default, the RAC does not broadcast for a name server if the preferred name servers do not respond. However, you can configure the RAC to broadcast requests for a name server by setting the **nameserver_broadcast** parameter to **Y**. This causes the RAC to broadcast three requests for a Domain Name Server, followed by three requests for an IEN-116 name server. You may want to use broadcast as a backup for a name server.

Using the RWHO Protocol

Berkeley UNIX hosts use the RWHO protocol to pass information about themselves to other hosts. This information includes the host's name, who is logged in, up time, and load factor. The RWHO daemon, **rwhod**, broadcasts this information and listens for RWHO messages from other hosts, storing what it receives in a file. The information can be displayed with the **rwho** and **ruptime** commands from a UNIX host.

The RAC uses the RWHO protocol as a name server. The RAC runs an **rwhod** that listens for broadcasts from other hosts, but does not broadcast information about itself. When the RAC receives an RWHO message, it stores the host name, status information, and the source address from the IP header as the host's Internet address in its host table.

Using only RWHO messages to build the host table is satisfactory for a small network in which all the hosts run **rwhod**. But, **rwhod** often is not used in networks primarily comprised of workstations because of the load it imposes on hosts. In large or heavily loaded networks, RWHO broadcasts can impose an excessive load on the network.

Some hosts send RWHO packets with incomplete source addresses in the IP header. The RAC is unable to store an IP address for these hosts, causing the host table to display the host's IP address as "_._._._".

If an **rwhod** forwards packets from one network to another, the IP address in the IP header is that of the forwarding host, not of the host whose name is in the data packet. This results in the RAC storing the wrong IP address for that host.

Because the RAC does not broadcast RWHO messages, RAC names never appear in host tables built exclusively from these broadcasts. In this case, the only way to access a RAC using the **telnet** command is with an IP address.

The **rwho** parameter defines whether or not the RAC listens for RWHO broadcasts. Setting the parameter to **N** disables the RAC's **rwhod** and prevents the RAC from using RWHO messages for building the host table. The default is **Y**.

Managing the Size of the Host Table

When the host table acquires a new entry after it is full, the RAC deletes the oldest, least-used entry to make room for the new one. The RAC's use of the host table is erratic if the table size is too small. Increasing the size of the table reduces this problem.

You modify the host table size using the **host_table_size** parameter. This parameter specifies the number of entries in the host table. You can specify the size as a number from **1** to **250**. Specifying the string **"unlimited"** sets no limit other than the size of memory available in the RAC. Alternatively, you can set the size to **none**, which forces the RAC to query the name server for each host name.

Minimum Uniqueness

Minimum uniqueness provides an ease-of-use feature which allows users to enter only the characters necessary to uniquely match an entry in the host table. However, users can force the RAC to select only an exactly matching host name by enclosing the name they enter in double quotes. For example:

```
annex: rlogin "widget"
```

If the host table contains the name *widgetslips*, and you want to log in to a host named *widget*, which is not in the host table, then entering *widget* without the quotes causes the RAC to select *widgetslips*. Entering the name enclosed in double quotes forces the RAC to query a name server, because an exactly matching name is not in the host table. The minimum uniqueness feature can be turned off by setting the **min_unique_hostnames** parameter to **N**.

Configuring Name Servers

The RAC uses various means of creating and maintaining the host table. This table includes the host names and the corresponding IP addresses of hosts known to the RAC. The host table is generated by querying a name server and/or listening for broadcasts from RWHO daemons running on other hosts.

The RAC adds entries to the host table when it receives:

- RWHO broadcast messages from other hosts.
- Responses from the Domain Name System (DNS) server and/or the IEN-116 name server to a query for an IP address.

If broadcasting is enabled, the RAC first makes three attempts for a DNS server followed by three attempts for an IEN-116 server.

RAC parameters allow you to configure a RAC to listen only for RWHO broadcasts or to query one or both types of name servers or use both means of building a host table (see *Using Name Servers on page 4-15*).

By default, the RAC builds the host table exclusively from RWHO broadcasts. Depending on what is available for your network, you can use a name server in conjunction with RWHO broadcasts or disable the use of RWHO in the RAC.

If the network is using a domain name server, you must add a resource record for each RAC to the domain server. If the network does not have any name servers, the RAC distribution provides a source for an IEN-116 server that you can install.

To determine if a specific system is running a name server, use the UNIX **netstat** command with the **-a** argument or the **-a** and **-n** arguments. An IEN-116 name server is displayed as listening on UDP port **name** (or **42**). A BIND server is displayed as listening on UDP or TCP port **domain** (or **53**).

Using RAC Security

The RAC provides a security system that allows you to implement as many security measures as the network requires. You can set up the security subsystem to use host-based security, local password protection, or a combination of the two. In addition to these security mechanisms, the RAC provides an administrative password that validates access through the administrative tools.

If unauthorized users can access your RAC, Bay Networks strongly suggests that you enable the security features after loading the host code and booting the unit.

For a detailed description of RAC security, see *Configuring Security on page 6-1*.

Installing the ACE/Server Software

If you are using the RAC's SecurID feature, you must install the ACE/Server software before installing the RAC software.

After compiling the ACE/Server installation, the file **sdconf.c** contains the network addresses for the ACE/Server host. See *Using the SecurID Card on page 6-128* and the *ACE/Server Manual* for more details.

Configuring LAT Services

The RAC can display, and connect to, currently available LAT services. Initially, all LAT functions in the RAC are disabled: the user does not have access to the **connect** and **services** CLI commands, and the administrator does not have access to the LAT-specific RAC parameters.

To enable the LAT functions:

1. **Enter the correct** lat_key **parameter value.**
2. **Configure the** disabled_modules **parameter not to include lat.**
3. **Reboot the RAC.**

The **lat_key** value is unique for each RAC. Since this value varies by port count, if the administrator changes the number of ports on the RAC, the **lat_key** must change and the RAC must be rebooted (contact your supplier to obtain a **lat_key** value).

When the administrator selects a RAC via **na** or **admin** that has LAT enabled, the RAC returns the normal configuration line with the string *W/LAT* included:

```
command: annex zippy
zippy: Remote Annex w/LAT Rx.x, 72 ports
```

Advertised Services

Advertised services are the announcements of resources (modems, for example) that LAT machines have available for use by the network clients. A service announcement carries the name and Ethernet address of the server offering the service, along with a current service rating. Every host that accepts communication sessions is a service provider. All service providers broadcast service announcements periodically (typically, once per minute).

A host can provide multiple services. When a user broadcasts a service request and there are multiple providers of that service, the RAC logs the user onto the host with the highest service rating. Typically, four factors account for a given service rating: 1) the most recent CPU idle time, 2) the CPU type, 3) the amount of memory, and 4) the number of available interactive slots.

The **%service** entries in the configuration file define the LAT services that the RAC advertises (see *Creating %gateway Entries in the Configuration File on page 3-19* for more details).

Learned Services

Learned services are the services that a LAT machine hears and stores from the network. All services received by a LAT machine are filtered based on the group codes in the service announcement and the access group code of the LAT receiving machine.

Each LAT machine has a group code mask that represents the allowable groups for users of that machine. To store a service announcement, there must be at least one group code common between the advertising machine and the receiving machine.

Group Codes

Service providers can be assigned a LAT group code. Group codes partition the LAT network logically into subsets. The RAC restricts clients to the assigned group code(s). Group codes can enhance both network security and network management.

Accessing LAT Services

The administrator must enable a set of the RAC's group codes that correspond to the site requirements. The RAC's group code is a security access mechanism designed to allow selective restriction of LAT services on the network. There are 256 group codes (0-255). Each group code is either enabled or disabled.

Each LAT service has an associated set of group codes. The users on a RAC will have access to a LAT service only if the service and the RAC have at least one enabled group code in common. For example, if the desired LAT service has group codes 1 and 3 enabled, the RAC must have either group code 1 or group code 3 enabled to access the service. If the RAC has only group code 0 enabled, the RAC users will not have access to the service. The RAC maintains information only for the services to which its users have access; the **services** command displays only the services to which RAC users have access.

Restricting Access to LAT Services

The **group_value** parameter specifies which remote group codes can access the local services offered by a particular RAC. To change the status of the RAC's group codes, the administrator must change the **group_value** parameter.

The **group_value** parameter displays the group codes that are enabled; initially, the value is *none*. Allowable values are 0 through 255. The administrator must determine which LAT services to enable, and set the group codes accordingly. The syntax is:

set annex group_value *group_range* **enable** | **disable**

The *group_range* must be integers (0-255) separated by either a comma, to indicate multiple group codes, or a hyphen, to indicate a range of group codes; spaces are not permitted as delimiters. For example, **set annex group_value** *0-10, 15, 20, 240* **enable** enables group codes 0 through 10, and group codes 15, 20, and 240. For convenience, specifying *all* indicates all group codes. Thus, **set annex group_value** *all* **enable** enables group codes 0 through 255. The administrator can disable group codes in the same way using **disable** instead of **enable**.

After LAT is enabled, the appropriate group codes are enabled, and the LAT parameters have been reset (using the **reset annex lat** command), the **services** command may show no services available for approximately 0-60 seconds. This occurs because LAT hosts broadcast the LAT services they offer periodically, and the RAC does not update its services table until it receives this broadcast.

Accessing LAT from a Virtual CLI

The **vcli_groups** parameter specifies which remote group codes are accessible to virtual CLI users. All virtual CLI users have the same group code. The syntax is:

set annex vcli_groups *group_range* **enable** | **disable**

Accessing LAT from a RAC Port

The **authorized_groups** parameter specifies which remote group codes are accessible to users on a particular RAC port. Each port has its own **authorized_groups** setting. The syntax is:

set port authorized_groups *group_range* **enable** | **disable**

Reverse LAT

The RAC advertises the services specified in the **service** section of the configuration file. Optionally, the RAC queues host-initiated connection (HIC) requests if the requested resource is not available, i.e., the port is in use.

Reverse LAT vcli

To enable the **vcli** service in the RAC, LAT must be configured for the network on which it is to run. Also, the **max_vcli** parameter must be enabled. As long as these conditions are met, the RAC can advertise the **vcli** service to the network.

The advertised service's *services* and *host* fields are set to the value of the **server_name** parameter. The RAC dynamically updates the *rating* field of the service advertisement based on the number of **vcli**s left. If **max_vcli** is set to unlimited, the rating is 255. The **max_vcli** parameter is the total number of vclis allowed in the RAC.

When a LAT user connects to the RAC **vcli** service, it is the same as a **telnet vcli**. If all **vcli**s are in use by Telnet and LAT users, the connection request is rejected.

Telnet-to-LAT Gateway

The Telnet-to-LAT gateway enables the system administrator to associate a unique IP address with a specific LAT service. This gateway allows a user to **telnet** to that unique IP address thereby connecting to the associated LAT service.

Before configuring this feature, configure your RAC as described in *RAC Internet Addressing on page 4-2*. LAT is operating properly on your RAC if you see your LAT network services appear when you execute the CLI **services** command and you can connect to them using the CLI **connect** command.

To set up the Telnet-to-LAT gateway, the system administrator must add a new entry to the **%gateway** section of the configuration file. The syntax for this new entry is:

annex *ip_addr*
    translate telnet *ip_addr* **to lat** *service_name* [**telnet_groups** *group_range*]
        [*host_name*[*port_type*]]
end

The *ip_addr* on the *annex-selector* line refers to the specific RAC offering the *gated* LAT service. The *ip_addr* on the *translate* line is the IP address that translates to a LAT service. Both *ip_addr* fields are specified in the standard dotted decimal notation.

The *group_range* indicates which remote group codes are accessible for this gateway. The *group_range* is specified as integers (0--255) separated by either a comma or a hyphen. The keyword **telnet_groups** is optional (it allows backward compatibility for users who already have Telnet-to-LAT gateways specified). If **telnet_groups** is not included, all groups are accessible (i.e., 0-255).

The *service_name* is the desired LAT service to which **telnet** connects; the *host_name* is the host that advertises the LAT service; and the *port_type* is the port type on the LAT host providing the LAT service. The *host_name* and *port_type* are optional. The *service_name*, *host_name*, and *port_type* can be a maximum of 16 characters. Any errors in the syntax are reported in the **syslog** file.

There must be an *annex-selector* line for each set of *translate* line(s) and the *ip_addr* on the *translate* line must be unique on the network and may appear only once in the **%gateway** section of the configuration file. Violating this rule may cause your network to go down or operate erratically (violation is analogous to defining multiple IP hosts with the same IP address).

Each time a translation entry in **%gateway** changes, the RAC specified in the *annex-selector* line must be rebooted. For example:

```
annex 192.9.200.245
     translate telnet 192.9.200.100 to lat modems node-a
     translate telnet 192.9.200.101 to lat modems
     translate telnet 192.9.200.102 to lat accting
end
```

The previous translations are defined only for the RAC with IP address 192.9.200.245. In the first translation, the IP address 192.9.200.100 connects to the LAT service **modems** on the host *node-a*. In the second translation, the IP address 192.9.200.101 connects to the LAT service **modems** on the host reporting the largest rating for **modems**. In the third translation, the IP address 192.9.200.102 connects to a LAT service called **accting** on the host reporting the largest rating for **accting**.

Figure 4-1 illustrates the RAC TCP/IP gateway.

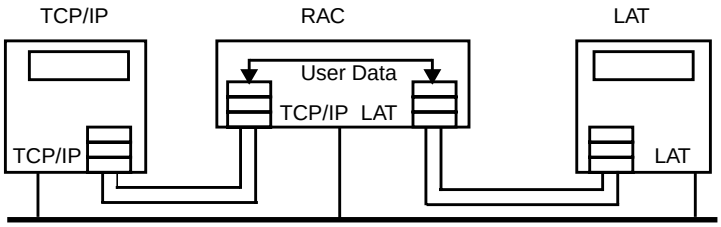

Figure 4-1. RAC TCP/IP Gateway

LAT-to-Telnet Gateway

The LAT-to-Telnet gateway allows a RAC to translate an advertised LAT service to a **telnet** connection for a specific IP address. This allows a LAT user to move transparently between the LAT, TCP/IP, and Telnet protocols.

To set up the LAT-to-Telnet gateway, the system administrator must add a translation directive to the **%gateway** section of the configuration file. This new entry has the following syntax:

annex *ip_address*
translate lat *service_name* [**identification** *id_string*] **to telnet**
ip_address [port_type]
end

For example, a translation entry for a UNIX machine named *frodo* running TCP/IP with the IP address 132.0.0.50 and a VMS host running LAT would look like this:

```
annex 132.0.0.35
     translate lat frodo identification 'Login service for
TCP/IP
     host 'frodo' to telnet 132.0.0.50
end
```

After making the above translation entry in **%gateway**, tyou must make sure that the RAC is running LAT and that the **group_value** parameter is configured so that the LAT host can use this service. Also, the RAC must be rebooted.

Using the above sample entry, once the RAC is rebooted, the LAT user sees an entry for *frodo* in the *services* display and the *host* field corresponds to the *service_name* set in the RAC 132.0.0.35. When the LAT user connects to *frodo*, the connection is made through the RAC 132.0.0.35.

Since LAT allows multiple hosts to offer the same service name, many RACs can offer a LAT-to-Telnet translation to the same TCP/IP host. The hosts that have access to the service will load-balance between RACs, based on the service ratings each RAC advertises for the service.

A **%service** entry is not required in the RAC configuration file.

Data-b Slot Support for LAT

The RAC LAT implementation now reports and responds to data-b slot messages. This feature is enabled on a per port basis using the **latb_enable** parameter; the default setting is **N**.

When a connection is established with a LAT host, the RAC sends that host a report of the port parameters via a report data-b slot message. If the RAC receives a set data-b slot message from a connected LAT host, it responds by configuring the port as commanded by the set data-b slot message.

Parameters changed via data-b slot messages are parity, baud rate, bits per character, and inband flow control.

Status via data-b slot messages supports the break signal at the local port. To get the RAC to forward this status to the host, disable the local break interpretation.

If the **latb_enable** parameter 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 terminal's cursor keys.

Miscellaneous LAT Parameters

The LAT-specific **na** parameters can be changed, but are not necessary to access LAT services; only the **group_value**, **vcli_groups**, and **authorized_groups** parameters are necessary for such access. Since the timer and limit values affect performance, take care when adjusting them. For convenience, tyou may want to change the **server_name** since this is the name by which other LAT hosts will refer to the RAC. After changing the appropriate LAT parameters, you must issue the **na** command **reset annex lat** to activate the new parameters.

For more details on using **na** and CLI commands, see the *Remote Access Concentrator Software Reference*.

Chapter 5
Configuring the WAN Interfaces, Global Ports, and Sessions

This chapter describes:

- Understanding call delivery
- Using the default call configuration
- Setting WAN interface parameters
- Understanding internal port handling
- Using global port parameters
- Configuring Session Parameter Blocks (SPBs)
- Setting the **mode** parameter
- Configuring Command Line Interpreter (CLI) sessions
- Understanding port differences
- Rotaries
- Dial-out networking

Understanding Call Delivery

A PRI call can arrive on any B channel on a RAC PRI line. During the call SETUP process between the telco switch and the RAC, the switch dynamically assigns a B channel to the call.

CAS (Channelized T1 or E1) calls are typically delivered in a similar fashion. The telco dynamically assigns a DS0 to each call, and the call can arrive on any DS0. However, unlike PRI switches, most CAS switches can be configured to deliver certain calls on specific DS0s.

Since the RAC cannot usually predict which B or DS0 channel the switch assigns to a particular call, providing the RAC with a static configuration for each B/DS0 channel is not often useful. Instead, the RAC relies on global WAN and port parameters (call defaults), and (optionally) Session Parameter Blocks (SPBs) to dynamically obtain parameter values for calls received.

Using the Default Call Configuration

When delivered to you, the RAC is configured to detect automatically the type of call - TA (V.120, V.110, or X.75), synchronous PPP, or modem - arriving on an ISDN PRI B channel or a CAS DS0 channel on either WAN interface. Any calls not recognized within a (modifiable) 5-second time-out period are treated as modem calls. This automatic detection feature allows you to keep operating costs low by purchasing a single dial-in number for all your remote access users.

Once the call type is detected, calls are handled as follows:

- TA and modem calls are placed in protocol-detection mode and directed accordingly to a PPP, ARAP, or terminal emulation (CLI) process. Because of its inherent lack of security, SLIP cannot be detected; it must be started by issuing the **slip** command at the CLI prompt.
- Synchronous PPP calls are directed to a PPP process.

For the RAC to be operational with this default configuration, the only requirements are that you set the switch type and any other WAN interface parameter whose factory defaults do not match the service options provided by the telco for your ISDN PRI or CAS line(s). See *Setting WAN Interface Parameters*, next.

Setting WAN Interface Parameters

There are two types of global WAN interface parameters:

- Generic parameters, such as the switch type, that apply to one or both WAN interface(s).
- B/DS0 channel parameters that apply to the channels on one or both WAN interface(s).

The factory defaults are identical for both WAN interfaces.

Setting Generic WAN Interface Parameters

To set generic WAN interface parameters:

1. **Use the CLI superuser** admin **command on the RAC itself, or use the** na **command on the UNIX host from which the RAC image was loaded. The examples in this section use** admin**.**
2. **Using** na **or** admin**, issue the** wan **command to specify the WAN interface for which you wish to reset parameter values (for example,** set wan=all**). All parameters set will apply to this WAN interface until you specify another.**
3. **Still using** na **or** admin**, use the** set wan **command to set parameter values. If you are setting only one parameter, you can specify the WAN interface to which it applies as part of the** set wan **command, instead of pre-defining the interface in Step 2.**
4. **Using na or admin, issue the** na **or** admin reset wan **command, or reboot the RAC.**

For complete information on **admin** and **na**, see the *Remote Access Concentrator Software Reference*.

You can also set SNMP variables to specify the generic WAN interface. See the *Remote Access Concentrator SNMP MIB Reference*.

Setting the Switch Type

The switch is the physical device, located at the telco's Central Office, that provides the RAC with PRI or CAS data transmission service. By default, the switch type is set to "", the null string. You must reset this value to the switch type(s) the telco is using for your WAN interfaces. *The RAC cannot operate properly until you define the type of switch in use*.

To specify the switch type, set the **switch_type** parameter. Table 5-1 shows the valid **switch_type** values for PRI. Table 5-2 shows the valid CAS **switch_type** values.

To set the switch type using **admin**:

1. **Invoke** admin**:**

```
annex: su
Password:
annex# admin
ANNEX-PRI I14.0.17, 48 async, 64 sync, 64 ta, 48 internal modem ports
admin : 
```

2. **Use the** set wan **command to specify the switch type and the interface(s) to which it applies. For example, enter:**

```
admin : set wan=1 switch_type DMS
        You may need to reset the appropriate port, Annex subsystem or
        reboot the Annex for changes to take effect.
admin : 
```

This sets the switch type to DMS for WAN interface 1. To specify WAN 2, enter **wan=2**; for both WANs, enter **wan=all**.

3. **Check to make sure that the switch type is set in nonvolatile memory:**

```
admin : show wan=1 switch_type

WAN Interface 1:
        _switch_type:*"DMS"
admin : 
```

The RAC uses asterisks to indicate parameters that have been changed from their defaults via the **admin** command.

4. **Quit** admin **and reboot the RAC from CLI superuser level:**

```
admin : q
annex# boot
boot file name:
warning:

        *** Annex (132.245.44.98) shutdown message from
        Annex (132.245.44.98) going down IMMEDIATELY
Connection closed by foreign host.
```

This causes the Automatic Firmware Download (AFD) software to download the image appropriate for the switch type. For more information on AFD, see *Automated Firmware Download (AFD) on page 3-52*.

(Rebooting is only required the first time you set the switch type. Thereafter, issuing the admin **reset wan** command is sufficient.)

5. **When the unit comes back up, use the CLI superuser** wan **command to check the actual switch type setting, not just the value saved in nonvolatile memory. This command queries the WAN modules for the information it displays.**

The following is a partial **wan** command display. For complete information on this command, see the *Remote Access Concentrator Software Reference*.

```
annex# wan
General WAN Statistics              Interface #1
-----------------------------------------------------
WAN Firmware Vers:                  A MGR=1.204
WAN Type:                           T1 CSU
WAN FDL Type:                       ATT
Switch Type:                        DMS
Analog Encoding:                    mu_law
WAN Interface Errors:               0
Accepted Incoming Calls:            3
Rejected Incoming Calls:            0
Accepted Outgoing Calls:            0
Rejected Outgoing Calls:            0
Normal Call Disconnects:            3
Abnormal Call Disconnects:          0
B Channels Currently Allocated:     0
Number Times WAN Fully Allocated:   0
```

Table 5-1. Valid PRI switch_type Values

Protocol	switch_type	Description
T1/PRI		Used in North America
	AT9	AT&T 5ESS#9 switch
	AT4	AT&T 4ESS support
	DMS	Nortel's DMS100 switch
	NI2	A switch supporting National ISDN2
E1/PRI	ETS	Used in Europe; ETSI
	ETS-NCRC4	Used in Europe; ETSI without CRC
	AU1	Used in Australia

Table 5-2. Valid CAS switch_type Values

Protocol	switch_type	Description
Channelized T1	UST1	Used in North America
	HKT1	Used in Hong Kong
Channelized T1 - R1	TWT1R1	Used in Taiwan
Channelized E1 - P7	SWE1P7	Used in Sweden

(continued on next page)

Table 5-2 Valid CAS switch_type Values

Protocol	switch_type	Description
Channelized E1 - R2	BBE1R2	CCITT BlueBookR2; used in most of Europe
	KRE1R2	Used in Korea
	BRE1R2	Used in Brazil
	IDE1R2	Used in Indonesia
	NZE1R2	Used in New Zealand
	PHE1R2	Used in the Philippines
	MYE1R2	Used in Malaysia
	THE1R2	Used in Thailand
	MXE1R2	Used in Mexico
	CNE1R2	Used in China
	ANE1R2	Used in St. Martin
	TRE1R2	Used in Turkey
	ILE1R2	Used in Israel

Additional Generic WAN Parameters

In addition to **switch_type**, there are several other generic WAN parameters you may need to set:

1. **To display all generic WAN parameter defaults using** admin**, issue the** show wan **command:**

CAS interfaces use all of the parameters shown, except **num_b_channels.** PRI interfaces, which are shown inside the dotted lines in the example, use only a subset of the CAS parameters. See Table 5-3 for explanations of all generic **wan** parameters.

2. **Specify the interface(s) for which you are setting parameters. For example, enter:**

```
admin : wan 1
admin :
```

This selects WAN interface 1.

3. **Set the parameters. For example, enter:**

```
admin : set wan framing d4
        You may need to reset the appropriate port, Annex subsystem or
        reboot the Annex for changes to take effect.
admin : set wan line_code hdb3
        You may need to reset the appropriate port, Annex subsystem or
        reboot the Annex for changes to take effect.
admin :
```

   This sets the **framing** parameter to **d4** and the **line_code** parameter to **hdb3** for WAN 1.

4. **Reset the WAN interface to save the changes; this is sufficient for resetting all WAN parameters except the** switch_type**, which requires a reboot the first time you set it.**

```
admin : reset wan
reset default wan interface set [y] ? y
admin :
```

Table 5-3. Parameters For the set wan Command

Parameter	Description
switch_type	A string specifying the type of switch provided by the telco for your WAN line. The default is "", a blank string. See Table 5-1 and Table 5-2.
dsx1_line_length	Does not apply to the RAC.
fdl_type	The Facilities Data Link format supported by the telco for the RAC.
num_b_channels	(PRI interfaces only) 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, **30** for E1/PRI connections. Valid values are **1**-**23** for T1/PRI, **1**-**30** for E1/PRI.

(continued on next page)

Table 5-3 Parameters For the set wan Command (continued)

Parameter	Description
buildout	A string defining the CSU transceiver line provided by the telco. Valid values are: **0db** (the default) **7.5db** **15db** **22.5db**
analog_encoding	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 being used, issue the superuser CLI **wan** command.
framing	(CAS interfaces only) The superframe format for which the WAN interface is provisioned. Valid values for Channelized T1 are: **esf** (extended superframe; the default) **d4** (superframe) Valid values for Channelized E1 are: **ddf** **mff_crc4** **mff_crc4_g706**

(continued on next page)

Table 5-3 Parameters for the set wan Command (continued)

Parameter	Description
line_code	(CAS interfaces only) The line code for which the WAN interface is provisioned. Valid values are: **b8zs** (the default) **ami** (Channelized T1 or E1) **hdb3** (Channelized E1 only)
dnis	(CAS interfaces only) Dialed Number Identification Service, which allows the called number on a given CAS line to be visible to the RAC. This permits you, for example, to map the dialed number to a particular service. To do this, enter the number in the **called_no** field of an SPB. See *Understanding WAN Sessions on page 5-24*. For the **dnis** argument, specify the number of DNIS digits that the telco is providing for the CAS interface. The default is **0**.
ani	(R2 CAS interfaces only) Automatic Number Identification, which allows the calling number to be visible to the RAC. This permits, for example, verification of a user's security based on his or her home phone number. To do this, enter the number in the **calling_no** field of an SPB that handles security. See *Understanding WAN Sessions on page 5-24*. Specify **y** if the telco is providing ANI, **n** if it is not.

(continued on next page)

Table 5-3 Parameters for the set wan Command (continued)

Parameter	Description
digit_width	(CAS interfaces only; used for debugging) Allows you to adjust the width of each digit, in milliseconds, generated by the RAC. The default is **0**; the maximum value is **255**.
inter_digit	(CAS interfaces only; used for debugging) Allows you to adjust the distance between each digit, in milliseconds, generated by the RAC. The default is **0**; the maximum value is **255**.
digit_power_1	(CAS interfaces only; used for debugging) The power level, in dBm, of the first tone of each digit generated by the RAC. The default is **0**; the maximum is **255**.
digit_power_2	(CAS interfaces only; used for debugging) The power level, in dBm, of the second tone of each digit generated by the RAC. The default is **0**; the maximum value is **255**.
busy_signal_bits	(CAS interfaces only) A string indicating 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** (see Table 5-4 on page 5-15) or use the **wan busyout** command (see the *Remote Access Concentrator Software Reference*). Or, you can arrange to have remaining DS0s busied-out automatically when the last available modem is used (see the **auto_busyout_enable** parameter, below.)

(continued on next page)

Table 5-3 Parameters For the set wan Command (continued)

Parameter	Description
local_phone_number	(CAS interfaces only) The telephone number of the RAC itself. Usually, this number is assigned by the switch. However, for some CAS protocols, such as R2, the number must be specified via the **local_phone_number** parameter.
auto_busyout_enable	(CAS interfaces only) Determines whether or not the RAC busies-out remaining DS0 channels when the last available modem has been used (e.g., because 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.

Channel Parameters

Parameters that apply to specific B or DS0 channels appear in Table 5-4. Use the **show wan b** or **show wan ds0 command** via **admin** or **na** to display these parameters, and **set wan b** or **set wan ds0** to change them. The following example shows the default settings for WAN interface 1. (The defaults are identical for both interfaces.)

```
admin : show wan=1 b

WAN Interface 1:

                        WAN B/DS0 Channel Parameters

      remote_address: 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
                      0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
                      0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
                      0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
                      0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
                      0.0.0.0 0.0.0.0 0.0.0.0
         ipx_network: 00000000 00000000 00000000 00000000
                      00000000 00000000 00000000 00000000
                      00000000 00000000 00000000 00000000
                      00000000 00000000 00000000 00000000
                      00000000 00000000 00000000 00000000
                      00000000 00000000 00000000
            ipx_node: 00-00-00-00-00-00 00-00-00-00-00-00 00-00-00-00-00-00
                      00-00-00-00-00-00 00-00-00-00-00-00 00-00-00-00-00-00
                      00-00-00-00-00-00 00-00-00-00-00-00 00-00-00-00-00-00
                      00-00-00-00-00-00 00-00-00-00-00-00 00-00-00-00-00-00
                      00-00-00-00-00-00 00-00-00-00-00-00 00-00-00-00-00-00
                      00-00-00-00-00-00 00-00-00-00-00-00 00-00-00-00-00-00
                      00-00-00-00-00-00 00-00-00-00-00-00 00-00-00-00-00-00
                      00-00-00-00-00-00 00-00-00-00-00-00
            sigproto: none none none none none none
                      none none none none none none
                      none none none none none none
                      none none none none none
           ringback: y y y y y y y y y y y y y y y y y y y y y y y y
admin : []
```

Table 5-4. Channel Parameters

Parameter	Description
remote_address	The IP address(es) to be associated with one or more B or DS0 channels. This allows you to automatically generate unique IP addresses for remote PPP or SLIP links. The syntax is one of the following: **set wan b**[**=***channel_range* | **all**] **remote_address** *ip_addr* [*increment*] or **set wan ds0**[**=***channel_range*|**all**] **remote_address** *ip_addr* [*increment*] An example follows this table. For complete information, see the *Remote Access Concentrator Software Reference*.
ipx_network	The IPX network address(es) to be associated with one or more B or DS0 channels. This allows you to generate unique network addresses for the remote ends of IPXCP links. The syntax is either: **set wan b**[**=***channel_range* | **all**] **ipx_network** *net_number* [*increment*] or **set wan ds0**[=*channel_range* | **all**] **ipx_network** *net_number* [*increment*] For complete information, see the *Remote Access Concentrator Software Reference*.
ipx_node	The IPX node numbers to be associated with one or more B or DS0 channels. This allows you to generate unique remote IPXCP node numbers. The syntax is either: **set wan b**[**=***channel_range* | **all**] *ipx_network net_number* [*increment*] or **set wan ds0**[**=***channel_range* | **all**] *ipx_network net_number* [*increment*] For complete information, see the *Remote Access Concentrator Software Reference*.

(continued on next page)

Table 5-4 Channel Parameters (continued)

Parameter	Description
sigproto	(CAS channels with switch type UST1) 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**.
ringback	(CAS channels with switch type UST1 or ETS) 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.**

Examples

The following are examples of the **set wan b** and **set wan ds0** commands:

- Example 1 sets remote addresses for the B channels. It assigns the address 132.254.22.1 to the first B channel and increments that address by 1 for each of the remaining B channels.
- Example 2 sets the signaling protocol to **wink_in** for all DS0 channels.

Example 1

```
admin : set wan=1 b remote_address 132.254.22.1 1
        You may need to reset the appropriate port, Annex subsystem or
        reboot the Annex for changes to take effect.
admin : reset wan=1
```

Example 2

Understanding Internal Port Handling

The RAC uses numbered, internal (logical) ports to manage the types of data that ISDN and CAS can carry on a B/DS0 channel. These numbered ports have no static association with the B or DS0 channels on which data arrives. They appear in displays of certain statistics but cannot be set by the administrator.

Internal Port Types

The RAC uses three internal port types:

- **ta** ports manage V.120, V.110, and X.75 calls.
- **syn** ports manage synchronous PPP calls.
- **asy** ports manage asynchronous voice calls. If no modems are installed, no internal asynchronous ports are supported.

Internal Port Numbering

Internal ports of types **ta** and **syn** are dynamically numbered within a given range, starting at 1, for each data type. The numbers are assigned in a circular fashion across the two WAN interfaces. When the maximum number (usually 64 for **ta** ports and more than 64 for **syn** ports) in the range has been assigned, the next incoming call is given the first free port number, beginning at 1.

The range of numbers for **asy** ports depends on the number of modems installed - 48 or 64. Assignment of modems is load balanced across the two WAN interfaces and modem numbering is random within the given range.

An internal port name consists of the port type and number, such as **ta1**, **syn10**, **asy4**.

Using Global Port Parameters

The RAC comes with global port parameters set to factory defaults. The RAC applies these parameters to the appropriate data type (**ta**, **asy**, **syn**) while a call is active for that data type on either WAN interface (see *Using the Default Call Configuration on page 5-2*.)

You can override global port parameter defaults by using the **set port** command via **na** or **admin**. Subsequently, you can reset to factory defaults any values you change.

Override only those global port parameter defaults that:

- Differ from your requirements.
- Apply to all WAN calls of a particular data type. To specify a parameter value that applies only to a subset of these calls, define that subset in a Session Parameter Block (SPB) and include the parameter settings in that SPB. Parameter values set in an SPB override global parameters set by **na** or **admin** (which in turn override the supplied defaults); see *Understanding WAN Sessions on page 5-24*.

Displaying Global Port Parameters

To display a global port parameter value, use the following syntax:

show port [**parameter_name** | *keyword*]

Table 5-5 describes the keywords for the **show port** command. Sample displays of **show port all** and **show port syn** follow the table. These displays show the factory defaults.

Table 5-5. Keywords for the show port Command

Keyword	Description
all	Displays all global port parameter values.
appletalk	Displays the global AppleTalk parameters.
editing	Displays the global CLI line editing parameters used with terminal emulation.
flow	Displays the global flow control parameters.
generic	Displays the basic global parameters.
ipx	Displays the global IPX parameters.
lat	Displays the global LAT parameters.
ppp	Displays the global PPP parameters.
security	Displays the global security parameters.
serial	Displays the global serial parameters.
slip	Displays the global SLIP parameters.
syn	Displays the global synchronous PPP parameters.
timer	Displays the global timer parameters.
tn3270	Displays the global tn3270 parameters.
vci	Displays the global VMS command interface parameters.

The following example uses the **admin** command **show port all** to display all asynchronous and TA global port parameters.

```
admin : show port all

global port:

                        Port Generic Parameters

                mode:*auto_detect                  location: ""
            term_var: ""                             prompt: ""
       cli_interface: uci                          data_bits: 8
           stop_bits: 1                              parity: none
   max_session_count: 3                    allow_broadcast: Y
broadcast_direction: port                       imask_7bits: N
          cli_imask7: Y                             banner: Y
       tcp_keepalive: 0                default_session_mode: interactive
dedicated_arguments: ""                    resolve_protocol: connect

                        Flow Control and Signal Parameters

  input_flow_control: eia                    input_start_char: ^Q
     input_stop_char: ^S                  output_flow_control: eia
    output_start_char: ^Q                    output_stop_char: ^S
  ixany_flow_control: N                             need_dsr: N
            v120_mru: 256

                        Port Timers and Counters

    forwarding_timer: off                    forwarding_count: 0
      cli_inactivity: off                    inactivity_timer: off
   input_is_activity: Y                    output_is_activity: N
 reset_idle_time_on: input                         long_break: Y
        short_break: Y                    autodetect_timeout: 30

                        Port Security Parameters

          user_name: ""                          cli_security: N
   connect_security: N                   port_server_security: N
      port_password: "<unset>"                    ipso_class: none
       ipx_security: N

                        "Login" User Parameters

 login_port_password: N                          login_timeout: N
```

(continued on next page)

The following example uses the **admin** command **show port syn** to display the synchronous PPP global port parameters.

```
admin : show port syn

global port:
           user_name: ""                        port_password: "<unset>
 ppp_username_remote: ""                  ppp_password_remote: "<unset>
   slip_ppp_security: N                 ppp_security_protocol: none
             ppp_ncp: all                              metric: 1
         subnet_mask: 0.0.0.0                         ppp_mru: 1500
    inactivity_timer: off                   input_is_activity: Y
  output_is_activity: N                    reset_idle_time_on: input
      net_inactivity: off                net_inactivity_units: minutes
             mp_mrru: 1500                  mp_endpoint_class: mac
 mp_endpoint_address: ""
admin :
```

Changing Global Port Parameter Values

Setting a Global Port Parameter

To change the value of a global port parameter, use the **set port** command via **admin** or **na**. The following example uses **na**:

1. **Log into a UNIX host and enter** na**:**

   ```
   % na
   Annex Network Administrator R14.0
   command:
   ```

2. **Specify one or more RACs:**

   ```
   command:annex 132.245.6.40 or
           annex 132.245.6.40,hobbes
           password:
   ```

3. **Set the global port parameter whose value you wish to change.**

   The following example sets the **allow_broadcast** parameter to **N** (the default is **Y**):

   ```
   command:set port allow_broadcast N
   ```

4. **Enter the** show port allow_broadcast **command to make sure the value is now set to** N**.**

The changed value is automatically saved in nonvolatile memory and remains set for all subsequent WAN calls and across re-boots of the RAC(s). Currently active calls are not affected.

Setting Multiple Port Parameters

To configure multiple global port parameters on multiple RACs:

1. **Define the global port parameters.**
2. **Use the** copy port **command to copy the parameters to other RAC sessions.**

The following example copies the parameter setting for **allow_broadcast** from one RAC to another RAC:

```
command: annex 132.245.6.40
command: set port allow_broadcast N
command: copy port 1@132.245.6.40 1@132.245.6.55
```

Setting All Parameters

You can also define all parameters, including global port parameters, for one RAC. Use the **write** command to create a script file on the specified UNIX host containing all the configuration data for that RAC. Finally, execute the **read** command for all RACs you want to configure.

The **write** command does not write passwords.

For descriptions of the parameters you can set, see the *Remote Access Concentrator Software Reference*.

Overriding Global Port Parameter Values

The same group of port parameters that you can set globally can also be set for a particular call or subset of calls. To do this, you include the parameter definition in a Session Parameter Block (SPB) that handles those calls. Parameter values set in SPBs override the global settings.

Understanding WAN Sessions

An ISDN PRI or CAS session begins when the RAC accepts a call and ends when the call terminates. By default, the RAC accepts all calls, automatically detecting TA and sync PPP calls and treating all others as modem calls (see *Using the Default Call Configuration on page 5-2*). To modify how the RAC handles calls, define one or more SPBs.

SPBs provide you with the flexibility to allow only synchronous PPP calls, to invoke different security options for TA calls than for other call types, or to negotiate different modem rates for different called numbers.

Configuring Session Parameter Blocks

You define SPBs in the **%wan** section of the configuration file on the host you use to download RAC software. By default, the file is named **config.annex** and is located in the **/usr/spool/erpcd/bfs** directory.

If you have a configuration file from an older model, such as the Remote Annex 5393, the software does not overwrite it. In this case, the SPB information is in a section labeled **%pri**.

SPB Sections

Each SPB in the **%wan** or **%pri** section must be defined within **begin_session** and **end_session** fields. The **begin_session** field lets you name an SPB within the configuration file.

An SPB has three sections:

- A section presenting call setup criteria. If the SETUP message that starts an incoming call meets *all* of these criteria, or if no criteria are specified, the call is handled by this SPB.
- A call handling section that manages all calls meeting the SPB setup criteria. This section specifies an action to be taken for the calls.
- A section containing per-session port parameter settings. All global port parameters can be overridden (as appropriate for the specific type of call) in this SPB section. If you do not need to change the value of a global port parameter, do not specify it here - i.e, if no parameter values are special to this session, you can omit this section of the SPB.

How SPBs Are Scanned

When it receives a call, the RAC tries to match the SETUP information elements of the call with setup criteria values defined in the SPBs. The RAC searches SPBs in the order that they appear in the configuration file, so the sequence in which you specify SPBs is important. You should order your SPBs from the most specific to the most generic.

When it scans the SPBs, the RAC uses the first SPB whose setup criteria are met by the incoming call. All criteria in an SPB must be met by the SETUP information elements in order for the RAC to consider the SPB a match.

Once the RAC finds a matching SPB setup criteria section for a particular call, it:

- Handles the call as specified in the call-handling section.
- Uses the per-session port parameter settings to form the dynamic parameter values that will be applied to the call.

If no SPBs are defined, or no matching SPBs are found, the RAC handles a call as described in *Using the Default Call Configuration on page 5-2*.

SPB Fields

Use the following format when entering an SPB into the configuration file. Table 5-6 describes all possible SPB fields. Unless otherwise noted, each field is optional.

```
# this is a comment line

begin_session                   <session_name>
calling_no                      <phone number>
called_no                       <phone number>
called_subaddress               <number>
bearer                          <voice or data>
detected                        <detection keyword>
call_action                     <action>
max_number_of_calls             <integer>
acp_log                         <yes or no>
rate56k                         <yes or no>
set                             <parameter_name setting>
end_session
```

Table 5-6. SPB Field Definitions

Field	Definition
begin_session	(Mandatory) Marks the beginning of an SPB and names it. The session name is an alphanumeric string of up to 12 characters. (The RAC will accept longer strings, but 12 characters is the recommended limit.) You can use this string with the CLI superuser **sessions** command to display an SPB.
calling_no	Specifies the telephone number that identifies the origin of the ISDN call. *Specify the entire number, including the area code, even if it would not normally be required to make the call.* You can use a dash to separate the area code from the rest of the phone number, or you can include the area code in parentheses. No wild card symbols (*) are permitted and white space is ignored. If this field is omitted, any calling number is permitted. For CAS calls, use the number provided by the Automatic Number Identification (ANI) service, if you have that service. Sometimes, the calling number is not available in the SETUP information, either because the telco did not have the equipment to deliver it or because the caller keeps his or her number private. If **calling_no** is specified, but no number is contained in the SETUP information, the SPB is not a match.

(continued on next page)

Table 5-6 SPB Field Definitions (continued)

Field	Definition
called_no	Specifies the number the user entered to dial into the RAC. *Specify the entire number, including the area code, even if it would not normally be required.* You can use a dash to separate the area code from the rest of the phone number, or you can include the area code in parentheses. No wild cards (*) are permitted. White space is ignored. For CAS calls, specify the number provided by the Dialed Number Identification Service (DNIS), if you have that service. Note: The ACP log file shows the called number delivered by the switch (for PRI protocols) or by the ANI service (for CAS protocols). The log file may only contain the final digits of the number. If this field is omitted, any called number matches this SPB.
called_subaddress	This field is appropriate only for end-to-end calls using an ISDN PRI line that the telco has provisioned for subaddressing.
bearer	Specifies the bearer capability of the call. Valid values are **voice** and **data**.

(continued on next page)

Table 5-6 SPB Field Definitions (continued)

Field	Definition
detected *keyword*	Specifies a keyword indicating how to handle calls detected as a result of a **call_action** field set to **detect** in another SPB. Permissible *keyword* values are: **valid**, which matches when either TA or synchronous PPP calls are detected. **modem**, which matches when neither TA nor synchronous PPP calls are detected during the *timeout* period defined in a **call action detect** field in a separate SPB. **v120**, which matches when TA calls (V.120 over HDLC) are detected. **sync_ppp**, which matches when synchronous PPP calls (PPP LCP Configure-Request over HDLC) are detected. **56**, which matches all calls when the line speed is 56 Kbps. **64**, which matches when the line speed is 64 Kbps. **any**, which matches any call, whether or not a **call_action** field is set to **detect** in another SPB. This is the default. You can combine keywords to produce a less-restrictive SPB than one containing a single keyword. For example, specifying **v120,modem** matches both V.120 and modem calls. To create a more restrictive scenario than you can in one SPB, use separate SPBs. For example, if you specify **v120** in one SPB, then **56** in a second SPB and **sync_ppp** in a third, V.120 calls will be handled by the first SPB, 56Kbps PPP calls will be handled by the second SPB, and 64Kbps will be handled by the third SPB.

(continued on next page)

Table 5-6 SPB Field Definitions (continued)

Field	Definition
call_action *keyword*	Defines how to handle the call. This field is mandatory, unless a **detect** action is already in effect for this call. Valid values for *keyword* are: **detect** [*timeout*], which attempts to recognize V.120 or synchronous PPP frames in the raw digital data delivered by the telco. If neither frame type is recognized in the number of seconds specified for *timeout*, which defaults to 5 seconds, the connection type is assumed to be analog modem. After detection, the SPBs are searched again for the presence of a **detected** field entry that matches the detected frame type and indicates how to handle the call. Do not use the **rate56k** or **set** fields (see below) in an SPB containing a **call_action** of **detect**. **reject**, which rejects the call. **modem**, which handles the call as a modem call. **v120**, which handles the call as a V.120 call. **sync**, which handles the call as a synchronous PPP call. The default is **detect**, which means that all calls are accepted; it is impossible to reject a call beyond this point.
max_number_of_ calls	Defines the maximum number of calls that this session handles simultaneously. Valid PRI values are **1** through **23** (in the U.S.) and **1** through **30** (in Europe). Valid CAS values are **1** through **24** (for Channelized T1) and **1** through **30** (for Channelized E1, R1, and R2). The default is the upper limit (23, 24, 30).
acp_log	Specifies whether or not the RAC forwards a call's SETUP information elements and status to the ACP log file. Valid values are **yes** and **no** (the default). Status is logged as *call accept*, *call reject*, or *call disc* (disconnect).

(continued on next page)

Table 5-6 SPB Field Definitions (continued)

Field	Definition
ds0	(CAS only; optional) Specifies a WAN interface, or a DS0 channel, or a WAN and a DS0 channel. The syntax is: **ds0** [*wan channel_number*] | [*channel_number*] For *wan*, specify **a** or **A** (for WAN 1) or **b** or **B** (for WAN 2). Specify *channel_number* as a digit from **1** through **24**. This feature allows you to apply an SPB to a particular WAN interface and/or DS0 channel. It is especially useful for CAS connections that do not support DNIS or ANI. For this feature to work, the telco must provide one or more phone numbers that terminate on specific DS0s. Users to whom you wish the SPB to apply must call these phone numbers.
rate56k	If set to **yes**, specifies a data rate of 56 Kbps for the B or DS0 channels, even if the bearer information in the incoming ISDN SETUP messages indicates a different rate. The default is **no**, which sets the data rate to the rate provided in the SETUP message. *Do not change this default unless you are in Europe or Australia and are having problems receiving calls from the U.S. In this situation, the telco sometimes fails to specify the correct data rate. In all other situations, use* **no**.
set	Specifies a port parameter setting that will be applied to the session. The syntax is: **set** [*parameter parameter_value*] You can specify multiple **set** commands. These settings will override the values in nonvolatile memory while the session is active, but they will not change the actual values in nonvolatile memory. Any parameters not specified in *set* fields are determined by the actual global (nonvolatile memory) settings.
end_session	Mandatory; ends the SPB.

SPB Configuration Procedure

Keeping the SPB format and field definitions in mind, perform the following steps to configure SPBs:

1. **Log into the load host and open the configuration file.**

   The configuration file is located in the install directory on the load host (the default file is **/usr/spool/erpcd/bfs/config.annex**). Use any system editor (e.g., **vi**, **textedit**) to open the file. For example, enter:

   ```
   cd /usr/spool/erpcd/bfs
   vi config.annex
   ```

2. **Using the editor, locate the** %wan **section of the file. Following a few lines of introduction, you should see sample SPBs. Read the explanatory text in the file to understand how these SPBs operate.**
3. **If you want to use any of the sample SPBs, remove the comment character (#) from the beginning of each line (except actual comment lines) in the samples.**
4. **Add any SPBs you need.**
5. **Save the file.**
6. **Activate your changes.**

   To activate changes, issue the following **na** command:

   ```
   command: reset annex session
   ```

   This causes the RAC to re-read the Session Parameter Blocks from the configuration file. Existing calls are not reset. No new calls are answered while the reset is in progress.

Sample SPBs

The example that follows shows three SPBs, which assume the following global port parameter settings:

- port **mode** is set to **auto_detect**
- **slip_ppp_security** is set to **y**
- **ppp_security_protocol** is set to **pap**
- **ppp_ncp** is set to **ipcp,atcp,ipxcp**

- **ppp_sec_auto** is set to **n**
- **cli_security** is set to **y**

The SPBs are:

```
begin_session do_detect
call_action detect
end_session

begin_session sync_users
detected sync_ppp
set ppp_ncp ipcp,atcp,ipxcp,mp
set mp_mrru 1500
set ppp_security_protocol chap
end_session

begin_session other_users
detected any
bearer data
end_session
```

The first SPB turns on automatic detection of the connection protocol. This causes the RAC to search the SPBs again, from the top, for an SPB containing a *detected* field. The second SPB contains this field and defines capabilities for synchronous PPP users. These users are allowed to use Multilink PPP but must be authenticated via CHAP. The third SPB places all other (TA and modem) users in **auto_detect** mode; they must use PAP for security and are not allowed to use Multilink PPP.

Setting the Mode Parameter

A parameter you often set in an SPB is the **mode** parameter, which dictates the protocol mode in which the session will operate. Valid modes are **arap**, **cli**, **ppp**, **slip**, **connect**, **rlogin**, **telnet**, and **auto_detect**.

If you use the automatic connection-protocol detection feature, the default **mode** is **auto_detect** for TA and modem calls, and **ppp** for synchronous PPP calls. If you do not use automatic protocol detection, the RAC applies the value set by the mode parameter to the calls that match the SPB containing the parameter. For information on automatic detection of connection protocols, see the **detect** and **call_action** fields in Table 5-6 on page 5-27, as well as *Using the Default Call Configuration on page 5-2*.

A session set to **arap** mode supports the AppleTalk Remote Access Protocol (ARAP).

A session set to **cli** allows access to the Command Line Interface, which, in turn, provides access to multiple hosts. From the CLI, users can also issue the **slip** or **ppp** command to switch to **slip** or **ppp** mode. By configuring a port this way, you can enforce CLI security before a user's session is placed in one of the two protocol modes.

A session set to **ppp** mode supports the synchronous Point-to-Point Protocol when a synchronous PPP call is detected or the **call_action** SPB field is set to **sync**. A session supports asynchronous PPP when a TA or modem call is detected or the **call_action** SPB field is set to **modem**. A session set to **ppp** also supports IPXCP.

A session set to **slip** mode supports the Serial Line Internet Protocol.

A session defined as **connect** allows the user to communicate with a LAT host via the **connect** command. Use this option in conjunction with the **dedicated_arguments** port parameter.

A session defined as **telnet** communicates via the **telnet** command. Use this option in conjunction with the **dedicated_arguments** port parameter.

For TA and modem calls, a session set to **auto_detect** mode automatically determines the protocol of the incoming data stream and handles the call accordingly. The protocols detected are CLI, PPP, and ARAP.

Although you usually set the **mode** parameter within an SPB, you can also set it globally via **na** or **admin**. For example, if all your users are connecting to the RAC via asynchronous PPP lines, you could issue the **na** command:

```
command: set port mode ppp
```

For information on setting port parameters globally, see *Changing Global Port Parameter Values on page 5-23*.

Configuring CLI Sessions

A RAC CLI session has the global or SPB **mode** parameter set to **cli**. When the session begins, the CLI prompt is displayed. The CLI prompt is **asy**# for modem connections, **ta**# for TA connections, and **pts**# for VCLI (e.g., **telnet**) connections). # is the connection number, starting at 1, for each connection type. For example, the first VCLI user receives the **pts**#1 prompt, and the second user receives **pts**# 2. The default CLI prompt is **annex:**

In a CLI session, the user has access to all permissible CLI commands. You can configure several options for a CLI session. When configuring CLI global or SPB parameters, consider the following:

- To limit the number of connections to other hosts that a user can activate simultaneously, set the **max_session_count**. Setting the value to **1** limits the user to one connection at a time. The default is **3** (with a maximum of **16**).

  

  Do not confuse the sessions (host connections) specified by **max_session_count** with the ISDN sessions configured via SPBs. The two types of sessions are unrelated.

- Set the **allow_broadcast** parameter to **N** if you want to disable the display of administrative messages generated with the **na** command **broadcast**.

- The **user_name** and **location** parameters are used for administrative information. The CLI **who** command displays this information. Also, the **user_name** is passed in the **rlogin** command's connection request. If the user does not have an account on the host under the same user name you defined with the **user_name** parameter, the user must issue the **rlogin -l** command.
- The **term_var** parameter is a string identifying the terminal emulation type. Any value defined for this parameter is passed with both the **telnet** and **rlogin** connection requests. If you define a terminal emulation type, it must be one that is valid for the host to which the user is connecting. The RAC uses this parameter internally for the **edit** command only.
- CLI activity timers provide simple security by resetting idle user connections. Limited resources, like dial-in modems, are released when not in use.

  The **inactivity_timer** specifies the amount of time in minutes that the remote user can be inactive before the RAC resets the connection. When this timer expires, all of the user's CLI connections to hosts are terminated. Allowable values for this parameter are **0** to **255**. The default is **0** (displays as **off**).

  Activity can be set to input (data received from the user), output (data sent to the user), or both. Set the **input_is_activity** parameter to **Y** and/or the **output_is_activity** parameter to **Y**.
- The **cli_inactivity** timer specifies the amount of time in minutes that a CLI connection can remain inactive, with no **telnet**, **rlogin**, or **connect** job running, before the RAC drops the connection. Allowable values for this parameter are **0** to **255**, and **immediate**. The **immediate** setting directs the RAC to drop the connection immediately after the last host connection closes. The default is **0**.

- Pressing an attention key notifies the RAC that the user wants to suspend an ongoing connection to a host and return to the CLI. The RAC provides three parameters for defining attention keys: **short_break**, **long_break**, and **attn_string**.
- Typically, parameters that display with the **show port editing** command define characters that provide CLI line editing functions. Some of these characters are passed as Telnet special characters with CLI-connected devices.

Port Differences for RACs and Remote Annexes

Please note the following differences between ports on the RAC and ports on Remote Annexes:

- You cannot use the **port** command to define one or more ports, although you do use the **set port** command to set parameters globally (across WAN interfaces).
- You cannot set the following generic port parameters:
  - **type_of_modem** (you can set this in an SPB, however)
  - **autobaud**
  - **ps_history_buffer**
  - **dedicated_address**
  - **dedicated_port**
- You cannot set the **mode** parameter to **dedicated**.
- Setting the **speed** parameter has no effect.
- Setting the **input_buffer_size** parameter has no effect.
- You cannot set the following serial networking protocol parameters:
  - **phone_number**
  - **slip_load_dump_host**
  - **slip_allow_dump** (always set to **N**)

The Port Server

The port server manages virtual ports for dial-in connections to the RAC.

You can connect to a RAC in several ways:

- Telnet ()
- Rlogin ()

The examples in this section use the **telnet** command. The **rlogin** command can be used in place of the **telnet** command except where noted.

Dial-in connections are managed by the port server; dial-out connections are managed by rotaries, which are a subset of the port server.

The RAC's port server accepts **telnet** and **rlogin** connection requests from users and applications on the network.

TCP Port Numbers

TCP port number 6000 is used when defining rotaries in a file (i.e., TCP port number 6000 is assigned to a single rotary). Issuing a **telnet** connection request to this TCP port results in directly attaching to the rotary (see *Rotaries on page 5-42* for more details).

TCP port number 7000 also provides a direct-mapped connection, but does not use the **telnet** protocol. This TCP port is a raw connection.

Virtual CLI (VCLI) Connections

The RAC can access the CLI through the port server from anywhere on the network, providing administrators with remote system management capabilities. The RAC creates a virtual CLI (VCLI) connection for the user when a CLI is requested at the port server prompt, or when TCP port number 5000 is included in the **telnet** command.

You can access any CLI command when using VCLI. However, of all the port characteristics, only the attention character (or character string) affects the VCLI connection. The attention character (or character string) provides access to the CLI while in a session with another host. You can define the attention character or character string using either the RAC parameter **attn_string** or the **stty attn** command (see the *Remote Access Concentrator Software Reference* for more details).

The RAC creates a new VCLI connection for each request it receives. You can limit the number of VCLI connections the RAC creates using the **max_vcli** RAC parameter. The only other limit on the number of VCLI connections that can be created is system resources.

Configuring Security

You can set up security for the port server and the VCLI connections.

Security for the Port Server

Port server security provides the option of configuring host-based security, local password protection, or both. Host-based security for the port server normally requires a user name and a password. Local password protection on the port server requires only a password.

As with security on CLI connections, local password protection can be used as the sole security mechanism or as a back-up to host-based security for occasions when the security servers are not available.

With port server security, the port server invokes the security mechanism when the user requests access to a specific port or rotary at the port server prompt. User validation occurs before the user is connected to the port to ensure that the user is authorized to connect to the selected port. If the user is not authorized, the port server notifies the user and prompts for another port.

If the user name or password is incorrect, the user is returned to the port identification prompt.

To use host-based security with the port server:

- Set the RAC parameter **enable_security** to **Y**.
- Set the port parameter **port_server_security** to **Y**.
- Create a password file (**acp_passwd**) on the security server(s); see *Encrypting Security Messages on page 6-59* for more details.

For local password protection with the port server, define a password for the **port_password** parameter.

The port parameter **port_password** is applicable for both CLI and port server connections.

Security for Virtual CLI Connections

The RAC establishes security for VCLI connections using host-based security, local password protection, or both. Host-based security validates the user name and user password. Local password protection validates only a password.

The VCLI security mechanism is similar to the port server security mechanism in that user validation is invoked after the user has requested access to the VCLI at the port server prompt. This ensures that the user is authorized to access the VCLI.

To set up host-based security on VCLI connections:

- Set the **enable_security** parameter to **Y**.
- Set the **vcli_security** parameter to **Y**.
- Create a password file (**acp_passwd**) on the security server(s); see *Encrypting Security Messages on page 6-59* for more details.

Local password protection requires only a password for validation. To set up this protection for the RAC:

- Set the parameter **enable_security** to **Y**.
- Set the parameter **vcli_security** to **N**.
- Define a password for all VCLI connections to the RAC using the **vcli_password** parameter.

Local password protection can be used as a back-up to host-based security:

- Set up host-based security for the VCLI connection.
- Define a VCLI connection password using the RAC parameter **vcli_password**. VCLI connections must adhere to any connection security defined for the RAC.

Rotaries

Once connected to the RAC, a user must either use the CLI or specify a "rotary" (which must be defined in the RAC configuration file). Rotaries are used for dial-out applications; each is a set of RAC virtual ports of the same type, on one or more RACs, grouped together so that they can be addressed by users and managed by a RAC as a single entity. A rotary allows the use of a name, rather than a number, for selecting a port.

When a user requests a rotary, the port server attaches the user to the first available virtual port.

Rotaries are defined in the **rotary** section of the configuration file. For more information on creating/using the configuration file and syntax rules for **rotary** entries, see *Configuring Hosts and Servers on page 4-1*.

Configuring Rotaries

When defining a rotary, you can:

- Name the rotary with a text string.
- Define multiple rotaries in a single file entry.

- Group together rotaries from multiple RACs.
- Assign auxiliary Internet addresses.
- Create multiple rotary definitions using the DNS server.
- Assign TCP port numbers.
- Configure visibility for the rotary.
- Configure the protocol for the rotary; this includes creating a raw rotary or forcing a binary mode connection for a rotary.
- Configure the phone numbers used by the rotary.
- Configure the port conversion type for the rotary.

Rotary Example

Following is an example of two simple rotary definitions in the RAC configuration file for a RAC:

```
%rotary
modems: phone=5551212 asy@123.456.789.1
ta_service: phone=5557777 ta@123.456.789.1
```

The first entry defines a rotary named *modems* that handles asynchronous modem calls on the RAC at Internet address 123.456.789.1. The second entry defines a rotary to handle TA calls on the same RAC.

Defining Multiple Rotaries with One Entry

You can include more than one RAC in a single entry in the **rotary** section of the RAC configuration file by separating the *ports@locations* fields with semicolons. The following entry defines a rotary named *modems* that resides on two different RACs. The rotary on *annex01* has seven ports; the rotary on the RAC with the Internet address 132.245.6.15 has one port.

```
modems: annex01; 132.245.6.15
```

When the user accesses *annex01* using a **telnet** command, the port server displays:

```
% telnet annex01
Trying...
Connected to annex01.
Rotaries Defined:
   modems
   cli
Enter Annex port name or number:
```

When the user accesses the RAC at 132.245.6.15, the port server displays:

```
% telnet 132.245.6.15
Trying...
Connected to 132.245.6.15.
Escape character is "^]".
Rotaries Defined:
   modems
   cli    -
Enter Annex port name or number:
```

Assigning Internet Addresses to Auxiliary Rotaries

An auxiliary address allows you to assign an Internet address for direct connection to the rotary. The user accesses the rotary by entering the unique auxiliary address in conjunction with the **telnet** or **rlogin** command. The auxiliary address must adhere to your network's standard addressing conventions.

Using an auxiliary address to access a rotary changes the port server's behavior. The port server does not display rotary names; instead, it attaches to the first available port in the rotary.

Ordinarily, when a user connects using **telnet** alone, the command line looks like this:

```
modems: annex01+132.245.6.80
```

Likewise, when a user connects using **rlogin** alone, the command line looks like this:

```
modems: annex01+132.245.6.80/513
```

```
modems: protocol=rlogin annex01+132.245.6.80
```

However, when a user connects using **telnet** in conjunction with an Internet address, the command line looks like this:

```
% telnet 132.245.6.80
Trying...
Connected to 132.245.6.80.
Escape character is "^]".
Attached to port 4.
```

You can also add the rotary name and the auxiliary address to **/etc/hosts** or to the host name database so users can access the rotary directly by name:

```
132.245.6.80 modems
```

This entry allows users to use the name *modems* to access the first available port in the rotary:

```
% telnet modems
Trying...
Connected to 132.245.6.80.
Escape character is "^]".
Attached to port 7.
```

Using the DNS Server to Define Multiple Rotaries

If you are using a Domain Name Service (DNS) server on the network, you can create an entry with multiple rotaries as described previously, assign Internet addresses to the rotaries, and create entries in the name servers database for the names of the rotaries. This allows users to request a rotary name using the **telnet** command.

With the DNS server, the **telnet** request attempts to connect to the first IP address returned by the name server. If that connection is unsuccessful, it moves on to the next connection, and so on, until a connection is successful. In the example below, one entry defines rotaries on two RACs:

```
modems:
       annex01+132.245.6.90;\
       annex05+132.245.6.91
```

In the DNS server's database, create two entries for the name *modems* with two different Internet addresses. For example, using a BIND name server:

```
modems   IN     A    132.245.6.90
         IN     A    132.245.6.91
```

When the user issues a **telnet** command to *modems*, **telnet** tries to locate an available port.

```
% telnet modems
Trying...
Connected to 132.245.6.91.
Escape character is "^]".
Attached to port 6.
```

Since a modem was not available on the first RAC, **telnet** automatically crossed over to the second RAC, *annex05*.

Configuring Visibility

Rotaries can be configured to be invisible to conceal the details of the connection, such as the rotary's name and the port to which a connection has been made. Ordinarily, the port server displays a (visible) rotary's name when users **telnet** or **rlogin** to the RAC's primary Internet address.

Visibility for rotaries is configured using the keyword **ps=** along with the options **visible** and **invisible**. Only rotaries that are accessible via an auxiliary Internet address or a TCP port in the 6000 range can be defined to be **invisible**. All other rotaries are always **visible**. Following is an example of an entry that makes the *HostC* rotary invisible:

```
HostC: ps=invisible annex01+132.245.6.80
```

Users who use **telnet** or **rlogin** to connect to *annex01* do not see the name; users who use **telnet** or **rlogin** to connect to *HostC* see the sequence illustrated in *Assigning Internet Addresses to Auxiliary Rotaries on page 5-44*.

Configuring the Protocol

Define the protocol between the port and the device using the keyword **protocol=** along with the arguments **telnet**, **rlogin**, **tstty**, **raw**, and **binary**.

protocol=telnet

Setting **protocol=telnet** configures **telnet** as the protocol between the port and the device. This is the default setting.

protocol=rlogin

Setting **protocol=rlogin** configures **rlogin** as the protocol between the port and the device. Another way to configure rlogin as the protocol is by specifying **TCP port 513** in the **rotary** section of the configuration file.

protocol=raw

Setting **protocol=raw** configures a raw rotary. A raw rotary passes data directly to and from the serial device -- no data processing occurs. By default, raw rotaries are invisible. Generally, raw rotaries are accessed by programs that use the system network facilities, such as the **socket** interface in BSD systems, to open a connection and to perform whatever functions are required for the device.

The setting **direct_camp_on=never** is the default for raw rotaries; **ask** cannot be used. Following is an example of a raw rotary consisting of ports on a RAC whose Internet address is 132.245.6.32. The rotary is accessed through TCP port 6300:

```
strip-record: protocol=raw\
              132.245.6.32/6300
```

protocol=binary

Setting **protocol=binary** configures a binary rotary. In this configuration, the RAC negotiates with the host to operate in **telnet binary** mode in both directions:

```
strip-record: protocol=binary 132.245.6.30
```

Assigning a Phone Number to a Rotary

A destination phone number is mandatory for each rotary on a RAC. Once a user connects to a rotary, the RAC automatically dials the specified destination phone number. The destination phone number can be specified using the keyword **phone=** . For example:

```
%rotary
modems: phone=5551212 asy@123.456.789.1
ta_service: phone=5557777 ta@123.456.789.1
```

It is also possible to create a rotary which prompts the user to enter the destination phone number in order to connect to a port. These rotaries are defined as:

```
%rotary
any_modem: phone=prompt asy@123.123.123.123/6078
any_ta: phone=prompt ta@123.123.123.123/6079
any_ta2: phone=prompt ta@123.123.123.123
```

Configuring Port Selection

The keyword **select=** defines the order in which the rotary selects ports. If **select=first**, the rotary selects the first available port in the *port_set*; **select=next** directs the rotary to keep track of the last port that was selected, and to start its search from that point.

In the example that follows, the user connects to the rotary *modems* and is attached to port 1; after disconnecting and then re-connecting to *modems*, the user is attached to port 2:

```
modems: select=next annex01
```

```
% telnet modems
Trying...
Connected to annex01.
Escape character is "^]".
Attached to port 1.
^]
telnet> quit
```

```
% telnet modems
Trying...
Connected to annex01.
Escape character is "^]".
Attached to port 2.
```

Specify the port type as the *ports* argument in the rotary definition, which has the following syntax:

rotary_name: [*keyword*] *ports@location* [; *ports@location*] ...

Table 5-7 gives the valid values for *ports* and *location*.

Since the RAC does not support devices attached to ports, users cannot configure a device by attaching to a port.

Table 5-7. Valid Values for *ports* Arguments in RAC Rotaries

Argument	Description
ports	One of the following values: **asy**, for asynchronous modem calls. **ta**, for TA calls operating at 64 kilobytes. **ta_56**, for TA calls operating at 56 kilobytes. **ta_64**, for TA calls operating at 64 kilobytes (the same as **ta**). **ta_voice,** for TA data-over-voice calls. This is referred to as Data Over Speech Bearer Service.

The *location* argument and *keywords* are the same as those defined in Chapter 4. Note that the **phone =** keyword is mandatory for RACs and optional for Remote Annexes.

The following example shows a user accessing this RAC via a **telnet** command and choosing the *modems* rotary:

```
telnet 123.456.789.1
Trying 123.456.789.1...
Connected to 123.456.789.1.
Escape character is '^]'.
<cr>
Rotaries Defined:
     modems:                                asy1-48
     ta_service:                            ta1-64
     cli                                    -

Enter Annex port name or number: modems
Attached to port asy1
```

If the RAC in the previous example had 24 internal modems instead of 32, the *modems* rotary would be displayed as *asy1-24*. Rotaries for *ta* calls always display as *ta1-64*, since there are 64 virtual *ta* ports.

Dial-Up Networking

In dial-up networking, users gain remote access to a local area network using modems and dial-up telephone lines. The RAC's implementation of protocols such as SLIP and PPP provides dial-in connectivity in a multi-protocol network. Using the RAC as a dial-up server, a remote user can dial into a modem connected to the RAC and become a directly connected network node; the RAC is transparent to the user. The RAC can also generate a call to the remote server and become a directly connected network node while remaining transparent to the user.

For details on using modems, see *Digital Modems on page 7-1*.

For details on using a SLIP link, see *Serial Line Internet Protocol* on page 10-1.

For details on using a PPP link, see *Point-to-Point Protocol on page 8-1*.

For details on filtering, see *Using Filters for Security on page 6-146*.

Dynamic Dialing

Dynamic dialing allows system administrators to define a database of information about a modem pool and a set of virtual dial-out routes. The RAC assigns each virtual dial-out route dynamically to ports in that pool. Each virtual dial-out route must be configured within the RAC configuration file to start the dial-out connection.

Dynamic dialing applies only to IP framing protocols.

Dial-out routes are defined in the RAC's configuration file. *Setting Up the Configuration File on page 3-12* provides a sample configuration file.

Dynamic dialing also provides:

- Dial-on-demand network routing by having the RAC's dial-out routes configured with the proper static network routes (thus, the RAC can route traffic to a particular network through a dial-out route's destination).
- The use of chat scripts; these are sequences of commands that are used to log into the remote system after the phone connection is established and before IP forwarding begins.
- The use of both the **netact** and **no_start** filter actions for dial-out connection interfaces. These actions will operate as described in *Using Filters for Security on page 6-146*. However, you cannot add a filter by using the CLI **filter** subcommand **add**. Instead, you specify a *filter* definition in the **dialout** section of the RAC configuration file. In this definition, omit both the word **add** and the name of the interface (e.g., asy7). For more details, see *Setting Up the Configuration File on page 3-15*.

A dynamic dialing route appears as a normal route to the end user. It has an entry in the route cache and is advertised by the RAC. (Advertising is enabled by default; see *rip_advertise* on page 11-55.) When a user tries to send traffic to its destination (e.g., using **telnet**), the connection protocol's process detects this traffic, establishes the phone connection by dialing into a modem, and then continues normal operation.

Network Inactivity

Dynamic dialing resets the dial-out line when no traffic occurs on the line for a certain length of time. Resetting the line terminates the phone connection, saving costs by stopping inactive users from keeping phone connections open.

With active RIP enabled (as it is by default) on a dialout route, RIP updates, sent every 30 seconds, are considered activity, and reset the inactivity timer. A filter must be applied to the **dialout** configuration to prevent this. A sample filter entry for a **dialout** configuration is:

```
filter in exclu proto udp src_port router netact
filter out exclu proto udp src_port router netact
```

Additionally, RIP updates are not sent over an inactive/quiescent dialout line, and therefore do not activate an inactive/quiescent dialout line.

Enabling Dynamic Dialing

To enable dynamic dialing:

1. **Configure the** digital_modem **section of the RAC configuration file.**

Several standard entries for the **digital_modem** section of the RAC configuration file are supplied with the software distribution. These entries, defined for use with the configuration file, are located in the file **/usr/annex/bfs/digital_modems.annex**.

If the modem you are using is contained within this file, using the **%include** *filename* command tells the parser that entries in the specified file are part of the configuration file. For example:

```
%digital_modem
%include digital_modems.annex
```

If the modem you are using is not contained within this file, you must edit the configuration file accordingly.

On each RAC, edit the configuration file to contain a **digital_modem** section similar to the sample entry. If the RAC boots from a host, the file resides on the host; for self-boot units, the file resides on the RAC.

2. **Reset the modem:**

```
annex# admin
Annex administration Remote Annex 13.3, 72 ports
admin: reset annex modem
```

3. **Configure the** dialout **section of the RAC configuration file. Table 5-8 on page 5-54 provides field definitions for the dialout entries.**

Each entry in the **dialout** section of the configuration file defines a dial-out route. The format of a **dialout** entry looks like this:

```
%dialout

global_timeout <time-out value>

# this is a comment line

begin_route      <route id>
local            <local address>
remote           <remote address>
mode             <slip or ppp>
ports            <port type/rotary>
phone            <phone number>
chat             <chat script list>
filter           <filter command>
disabled         <time interval>
advertise        <Y or N>
set              <parameter_name setting>
set              <parameter_name setting>
end_route
```

The mandatory fields for a **dialout** entry are: *begin_route*, *remote*, *mode* (**slip** or **ppp**), *ports*, *phone*, and *end_route*.

When defining dial-out routes in the **config.annex** file, specify a port type or a rotary name for the *ports* field in a dial-out entry. Table 5-8 describes the valid port types.

Table 5-8. Valid Port Types for ports Field in a Dial-out Entry

Port Type	Description
asy	For asynchronous modem calls.
ta	For TA calls operating at 64 kilobytes.
ta_56	For TA calls operating at 56 kilobytes.
ta_64	For TA calls operating at 64 kilobytes (the same as **ta**).
ta_voice	For TA data-over-voice calls. This is referred to as Data Over Speech Bearer Service.
syn	For synchronous PPP calls operating at 64 kilobytes.
syn_56	For synchronous PPP calls operating at 56 kilobytes.
syn_64	For synchronous PPP calls operating at 64 kilobytes (same as **syn**).
syn_voice	For synchronous PPP data-over-voice calls. This is referred to as Dial Out Speech Bearer Service.

If you specify a rotary for ports, the telephone number defined for the rotary takes precedence over the phone number specified in the **dialout** entry.

Using separate *set* field entries, set any port parameters that you want included in the **dialout** entry (for more details, see Table 5-8 on page 5-54).

All parameter settings for the global port are in effect for dial-out routing configurations, unless the parameters are overridden by *set* commands in the **dial-out** entry. To see the asynchronous and TA global port settings, issue the superuser **admin** or **na** command **show port all**. To see the synchronous PPP settings, issue the superuser **admin** or **na** command **show port syn**.

Any port parameter not set in the **dialout** entry will use the current settings in non-volatile memory.

Set the following parameters for each port type to which the **dialout** entry applies:

a) Set the **mode** parameter to **slave**, **adaptive**, or **auto-adapt**.

b) Set **type** as appropriate for the attached equipment.

c) Set **speed** as appropriate for the attached equipment.

d) Set the **type_of_modem** parameter to match the *type_of_modem* field entry in the **modem** section of the RAC configuration file.

e) Set **control_lines** as appropriate for the attached equipment.

f) Set **input_flow_control** as appropriate for the attached equipment.

g) Set **output_flow_control** as appropriate for the attached equipment.

For more details, see *Sample Configurations for Dial-Out Routing on page 5-56*. For more details on RAC configuration parameters, see the *Remote Access Concentrator Software Reference*.

4. **Issue a** reset annex dialout **command.**

If a SLIP or PPP line is using both dynamic dialing and network inactivity, network inactivity only goes into effect after dynamic dialing has been activated.

5. **Initiate a dynamic dialing session:**

a) On the terminal connected to each RAC, issue the CLI superuser **modem -a** command to verify the modem type and each of the strings defined for the modem.

b) On each RAC, issue a CLI **netstat -r** command to verify that there is a route (e.g., do2).

c) When outbound traffic is detected, the RAC initiates a dial-out connection based on the information in the **dialout** section of the RAC configuration file. Debug level syslogging tracks the connection attempt.

d) If the network inactivity timer expires before the dialout connection is established, the RAC breaks the connection. Any subsequent activity across the link (that has not been filtered out) re-establishes the dynamic dial connection.

See *Sample Configurations for Dial-Out Routing on page 5-56* for an example of a final configuration for two RACs.

For details on filtering, see *Using Filters for Security on page 6-146*.

For more details on dial-up SLIP, see *Serial Line Internet Protocol* on page 10-1.

For more details on dial-up PPP, see *Point-to-Point Protocol on page 8-1*.

For more details on editing the configuration file, see *Using the Configuration File on page 3-1*.

For more details on RAC configuration parameters, see the *Remote Access Concentrator Software Reference*.

Sample Configurations for Dial-Out Routing

This section illustrates two RACs, RAC A and RAC B, configured for dial-out routing. Figure 5-1 depicts the two units. The text that follows the figure shows:

- The dial-out definitions that would appear in the RAC configuration files.
- The SPBs required in order for the RACs to handle dial-out calls from each other.

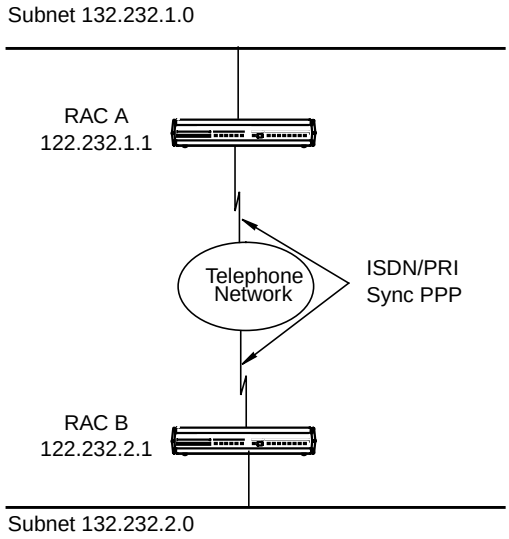

Figure 5-1. RACs to Be Used for Dial-out

RAC A **dialout** configuration:

```
%dialout
annex 132.245.1.1

begin_route   1
mode          ppp
local         122.232.1.1
remote        122.232.2.1
set           net_inactivity 20
phone         16175551234
```

(continued on next page)

```
set             do_compression Y
set             allow_compression Y
set             net_inactivity_units minutes
set             subnet_mask 255.255.255.0
set             rip_sub_advertise Y
set             rip_sub_accept Y
set             rip_advertise all
set             rip_accept all
advertise       Y
ports           syn
set             ppp_ncp ipcp
filter          in excl proto udp src_port router netact
filter          out excl proto udp src_port router netact
end_route
end
```

In the previous example:

- The subnet mask of 255.255.255.0 applies to the remote end of the PPP link.
- The **ppp_ncp** parameter *must* be set to **ipcp** in order for the filters (defined next) to operate; filters apply to IP packets only.
- The two filters at the end of the dial-out entry prevent RIP updates (generated every 30 seconds) from being considered activity. Without these filters, RIP updates could activate a dial-out connection that might (and should) otherwise time out.

RAC B **dialout** configuration:

```
%dialout
annex 132.245.2.1
begin_route   1
mode          ppp
local         122.232.2.1
remote        122.232.1.1
set           net_inactivity 20
phone         16175554321
set           do_compression Y
set           allow_compression Y
set           net_inactivity_units minutes
set           subnet_mask 255.255.255.0
set           rip_sub_advertise Y
set           rip_sub_accept Y
set           rip_advertise all
set           rip_accept all
advertise     Y
ports         syn
set           ppp_ncp ipcp
filter        in excl proto udp src_port router netact
filter        out excl proto udp src_port router netact
end_route
end
```

The following SPB is defined in RAC A's configuration file. This SPB handles RAC B's dial-out calls when they arrive at RAC A.

```
%pri

begin_session sync
called_no 16175554321
call_action sync
set mode ppp
end_session
```

The framework for this SPB is included in the default RAC configuration file. To activate the SPB, remove the comment character (#) from the beginning of each line that is not a comment, and specify the called number, so that the SPB resembles the example above. Then save the file and issue a **reset annex session** command from **na** or **admin**.

The following SPB is defined in RAC B's configuration file to handle Router A's dial-out calls when they arrive at RAC B.

```
%pri

begin_session sync
called_no 16175551234
call_action sync
set mode ppp
end_session
```

Parameter settings in SPBs apply to incoming calls only and have no effect on outgoing calls. However, you must define an SPB to handle the incoming call on the remote side of a dial-out route if the destination is another RAC. For more information on SPBs, see *SPB Configuration Procedure on page 5-32*.

Displaying Dynamic Dialing Routes in the Routing Table

The CLI **netstat -r** command displays statistics and information about all available routes in the routing table; dynamic dialing routes that do not have a phone connection established appear without a **U** in the *Flags* field. A route is comprised of a destination host or network and the gateway through which data is forwarded.

In the following example, routes *do14* and *do16* are dynamic dialing routes waiting for activation:

```
annex: netstat -r
Routing tables
Destination     NextHop        Flags  Usage   UseCount Mtr Interface
127.0.0.0/8     *              UI     fixed   0        2   lo0
132.245.1.0/24  132.245.44.22  UR     -114    0        3   en0
132.245.33.0/24 *              QI     fixed   147      1   do14
132.245.34.0/24 *              QI     fixed   0        2   do16
132.245.44.0/24 *              UHF    fixed   838      2   asy10(slip)
```

In the following example, dynamic dialing route *do14* has been dialed and used:

```
annex: netstat -r
Routing tables
Destination     NextHop        Flags  Usage   UseCount Mtr Interface
127.0.0.0/8     *              UI     fixed   0        2   lo0
132.245.1.0/24  132.245.44.22  UR     -114    0        3   en0
132.245.34.0/24 *              QI     fixed   0        2   do16
132.245.44.0/24 *              UHF    fixed   838      2   asy10(slip)
132.245.33.0/24 *              UI     fixed   1        1   asy14(ppp)
```

Chapter 6
Configuring Security

RACs provide comprehensive security features that assist you in securing your network from unauthorized access. Using these features, you can select from among the following types of security:

- Local password protection (where the passwords are stored on the RAC).
- Host-based security (where at least one host on the network is functioning as a security server).

If unauthorized users can access your RAC, we strongly suggest that you enable security after loading the host code and booting the unit.

Enabling Security

Security for the RAC is disabled by default. To use any security feature, including local password protection, you *must* enable security for the RAC by setting the **enable_security** parameter to **Y**. If the **enable_security** parameter is set to **N**, no security is used, and no logging is performed regardless of any other parameter settings. The administrative password used for access to administrative tools is unaffected by this parameter, however; the admin password is always enabled.

The **enable_security** parameter does not take effect until the RAC is either rebooted or reset.

Setting Security-Related Parameters

You need to determine which global security parameters to change and which to set for specific sessions (by creating or editing Session Parameter Blocks (SPBs) within the RAC configuration file). For more details, see *Configuring the WAN Interfaces, Global Ports, and Sessions on page 5-1*.

Configuring Local Security

Local Password Protection

Local password protection allows you to assign a password that a user must enter before accessing a RAC. Because this password is stored locally on the RAC, it does not require a remote security server. Local password protection can be used as a back-up security mechanism in case the host-based security servers are unavailable. It validates access from either the device or the network. Local password protection supports **cli**.

Local password protection can be defined for access through virtual ports (i.e., sessions). Local password protection does not provide logging of security events to the security server. If event logging is enabled, user activities can be logged to **syslog** with local password protection (see *Using Event Logging on page 4-3*).

The passwords set in the following parameters are stored on the RAC and do not involve the use of a security server:

- **password**
- **port_password**
- **ppp_password_remote**
- **vcli_password**

Implementing Local Virtual CLI Password Protection

Local password protection can be implemented for the RAC in either of two ways:

- Upon virtual CLI (VCLI) connection.
- Upon access through administrative utilities.

The **vcli_password** parameter defines a local password for VCLI connections. The user enters a password only, and does not need to provide a user name.

To configure the RAC for local VCLI password protection:

1. **Enable local security by setting the** enable_security **parameter to** Y**.**
2. **Disable VCLI remote security by setting the** vcli_security **parameter to** N**.**
3. **Define a password using the** vcli_password **parameter.**

The RAC behaves as follows:

- If the **vcli_password** parameter is not set (**"<unset>"**) and the **vcli_security** parameter is set to **N**, the RAC prompts for the password specified by the **password** parameter.
- If the **password** parameter is not set (**"<unset>"**), the RAC fails the VCLI attempt.
- If the **vcli_security** parameter is set to **N** and the **vcli_password** parameter is set (**"<set>"**), the RAC prompts for the password specified in **vcli_password**.
- If the **vcli_security** parameter is not set (**N**) and the **vcli_password** parameter is not set (**"<unset>"**), the RAC does not perform a security check for VCLIs.
- If the **vcli_security** parameter is set to **Y**, the **vcli_password** parameter is not set (**"<unset>"**), and the **password** parameter is not set (**"<unset>"**), the RAC denies access to the VCLI if the security server is unreachable.

You can also use **vcli_password** as a back-up to host-based security. When local VCLI password protection is used as a back-up, the RAC first accesses the security server to validate a CLI connection request. If no response is received from a security server, the RAC requests the local VCLI password. The user can enter either the VCLI password or the RAC administrative password.

To set up the local VCLI password for back-up security:

- Enable security by setting the **enable_security** parameter to **Y**.
- Enable VCLI security by setting the **vcli_security** parameter to **Y**.
- Define a password using the **vcli_password** parameter.
- Define a security server host using the **pref_secure1_host**, **pref_secure2_host**, or **security_broadcast** parameter (e.g., 0.0.0.0).
- Create a **password** file on the security server (see *Creating User Password Files on page 6-72*).

If the remote server(s) fail:

- Access is permitted only through the VCLI password.
- No access is permitted if the **vcli_password** parameter is not set.

The **show annex** command does not display the value of the **vcli_password** parameter. Instead, it displays "<set>" or "<unset>".

Administrative Password

The RAC administrative password protects the administrative tools; the default administrative password is the RAC's IP address. When the **show annex** command displays the password as **"<unset>"**, use the default administrative password for:

- Access to superuser CLI commands.
- Access to sessions locked with the CLI **lock** command.
- Access to a virtual CLI connection through local password protection (if the **vcli_security** parameter is set to **N**).

Modifying the assigned administrative password enables password protection on access to **na** and **admin**.

The administrative password validates access to a RAC through **na** only when security is enabled and the password is defined. Also, it can be used as the VCLI password and to override the password assigned with the CLI **lock** command.

The administrative password never displays. If you forget the modified password, you can reset it only by erasing the RAC's non-volatile memory using the ROM Monitor **erase** command, and re-entering all parameters.

As a safeguard against losing the unit's current configuration, use the **na** command **write** to save the RAC and global port parameter settings; if necessary, you can restore these settings using the **read** command (for more details on using these commands, see *The na and admin Utilities on page 2-11*).

Protecting the Superuser CLI

A RAC administrative password is required for access to the superuser CLI. The default password is the RAC's IP address. There are two ways to change the password:

- Using the superuser CLI **passwd** command.
- Changing the **password** parameter using **na** or **admin**.

Using either method, the new password takes effect immediately for access to the superuser CLI. Reset the password to the RAC's IP address by either:

- Using **na** or **admin** to set the **password** parameter to the null string:

  `command: set annex password ""`

- Using the superuser CLI **passwd** command and pressing **Return** in response to the prompt for a new password.
- Erasing all parameters using the ROM monitor **erase** command.

Protecting Resources from Unauthorized Access

When terminals are connected to a network, they provide users with the potential for unauthorized access to network resources. In addition to the available security schemes, the RAC provides timers that can terminate a session. The default **cli_inactivity** parameter (default setting: **off**) sets the CLI inactivity timer. When enabled, the RAC terminates the session after the amount of time specified in this parameter has elapsed or the last session is completed.

Users can protect their login sessions using the CLI **lock** command if they do not want to log out when leaving the terminal unattended.

Protecting the na Utility from Unauthorized Access

When using the **na** utility, users can access RAC parameters and obtain useful information, or reconfigure and reboot RACs. Protecting **na** involves UNIX superuser protection and the RAC administrative password.

Upon installation, **na** is owned by root and executable by all. Only a superuser can execute the **set**, **reset**, **broadcast**, **dumpboot**, **boot**, **read**, and **copy** commands.

Enhancing Password Security

The following sections describe how to configure the RAC to record password histories and blacklist users. It also explains how to view and manage the database in which password histories and blacklisting information is kept.

Overview of Password History and Aging

You can configure ACP to save the encrypted form of the previous passwords a user has set. This applies to passwords set by using the **ch_passwd** utility on the security host or by responding to a RAC prompt when a password expires. These passwords are stored in the UNIX database **acp_dbm** on the security host, where they are keyed on user names. If a user tries to reset his or her password to one of the stored values, ACP will reject it and display an error message.

Benefits of Password Histories

The password history mechanism helps protect against off-line, "dictionary" attacks. In this kind of attack, a user obtains the encrypted **acp_passwd** (or **/etc/passwd**) file. The user then tries to crack the passwords by taking a dictionary of words, encrypting the words (using *salted* DES encryption) and comparing them to the encrypted passwords.

Benefits of Password Aging

The longer a password is in effect, the more time an attacker has to crack its encryption. Consequently, the password history feature is most effective when used in conjunction with password aging. If password aging is enabled:

- The user *must* change passwords when a predefined amount of time has elapsed. If the user never changes passwords, there is no password history to record.
- The user *cannot* change passwords until the predefined amount of time has elapsed. This prevents potential intruders from changing passwords in rapid succession in an attempt to cycle the old passwords out of the password history and use them again.

Password aging is enabled through the use of a **shadow** file in conjunction with a **passwd** file. By default, **erpcd** uses the **acp_passwd** file alone, so password aging is initially disabled. When only the **passwd** file is used (a Berkeley standard), that file contains both the user names (UIDs) and the encrypted passwords. The **passwd/shadow** form (used with UNIX System V) contains an *x* in place of a password in the **passwd** file and saves the encrypted passwords in a separate **shadow** file.

If your UNIX is based on System V and you want to use the password history feature, choose the **passwd/shadow** scheme. Use the *convert* program, located in the **erpcd** directory, to change the integrated **passwd** form to the **passwd/shadow** form (and vice-versa).

If your UNIX is based on a Berkeley BSD system, password history is disabled by default. To enable it, change the value of the *STORED_PASS #define* statement in **acp_policy.h**, as described in the following section.

Enabling and Configuring Password Histories

To turn on the password history feature and (optionally) enable aging via **shadow** files:

1. **Use** na **or** admin **to set the** enable_security **parameter to** Y **for the RAC(s) you are configuring.**
2. **Use** na **or** admin **to define a security host for the RAC(s) you are configuring. (See** ***Configuring the Security Server on page 6-57*.****)**
3. **Log into the security host as** root**.**
4. **Go (**cd**) to the installation directory (typically** /usr/annex**).**
5. cd **to the** src/erpcd **directory, which is within the installation directory.**
6. **In the** erpcd **directory, use a text editor to modify the** acp_policy.h **file.**
   - If you are using a **shadow** file, uncomment the following line in **acp_policy.h**:

```
/* #define USESHADOW 1 */
```

To uncomment the previous line, delete the slashes and asterisks at the beginning and end of the line, so that the line is as follows:

```
#define USESHADOW 1
```

- Look for the following lines to define STORED_PASS, which are already uncommented. The lines define the number of passwords that are stored to prevent them from being re-used. The variable is initialized to 6 for **passwd**/**shadow** files and 0 for **passwd** files alone. A value of 0 disables password history.

  ```
  #ifdef USESHADOW
  #define STORED_PASS 6
  #else
  #define STORED_PASS 0
  #endif
  ```

  You can store up to 12 passwords in a **shadow** file.

  If you are using a **passwd** file alone and you want to enable password history, change the value of the second STORED_PASS from 0 to a number from 1 through 12.

  Specifying a non-zero value for either of the above STORED_PASS variables turns on the recording of password histories in **acp_dbm**.

- The final variable related to password history is MAX_STORED_PASS. It defines the absolute maximum number of login failures that can occur before a user is blacklisted. It is best not to change this variable, which is set to 12. If you must change it, follow the instructions in **acp_policy.h**.

7. **If you are using only the password history feature, follow the instructions in Steps 8 through 11, below. If you are also using blacklisting, wait to do this until you have configured both features.**

8. **From the** /usr/annex/src **directory, recompile** erpcd**:**

```
# cd /usr/annex/src
# make install
```

   This automatically rebuilds **erpcd** and any other files that need to be recompiled because of the changes you have made. In addition, the following message is displayed:

```
WARNING: If you have called “make install” yourself,
then in directory /usr/annex you will have to copy
erpcd.new to erpcd. Make sure the erpcd daemon is not
running when that is done.

If you used the installation script called “make
install” then the copy will be done for you.
```

9. **If** erpcd **is running on the host and the host is running Berkeley BSD UNIX, kill the existing** erpcd **process as follows (your process number will vary):**

```
# ps -ax | grep erpcd
25493 ? IW 0:00 ./erpcd
25797 p1 S 0:00 grep erpcd
# kill -9 25493
```

   To kill the process on a System V host, substitute the following for the first line above:

```
# ps -ef | grep erpcd
```

10. cd **to directory /usr/annex and copy** erpcd.new **to** erpcd**:**

```
# cd /usr/annex
# cp erpcd.new erpcd
```

11. **Now, restart** erpcd**.**

```
# ./erpcd
```

After completing Steps 1 through 11, the **acp_dbm** database is created automatically the first time a user changes a password via the **ch_passwd** utility. To list the users for which password histories exist, go to the security host’s install directory (default is **/usr/annex**) and issue the **acp_dbm -l** command:

```
# cd /usr/annex
# acp_dbm -l
List of users currently present in the acp_dbm:
                    hobbes
                    fritz

#
```

In the previous example, password histories are saved for users **hobbes** and **fritz**.

Overview of Blacklisting

A user account is considered under attack, and therefore blacklisted, when either (or both) of the following occurs:

- A configurable number (default is 5) of consecutive failed login attempts is exceeded. In other words, if you use the default, a user is blacklisted on the sixth consecutive failed login attempt.
- A configurable number (default is 10) of non-consecutive failed login attempts is exceeded within a configurable period of time (default is 26 weeks). If you use the defaults, a user is blacklisted when the eleventh failed login attempt occurs within a period of 26 weeks.

Blacklisting enhances security by limiting the number of passwords an on-line attacker can try before the user account is automatically disabled. At this point, no one can log in with the blacklisted user name, even if someone enters the “correct” password. However, the failed login message is the same before and after blacklisting, so the user does not know that the account has been disabled.

The system administrator is informed when blacklisting occurs. First, a record is created in the ACP log file indicating that the user ID has been blacklisted. This record remains unless and until you delete it manually. Second, when you invoke the **acp_dbm** utility, it immediately displays a warning identifying any blacklisted users. See *Viewing and Managing the acp_dbm Database on page 6-14*.

The data necessary for blacklisting is kept in the **acp_dbm** database, keyed on the user name. If password history and blacklisting are configured, this database is created automatically the first time a user changes passwords or attempts to login and fails. The absence of an **acp_dbm** database indicates that no password histories exist and no failed login attempts have occurred.

Blacklisting makes the RAC susceptible to denial-of-service attacks. To disable a user account, a saboteur need only make a few failed login attempts. In the extreme case, a saboteur who obtains a list of employee user names could create a shell script that would automatically disable all user login capabilities.

Configuring Blacklisting

You can configure blacklisting in one of two ways:

- By editing *#define* statements in the **acp_policy.h** file.
- By issuing the **erpcd** command with the **-b**, **-x**, and **-g** options.

The **erpcd** syntax is:

erpcd [**-b***max_con*] [[**-x***max_total*] [**-g***period*]]

Do not enter any space between an option (e.g., **-b**) and the value you specify with it (e.g., *max_con*).

The **erpcd** options override the **acp_policy.h** variables. Table 6-1 describes the options and their **acp_policy.h** equivalents.

For information on how to edit and rebuild **erpcd** and the other files that have changed in the **acp_policy.h** file to put the modifications into effect, see Steps 6 through 11 in *Enabling and Configuring Password Histories on page 6-8*.

Once you have configured and activated blacklisting, **erpcd** automatically creates the **acp_dbm** database the first time a user makes an unsuccessful login attempt. To monitor the blacklist status of one or more users, go to the directory (on the security host) that contains **erpcd** and use the **acp_dbm** utility (see *Viewing and Managing the acp_dbm Database on page 6-14*).

Table 6-1. The erpcd Options and acp_policy.h Variables

erpcd Option	Equivalent acp_policy.h Variable	Description
-b*max_con*	MAX_BL_CON	The number of consecutive login failures allowed before a user is blacklisted. Valid values are 0-8. A value of 0 enables blacklisting upon any login failure (not recommended). 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*	MAX_BL_ NONCON	The number of non-consecutive login failures allowed before a user is blacklisted. Valid values are 0-20. A value of 0 enables blacklisting upon any login failure (not recommended). 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.

(continued on next page)

Table 6-1 erpcd Options and acp_policy.h Variables

erpcd Option	Equivalent acp_policy.h Variable	Description
-g*period*	MAX_BL_ PERIOD	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 0-52. 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 disabled.

Viewing and Managing the acp_dbm Database

The **acp_dbm** utility lets you manage and display information about password histories and blacklisting from the **acp_dbm** database. To use this utility, you must log in with a user ID of **root** or have superuser privileges. If neither is the case, **acp_dbm** immediately exits on invocation and displays the message:

```
You must have root privilege to run acp_dbm.
```

Execute the **acp_dbm** utility from the directory containing **erpcd**. Upon execution, **acp_dbm** immediately sends a warning message to standard output for each user on the blacklist. The message format is:

```
Warning: Annex user userid may be under attack; all logins
for this account have been disabled.
```

In this message, *userid* is the user name for the account that is blacklisted.

The syntax for the **acp_dbm** utility is:

acp_dbm [**-s** *username*] [**-c** *username*] [**-d** *username*] [**-l**]

Table 6-2 explains the options.

Table 6-2. Options for the acp_dbm Utility

Option	Description

-s *username*	Sends information about *username* from the **acp_dbm** database to standard output. The output (after the initial warning message) shows the user name, the total number and type of failures, and the date and time of each failure. The following is an example:
	`User name: hobbes`
	`Total number of consecutive failed login attempts: 2`
	`Login failure on Tue Dec 12 12:49:49 1995`
	`Login failure on Mon Dec 11 11:25:10 1995`

-c *username*	Clears *username* from the blacklist and deletes all records of login failures for *username*. Does not clear the password history or any other information about *username* in the **acp_dbm** database. Before using this option, investigate the account thoroughly so that you are confident it is not under attack.
-d *username*	Deletes the user record from the **acp_dbm** database. Use this option, rather than **-c**, to delete the **acp_dbm** user account entirely. This option does not delete references to *username* in any other ACP files, such as **acp_userinfo** and **acp_passwd**. You must explicitly remove the user name from these files to delete the user completely.
-l	Lists all the user names contained in **acp_dbm**, including those with password histories.

Deleting the acp_dbm Database

The only way to delete the **acp_dbm** database is via the UNIX **rm** command.

Error Handling for Password Histories and Blacklisting

The following error conditions can occur:

- If **erpcd** cannot read or write to the **acp_dbm** database or detects incorrect protection, the event is syslogged at level LOG_CRIT and all users are denied access until **erpcd** can read and write to **acp_dbm**.

  If the wrong protection is detected, the syslogged message is:

  ```
  Security problem: Wrong protection (not 600) on acp_dbm database.
  ```

  If **erpcd** cannot read or write to **acp_dbm**, the message is:

  ```
  Cannot [read from | write to] acp_dbm database.
  ```

- If the **acp_dbm** utility fails to read or write the **acp_dbm** database, it generates the following message:

  ```
  acp_dbm: Error [reading from | writing to] acp_dbm database.
  ```

  If the utility detects the wrong protection, it generates the following message:

  ```
  acp_dbm: Wrong protection (not 600) on acp_dbm database.
  ```

- If the **ch_passwd** utility fails to read or write the **acp_dbm** database, **ch_passwd** generates the message:

  ```
  ch_passwd: Error [reading from | writing to] acp_dbm database.
  Notify System Administrator. Password change cancelled.
  ```

  If **ch_passwd** detects the wrong protection, it generates the message:

  ```
  ch_passwd: Wrong protection (not 600) on acp_dbm database.
  Notify System Administrator. Password change cancelled.
  ```

Configuring RADIUS Security

RADIUS is an IETF-developed protocol that defines a communication standard between a Network Access Server (NAS), a RAC in this case, and a host-based communication server.

RADIUS operates in three modes:

- RADIUS Authentication includes authentication of the dial-up user to the RADIUS server, as well as authentication of the RADIUS server to the NAS. RADIUS supports the authentication modes PAP (Password Authentication Protocol), CHAP (Challenge Handshake Authentication Protocol), and user name/password validation. Authorization is sent back by the server.
- RADIUS Accounting defines a communication standard between a NAS and a host-based accounting server. It records duration of service, packet throughput, and raw throughput.
- RADIUS Authorization; a user's authorization information is supplied by the RADIUS server.

The RAC's software includes a native RADIUS client; a RADIUS server is available from Bay Networks separately, or you can use any other RADIUS server. You can use the RADIUS client independently, with the RAC's security regime set to RADIUS, or you can use erpcd as a proxy RADIUS client, running under the ACP security regime.

Note that the capabilities provided by the proxy RADIUS client running under the ACP security regime are a subset of those provided by the native RADIUS client; any transactions which take place between the proxy RADIUS client and ACP also take place when the security regime is RADIUS.

Using erpcd As a Proxy RADIUS Client

To configure security using erpcd as a proxy RADIUS client:

1. **Set** enable_security **to** Y **and set** auth_protocol **to** acp**.**
2. **Configure the** erpcd.conf **file on the pref_secure1_host. This causes the RAC to send ACP authentication and authorization requests to erpcd. erpcd converts these requests to RADIUS format, and sends the reformatted reports to the RADIUS server configured in the** erpcd.conf **file.**
3. **Reset the RAC in order for these changes to take effect.**

Using the RAC's Native RADIUS Client

To configure security using the RAC's native RADIUS client:

1. **Set** enable_security **to** Y **and set** auth_protocol **to** radius**.**
2. **Reset the RAC in order for this change to take effect.**

RADIUS Parameters

The following **admin**/**na** parameters support the RAC's embedded RADIUS capability.

Refer to the *Remote Access Concentrator Software Reference* for a complete description of each parameter.

- **address_origin -** This parameter specifies the server from which the dial-in user receives a network address.
- **auth_protocol -** This parameter indicates which authentication regime will be used, **acp** (the default) or **radius**. This parameter must be set to **acp** in order for erpcd to be used as a proxy RADIUS client.

- **enable_radius_acct** - This parameter, when set to **Y**, enables RADIUS accounting when security is enabled and the security regime is RADIUS. The RADIUS accounting server runs on the same host as the RADIUS authentication server.
- **enable_security** - This parameter must be set to **Y** in order for any security regime to work. The parameter's default value is **N**.
- **pref_secure1_host** - This parameter must be set to the IP address of the primary RADIUS and RADIUS accounting server.
- **pref_secure2_host** - This parameter must be set to the IP address of the secondary RADIUS and RADIUS accounting server.
- **radius_acct_level** - The level of RADIUS accounting used (**basic** or **advanced**).
- **radius_acct_port** - The number of the UDP port on which the RADIUS accounting server listens. The default value is **1813**. (Older RADIUS servers use port **1646**.)
- **radius_auth_port** - The number of the UDP port on which the RADIUS server listens. The default value is **1812**. (Older RADIUS servers use port **1645**.)
- **radius_port_encoding** - This parameter controls the format in which RADIUS accounting information is reported to the RADIUS server. The possible values for this parameter are **device** (the default) and **channel**.
- **radius_retries** - The number of times the RADIUS client retries sending access-request/accounting-request packets before trying the secondary host. The default value is **10**.
- **radius_secret** - This string defines the RADIUS shared secret for the RAC. By default, the shared secret is unset.

  

  The value of **radius_secret** should not be exposed on the network in clear text form.

- **radius_timeout** - The retransmission timer for RADIUS access-request/accounting-request packets. The default value is **4**.

RADIUS Attributes

RADIUS tracks various pieces of data using attributes. The RAC supports a number of standard RADIUS attributes, plus a number of Bay Networks vendor-specific attributes (VSAs) which are equivalent to entries in various files used by the ACP security regime.

Supported RADIUS Standard Attributes

- **User-Name (1)**
- **User-Password (2)**
- **CHAP-Password (3)**
- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Service-Type (6)**
- **Framed-Protocol (7)**
- **Framed-IP-Address (8)**
- **Framed-IP-Netmask (9)**
- **Framed-Routing (10)**
- **Filter-Id (11)**
- **Framed-MTU (12)**
- **Framed-Compression (13)**
- **Login-IP-Host (14)**
- **Login-Service (15)**
- **Login-TCP-Port (16)**
- **Unassigned (17)**
- **Reply-Message (18)**
- **Callback-Number (19)**

- **Callback-Id (20)**
- **Unassigned (21)**
- **Framed-Route (22)**
- **Framed-IPX-Network (23)**
- **State (24)**
- **Class (25)**
- **Vendor-Specific (26)**
- **Session-Timeout (27)**
- **Idle-Timeout (28)**
- **Termination-Action (29)**
- **Called-Station-Id (30)**
- **Calling-Station-Id (31)**
- **NAS-Identifier (32)**
- **Proxy-State (33)**
- **Login-LAT-Service (34)**
- **Login-LAT-Node (35)**
- **Login-LAT-Group (36)**
- **Framed-AppleTalk-Link (37)**
- **Framed-Apple-Talk-Network (38)**
- **Framed-AppleTalk-Zone (39)**
- **CHAP-Challenge (60)**
- **NAS-Port-Type (61)**
- **Port-Limit (62)**
- **Login-LAT-Port (63)**

Supported RADIUS Accounting Attributes

- **Acct-Status-Type (40)**
- **Acct-Delay-Time (41)**
- **Acct-Input-Octets (42)**
- **Acct-Output-Octets (43)**

- **Acct-Session-Id (44)**
- **Acct-Authentic (45)**
- **Acct-Session-Time (46)**
- **Acct-Input-Packets (47)**
- **Acct-Output-Packets (48)**
- **Acct-Terminate-Cause (49)**
- **Acct-Multi-Session-Id (50)**
- **Acct-Link-Count (51)**

Bay Networks Vendor-Specific Attributes (VSAs)

These attributes enable RADIUS to emulate the behavior of the ACP security regime.

- **Annex-Filter (VSA Bay Networks 28)**
- **Annex-CLI-Command (VSA Bay Networks 29)**
- **Annex-CLI-Filter (VSA Bay Networks 30)**
- **Annex-Host-Restrict (VSA Bay Networks 31)**
- **Annex-Host-Allow (VSA Bay Networks 32)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**
- **Annex-Local-IP-Address (VSA Bay Networks 35)**
- **Annex-Tunnel-Type (VSA Bay Networks 36)**
- **Annex-Tunnel-Medium-Type (VSA Bay Networks 37)**
- **Annex-Tunnel-Client-Endpoint (VSA Bay Networks 38)**
- **Annex-Tunnel-Server-Endpoint (VSA Bay Networks 39)**
- **Annex-Tunnel-Id (VSA Bay Networks 40)**
- **Annex-Tunnel-Connection-Id (VSA Bay Networks 41)**
- **Annex-Callback-Port-List (VSA Bay Networks 42)**

The RADIUS Dictionary File

A reference RADIUS dictionary file is included in the distribution kit and is placed in the security files area. The dictionary file defines keywords, types, and values for RADIUS attributes and their corresponding code points. The file is in a format that is used as input by some RADIUS servers to parse messages and write text output files. You may have existing dictionaries with differences in the keyword names, and you may want to evaluate the impact to your databases and output reports.

The file that Bay Networks provides includes the latest IETF definitions of the RADIUS protocol at the time of release; it includes all attributes and values that are needed to support Bay's RAC implementation. It is not necessary to use our definitions directly, but other dictionaries may have to be extended to cover our usage.

This file can be used as a reference to add or change existing RADIUS dictionaries as needed. Since it is in the format of some of the popular RADIUS servers, in some cases it can be used as a direct replacement. However, you should review the dependencies and make a decision on how to apply the differences.

A partial listing of the dictionary contents is shown below:

```
ATTRIBUTE       User-Name           1       string
ATTRIBUTE       Password            2       string
ATTRIBUTE       CHAP- Password      3       string
ATTRIBUTE       NAS-IP-Address      4       ipaddr
ATTRIBUTE       NAS-Port            5       integer
ATTRIBUTE       Service-Type        6       integer
ATTRIBUTE       Framed-Protocol     7       integer
ATTRIBUTE       Framed-IP-Address   8       ipaddr
<...>
```

(continued on next page)

```
#          User Service Types
VALUE      Service-Type      Login-User           1
VALUE      Service-Type      Framed-User          2
VALUE      Service-Type      Callback-Login-User  3
VALUE      Service-Type      Callback-Framed-User 4
VALUE      Service-Type      Outbound-User        5
VALUE      Service-Type      Administrative-User  6
VALUE      Service-Type      NAS-Prompt           7
VALUE      Service-Type      Authenticate-Only    8
VALUE      Service-Type      Callback-NAS-Prompt  9
<...>
#          Framed Protocols
VALUE      Framed-Protocol   PPP                  1
VALUE      Framed-Protocol   SLIP                 2
VALUE      Framed-Protocol   ARAP                 3
VALUE      Framed-Protocol   Gandalf-SL/MLP       4
VALUE      Framed-Protocol   IPX/SLIP             5
```

Configuring RAC Functions Using RADIUS

RAC functions can be configured by setting the values of RADIUS attributes on the RADIUS server. This section details what RADIUS attributes must be set to enable various RAC functions.

In the descriptions that follow, note that numbers for packet types appear in braces **{1}**, numbers for attributes appear in parentheses **(1)**, and numbers for enumerations appear in brackets **[1]**.

End User and Session Identification

The RAC provides identification information to the RADIUS server via attributes included in Access-Request packets. Access-Request packets includes the following RADIUS attributes:

- **User-Name (1)**
- **User-Password (2)**
- **CHAP-Password (3)**
- **NAS-IP-Address (4)**
- **NAS-Port-Type (61)**
- **NAS-Port (5)**
- **Service-Type (6)**
- **Framed-Protocol (7)**
- **Called-Station-Id (30)**
- **Calling-Station-Id (31)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**

Automatic Connection

RADIUS can be configured to connect a user to a specific service automatically when the user calls in. The RAC port must be in CLI mode (through the port parameter **mode** set to **cli** or **auto_detect**). The RADIUS attributes **Service-Type (6)**, **Framed-Protocol (7)**, and **Login-Service (15)** determine the service to which the user is connected.

Table 6-3 shows the services to which a user can be connected automatically based on a given value of **Service-Type (6)** and **Framed-Protocol (7)** or **Login-Service (15)**.

Table 6-3. RAC Automatic Services

Service-Type (6)	Framed-Protocol (7)	Login-Service (15)	Automatic Service
Login [1]/Callback [3]	n/a	**Telnet [0]**	telnet
Login [1]/Callback [3]	n/a	**Rlogin [1]**	rlogin
Login [1]/Callback [3]	n/a	**LAT [4]**	connect
Login [1]/Callback [3]	n/a	any except telnet, rlogin, or LAT	CLI
Framed [2]/Callback [4]	**PPP [1]**	n/a	ppp
Framed [2]/Callback [4]	**SLIP [2]**	n/a	slip
Framed [2]/Callback [4]	**ARAP [3]**	n/a	arap
Administrative [6]	n/a	n/a	CLI super-user
NAS Prompt [7]/Callback [9]	n/a	n/a	CLI

telnet/rlogin

The **Login-IP-Host (14)** attribute specifies the internet address to be connected to via telnet or rlogin. If **Login-IP-Host (14)** is set to **0xff**, the user is prompted for a host. If **Login-IP-Host (14)** is set to **0**, the user is connected to the address stored in the RAC port parameter **dedicated_arguments**. If this method is used, the **Login-TCP-Port (16)** attribute is ignored.

The **Login-TCP-Port (16)** attribute specifies the destination TCP port for the telnet or rlogin session. The default port for telnet is 23, and the default port for rlogin is 513.

LAT/connect

The **Login-LAT-Node (35)** attribute specifies the LAT node to connect to via the CLI **connect** command. Alternatively, the **Login-LAT-Service (34)** attribute can be specified to restrict the user to a particular service pool. The **Login-LAT-Port (63)** attribute can be used to specify the LAT port to connect to on the remote node or service. The **Login-LAT-Group (36)** attribute is a bit mask of the LAT groups the user will have access to.

Service Hint and Restriction

If the RAC port is not in CLI mode (whether the RAC port auto-detected a framing protocol /slave or the port was configured for framing/slave) then the user will be restricted to the profile returned by RADIUS. If the RADIUS server returns a specific **Service-Type (6)** or **Framed-Protocol (7)**, but the port is not running that service or protocol currently, the user is rejected, and the reason is logged by RADIUS Accounting.

Table 6-4 shows the required port modes or services which correspond to particular combinations of values for **Service-Type (6)** and **Framed-Protocol (7)**.

Table 6-4. RAC Port Mode/Service Restrictions

Service-Type (6)	**Framed-Protocol (7)**	Required Port Mode/Service
Login [1]/Callback [3]	any	cli
Framed [2]/Callback [4]	**PPP [1]**	ppp
Framed [2]/Callback [4]	**SLIP [2]**	slip
Framed [2]/Callback [4]	**ARAP [3]**	arap
Framed [2]/Callback [4]	unspecified	ppp, slip, or arap
Outbound [5]	any	slave
Administrative [6]	any	cli
NAS-Prompt [7]/Callback [9]	any	cli
unspecified	any	unrestricted

Dialback Services

For the Callback service types (**Service-Type (6) = Callback-Login [3]**, **Callback-NAS-Prompt [9]**), the RAC will dial back to the user with the phone number specified in the **Callback-Number (19)** attribute. If this attribute is not returned by the RADIUS server, the user is prompted for the number. Specifying the callback number in the RADIUS server is more secure than having the user provide it at the prompt.

The RAC dials back to the user on the same channel that he or she dialed in on. Dialback calls are re-authenticated (and re-authorized) as if they were new calls.

Session Timeout

You can restrict the user to a specified dial-in length using the **Session-Timeout (27)** attribute. The value of **Session-Timeout (27)** is equal to the number of seconds the user is allowed to be dialed-in before the RAC unilaterally terminates the user's session. This feature is identical to the *max_logon* feature in ACP.

Idle Timeout

You can use the **Idle-Timeout (28)** attribute to time out the user's session once the corresponding port stops receiving or transmitting data. The value of the **Idle-Timeout (28)** is equal to the number of seconds the session can be idle before the RAC unilaterally terminates the session. This feature is identical that provided by the RAC port parameter *inactivity_timer.*

CLI Scripting

The user can be configured through RADIUS to execute a CLI script upon gaining access. This feature uses the **Annex-CLI-Command (VSA Bay Networks 29)** attribute to specify a list of CLI commands to run, with each command in a separate attribute. The commands will be executed in the order in which they are received. Note that protocol commands (ppp, slip, arap) will end the script, even if later commands are specified. Note also that the **...** command also will end the script, and is interpreted to mean that the user remains at the NAS prompt. This feature is identical to the **clicmd** entry in the **acp_userinfo** file used with the ACP security regime.

CLI Command Filtering

Certain CLI commands can be made unavailable for the user. This feature uses the **Annex-CLI-Filter (VSA Bay Networks 30)** attribute to specify a list of CLI commands that the user will not have access to. Each command filtered off must be specified in a separate attribute. Entering commands present in this list generate the error message "CLI: Command not found". This feature is identical to the **climask** entry in the **acp_userinfo** file used with the ACP security regime.

CLI IP Host Filtering

The user can be prevented from gaining access, via *rlogin* or *telnet*, to a specific host or host-transport port combination. This is done with a list of **Annex-Host-Restrict (VSA Bay Networks 31)** and **Annex-Host-Allow (VSA Bay Networks 32)**. The values of each of these attributes is a composite string value. The first four bytes contains, in network order, the IP Address which the user should be specifically restricted from using or allowed to use. Trailing bytes that are zero are interpreted to match all values of that byte. Thus, 132.245.0.0 means everything on the 132.245.0.0 subnet, while 0.0.0.0 means every host on the entire WAN. The remainder of the string is a printable comma-delimited list or dash-delimited range of TCP or UDP ports that the user is restricted from using or allowed to use. For example, "23,101" would restrict/allow usage of ports 23 and 101, while "17-105" would restrict/allow usage of ports 17 through 105, inclusively. **This feature is identical to the acp_restrict file used with the ACP security regime.**

To determine if a user can gain access to a host-port, each attribute is processed in the order in which it was received. Processing stops when a host-port match is found. The user is restricted access if the attribute that matched was an **Annex-Host-Restrict (VSA Bay Networks 31)** attribute. Otherwise, the user is allowed access.

Raw Outbound Service

The RAC is capable of allowing raw outbound access to a RAC port via telnet. In order to use this, the user must either have no **Service-Type (6)** specified or have **Service-Type (6)** = **Outbound**. This is analogous to the slave port mode.

Framed Protocol Service

The RAC is capable of providing a dial-in user with framed protocol service (PPP, SLIP, or ARAP). In order to use this, the user must either have no **Service-Type (6)** specified or have **Service-Type (6) = Framed [2]**.

Disabling Routing

The framed protocol user can disable routing packets across the link with the **Framed-Routing (10)** attribute. The default behavior is to send and listen for routing packets across a link to a different subnet, but not a link to the same subnet. Note that since this attribute does not specify which network layer protocol or framing protocol to do this for, the RAC assumes that the attribute applies to all framing protocols (PPP, SLIP) but only IP.

IP Network Mask

When IP is running over the link, the IP network mask for the dial-in side can be configured with the **Framed-IP-Netmask (9)** attribute. In this way, the RAC can know when a packet should be forwarded across the dial-up link. The default is for the RAC to assume that the network mask of the remote link is identical to the RAC's network mask.

IP Static Route Configuration

When IP is running over the link, then static routes for that link can be configured with the **Framed-Route (22)** attribute. Note that the RAC will accept the non-standard format for this attribute used by the Nautica RADIUS server as well as the standard format.

Filtering Services

The RAC supports filtering of IP, TCP, and UDP packets.

The **Annex-IP-Filter (VSA Bay Networks 28)** attribute allows the RADIUS server to specify a packet filter in the RAC packet filter format, since RADIUS does not define a specific filter format. The RADIUS server can then be configured to upload specific packet filters based on the user profile.

The RAC also supports the **Filter-Id (11)** attribute in conjunction with the **Annex-Filter (VSA Bay Networks 28)** attribute. Upon receiving a **Filter-Id (11)** attribute, the RAC will initiate another Access-Request. The RAC will then wait for an **Access-Accept {2}** with the list of the actual filters supplied in **Annex-Filter (VSA Bay Networks 28)** attributes. This method requires the existence of a corresponding "pseudo-user" in the RADIUS server. **Nested Filter-Id (11)** attributes are not permitted.

The algorithm for application of filters is thus: the **Access-Accept {2}** packet is parsed. If an **Annex-Filter (VSA Bay Networks 28)** attribute is present, the filter is applied to the user's session. If a **Filter-Id (11)** attribute is present, then the RAC will make a "pseudo-user" request as described in the previous paragraph. The response to the "pseudo-user" request MUST contain only **Annex-Filter (VSA Bay Networks 28)** attributes; other attributes will be ignored by the RAC. Note that the order of processing the filter attributes is not important, because the order of the RAC filters is not important.

Maximum Transmission Unit

The maximum transmission unit size (from the RAC to the remote peer) can be set with the **Framed-MTU (12)** attribute, which is supported for SLIP and PPP, but not for ARAP. The value of this attribute is overridden by PPP, however, if the RAC receives a *ppp_mru* value from the remote peer. Note that **Framed-MTU (12)** must be at least 576 bytes for IPX traffic and 599 bytes for AppleTalk traffic.

Network Layer Compression Protocols

Network layer compression protocols can be configured using the **Framed-Compression (13)** attribute. Only Van-Jacobson TCP/IP header compression is supported by the RAC.

PAP

PAP works as described in RFC 2058.

CHAP

CHAP works as described in RFC 2058. Note that the RAC will send the CHAP challenge in *both* the **CHAP-Challenge (60)** attribute and the authenticator.

MP

The maximum number of MP links allowed for the user can be configured with the **Port-Limit (62)** attribute. If not specified, the default number of links allowed is one. This attribute is only recognized for the first link of an MP bundle. For subsequent links, this attribute is ignored.

L2TP

L2TP tunnels a user's PPP session over to another node where it is treated as if it were a local PPP session. L2TP is used to implement both DVS and MMP.

L2TP uses CHAP for its peer authentication. Operation of this is the same as for regular PPP CHAP but with one exception: the CHAP Identifier is set to be the low order byte of the CHAP Challenge. Thus RADIUS can be used to authenticate L2TP tunnels using its existing CHAP mechanism. The L2TP Access-Request will contain the following attributes:

- **User-Name (1)**
- **CHAP-Password (3)**
- **NAS-IP-Address (4)**
- **NAS-Port (5)** = VPN[5000] + index
- **Service-Type (6)**
- **Acct-Delay-Time (41)**
- **CHAP-Challenge (60)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**
- **Annex-Tunnel-Type (VSA Bay Networks 36)**
- **Annex-Tunnel-Medium-Type (VSA Bay Networks 37)**
- **Annex-Tunnel-Client-Endpoint (VSA Bay Networks 38)**
- **Annex-Tunnel-Server-Endpoint (VSA Bay Networks 39)**
- **Annex-Tunnel-Id (VSA Bay Networks 40)**

IPCP

The user's IP address (the remote peer's address) can be configured using the **Framed-IP-Address (8)** attribute if the *address_origin* port parameter is set to *acp* or *auth_server*. If an address is returned by the RADIUS server, then the RAC will *insist* on using that address, or it will not allow IPCP to come up. If, however, 255.255.255.255 is specified, then the RAC will allow the peer to set the address. If 255.255.255.254 is specified, then the RAC will get the address using DHCP. If the attribute is not specified, then the RAC will fall back to the *remote_address* admin port parameter.

The local port IP address can be configured using the **Annex-Local-IP-Address (VSA Bay Networks 35)** attribute if the admin *address_origin* port parameter is set to *acp* or *auth_server*. If an address is returned by the RADIUS server, then the RAC will *insist* on using that address, or it will not allow IPCP to come up. If, however, **255.255.255.255** is specified, then the RAC will allow the peer to set the address. If **255.255.255.254** is specified, then the RAC will get the address using DHCP. If the attribute is not specified, then the RAC will fall back to the *local_address* port parameter.

IPXCP

The user's IPX network number can be configured using the **Framed-IPX-Network (23)** attribute. Note that the RAC will *insist* on using the address returned by RADIUS, or it will not allow IPXCP to come up. If, however, **4294967294** is specified, the RAC will assign an address from the local port parameter *ppp_ipx_network*.

SLIP

The user's IP address (the remote peer's address) can be configured using the **Framed-IP-Address (8)** attribute if the admin *address_origin* port parameter is set to *acp* or *auth_server*. If an address is returned by the RADIUS server, then the RAC will use that address. If, however, **255.255.255.254** is specified, then the RAC will get the address using DHCP. If the attribute is not specified, then the RAC will fall back to the *remote_address* admin port parameter.

The local port IP address can be configured using the **Annex-Local-IP-Address (VSA Bay Networks 35)** attribute if the admin *address_origin* port parameter is set to *acp* or *auth_server*. If an address is returned by the RADIUS server, then the RAC will use that address. If, however, **255.255.255.254** is specified, then the RAC will get the address using DHCP. If the attribute is not specified, then the RAC will fall back to the *local_address* admin port parameter.

Challenge-Response Mechanisms

The RAC will support the standard RADIUS Challenge-Response Mechanisms through the use of the **Access-Challenge {4}** packet type and the **Reply-Message (18)** and **State (24)** attributes. That is, if the RAC receives an **Access-Challenge {4}** response, it will prompt the user for information, with the prompt as the string value of the **Reply-Message (18)** attribute. The RAC will then send a new **Access-Request {1}** packet, with the user's response encoded in the **User-Password (2)** attribute, and with the **State (24)** attribute returned unmodified. This dialog may continue indefinitely until the RADIUS server determines that the user should be allowed access or not.

Accounting

This section describes the RADIUS Accounting features that the RAC supports. Note that the RADIUS **Accounting-Request {4}** packets will include the actual values of RADIUS attributes used, and not necessarily the values returned in the **Access-Accept {2}** packet. For example, this means that for a PPP user, the **Framed-IP-Address (8)** attribute will be the address actually negotiated during IPCP startup, and not necessarily the address returned by RADIUS. Note that this address will actually be included in the **Accounting-Request {4}** packet when **Acct-Status-Type (40)** = **IPCP-Start [VSE Bay Networks 3]**, because the actual negotiated address is not known at authentication time.

Events

This section describes the events that trigger RADIUS Accounting from the RAC. Each RADIUS Accounting log will be queued for transmission, and once transmitted, re-queued for acknowledgment. When the RAC receives the corresponding Accounting-Response, the Accounting-Request log is purged. Periodically, the RAC will re-transmit the Accounting-Request logs that have been sitting on the acknowledgment queue. Once the RAC determines that it has too many logs, the RAC will start syslogging the oldest ones and purging them from memory, clearing room for newer events to be held in buffers.

User Login

The RAC will log whenever a user is granted access to the RAC. In this case, **Acct-Status-Type (40)** = **Start [1]**. The RAC will also include the following attributes, when applicable, in this log:

- **User-Name (1)**
- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Service-Type (6)**

- **Framed-Protocol (7)**
- **Login-IP-Host (14)**
- **Login-Service (15)**
- **Login-TCP-Port (16)**
- **Callback-Number (19)** - Only if this starts a dialback session
- **Class (25)**
- **Called-Station-Id (30)**
- **Calling-Station-Id (31)**
- **Login-LAT-Service (34)**
- **Login-LAT-Node (35)**
- **Login-LAT-Group (36)**
- **Acct-Delay-Time (41)**
- **Acct-Session-Id (44)**
- **Acct-Authentic (45)**
- **Acct-Multi-Session-Id (50)**
- **Acct- Link-Count (51)**
- **NAS-Port-Type (61)**
- **Login-LAT-Port (63)**
- **Annex-CLI-Command (VSA Bay Networks 29)**

User Logout

The RAC will log whenever a user's RAC session completes. In this case, **Acct-Status-Type (40)** = **Stop [2]**. The RAC will also include the following attributes, when applicable, in this log:

- **User-Name (1)**
- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Service-Type (6)** - Use Callback type if the user will be called back next, otherwise normal
- **Framed-Protocol (7)**

- **Callback-Number (19)** - Only if the user will be called back next .
- **Class (25)**
- **Acct-Delay-Time (41)**
- **Acct-Input-Octets (42)**
- **Acct-Output-Octets (43)**
- **Acct-Session-Id (44)**
- **Acct-Session-Time (46)**
- **Acct-Input-Packets (47)**
- **Acct-Output-Packets (48)**
- **Acct-Terminate-Cause (49)**
- **Acct-Multi-Session-Id (50)**
- **Acct-Link-Count (51)**
- **NAS-Port-Type (61)**

NAS Reboot Up

The RAC will log whenever the RAC has booted and has come up. In this case, **Acct-Status-Type (40)** = **Accounting-On [7]**. The RAC will include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **Acct-Delay-Time (41)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**

NAS Reboot Down

The RAC will log whenever the RAC is about to go down and reboot. In this case, **Acct-Status-Type (40)** = **Accounting-Off [8]**. The RAC will include the following attributes, when applicable in this log:

- **NAS-IP-Address (4)**

- **Acct-Delay-Time (41)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**

NAS Accounting Start

The RAC will log whenever the RAC starts RADIUS Accounting. This will happen when security is turned on and reset after initially being off. In these cases, **Acct-Status-Type (40) = Accounting-Restart [VSE Bay Networks 6]**. The RAC will include the following attributes, when applicable in this log:

- **NAS-IP-Address (4)**
- **Acct-Delay-Time (41)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**

NAS Accounting Stop

The RAC will log whenever the RAC stops RADIUS Accounting. This will happen when security is turned off and reset after initially being on. In these cases, **Acct-Status-Type (40) = Accounting-Shutoff [VSE Bay Networks 7]**. The RAC will include the following attributes, when applicable in this log:

- **NAS-IP-Address (4)**
- **Acct-Delay-Time (41)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**

User Reject

The RAC will log whenever the RAC rejects the user based on security criteria. In this case, **Acct-Status-Type (40)** = **User-Reject [VSE Bay Networks 1]**. The RAC will also include the following attributes in this log:

- **User-Name (1)**
- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Service-Type (6)**
- **Framed-Protocol (7)**
- **Class (25)**
- **Called-Station-Id (30)**
- **Calling-Station-Id (31)**
- **Acct-Delay-Time (41)**
- **Acct-Authentic (45)**
- **NAS-Port-Type (61)**
- **Annex-Product-Name (VSA Bay Networks 33)**
- **Annex-SW-Version (VSA Bay Networks 34)**

Call Start

The RAC will log whenever a 5399, 5393 or RA6300 accepts an incoming call. In this case, **Acct-Status-Type (40)** = **Call-Start [4]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Called-Station-Id (30)**
- **Calling-Station-Id (31)**
- **Acct-Delay-Time (41)**
- **NAS-Port-Type (61)**

Call Reject

The RAC will log whenever it rejects an incoming call before user authentication. In this case, **Acct-Status-Type (40)** = **Call-Reject [VSE Bay Networks 2]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Called-Station-Id (30)**
- **Calling-Station-Id (31)**
- **Acct-Delay-Time (41)**
- **NAS-Port-Type (61)**

Call Stop

∑The RAC will log whenever it detects an end to a call. In this case, **Acct-Status-Type (40)** = **Call-Stop [5]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Class (25)**
- **Called-Station-Id (30)**
- **Calling-Station-Id (31)**
- **Acct-Delay-Time (41)**
- **NAS-Port-Type (61)**

IPCP Start

The RAC will log whenever a PPP session starts up IPCP. The log will contain the negotiated IP address. In this case, **Acct-Status-Type (40)** = **IPCP-Start [VSE Bay Networks 3]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Service-Type (6)**
- **Framed-Protocol (7)**
- **Framed-IP-Address (8)**
- **Class (25)**
- **Acct-Delay-Time (41)**
- **Acct-Session-Id (44)**
- **Acct-Multi-Session-Id (50**)
- **NAS-Port-Type (61)**
- **Annex-Local-IP-Address (VSA Bay Networks 35)**

IPXCP Start

The RAC will log whenever a PPP session starts up IPXCP. The log will contain the negotiated IPX address. In this case, **Acct-Status-Type (40)** = **IPXCP-Start [VSE Bay Networks 4]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Framed-IPX-Network (23)**
- **Class (25)**
- **Acct-Delay-Time (41)**
- **Acct-Session-Id (44)**
- **Acct-Multi-Session-Id (50)**
- **NAS-Port-Type (61)**

ATCP Start

The RAC will log whenever a PPP session starts up ATCP. In this case, **Acct-Status-Type (40)** = **ATCP-Start [VSE Bay Networks 5]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)**
- **Class (25)**
- **Acct-Delay-Time (41)**
- **Acct-Session-Id (44)**
- **Acct-Multi-Session-Id (50)**
- **NAS-Port-Type (61)**

Tunnel Start

The RAC will log whenever an L2TP tunnel is established with another node. When an L2TP tunnel is established, the log will contain **Acct-Status-Type (40)** = **Tunnel-Start [VSE Bay Networks 8]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **Acct-Delay-Time (41)**
- **Annex-Tunnel-Type (VSA Bay Networks 36)**
- **Annex-Tunnel-Medium-Type (VSA Bay Networks 37)**
- **Annex-Tunnel-Client-Endpoint (VSA Bay Networks 38)**
- **Annex-Tunnel-Server-Endpoint (VSA Bay Networks 39)**
- **Annex-Tunnel-Id (VSA Bay Networks 40)**

Tunnel Stop

The RAC will log whenever an L2TP tunnel is destroyed. When an L2TP tunnel is destroyed, the log will contain **Acct-Status-Type (40)** = **Tunnel-Stop [VSE Bay Networks 9]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **Acct-Delay-Time (41)**
- **Annex-Tunnel-Type (VSA Bay Networks 36)**
- **Annex-Tunnel-Medium-Type (VSA Bay Networks 37)**
- **Annex-Tunnel-Client-Endpoint (VSA Bay Networks 38)**
- **Annex-Tunnel-Server-Endpoint (VSA Bay Networks 39)**
- **Annex-Tunnel-Id (VSA Bay Networks 40)**

Tunnel Reject

The RAC will log whenever it rejects L2TP tunnel establishment with a peer. When an L2TP tunnel is rejected, the log will contain **Acct-Status-Type (40)** = **Tunnel-Reject [VSE Bay Networks 10]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **Acct-Delay-Time (41)**
- **Annex-Tunnel-Type (VSA Bay Networks 36)**
- **Annex-Tunnel-Medium-Type (VSA Bay Networks 37)**
- **Annex-Tunnel-Client-Endpoint (VSA Bay Networks 38)**
- **Annex-Tunnel-Server-Endpoint (VSA Bay Networks 39)**
- **Annex-Tunnel-Id (VSA Bay Networks 40)**

MP Start

The RAC will log whenever an MP bundle is created. For MMP, this will only be logged on the LNS. In this case, **Acct-Status-Type (40)** = **MP-Start [VSE Bay Networks 13]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)** = **MP [6000]** + index
- **Acct-Delay-Time (41)**
- **Acct-Multi-Session-Id (51)**
- **NAS-Port-Type (61)** = **Virtual [5]**

MP Stop

The RAC will log whenever an MP bundle is destroyed. For MMP, this will only be logged on the LNS. In this case, **Acct-Status-Type (40)** = **MP-Stop [VSE Bay Networks 14]**. The RAC will also include the following attributes, when applicable, in this log:

- **NAS-IP-Address (4)**
- **NAS-Port (5)** = **MP [6000]** + index
- **Acct-Delay-Time (41)**
- **Acct-Multi-Session-Id (51)**
- **NAS-Port-Type (61) = Virtual [5]**

Time Stamps and Session Duration

Each RADIUS Accounting record is queued with a time-stamp of when the event occurred. Whenever a RADIUS Accounting-Request is issued by the RAC, the RAC will record the difference in time (now - occurrence) and place the result in the **Acct-Delay-Time (41)** attribute.

Each session in the RAC retains a time-stamp of the start of the session. When the session ends, the RAC will record the difference in time (finish - start) and place the result in the **Acct-Session-Time (46)** attribute.

Session Throughput

Each session in the RAC will keep track of the packet and byte throughput of each session associated with a physical line. When the session ends, the RAC will record these statistics and place the results in the **Acct-Input-Octets (42)**, **Acct-Output-Octets (43)**, **Acct-Input-Packets (47)** and **Acct-Output-Packets (48)** attributes.

Session Tagging

Each session in the RAC has a unique Session Identifier. This identifier is an eight-digit upper case hexadecimal number. For the initial session, the first four digits will be random, the next three digits will be zero, and the final digit will be one. Subsequent sessions will increment the previously used session id as its own. This identifier will be placed in the **Acct-Session-Id (44)** attribute.

Each MP bundle will also have a unique MP Bundle Identifier equal to the Session Identifier of the first link of the bundle. This identifier will be placed in the **Acct-Multi-Session-Id (50)** attribute.

Multi-Session Link Count

Each MP session records the number of links it has used in the **Acct-Link-Count (51)** attribute.

Authentication Method

RADIUS Accounting only logs users that are authenticated via RADIUS. This means that **Acct-Authentic (45)** = **RADIUS [1]** in each **Accounting-Request {4}** packet when **Acct-Status-Type (40)** = **Start [1]**.

Termination Reason

Termination reason reporting is supported by the **Acct-Terminate-Reason (49)** attribute. Refer to the Remote Access Concentrator Software Reference for a complete description of this attribute.

Access State

The RAC supports the RADIUS standard way to preserve RADIUS server state from the **Access-Accept {2}** to the **Accounting-Request {4}** packets. That is, if the RAC is delivered a **Class (25)** attribute in the **Access-Accept {2}**, then the RAC will echo the attribute in its **Accounting-Request {4}** packet. In this way, the RADIUS server can maintain a state relationship from when it granted access to when it get recorded access.

Configuring ACP Security

Guidelines for Creating ACP-Related Files

To create ACP-related files for use with the RAC that will not impede its operation, observe the following guidelines:

- Do not specify ports by number, range of numbers or groups of numbers (e.g., **ports=1-10**) in security profiles.
- Do specify ports, when appropriate, by port type (e.g., **ports=ta**) in security profiles.
- Do not specify port passwords within the **acp_passwd** file.
- Use the correct RAC name or IP address when specifying a RAC within the **acp_dialup** file but do not specify ports by number, range, or group.

Information for Users of Remote Annexes

Read this section thoroughly if RACs are installed in an environment with Remote Annexes.

If you are using Remote Annex products, do not use your existing ACP-related files with the RAC without modification.

Using your existing ACP-related files with the RAC, modifying them only to specify the RAC's IP address (when required), affects the operation of the RAC in the following ways:

- Specified ports (e.g., **ports=1**), port ranges (e.g., **ports=1-10**), and port groups (e.g., **ports=1,3,6,7,12,17**) within profile criteria of your existing ACP-related files will be recognized and applied by the RAC to internal RAC asynchronous ports (**asy** ports).

  However, the RAC ignores ports specified by number, range of numbers, or group of numbers for internal terminal adapter (**ta**) ports and synchronous PPP (**syn**) ports and applies profile criteria to all RAC **ta** and **syn** ports.
- Port passwords specified in the **acp_passwd** file affects RAC operation negatively since the passwords are matched to the equivalent RAC internal asynchronous port. As a result, asynchronous modem connections made to a RAC are subject to random password associations (i.e., each prompt asking for a different password).

Editing Existing ACP Files

When editing existing ACP files for use with the RAC, keep in mind that ACP profile criteria cannot be port specific and that you need to modify your existing files accordingly. For example, an **acp_userinfo** file user-end block that uses **ports=** for an RA 4000 on a per-port basis, as shown below, should be modified for the RAC:

```
user group=eng;ports=1-5
climask telnet end
end
```

Modifying the file as shown applies the **climask** to all modem connections on any server, not just ports 1-5:

```
user group=eng;ports=asy
climask telnet end
end
```

If the former were used in an environment with the RAC, the **climask** would inconsistently apply to modem connections because the dynamic allocation of ports is not constrained to ports 1-5.

About Host-Based Security

The Access Control Protocol (ACP) provides host-based security in which a UNIX or Windows NT host on the network is defined as a security server. You can modify the host-based software to implement a security policy that fits the needs of your environment.

Overview of Host-Based Security

ACP security has three requirements: 1) at least one UNIX or Windows NT host on the network must act as a security server running RAC security software; 2) security must be enabled on the RAC (the **enable_security** parameter is set to **Y**); and 3) a security regime, such as **acp** or **securid**, must be defined for authenticating RAC users.

The security server maintains a database of files that reside by default in the directory **/usr/annex**. These files include:

- **acp_keys** (encryption key information).
- **acp_dialup** (user names and addresses for dial-up connections).
- **acp_group** (user-group associations for security).
- **acp_regime** (security authentication system and associated password file name).

- **acp_passwd** (security passwords).

  

  Do not specify port passwords for the RAC.

- **acp_userinfo** (initial login environment and start-up CLI commands).
- **acp_restrict** (restricted hosts and host ports).
- **acp_logfile** and **acp_logfile.*Annex_IPaddress*** (security audit trails).

  

  The contents of these files should match on all security servers (except for **acp_logfile**).

The following sections describe these aspects of ACP security:

- *Basic ACP Configuration on page 6-52*.
- *Encrypting Security Messages on page 6-59*.
- *Defining Security Profiles on page 6-61*.
- *Using include Files in the acp_userinfo File on page 6-99*.
- *Dynamic Allocation of Network Addresses on page 6-142*.
- *Using AppleTalk Security on page 6-133*.
- *Using IPXCP Security on page 6-135*.
- *Using PPP Security on page 6-135*.
- *Using Kerberos Authentication on page 6-112*.
- *Using the ACE/Server on page 6-127*.
- *Using SafeWord AS Security on page 6-119*.
- *Configuring Security for the RAC FTP Daemon on page 6-152*.
- *Configuring the IP Basic Security Option (IPSO) on page 6-126*.
- *Logging Security Events on page 6-153*.
- *The parser of the acp_userinfo file generates log messages if an error is detected when processing a user's profile. on page 6-154*.

Basic ACP Configuration

This section outlines procedures for configuring the basic ACP features and describes what happens in each case if ACP goes down.

CLI Security

You can set up host-based security for CLI connections in which users must provide a valid user name and password before they are granted access to a CLI:

1. **Set the** cli_security **parameter to** Y**, so that the RAC uses ACP.**
2. **Define a security server using the** pref_secure1_host**,** pref_secure2_host**, or** security_broadcast parameter **(see *Configuring the Security Server on page 6-57*).**
3. **Create entries in the** acp_regime **file defining the authentication systems to be used and the conditions under which to use them.**

   The install program creates the **acp_regime** file, then prompts you for a default regime and (in some cases) a password file name, which it enters into **acp_regime**. Subsequently, you can add to and/or change the contents of this file. (See *Configuring the acp_regime File on page 6-70*.)
4. **Create entries in the appropriate password files (see *Creating User Password Files on page 6-72*).**
5. **(Optional) Configure encryption for security messages (see *Encrypting Security Messages on page 6-59*).**

If ACP is down, the RAC acts as follows:

- First, the RAC prompts for the password specified in the **port_password** parameter. If the **port_password** parameter is not set (**“<unset>”**), the RAC does not connect the user to the CLI.

- If the **cli_security** parameter is set to **N** and the **port_password** parameter is set (**“<set>”**), the RAC prompts for the password specified in **port_password**.
- If **cli_security** is set to **N** and the **port_password** parameter is not set (**“<unset>”**), the RAC does not perform a security check for CLI connections and allows unrestricted access to the CLI.

Virtual CLI Security

You can set up host-based security for virtual CLI (VCLI) connections in which users must provide a valid user name and password before they are granted access to a virtual CLI:

1. **Set the** vcli_security **parameter to** Y **so that the RAC will use ACP.**
2. **Define a security server using the** pref_secure1_host**,** pref_secure2_host**, or** security_broadcast **parameter (see *Configuring the Security Server on page 6-57*).**
3. **Create entries in the** acp_regime **file defining the authentication systems to be used and the conditions under which to use them.**

   The install program creates the **acp_regime** file, prompts you for a default regime and (in some cases) password file name, and then enters them into **acp_regime**. Subsequently, you can add to and/or change the contents of this file (see *Configuring the acp_regime File on page 6-70*).
4. **Create entries in the appropriate password files (see *Creating User Password Files on page 6-72*).**
5. **(Optional) Configure encryption for security messages (see *Encrypting Security Messages on page 6-59*).**

If ACP is down, the RAC acts as follows:

- First, the RAC prompts for the password specified in the **vcli_password** parameter (see *Implementing Local Virtual CLI Password Protection on page 6-2*).
- If the **vcli_password** parameter is not set (**"<unset>"**) and the **vcli_security** parameter is set to **N**, the RAC prompts for the password specified by the **password** parameter.
- If the **password** parameter is not set (**"<unset>"**), the RAC fails the VCLI attempt.
- If the **vcli_security** parameter is set to **N** and the **vcli_password** parameter is set (**"<set>"**), the RAC prompts for the password specified in **vcli_password**.

Connection Security

You can authorize or deny IP or CLI access to specific hosts, host ports, or networks for a particular user, group, time of day, or protocol in use.

1. **Define a security server using the** pref_secure1_host**,** pref_secure2_host**, or** security_broadcast **parameter (see *Configuring the Security Server on page 6-57*).**
2. **Set the** connect_security **parameter to** Y**, so that the RAC uses ACP on a CLI connection (via** telnet **and/or** rlogin**).**
3. **Configure the** acp_restrict **file on the security server (see *Limiting Access to Hosts via acp_restrict on page 6-94*).**

   For CLI **telnet** or **rlogin** connections, ACP checks **acp_restrict** to see whether or not access should be granted to the user. For SLIP and IP over PPP connections, the **acp_restrict** file controls access by creating filters based on your input.
4. **(Optional) Configure encryption for security messages (see *Encrypting Security Messages on page 6-59*).**

SLIP and PPP Security

Configure access to a SLIP or PPP link from the RAC as follows:

1. **Set the** mode **parameter to** cli **and have the user issue the** slip **or** ppp **command from the CLI.**

   If the **mode** parameter is set to **slip**, the RAC does not perform a security check.

2. **If you want authentication performed on the CLI connection (rather than authenticating when the user issues the** slip **or** ppp **command), set the** slip_ppp_security **and** cli_security **parameters to** Y**. Then proceed to step 4.**

3. **To have authentication performed when the user issues the** slip **or** ppp **command (rather than authenticating when the CLI connection is made) set the** cli_security **parameter to** N **and the** slip_ppp_security **parameter to** Y**.**

4. **Define a security server using the** pref_secure1_host**,** pref_secure2_host**, or** security_broadcast parameter **(see *Configuring the Security Server on page 6-57*).**

5. **Create entries in the** acp_regime **file defining the authentication systems to be used and the conditions under which to use them.**

   The install program creates the **acp_regime** file, then prompts you for a default regime and password file name, which it enters into **acp_regime**. Subsequently, you can add to and/or change the contents of this file. (See *Configuring the acp_regime File on page 6-70*.)

6. **Create entries in the appropriate password files (see *Creating User Password Files on page 6-72*).**

7. **(Optional) Configure encryption for security messages (see *Encrypting Security Messages on page 6-59*).**

If ACP is down, the **slip** or **ppp** command fails.

The RAC never uses local security with the **slip** or **ppp** command.

Port Server Security

You can set up security for port servers in which users must provide a valid user name and password before they are granted access to an outgoing port:

1. **Set the** port_server_security **parameter to** Y **so that the RAC will use ACP.**
2. **Define a security server using the** pref_secure1_host**,** pref_secure2_host**, or** security_broadcast **parameter (see *Configuring the Security Server on page 6-57*).**
3. **Create entries in the** acp_regime **file defining the authentication systems to be used and the conditions under which to use them.**

   The install program creates the **acp_regime** file, then prompts you for a default regime and password file name, which it enters into **acp_regime**. Subsequently, you can add to and/or change the contents of this file. (See *Configuring the acp_regime File on page 6-70*.)
4. **Create entries in the appropriate password files (see *Creating User Password Files on page 6-72*).**
5. **(Optional) Configure encryption for security messages (see *Encrypting Security Messages on page 6-59*).**

If ACP is down, the RAC acts as follows:

- If the **port_server_security** parameter is set to **Y**, the RAC prompts for the password specified in the **port_password** parameter.

- If the **port_password** parameter is not set (**“<unset>”**), the RAC fails the port connection attempt.
- If the **port_server_security** parameter is set to **N** and the **port_password** parameter is set (**“<set>”**), the RAC prompts for the password specified in **port_password**.
- If the **port_server_security** parameter is set to **N** and the **port_password** parameter is not set (**“<unset>”**), the RAC does not perform a security check for port connections.

Configuring the Security Server

The ACP security server software is provided as part of the expedited remote procedure call daemon (**erpcd**) software. Included with the software is the **eservices** file that has two entries: one for the block file server (**bfs**) and one for ACP.

The **erpcd** process must be running; **erpcd** requires the **/etc/services** file to have an entry for *erpc 121/udp*.

Setting Up a Security Server

To set up a security server, you must install the file server software on a host and delete the # symbol in front of the ACP entry in the **eservices** file. For example:

```
# erpcd remote programs
#
# prog no. verlo       verhi     name
#
  1        0           99        bfs
  3        0           99        acp
```

Specifying the Security Hosts

The **pref_secure1_host** and **pref_secure2_host** parameters specify the preferred security hosts. The RAC first queries the **pref_secure1_host** for user validation. If a response is not received within the time defined in the **network_turnaround** parameter, the RAC repeats the query several times. If the RAC still does not receive a response, it queries the host defined in the **pref_secure2_host** parameter. If a response is not received from the second security host within the allowable time limit, and the **security_broadcast** parameter is set to **Y**, the RAC broadcasts to the network for another host with **erpcd** running to authorize the access request. If the **security_broadcast** parameter is set to **N**, the RAC denies the authentication request.

The **network_turnaround** parameter specifies the amount of time in seconds in which the RAC expects a response from the security servers. To reduce the possibility of a retry, the network turnaround time should be long enough to allow for a network transmission to the security server and transmission back to the RAC; unfortunately, if this period of time is too long, the RAC will attempt multiple retries before sending a query to the second security server.

Disabling Broadcasting for Security Servers

The RAC broadcasts to the network for a security server if:

- The **security_broadcast** parameter is set to **Y**.
- The **pref_secure1_host** and **pref_secure2_host** parameters do not respond.

Setting the **security_broadcast** parameter to **N** disables RAC broadcasting. If the hosts defined in the **pref_secure1_host** and **pref_secure2_host** parameters do not respond, the RAC refuses the connection request.

Encrypting Security Messages

Messages between the security server and the RAC are encrypted if the RAC parameters **enable_security** and **acp_key** are set. The parameters do not take effect until the RAC is either rebooted or RAC security is reset.

The **acp_key** parameter specifies the encryption key the RAC uses to exchange messages with the security server. The security server maintains the encryption key for each RAC in the **acp_keys** file (see *Creating the acp_keys File on page 6-59*).

The encryption key also validates the security host: the host must know the RAC's ACP key for the RAC to consider the host valid. Without the appropriate key, the RAC denies the user's request even if the host is defined as a preferred security host.

The **show annex** command does not display the value of the **acp_key** parameter. Instead, it displays **"<set>"** or **"<unset> "**.

Creating the acp_keys File

The security server maintains the encryption key for each RAC in the **acp_keys** file. Each entry in this file contains a list of RAC names or IP addresses separated by commas and an encryption key for those RACs. The RAC or the list of RACs and the key are separated by a colon. The order of placement in the file is important, as the file is read sequentially.

When the security server receives an encrypted message from the RAC, the server tries to match that key against the key assigned to the RAC in the file. If no match exists, the RAC and the server cannot communicate.

The syntax rules for the **acp_keys** file are:

- Any part of an IP address in the list can be specified with an asterisk (*).
- A backslash (\) is used to continue a line.

- Any ASCII character except spaces and tabs are valid encryption keys (keys are case sensitive).
- Each key can contain a maximum of fifteen characters.

RACs with no entries are assumed to have no key set. Since wildcards are valid, some entries in the file may require an explicit “no key” declaration:

```
annex01, annex02:   seKret2
#131.21 net Annexes have the same key except for 3 Annexes
131.21.2.1, 131.21.2.2:
131.21.1.1: SpeciaL
131.21.*: Gub-Net
```

In the following example, the first three entries specify *insomniac-1* as the key for the RAC whose IP address is 132.245.6.15, no encryption for the RAC whose IP address is 132.245.6.75, and *Piano* as the key for all other RACs on the 132.245.6 subnet. The last entry specifies *gl12ch* as the key for *annex01*, *annex02*, and *annex03*. Each **acp_key** parameter for the RACs listed in the example must be identical to the key included in the **acp_keys** file.

```
132.245.6.15:insomniac-1
132.245.6.75:
132.245.6.*:Piano
annex01,annex02,annex03:gl12ch
```

Changing the value of the **acp_key** parameter on any RAC requires the same change to the **acp_keys** file on the security server. The recommended order for changing the ACP encryption key on a RAC is:

1. **Edit the** acp_keys **file on all security server hosts.**
2. **Change the value of the** acp_key **parameter for all affected RACs.**
3. **Update the cache by sending the** erpcd **on all security server hosts a HUP signal with** kill**.**

   `kill -HUP` `<pid_number>`
4. **Reset the security subsystem for all affected RACs using the** na **command** reset annex security**.**

Defining Security Profiles

The expedited remote procedure call daemon (**erpcd**) that implements ACP permits you to define different security profiles for different users, groups of users, or for other *connection conditions*, such as the time of day or the date. Specifically, you can use the **acp_regime**, **acp_userinfo**, and **acp_restrict** files to create diverse security profiles based on any combination of the *profile criteria* shown in Table 6-5.

Table 6-5. Profile Criteria

Criterion	Description
username	The user's userid.
group	The name of a group to which the user belongs, as defined in the **/etc./groups** or **acp_group** file; see *Creating User Groups on page 6-69*.
time	The day of the week and/or the time of day.
protocol	The connection protocol (e.g., PPP, CLI).
annex	The name or IP address of the RAC on which the connection is made.
port type	The RAC internal port type on which the connection is made.

Overview of Security Profile Criteria

Security profile criteria specify the connection conditions that must be met in order for the RAC to:

- Use a particular security regime for authentication (in **acp_regime**).
- Define the user environment that will be in effect upon login (in **acp_userinfo**).
- Permit or restrict access to hosts or host ports (in **acp_restrict**).

Together, the security regime, user environment, and host access restrictions define the security profile.

A profile criterion begins with one of the keywords listed in Table 6-5 on page 6-61. The keyword is followed by an = sign, which is followed by a value. No space is permitted before or after the = sign. The syntax is:

keyword=value

To enter more than one criterion, separate the criteria with semicolons(;). Keep the criteria on one line. Use the backslash (\) continuation symbol to extend the line beyond the right margin, if necessary. No spaces are allowed on either side of the semicolon or within the *value* field, with the exception of data for the **time** criterion (see *Time on page 6-65*). A particular keyword may appear only once in a line of criteria. An entire line of profile criteria is called a *profile criteria specification*. The following examples could appear in any or all of the **acp_regime**, **acp_userinfo**, and **acp_restrict** files.

For the **acp_regime** and **acp_restric**t files, the entry looks like this:

```
username=chris;time="9:00am-10:30pm Monday-Friday";annex=annex03
```

For the **acp_userinfo** file, the entry looks like this:

```
user username=chris;time="9:00am-10:30pm Monday-Friday";annex=annex03
```

A profile criteria specification cannot exceed 80 characters.

After the profile criteria specification, you specify the security measure(s) to be applied if the criteria are met. The following is an example from the **acp_userinfo** file:

```
user username=chris;time="9:00am-10:30pm Monday-Friday";annex=annex03
climask ppp end
end
```

When user *chris* connects to *annex03*, **erpcd** records all the conditions related to the connection - the user id and any group associations (as defined in the **acp_group** or **/etc/group** file), the RAC and port that *chris* connects to, the time of connection, and the connection protocol - CLI, PPP, or SLIP. **erpcd** saves these connection conditions for comparison with profile criteria specifications in the **acp_regime**, **acp_userinfo**, and **acp_restrict** files.

All of the profile criteria in a specification must be met in order for **erpcd** to consider that the specification *matches* the connection conditions recorded. The specification for Chris is considered a match if he gives a user name of *chris* and connects to *annex03* between 9:00 A.M. and 10:30 P.M Monday through Friday. If all of these criteria are met, user *chris* is prevented (via the *climask* entry) from issuing the **ppp** command while he is logged into *annex03*.

Note that, in the example cited, the profile criteria specification replaces the **user name** field in **acp_userinfo**. The **user** field can be specified instead, for compatibility with earlier RAC releases. For more information, see *Creating the acp_userinfo File on page 6-74*.

One Match Per File

You can enter an unlimited number of profile criteria specifications in each of the **acp_regime** and **acp_userinfo** files. However, for any single set of connection conditions, **erpcd** uses only the first matching specification it finds in each file. Consequently, the placement of profile criteria specifications is important. For example, suppose that user *chris* belongs to a group named *engineering* and that the first line in **acp_regime** specifies that the *engineering* group should be authenticated via Kerberos, while the second line specifies that user *chris* should be authenticated by SecurID. The result is that *chris* is authenticated by Kerberos, since a match for the group entry is found first.

The first-match algorithm is also true for **acp_restrict** entries that apply to CLI (**telnet** and **rlogin**) connections. However, **acp_restrict** entries for PPP and SLIP are treated differently (see *Limiting Access to Hosts via acp_restrict on page 6-94*).

The Resulting Security Profile

Once **erpcd** has found all the matching profile criteria in **acp_regime**, **acp_userinfo**, and **acp_restrict** (using the one-match-per file rule where appropriate) for a given set of connection conditions, the result is a single security profile.

Profile Criteria Syntax

The following sections give the purpose and syntax for each of the different criteria you can include in a profile criteria specification. Additional information and examples are supplied in the sections on **acp_regime**, **acp_userinfo**, and **acp_restrict**.

Username and Group Criteria

The **username** criterion lets you control security based on the RAC User ID (the name the user specifies at login). The **username** criterion supports user IDs up to 128 characters long.

The **group** criterion lets you control security based on a user's membership in a group. You assign users to groups via either the **acp_group** file or the **/etc/group** file (see *Creating User Groups on page 6-69*). When a **group** profile criterion is specified, **erpcd** checks the **acp_group** file to find the users belonging to the group. If it cannot find an **acp_group** file, **erpcd** looks in the **/etc/group** file.

A wildcard (*) can be used to represent as many of the final characters in a **user name** or **group** as can be removed and still leave the name unique. The following are examples of **user name** and **group** criteria:

```
username=fritz
username=fri*
username=frank
username=fra*
group=finance
group=fi*
group=fun
```

The following designates all users:

```
username=*
```

Time

The **time** criterion lets you control security based on the day of the week, the date, and the time of day. The following are the four possible syntaxes:

time="*day***"**
time="*time1***-***time2 day1***-***day2***"**
time="*time1 day1* **-** *time2 day2***"**
time="*time1 date1* **-** *time2 date2***"**

Enclose the **time** criterion in quotation marks and specify the arguments as follows:

- For *day*, specify a weekday, e.g., Sunday or Monday. The time criterion will apply to that entire day. Weekday specifications observe minimum uniqueness and are not case sensitive.
- For *time1*, specify the beginning of a time range; for *time2*, specify the end of a time range. Use *hh***:***mm*[**am**|**pm**], where *hh* is the hour and *mm* is the minutes, as the format for each end of the range. If you do not include **am** or **pm**, the RAC assumes you are using military (24-hour) notation. Both ends of a range must use the same type of notation - you cannot use military time for one part of a range and **am** or **pm** for the other.

  To indicate midnight, specify either 12:00am or 00:00. Specify noon as 12:00pm or 12:00. To indicate a 24-hour range, use either 00:00 - 23:59 or 12:00am - 11:59pm.

  Be sure to include the colon and minutes (**:***mm*) after the hour (*hh*). For example, 9:00am - 5:00pm is valid; 9am - 5pm is not.

You cannot specify time ranges without also specifying either a range of days or a range of dates. A time range with only a single day or date is not permitted. For example, *time= 9:00am - 5:00pm Sunday* is invalid. The correct usage would be *9:00am - 5:00pm Sunday - Sunday.*

- For *date1*, specify the beginning of a month and day range, e.g., January15, February10; for *date2* specify the end of a month and day range. Month specifications observe minimum uniqueness and are not case sensitive.

The following are examples:

```
time=”9:00am-5:00pm Monday - Friday”
time=”9:00-22:00 Sunday - Sunday”
time=”Wed”
time=”8:00AM Friday - 6:35PM Friday”
time=”10:30 Nov 30 - 21:30 Nov 31”
```

The **time** criterion applies to initial access by the user. For instance, in the first example above, the criterion is met if the user logs in at any time between 9:00 A.M. and 5:00 P.M. on Monday through Friday of any week in any month.

annex and port type

The **annex** and **port type** criteria let you control security based on the RAC and RAC port type that the user tries to access. You can use an asterisk (*) symbol as a wild card in place of a RAC name or the host port of a RAC IP address. The following are valid **annex** and **port type** specifications:

```
annex=5399rac03;ports=asy
ports=syn
annex=192.17.5*
annex=*
```

The first example specifies all internal **asy** ports (asynchronous ports) on *rac5399_03*.

In the second example, *annex*=* is implied and the user can access all internal **syn** ports (synchronous PPP Ports) on all RACs.

In the third and fourth examples, *ports*=* is implied. The fourth example specifies all ports on all RACs, which is the default.

You cannot abbreviate the **ports** keyword.

Protocol

The **protocol** criterion lets you control security based on the protocol used to attempt access to a host or host port. Valid values are:

- **slip**
- **ppp**
- cli (for telnet and rlogin)

Specify a protocol criterion using the syntax:

protocol=*protocol_name*

To specify more than one protocol, you must specify multiple security profile definitions. For example, to specify both PPP and SLIP, enter:

```
protocol=ppp
protocol=slip
```

The default is any protocol.

Overview of Files Used to Define Security Profiles

Following are the files you use to define security profiles:

- **acp_group** or **/etc/group**. If you intend to assign different security profiles to different groups of users, you must first define the groups in the **acp_group** or **/etc/group** file.
- **acp_regime**. An initial **acp_regime** file is created by the RAC **install** program. It is based on answers you supply to prompts from **install,** and it contains a single authentication scheme, such as **acp**, to be used for authenticating all RAC users. It also contains the name of a password file, if the regime is **acp** or **kerberos**.

  You can modify the initial **acp_regime** file so that different authentication schemes are used when particular criteria are met.

  

  Do not confuse ACP, the RAC's Access Control Protocol that controls all host-based security, with **acp**, one of several authentication systems (regimes) that can be used with ACP.

- **acp_userinfo**. This file allows you to configure login environments based on a single user id or one or more profile criteria. Configurable aspects of login environments include CLI commands to be executed at start-up, CLI commands not permitted during a login session, filter and route definitions, a CHAP secret token, and various AppleTalk session characteristics. You can also use **acp_userinfo** to deny login access.
- **acp_restrict**. You can use this file to restrict access to hosts or host ports based either on the RAC that attempts the connection or on specified access criteria.

The following sections describe these files in detail.

Creating User Groups

One of the most useful aspects of customizing security is the ability to apply different access rights and restrictions to different groups of users. To associate individual users with one or more groups, you create entries in the **/etc/group** or **acp_group** file.

The **/etc/group** file already exists on most UNIX systems, so you may prefer to add entries to this file rather than entering them in **acp_group**, which you must create. Either file must reside in the install directory (default is **/usr/annex**) on the UNIX security host.

The **erpcd** process looks for **acp_group** first, only using **/etc/group** if it cannot find **acp_group**. To designate that **erpcd** should use **/etc/group** rather than **acp_group**, see *Changing the Expected File Names Used by ACP on page 6-102*.

The **acp_group** file must have the same format as the **/etc/group** file. The following systems do not support the **acp_group** file: Ultrix, FreeBSD, and BSDI. On these systems, you must use the **/etc/group** file.

An **/etc/group** or **acp_group** file contains a one-line entry for each group. To retain compatibility with **/etc/group**, the **acp_group** file includes passwords and group ID fields, although ACP does not use them. You must specify a value in each of these two fields, although what you choose to enter is arbitrary. The format for an ACP group entry in **acp_group** or **/etc/group** is:

groupname:*password*:*groupid*:*userlist*

The *groupname* field specifies the name of the group; the *userlist* field is a comma-separated list of user names belonging to the group. There is no arbitrary limit to the number of names in *userlist*. Fields are separated by the colon (:) character.

Following are two sample **acp_group** (or **/etc/group**) entries:

```
accounting:p:g:kim,herbert,sam,louise,bill
engineering:p:g:dilbert,jim,sharon,scott,john,liza,\
carrie,edna,dena,caroline,marsha,sue,don,phil,eric,\
dan,fritz,jeremiah,amy
```

The *p* and *g* in the previous examples are place holders for the values that UNIX requires but ACP ignores.

The second example above is one long line. Although word wrapping may occur, the vi editor will see it as one line of input.

Configuring the acp_regime File

The initial security regime that the RAC uses to authenticate all users is defined in the **acp_regime** file. This file is created the first time the network administrator runs the RAC install program. The program prompts for a security regime and, if the regime requires it, a password file name. The regimes from which you can choose are **acp**, **securid**, **safeword**, **kerberos**, **native**, and **none** (see Table 6-6). Password files are required for **acp** and **kerberos**; the defaults are **acp_passwd** and **/temp/tkt_erpcd_**. Both the **acp_regime** file and the password files (if any) must be stored in the installation directory, which defaults to **/usr/annex**.

The **acp_regime** file created by **install** has the following format:

:*initial_regime*[:*initial_password_file*]

Once the **acp_regime** file has been created by the **install** program, you can modify the file to specify more than one regime and to include profile criteria that determine the conditions under which different regimes are used. The syntax for an **acp_regime** entry is:

[*profile_criteria*]**:***regime*[**:***password_filename*]

Each field in the entry must be separated by the colon (:) character; a space may follow but not precede the colon. The syntax for *profile_criteria* is:

keyword=value[**;***keyword=value*;...]

If profile criteria are omitted, the specified regime applies to any user logging in under any circumstances. Profile criteria are explained in *Overview of Security Profile Criteria on page 6-61*. Valid regimes are explained in Table 6-6.

The *password_filename* field is valid only for the **acp** and **kerberos** regimes. If you specify the **acp** or **kerberos** regime but supply no *password_filename*, the default is used (see Table 6-6). If the file is not found, an error message is logged and access is denied.

Table 6-6. Authentication Regimes

Regime	Description
acp	ACP authentication, using the password file you specify. Default is the **acp_passwd** file.
safeword	SafeWord authentication.
kerberos	Kerberos authentication, using the ticket-directory prefix you specify. Default is **/temp/tkt_erpcd_**.
native	Authentication via the native operating system of the security server. For UNIX, native means the **/etc/passwd** file is used for authentication.
none	No authentication is performed; the user is unconditionally authenticated.
securid	SecurID authentication.

The following is a sample **acp_regime** file:

```
username=jack;time="9:00am-10:00pm Tuesday - Thursday":securid
group=finance:acp:special_acppw
:acp
```

Given this sample, **erpcd** uses SecurID to authenticate user *jack* if he logs in between 9:00 A.M and 10:00 P.M. on the specified day.

Next, **erpcd** looks in the **acp_group** (or **/etc/group**, if **acp_group** does not exist) file to find the members of the group named *finance*. If one of these users tries to log in at any time on any day or date, **erpcd** attempts to authenticate that user via the **acp** regime, using the **special_acppw** password file (which must reside in the RAC install directory). Even if user *jack* is defined in *finance*, if he logs in between 9:00 A.M. and 10:00 P.M., **erpcd** nevertheless tries to authenticate him via SecurID, since the profile criteria specification that begins with **username** is matched first.

Finally, any users whose login characteristics do not match the first two profile criteria specifications are authenticated via ACP, using the default password file, **acp_passwd**.

Creating User Password Files

Password Files for the acp Regime

If the security regime defined is **acp**, **erpcd** prompts the user for a user name and password. The RAC validates this information by instructing the security server to compare these entries against entries in the password file specified in **acp_regime**. If no password file is specified, ACP uses **acp_passwd**, which must reside in the install directory (default is **/usr/annex**). In either case, if a match is found, the user is granted access; otherwise, the user is denied access.

A typical session looks like this:

```
Annex Command Line Interpreter * Copyright 1988, 1997 Bay Networks
Checking authorization, Please wait...
Annex username: kate
Annex password:
Permission granted
annex:
```

The **acp_passwd** file uses the same format as the **/etc/passwd** file. The easiest way to create this password file is to copy the **/etc/passwd** file to **acp_passwd**. One advantage to creating the **acp_passwd** file this way is that you can merge **/etc/passwd** files from different hosts into one file on the security server, thus allowing you to create a network-wide password file.

If you are using a System V.4 or V.5 host, use the **/etc/shadow** file rather than the **/etc/passwd** file.

Not all password files work with ACP. For example, you can not merge SCO UNIX password files into the **acp_passwd** file.

Non-superusers can change their passwords only if the *username* in the **acp_passwd** file matches the *username* in the **/etc/passwd** (or **/etc/shadow**) file on the host.

After creating this entry, use the **ch_passwd** command to enter the port password:

```
% ch_passwd 132.245.33.11.1
New password: <password>
```

This port password is independent of the port parameter **port_password**. The port parameter is used only for local security.

The ACP prompts appear as follows:

```
Annex username:
Annex password:
Port password:
```

Password File for the Kerberos Regime

If **kerberos** is defined in **acp_regime**, **erpcd** validates the user name and password by comparing them to entries in the password file specified in **acp_regime**. If no password file is specified, **erpcd** looks for **/temp/tkt_erpcd_** in the install directory (default is **/usr/annex**). If **erpcd** does not find a match in that file, the user is denied access to the RAC. For more information, see *Using Kerberos Authentication on page 6-112*.

Password Files for Other Regimes

For information on passwords used with third-party systems other than Kerberos, see the following sections:

- *Using the ACE/Server on page 6-127*.
- *Using SafeWord AS Security on page 6-119*.

Password Histories and Blacklisting

You can enhance security for passwords by configuring the RAC to record password histories and to blacklist users who have a configurable number of failed login attempts.

Creating the acp_userinfo File

The **acp_userinfo** file resides in the install directory and is maintained by the network administrator. The file primarily defines aspects of the user login environment. This environment can be defined on the basis of profile criteria and/or a user ID (user name).

The information from **acp_userinfo** is loaded into the **erpcd** internal database. To update the database, send a USR1 signal to **erpcd** (**kill -USR1** *pid*). When updating **acp_userinfo**, it is a good idea to check syntax using the **erpcd -u** *filename* command.

To create entries in the **acp_userinfo** file, use the following format, which is referred to as a **user...end** block:

user username={*name*| *profile_criteria*}
entry
:
:
end

The syntax for *profile_criteria* is:

keyword=*value* [**;***keyword*=*value***;**...]

Entering profile criteria is described in detail in *Profile Criteria Syntax on page 6-64*.

If you use the *name* argument instead of *profile_criteria*, specify a valid user ID. This argument is supported for compatibility with Release 10.1 and earlier releases but is treated as if it were the profile criterion **username**=*name*. In searching **acp_userinfo**, **erpcd** looks only for a first match, whether that match is a single userid or all the criteria in a profile criteria specification.

The following is an example of a user-end block:

```
user username=jill
      climask slip ppp end
end
user group=finance;time="8:00AM-6:00PM Monday-Wednesday"
      clicmd ppp end
end
user group=finance
      deny
end
```

In the above example, **user username=jill** can also be specified as **user jill.**

In this example, even if user *jill* is a member of the *finance* group and meets all of the criteria in that profile criteria specification, *jill* is not permitted to use **slip** or **ppp**, since the first match found is the userid *jill*. The remainder of the example specifies that the finance group is allowed to connect only if its members log in between 8:00 A.M. and 6:00 P.M. on the specified days. The CLI port they are connected to will be converted to **ppp** mode after the group members have been authenticated. At any other time, they are denied access.

You can specify the following *entry* options (the following subsections discuss these options in detail):

- **accesscode**
- clicmd
- climask
- deny
- filter
- route
- at_zone
- **at_connect_time**
- at_nve_filter
- **at_passwd**
- **chap_secret**

accesscode

The **accesscode** is a string for which the user is prompted. Specify **accesscode** at the beginning of an accesscode entry. For each user, or for conditions that meet profile criteria, you can define one or more accesscode entries in the **acp_userinfo** file. Depending on the contents of the entry, one of several actions can occur, including dial-back. Each entry can include a phone number, inbound and outbound modem pools, and a job name (see Table 6-7). The syntax is:

accesscode *code*
accesscode_entry
end

Table 6-7. Entries for accesscode in the acp_userinfo File

Entry	Description
code	A character string defined by the administrator. The user is prompted for this string (after the **user name** and **password** prompts) when logging onto a port defined for dial-back security.
accesscode_entry	A list of one or more of the **accesscode** entries: **phone_no**, **in_pool_name**, **out_pool_name**, **job**.
phone_no	Specifies the dial-back phone number with the format: **phone_no** *phone_no* *phone_no* is the phone number to be called. If this optional parameter is not specified, the user is prompted for the dial-back phone number. System administrators are encouraged to specify this entry to avoid compromising system security. Any characters accepted by the modem can be used here. Notice that the escape character (\) must precede each special character (* , # @ ! ; =).

(continued on next page)

Table 6-7 Entries for accesscode in the acp_userinfo File (continued)

Entry	Description
in_pool_name	Specifies the name of the inbound modem pool with the format: **in_pool_name** *pool_name* *pool_name* is the name of an inbound modem pool. For the dial-back request to be initiated, the designated port type must be defined for the inbound pool.
out_pool_name	Specifies the name of the outbound modem pool with the format: **out_pool_name** *pool_name* *pool_name* is the name of an outbound modem pool. For the dial-back request to be initiated, the designated port type must be defined for the outbound pool.
job	Defines a specific CLI command. The default is the CLI. Each **accesscode** can have up to one job record, using the format: **job** *command* [*argument...*]**end** *command* is a CLI command name, e.g., **rlogin**. *argument...* is an option list of command-specific arguments, e.g., to remotely log the user *Morse* into the host *amos*, the job entry is: `job rlogin amos -1 Morse end`

The following example illustrates **accesscode** entries in the **acp_userinfo** file. When logging in, the user *cobb* is prompted for a user name, password, and **accesscode**. If *cobb* enters the information at each prompt, the RAC determines whether or not *cobb's* access is via the inbound modem pool. If so, one of the following occurs:

- If cobb enters *access* at the **accesscode** prompt, the RAC calls *cobb* back at the number *9-765-4321* and then logs *cobb* into the host *calvin*.
- If *cobb* enters *promptphone* at the **accesscode** prompt, the RAC prompts for a phone number, drops the connection, and calls *cobb* back via the outbound modem pool. Then the Annex prompt is displayed.
- If *cobb* enters *direct* for the accesscode, the Annex prompt is displayed and no dial-back occurs.

```
user cobb
     at_passwd        nedry
     at_zone          bn-33net bn-55net end
     accesscode       access
          phone_no        9\,7654321
          in_pool_name    inbound
          out_pool_name   outbound
          job             rlogin calvin -1 cobb end
     end

     accesscode           direct
          in_pool_name    inbound
     end

     accesscode       promptphone
          in_pool_name    inbound
          out_pool_name   outbound
     end
end

pool inbound
     ports            asy@hobbes
     ports            asy@simon
end
pool outbound
     ports            asy@hobbes
     ports            asy@simon
end
```

clicmd

For a single user or for conditions that meet profile criteria, you can define one or more CLI commands and macros in the **acp_userinfo** file. These commands are executed, in the order in which they are specified, if the profile criteria are met or the user name matches the user ID supplied at login. If the RAC detects an error in a command, **erpcd** stops sending commands, syslogs an error, and denies access to the user.

Table 6-8 describes the **clicmd** entry. The syntax is:

clicmd *CLI_command* **end**
clicmd ... end

For **clicmd** to work, the **cli_security** parameter must be set to **Y**.

Table 6-8. Arguments For the clicmd Entry in the acp_userinfo File

Argument	Description
CLI_command	Any user or superuser CLI command, or the name of a macro previously defined for the RAC. Only one command or macro is allowed per **clicmd** entry, although a **user...end** block can contain multiple **clicmd** entries. After the final command in a **user...end** block executes, the CLI session ends. To continue the session, use the **clicmd** with the ellipses (**...**) argument, explained next. (For descriptions of the CLI commands, see *CLI Commands on page 2-3*.
...	Specifies that the CLI session should not end when the last **clicmd** entry for a given user has been executed. Subsequent commands in the same **user...end** block are ignored.

The **clicmd** entry is useful for configuring dedicated connections. In the following example, if user *kip* logs in at any time between 9:00 A.M. and 5:00 P.M. on the specified days, the RAC executes the **ppp** command (after authenticating *kip* at the CLI level). The port to which *kip* is connected is thereby converted from CLI to PPP mode. When the PPP link goes down, *kip* is disconnected from the RAC.

```
user username=kip;time="9:00am - 5:00pm Tuesday-Friday"
     clicmd ppp end
end
```

In the following example, the RAC does not disconnect *kip* when the PPP link terminates.

```
user username=kip;time="9:00am - 5:00pm Wednesday-Friday"
     clicmd ppp end
     clicmd ... end
end
```

climask

For a single user or for conditions that meet profile criteria, you can define a CLI command mask in the **acp_userinfo** file that limits which CLI commands the user(s) can execute (see *Masking CLI Commands on page 6-107*). Table 6-9 describes the entry for **climask** in the **acp_userinfo** file. The syntax for adding the CLI command mask to a user profile is:

climask *command_list* **end**

Table 6-9. Entry For climask in the acp_userinfo File

Entry	Description
command_list	A list of user-level CLI commands, separated by spaces, that are ***not*** available to the user. Valid values are **bg**, **call**, **fg**, **hangup**, **help**, **hosts**, **jobs**, **kill**, **netstat**, **rlogin**, **stats**, **stty**, **telnet**, **who**, **lock**, **su**, **slip**, **connect**, **services**, **ppp**, **arap**, **ipx**, and **none** (the default). The list of restricted command names is sent to the RAC and the user is prevented from executing those CLI commands. Do not specify the same command as both a **clicmd** and a **climask** in a given **acp_userinfo** entry (for more details on CLI commands, see *CLI Commands on page 2-3*).

The **climask** entry allows minimum uniqueness for command names. If you specify an ambiguous command name, **climask** generates a warning but cannot prevent the user from issuing the command.

The following is an example of **climask**:

```
user username=sam;time="9:00am-10:30pm Friday-Monday"
     climask ppp arap end
end
```

If user *sam* logs into any RAC between 9:00 A.M. and 10:30 P.M. on the specified days, he cannot issue the **ppp** or **arap** command. In all other situations, this particular **user...end** block is ignored. For example, if *sam* logs into a RAC at 11:00 PM, the entry is ignored.

deny

For a single user or for conditions that meet profile criteria, you can deny access to the RAC in the **acp_userinfo** file. If the profile criteria are met or the user name in the user entry matches the user ID supplied at login, ACP refuses access to the RAC. Table 6-10 describes the entry for **deny** in the **acp_userinfo** file. The syntax is:

deny

Table 6-10. Entry for deny in the acp_userinfo File

Entry	Description
deny	A keyword indicating the user will be denied access to the RAC. If used, **deny** should be the only entry in the **user...end** block. A message is logged in the ACP log file indicating why access is being denied. For CLI users, a message is displayed.

The following is an example of using **deny** in the **acp_userinfo** file:

```
user username=liza
     deny
end
user group=eng;time=”9:00am -10:30pm Saturday-Sunday”
     clicmd ppp end
end
```

In this example, even if user *liza* is a member of the *eng* group, she is denied access, since **erpcd** finds the match with the user ID first.

In the following example, users cannot connect to any RAC between 11:00 PM and 12:00 PM on any of the specified days:

```
user time=”11:00pm - 12:00pm Saturday-Wednesday”
     deny
end
```

filter

For a single user or for conditions that meet profile criteria, you can define one or more IP filters in the **acp_userinfo** file. These filters can apply to PPP and/or SLIP packets. Table 6-11 describes the entry for **filter** in the **acp_userinfo** file. The syntax is:

filter *filter_definition* **end**

Filters are session-specific; they are dynamically applied to each internal port for the duration of a session based on the entries in the **acp_userinfo** file.

Table 6-11. Entry for filter in the acp_userinfo File

Entry	Description
filter_definition	Defines a filter to apply to the port on which the user logs in. You can enter only one filter per line but multiple filters are allowed within one **user...end** block. Filters are applied in the order in which they are specified. Unlike the **filter** command, a filter specification in **acp_userinfo** does not start with the word **add** (since it is assumed that you are adding a filter) and does not contain the name of the login interface (since that is known).

You can also restrict the transmission and reception of SLIP and IP over PPP packets by using the **acp_restrict** file. Using **acp_restrict** for this purpose can be easier than using **acp_userinfo** because you do not have to enter actual filters in **acp_restrict**. Instead, you enter user-friendly statements from which filters are created for you.

Any filters you enter in **acp_userinfo** or arrange to have generated by **acp_restrict** will be combined with, and interpreted according to the algorithm used for, filters created by the superuser **filter** command.

The following example creates a filter that discards any IP packets destined for address *132.245.4.33* - if transmission of such packets is attempted on the port from which user *sam* logs in.

```
user username=sam
     filter output include dst_address 132.245.4.33 discard end
end
```

Like all other **acp_userinfo** entries (except **deny**), the **filter** entry can be accompanied by other entries within the same **user...end** block. In the following example, not only is the above filter created, but a pre-defined macro named *special_setup* and the CLI command **ppp** are also executed for user *sam*.

```
user username=sam
     clicmd special_setup end
     filter output include dst_address 132.245.4.33 discard end
     clicmd ppp end
end
```

route

For a single user or for conditions that meet profile criteria, you can define one or more IP routes in the **acp_userinfo** file. You can enter only one route per line, but multiple routes are allowed within one **user...end** block.

Routes in **acp_userinfo** are entered into the routing table when their interfaces become active, but they are not entered into the route cache. You cannot use a **route** entry in **acp_userinfo** to define a default route. The syntax for the **route** entry is:

route [**-h**] *dest mask gateway* [*metric*] **end**

Table 6-12. Arguments forThe route Entry in the acp_userinfo File

Entry	Description
-h	Defines the route as hardwired.
dest	Specifies the destination address of the route.
mask	Specifies the subnet mask for the destination address. You can enter the mask in dotted decimal notation, e.g., 255.255.255.0, or you can specify the mask by appending /*n* to the destination address, where *n* is the number of 1 bits in the mask, from left to right. For example, appending /24 specifies 255.255.255.0 as the subnet mask.
gateway	Specifies the IP address of the gateway (router) that is the next hop for the route. If you specify an asterisk (*) for gateway, the RAC uses the port's remote address as the gateway.
metric	Specifies the number of hops to the destination. Values range from **1** through **15**; the default is **1**.

Typically, a **route** entry in **acp_userinfo** is used when a router attached to a small network dials into the RAC but does not want to incur the overhead of running a routing protocol itself. Consider the configuration in Figure 6-1.

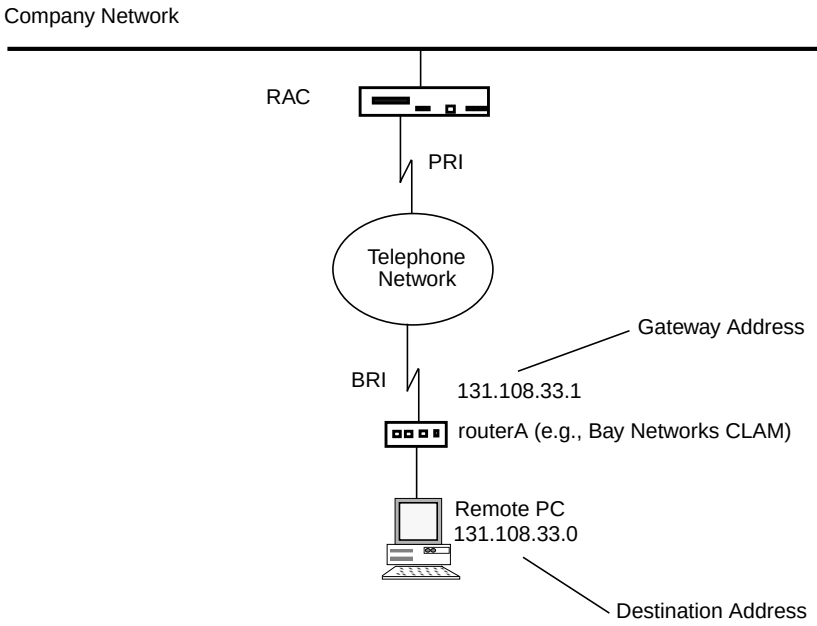

Figure 6-1. Sample Configuration for a route Entry in acp_userinfo

Given the configuration in Figure 6-1, the following example defines a route on the RAC that will be used for routerA (e.g., Bay Networks CLAM). This route allows packets to be sent back and forth between the company network and the remote PC with the IP address *131.108.33.0*. The destination address is *131.108.3.0*, using a subnet mask of *255.255.255.0*. The gateway address is *131.254.33.1*, and the metric for the route is 1 (the default).

```
user username=routerA;annex=RAC
     route 131.108.33.0/24 131.254.33.1 1 end
end
```

at_zone

For a single user or for conditions that meet profile criteria, you can define AppleTalk zone list entries in the **acp_userinfo** file. This zone list consists of zone names. Table 6-13 lists the entry for **at_zone** in the **acp_userinfo** file. The syntax is:
at_zone *zone...* **end**

Table 6-13. Entry for at_zone in the acp_userinfo File

Entry	Description
zone	A list of one or more ASCII character strings. You can have any number of zones specified in a zone list, subject to the following constraints: • A zone identifier cannot contain non-printable characters. • An individual zone identifier cannot exceed 32 characters in length. • The combined length of the entire zone list cannot exceed 524-*n* characters, where *n* is the number of zones in the list. • The reserved keyword **end** cannot appear as a zone argument. • A string containing a space must be enclosed in double quotation marks.

The following example illustrates **at_zone** entries in the **acp_userinfo** file. When logging in using ARA, user *cobb* is assigned to zones *bn-33net* and *bn-55net*.

```
user username=cobb
     at_zone bn-33net bn-55net end
end
```

The next example shows an **at_zone** entry that uses profile criteria. When logging in via ARA between the hours of 8:00 A.M. and 6:00 P.M, user *hobbes* is assigned to zones *bn-11net* and *bn22-net*.

```
user username=hobbes;time="8:00am-6:00pm Sunday-Wednesday"
     at_zone bn-11net bn-22net end
end
```

at_connect_time

The **acp_userinfo** file can have an ARA connect timer defined; **at_connect_time** defines the maximum amount of time, in minutes, that an ARA connection can remain open. You can specify **at_connect_time** for a single user or for conditions that meet profile criteria. Table 6-14 defines the argument for **at_connect_time** entries in the **acp_userinfo** file. The syntax is:

at_connect_time *time_value*

For example:

at_connect_time *120*

Table 6-14. Entry for at_connect_time in the acp_userinfo File

Entry	Description
time_value	The format for this argument is <*minutes*>.

at_nve_filter

NVE filtering controls a remote access Apple user's view of network resources: when using *Chooser* to select resources, only the resource set defined for the user by the administrator is visible. The administrator can specify the NVE filter on a per-user basis or for conditions that meet profile criteria. This feature complements the existing zone list by offering a higher level of control.

The administrator uses the **nve_filter** entry in the **acp_userinfo** file to specify a finite list of filters. Only one **nve_filter** entry per user or per profile criteria specification is permitted. The entry uses the format:

at_nve_filter [**include**|**exclude**] *tuple,tuple tuple...* **end**

Table 6-15. Entries for at_nve_filter in the acp_userinfo File

Entry	Description
include| **exclude**	The **include** or **exclude** qualifier controls how filters are used: **include** filters allow only matching answers; **exclude** filters discard matching answers and allow non-matching answers. There is a 10-filters-per-user limit. The default is **include**.
tuple	A three-part string that identifies all network resources. The three parts of a *tuple* are: object, type, and zone. The format of a *tuple* is: **object:type@zone** with asterisks as wild cards. Any *, @, or : used as an NVE character within a *tuple* must be preceded by the Escape (\) character. Characters in a *tuple* are case-insensitive. Each field of an entity can contain up to 32 characters.

Following are sample **acp_userinfo** entries, including **nve_filter** information, for two users. User *frick* is allowed access only to the resources of her office Macintosh named *Frick CPU*. User *frack* cannot access *frick*'s machine, nor is she allowed access to any sales resources.

```
user username=frick
     at_passwd klot
     at_nve_filter include Frick\CPU:*@eng end
end

user username=frack
     at_passwd curly
     at_nve_filter exclude Frick*:*@ *:*@sales end
end
```

Like all other **acp_userinfo** entries, **nve_filter** information is syntax-checked by **erpcd**. Any errors cause the entire filter to be discarded, and an error message is generated.

This method of limiting NBP traffic is not secure, and can be circumvented by a person willing to write code to probe the network without using NBP. Also, this feature has no local RAC security equivalent.

at_passwd

Each registered AppleTalk user (as opposed to a guest) must have a password defined in the **acp_userinfo** file. You can define a password for a single user or for conditions that meet profile criteria (e.g., membership in a group). Table 6-16 defines the argument for the **at_passwd** entry in the **acp_userinfo** file. The syntax is:

at_passwd *string*

Table 6-16. Entry for at_passwd in the acp_userinfo File

Entry	Description
string	A string of up to nine alphanumeric characters (the un-encrypted password). Punctuation marks are permitted; precede spaces and hex values with a backslash (\).

The following example illustrates an **at_passwd** entry in the **acp_userinfo** file:

```
#Set up the user entry
#

user username=cobb
     at_passwd ned\ ry
end
```

A guest entry in the **acp_userinfo** file looks like this:

```
#Set up a guest user entry that allows guests to connect
#for 1 hr.and hides our file servers

user username=<Guest>

     at_connect_time 60:00
     at_nve_filter exclude

        *:AFPServer@*

     end

end
```

The *Guest* entry is case-sensitive. If it is entered incorrectly, guests can log in with no restrictions because the **at_guest** parameter for this port is set to **Y**.

chap_secret

A secret token that enables CHAP authentication for PPP is defined in **chap_secret** entries in the **acp_userinfo** file. You can define the token for a single user or for conditions that meet profile criteria (e.g., membership in a group). Table 6-17 defines the entry for **chap_secret**. The syntax is:

chap_secret *secret_token*

Table 6-17. Entry for chap_secret in the acp_userinfo File

Entry	Description
secret_token	A string from 1 to 32 bytes long; 16 bytes is recommended due to the operation of the MD5 encryption algorithm.

The following example illustrates a **chap_secret** entry in the **acp_userinfo** file:

```
user username=smith
     chap_secret achapsecrettoken
end
```

For more details on CHAP and secret tokens, see *Challenge-Handshake Protocol (CHAP) on page 6-137*.

Limiting Access to Hosts via acp_restrict

The erpcd utility can restrict any CLI (i.e., **telnet** and **rlogin**), SLIP, or PPP request for IP access to a specific host or host-port combination. This security mechanism uses a host-resident file that lists the hosts and host ports to which access is restricted and specifies the RAC or the profile criteria to which the restrictions apply. By default, there are no host or host-port restrictions.

Host access security for CLI ports is enabled by setting the port parameter **connect_security** to **Y**.

Hosts or ports not listed in **acp_restrict** are considered unrestricted.

When a user issues a connection command or a SLIP or PPP link becomes active, the RAC, using **erpcd**, checks a restrict file for permission to connect to that host. **erpcd** expects the restrict file to be **acp_restrict** (located in the installation directory), which is an ASCII file that you create with any text editor. Table 6-18 describes the arguments in each entry. The entry format is:

annex | *profile_criteria*: *restricted host* [[*ports*]] [,*restricted host* [[*ports*]], ...]
annex | *profile_criteria*~ *unrestricted host* [[*ports*]] [, *unrestricted host* [[*ports*]],...]

Table 6-18. Arguments in the acp_restrict File Entries

Argument	Description
annex	The name or IP address of the RAC initiating the access. This argument is supported for backward compatibility with Release 10.1 and earlier releases but is treated as if it were the profile criterion **annex=***annex* (*profile_criteria* are described next).
profile_criteria	One or more *keyword=value* pairs, separated by semicolons (;), specifying the conditions under which the specified hosts will be restricted or unrestricted. For information on entering *profile_criteria*, see *Profile Criteria Syntax on page 6-64*.
: (colon)	Indicates that the hosts listed in the same entry are restricted. White space may follow, but not precede, the colon.
~ (tilde)	Indicates that the hosts are unrestricted. White space may follow, but not precede, the tilde. For PPP and SLIP connections, hosts specified as unrestricted imply that all other hosts are restricted. (For a CLI connection, hosts specified as unrestricted have no implications for other hosts.)
restricted host	The name or IP address of a restricted host (including RACs). The list of restricted hosts is separated by commas; white space is not allowed. You can use an asterisk (*) as a wild card in place of a host name or the host part of an IP address.

(continued on next page)

Table 6-18 Arguments in the acp_restrict File Entries (continued)

Argument	Description
unrestricted host	The name or IP address of an unrestricted host (including RACs). The list of unrestricted hosts is separated by commas; white space is not allowed. You can use an asterisk (*) as a wild card in place of a host name or the host part of an IP address.
[*ports*]	One or more TCP or UDP ports on *restricted host* or *unrestricted host*. To specify multiple ports, separate them with commas or specify them as a range separated by a hyphen (-). Enclose the port(s) in square brackets ([ ]). White space and wild cards are not allowed. The default is any TCP or UDP port.

Following are two restricted-host entries:

```
annex01: hosta,hostb,hostf,132.245.6.23
annex02: hostc,132.245.6.15,hostf,132.245.6.23,\
                    hosth,annex01
```

The first entry prevents SLIP, PPP, and CLI connections from *annex01* to any port on *hosta*, *hostb*, *hostf*, or the host at IP address 132.245.6.23. The second entry prevents SLIP, PPP, and CLI connections from *annex02* to any port on *hostc*, *hostf*, *hosth*, the host at IP address 132.245.6.15, the host at IP address 132.245.6.23, and *annex01*.

In the next example, which shows the use of profile criteria, user *carl* is blocked from using **telnet** or **rlogin** to access hosts *atlas* and *steam*:

```
username=carl;protocol=cli:atlas,steam
```

For profile criteria entries in which the only protocol specified is CLI, as in the previous example, **erpcd** searches the file in sequential order and uses only the first entry whose profile criteria are met. For these types of entries, order of placement in the file is important. If permission is granted to a CLI connection request, the user follows the normal login procedure. If the request is denied, the message *Permission denied* is displayed and the session (job) is aborted.

For profile criteria specifications that explicitly specify SLIP or PPP (or implicitly specify them, by not specifying *any* protocol), filters are automatically generated to restrict SLIP and/or PPP connections if either protocol becomes active. Consider the following example:

```
username=*;protocol=slip: finance
```

In this example, all SLIP users on all RACs are denied access to host *finance* but are allowed access to all other hosts and host ports.

Given an address of 132.245.11.4 for host *finance*, the filters generated to effect these restrictions are:

```
in include address_pair 132.245.11.4 * discard
out include address_pair 132.245.11.4 * discard
```

In the next example, the members of the group *mail_only* who connect using the PPP or SLIP protocol (as opposed to the CLI) may access the SMTP port (25) on host *mailhub* and the DNS server port (53) on the host *dns_srv*, but they cannot access anything else.

```
group=mail_only;protocol=slip~ mailhub[25], dns_srv[53]
group=mail_only;protocol=ppp~ mailhub[25], dns_srv[53]
```

To put these restrictions into effect, the RAC generates the following four filters, in which *132.245.33.1* is the address of *mailhub* and *132.245.33.2* is the address of *dns_srv*.

```
in exclude address_pair 132.245.33.1 * port_pair 25 * discard
out exclude address_pair 132.245.33.1 * port_pair 25 * discard
in exclude address_pair 132.245.33.2 * port_pair 53 * discard
out exclude address_pair 132.245.33.2 * port_pair 53 * discard
```

Filtering Restrictions

IP filtering can handle the following two cases:

- One or more hosts cannot be reached and all other hosts can.
- One or more hosts can be reached and all other hosts cannot.

However, IP filtering cannot handle the next two cases:

- A subset (e.g., a subnet or subnet group) of hosts can be reached, except for a few hosts in the subset, and all other hosts cannot be reached.
- A subset of hosts cannot be reached, except for a few hosts in the subset, and all other hosts can be reached.

For example, you cannot use **acp_restrict** to allow a user named *martha* to access all hosts on her home network (132.245.0.0), except for the finance machine at IP address 132.245.77.1, and also deny her access to hosts outside the 132.245.0.0 network. The **acp_restrict** entries for this are:

```
user martha: 132.245.77.1
user martha~ 132.245.*
user martha: *
```

If such an entry is found, a *syslog* message is generated and the user is denied access.

In addition, **acp_restrict** cannot create filters from host names containing wild cards, e,g., annex*.

Finally, filters apply to IP packets only; IPX and AppleTalk packets cannot be filtered.

Using include Files in the acp_userinfo File

To reduce the task of repeating several **job** and **climask** lines in the **acp_userinfo** file, you can create an **include** file. Nested **include** files are not allowed; the only commands allowed in the **include** file are **job** and **climask**. The syntax is:

include *filename*

Table 6-19. Argument for the include File

Argument	Description
filename	The name of a file located in the same directory as **acp_userinfo**.

A log message is written to the ACP log file if the **acp_userinfo** file references an **include** file that could not be opened.

Modifying the Supplied Security Application

You can modify the supplied security policy to create a security scheme that meets the needs of your network. Some simple modifications involve changing system definitions in the file */annex_root*/**src/erpcd/acp_policy.h**. More elaborate security policies may require modifying or replacing functions in the file */annex_root*/**src/erpcd/acp_policy.c**.

Do not change the function declarations or the description of the interface; these are fixed by the calls made into this library. Before making even the smallest change, save the base version of the file requiring modification.

If you modify the default policy, you must re-compile **erpcd**, kill the current version, and start the new version (see *Modifying the Code on page 6-109*).

Disabling User Name and Password Validation

When security is enabled, users must provide a user name and password. You can disable this policy by modifying the */annex_root*/**src/erpcd/acp_policy.h** file.

To disable the user name requirement, change the line that defines user validation from:

```
#define USER_VALIDATION 1   to   #define USER_VALIDATION 0
```

Messages are logged to the security server host when users access the CLI, but the message does not include a user name.

To disable the port password requirement, make sure the following line is commented out (i.e., enclosed in asterisks), as follows:

```
/* #define PORT_PASSWORD 1 */
```

Linking NIS Password File Verification to ACP

You can enable several options in the **acp_policy.h** file by removing the slash (/) and asterisk (*) at the beginning and the end of the definition line.

To use the NIS password file for verification through ACP, change (uncomment) the following lines:

```
/* #define NATIVEPASSWD 1 */ to #define NATIVEPASSWD 1
/* #define NATIVESHADOW 1 */ to #define NATIVESHADOW 1
```

You can change several other options in the same way:

```
/*
     * Uncomment this line to select the use of the\
     * standard syslog(3) facility in addition to or in\
     * place of the logfile -- the value of "USE_SYSLOG"\
     * is used to identify the daemon.(Comment the
     * second line out to disable the normal acp log file.)
*/
/* #define USE_SYSLOG "annex" */
#define USE_LOGFILE 1

/*
     * Uncomment this line to use decoded Annex peer names,\
     * rather than numeric IP addresses, in the log file
     * and in syslogging.
*/
/* #define USE_ANAME 1 */
```

Modifying Message Formats in the ACP Log File

The USE_SECONDS option in the **acp_policy.h** file enables messages in the ACP log file to use a *seconds-since-1970* (ten decimal digits) format. This format is most useful for automatic ACP log file parsing programs since these programs frequently need to do comparisons and arithmetic on dates. This option is disabled by default.

You can enable USE_SECONDS by changing (uncommenting) the following line:

```
/* #define USE_SECONDS 1*/ to #define USE_SECONDS 1
```

The standard message format in the ACP log file is:

<annex_name>:*<logid>*:#*<port>*:*<yymmdd>*:*<hhmmss>*:*<service>*:\
<event>:<pkts in>:<pkts out>:<bytes in>:<bytes out>:<msg>

When USE_SECONDS is enabled, the message format in the ACP log file is:

<annex_name>:*<logid>*:#*<port>*:*<seconds_since_1970>*:\
<service>:<event>:<pkts in>:<pkts out>:<bytes in>:\
<bytes out>:<msg>

Changing the Expected File Names Used by ACP

The supplied policy uses names for various files. For example: **acp_passwd**, **acp_keys**, **acp_restrict**, and **acp_logfile**. You can change the names of any of these files in the */annex_root*/**src/erpcd/acp_policy.h** file.

If you decide to use either an existing system or a network-wide password file instead of the **acp_passwd** file, change the following lines in the **acp_policy.h** file:

```
#define ACP_PASSWD (str) \
   sprintf(str,"%s/acp_passwd",install_dir)

#define ACP_PTMP (str) \
   sprintf(str,"%s/acp_ptmp",install_dir)
```

To change only the filename:

```
#define ACP_PASSWD (str) \
   sprintf(str,"%s/new_filename",install_dir)

#define ACP_PTMP (str) \
   sprintf(str,"%s/new_tempfile",install_dir)
```

To change the full pathname:

```
#define ACP_PASSWD (str) \
   sprintf(str,"new_path/new_filename")

#define ACP_PTMP (str) \
   sprintf(str,"new_path/new_tempfile")
```

The *new_filename* is the name of the new password file, and the *new_tempfile* is a temporary file used by the **ch_passwd** command. Since you do not need the temporary file if you are using an existing system file, comment out the line for the temporary file.

The **install_dir** is defined in the file */annex_root*/**src**/**make.config** with the leading quote supplied by the makefile. Since the trailing quote is required by the two strings, double quote the names for the new password and temporary files.

You can change the names of several other files in the **acp_policy.h** file in the same way:

```
#ifdef NATIVESHADOW

#define ACP_SHADOW(str)\
     strcpy(str,"/etc/shadow")
#define ACP_STMP(str)\
     strcpy(str,"/etc/shadow.tmp")
#define ACP_LOCKFILE(str)\
     strcpy(str,"/etc/.pwd.lock")
#define ACP_GROUP(str)\
     strcpy(str,"/etc/group")
#else

#define ACP_SHADOW(str)\
     sprintf(str,"%s/acp_shadow",install_dir)

#define ACP_STMP(str)\
     sprintf(str,"%s/acp_stmp",install_dir)

#define ACP_LOCKFILE(str)\
     sprintf(str,"%s/.pwd.lock",install_dir)

#define ACP_GROUP(str)\
     sprintf(str,"%s/acp_group",install_dir)

#endif

/*   define pathname of accounting file*/

#define ACP_LOGFILE(str) \

     sprintf(str,"%s/acp_logfile",install_dir)

/*   define pathname for restrictions file*/
```

(continued on next page)

```
#define ACP_RESTRICT(str) \
     sprintf(str,"%s/acp_restrict",install_dir)
/*   define pathanme for annex acp_keys file */
#define ACP_KEYS(str) \
     sprintf(str,"%s/acp_keys",install_dir)
/*   define pathanme for annex dialup addresses file */
#define ACP_DIALUP(str) \
     sprintf(str,"%s/acp_dialup",install_dir)
/*   define pathname for user profile file */
#define ACP_USERINFO(str) \
     sprintf(str,"%s/acp_userinfo",install_dir)
#define ACP_ESERVICES(str) \
     sprintf(str,"%s/eservices",install_dir)
```

In the same way, you can also change the expected prompts for default applications:

```
#ifndef SECURID_CARD
#define ACP_USERPROMPT "Annex username: "
#define ACP_PASSPROMPT "Annex password: "
#define ACP_PERMGRANTD "\nPermission granted\n"
#define ACP_PERMDENIED "\007\nPermission denied\n"
#define ACP_INCORRECT "\nUsername/Password Incorrect\n"
#else
#define ACP_USERPROMPT "Username: "
#define ACP_PASSPROMPT "Enter PASSCODE: "
#define ACP_PERMGRANTD "\nPASSCODE accepted\n"
#define ACP_PERMDENIED "\007\nAccess Denied\n"
#define ACP_INCORRECT "\nUsername/PASSCODE Incorrect\n"
#endif
```

(continued on next page)

```
#define ACP_TIMEDOUT "\007\nLogin Timed Out\n"
#define ACP_WARNING "\007\nYour password will expire \
     in %ld days unless changed.\n"
#define ACP_WARNINGM "\007\nYour password expires after\
     tomorrow unless changed.\n"
#define ACP_WARNINGT "\007\nYour password expires after \
     today unless changed.\n"

#define ACP_AWARNING "\007\nYour account will expire in\
     %ld days.\n"
#define ACP_AWARNINGM "\007\nYour account expires after\
     tomorrow.\n"
#define ACP_AWARNINGT "\007\nYour account expires after\
     today.\n"
#define ACP_EXPIRED "Your password has expired.\n"
#define ACP_NEWPASS "Enter a new password:  "
#define ACP_NEWPASS2 "Re-enter new password:  "

#define ACP_PASSMATCH "Entered passwords do not match.\
     Try again.\n"
#define ACP_ACCESSCODEPROMPT "Access Code: "
#define ACP_PHONEPROMPT "Telephone Number: "
#define ACP_DIALBACKGRANTD "\nRequest accepted,dialback in\
     progress\n"
#define ACP_CLINODIALBACK "\nPermission granted, no\
     dialback\n"

/*   define messages used by Securid Card application*/

#ifdef SECURID_CARD
#define ACP_NEXTCODEPROMPT "Enter next card code: "
#define ACP_PINCHAR "characters"
#define ACP_PINDIGIT "digits"
#define ACP_PINSIZE "%d"
#define ACP_PINSZRANGE "%d to %d"
#define ACP_NEWPINPROMPT "Enter your new PIN containing %\
     %s,\n"
```

(continued on next page)

```
#define ACP_OR "\t\tor\n"
#define ACP_NEWPIN_2 "Press Return to generate new PIN and\
     display it\n"
#define ACP_NEWPIN_3 "<Ctrl d> to leave your card in New-PIN\
     mode.\n"
#define ACP_SYSGENPIN "\t\t%s\n"
#define ACP_PINREENTRY "Please re-enter PIN: "
#endif#ifdef PORT_PASSWORD
/* only if PORT_PASSWORD is set and a port password exists
in acp_passwd */
#define ACP_PORTPROMPT "Port password: "
#endif

/* miscellaneous defines for default application */

#define INPUT_TIMEOUT 30
#define INPUT_POLL_TIMEOUT 3
#define RETRIES_MAX 3
```

Locking the ACP Log File

To prevent two or more host processes from logging a record simultaneously, the RAC **erpcd** code uses the host system call **lockf** to lock the ACP log file. This lock prevents other processes from writing the file until the file update is complete.

There are two ways to use the system **lockf** call. You can select either mechanism via a switch in the **acp_policy.h** file. The following explanation of the switch resides in this file. The default method, T_LOCK, is reliable but not very efficient; F_LOCK is more efficient but does not work on all hosts (some host manufacturers have issued patches that resolve this issue).

```
/*
* Uncomment this line to select the F_LOCK method to lock the
* ACP log file for updating.
*
* A file must be locked for update in order to block other
* processes from writing to it simultaneously.
*
* F_LOCK - Passing the F_LOCK as the cmd value when making
* system lockf call is the most efficient and preferred manner
* to lock a file for exclusive write access. In this scenario
* a process is put to sleep until the resource is available.
* Once available the process is preempted owning the resource.
*
* T_LOCK - When the T_LOCK cmd argument is passed, the process
* must repeatedly send the lockf call the until the resource
* is available.Once available the system call returns a
* success and the resource is acquired.
*
* The F_LOCK cmd has been determined to be faulty on many
* hosts. Failures can not be narrowed down to any particular
* hardware manufacturer or UNIX system. There are to many OS
* revs and variables to sense the correct lockf method to
* use at installation time. The default, T_LOCK was chosen
* simply because it has been proven reliable. SEE
* 'log_message()'
*/
/* #define USE_F_LOCK 1 */
```

Masking CLI Commands

When the security subsystem is enabled, you can mask (disable) user access to specific CLI commands by modifying the CLI_MASK line in the **acp_policy.h** file.

To disable **rlogin** and **telnet** for all users that enter the system through ACP security, modify the definition line to read:

```
#define CLI_MASK (unsigned long) (MASK_RLOGIN | MASK_TELNET)
```

To disable the CLI **who** and **su** commands for all users that enter the system through ACP security, modify the definition line to read:

```
#define CLI_MASK (unsigned long) (MASK_WHO | MASK_SU)
```

You can extend this to any set of commands by adding masks to that line separated by the vertical bar (|).

If the user enters the masked command, the CLI displays an error message. The superuser CLI commands cannot be masked individually. They can all be disabled by masking the **su** command.

Superuser CLI mode overrides ACP command masking.

You can disable several other CLI commands in the same way:

```
/*   define bit to disable each maskable CLI command*/

#define MASK_BG       0x00000001
#define MASK_CALL     0x00000002
#define MASK_FG       0x00000004
#define MASK_HANGUP   0x00000008
#define MASK_HELP     0x00000010
#define MASK_HOSTS    0x00000020
#define MASK_JOBS     0x00000040
#define MASK_KILL     0x00000080
#define MASK_NETSTAT  0x00000100
#define MASK_RLOGIN   0x00000200
#define MASK_STATS    0x00000400
#define MASK_STTY     0x00000800
#define MASK_TELNET   0x00001000
#define MASK_WHO      0x00002000
#define MASK_LOCK     0x00004000
#define MASK_SU       0x00008000
#define MASK_SLIP     0x00010000
#define MASK_CONNECT  0x00020000
#define MASK_SERVICES 0x00040000
```

(continued on next page)

```
#define MASK_PPP      0x00080000
#define MASK_ARAP     0x00100000
#define MASK_NONE     0x80000000
```

After changing the code, **cd** to the **/src** directory and recompile **erpcd**.

For more specific command disabling, e.g., by user name, you must edit the distribution policy file */annex_root*/**src/erpcd/acp_policy.c**.

Modifying the Code

You can create a more elaborate security policy application by modifying the code in the files */annex_root*/**src/erpcd/acp_policy.c** and */annex_root*/**src/erpcd/acp_policy.h**. The program that executes ACP starts a new version of itself each time a security request is received from a RAC. A call is made to an ACP remote procedure, which makes calls to functions in the ACP library to prompt for user names, passwords, etc. When ACP gathers the information required to perform the authorization algorithm, it again calls functions in the library to grant or deny the request. The program then exits.

The distribution policy file **acp_policy.c** is documented in the form of *C* programming language comments. The file **policy.doc** provides a complete description of the available library functions.

Re-Compiling erpcd

You must re-compile **erpcd** if you modify the supplied policy and the **ch_passwd** utility if you changed the name of the ACP password file from **acp_passwd**. The source files are in */annex_root*/**src/erpcd**, where *annex_root* is the directory to which the RAC's source code was copied. To re-compile:

1. **cd to** /annex_root/src**.**
2. **To re-compile only** erpcd**, enter the command:**

   # **`make erpcd`**
3. **To re-compile both** erpcd **and** ch_passwd**, enter the command:**

   # **`make all`**
4. **To install, enter the command:**

   # **`make -f ../make.config -f Makefile install`**

   This saves the old version of **erpcd** as **OLDerpcd** in the installation directory.
5. **Kill the current** erpcd **and start the new one.**

Using the ch_passwd Utility

The **ch_passwd** utility enables users to change their passwords when accessing a RAC through the Access Control Protocol (ACP) security system. This utility affects only passwords in the **acp_passwd** or **acp_shadow** file. Table 6-20 describes the supported argument for **ch_passwd**.

To change a RAC user password, the *username* in the **acp_passwd** file must match the *username* in the **/etc/passwd** (or **/etc/shadow**) file on the ACP host.

If ACP is configured to record password histories, it saves the passwords set via the **ch_passwd** command. ACP keeps these passwords in the **acp_dbm** database on the security host, keyed by user name. The value of the STORED_PASS variable in **acp_policy.h** determines the number of passwords saved. This variable is initialized to 6 for **passwd**/**shadow** files and 0 for **passwd** files alone. A value of 0 disables password history. For more information, see *Enabling and Configuring Password Histories on page 6-8*.

The **ch_passwd** utility first prompts for the old password, and then for the new one. The syntax is:

ch_passwd

A superuser can change the password for any user. The syntax is:

ch_passwd [*username*] [**-s** *directory*]

If you change the name of the ACP password file, you must recompile both **erpcd** and the **ch_passwd** utility. The source files for both are provided with the RAC software distribution and are located in the */annex_root*/**src**/**erpcd** directory. For instructions on recompiling both, see *Configuring the Security Server on page 6-57*.

Table 6-20. Supported Argument for ch_passwd

Argument	Description
-s *directory*	Specifies the directory for the security files (**acp_passwd** and, if configured, **acp_shadow**); defaults to the defined install-annex directory (usually **/etc/annex/**).

Configuring Third-Party Security Regimes

Using Kerberos Authentication

The default ACP configuration authenticates a user by checking the user name and password against entries in the **acp_passwd** file. You can configure ACP to use Kerberos instead of the default authentication process.

When building the ACP/**erpcd** process, a Kerberos library routine (**libkrb.a**) is linked with the ACP code. ACP prompts the user for a user name and password. However, instead of validating the user name and password via the **acp_passwd** file, ACP opens a connection to the Kerberos server and passes the user name and password to the Kerberos library routine for authentication. The Kerberos library routine returns a ticket to ACP indicating whether or not the user is authenticated.

If the Kerberos server authenticates the user, it encrypts the ticket with the user's password before returning it to ACP. If the Kerberos server rejects the user, it returns an error code, and ACP refuses the login attempt. In either case, ACP calls a separate Kerberos routine to destroy the returned ticket after the validation process.

Enabling Kerberos Authentication

To enable Kerberos authentication, you must rebuild the **erpcd** process, and then use this process instead of the default version. To rebuild **erpcd**:

1. **Edit the** make.config **file in the /annex_root/src directory and look for the keyword CFG_STUBLINKING, at the bottom of the file. The line will look like this:**

   ```
   CFG_STUBLINKING = L. -lstubs
   ```

2. **Modify the line in Step 1 to include the** libkrb.a **file, as follows:**

   ```
   CFG_STUBLINKING = Kerberos_lib_path/libkrb.a
   ```

   For *Kerberos_lib_path*, specify the name of the directory containing the **libkrb.a** file. This file is located in the directory in which Kerberos was installed.

3. **Rebuild** erpcd **(see *Re-Compiling erpcd on page 6-110*).**
4. **Install the new** erpcd **in the usual place (saving the old version as a back-up in case of problems).**
5. **Terminate the executing** erpcd **and start up the new version.**

If both the primary and secondary ACP servers are defined, it is important that both the primary and secondary ACP servers support Kerberos authentication for consistency.

Configuring the RAC for Use with Kerberos Authentication

To configure the RAC for use with Kerberos authentication, you must set the parameters as indicated in Table 6-21.

Table 6-21. Kerberos Parameter Settings

Parameter	Setting
enable_security	Yes
security_broadcast	No
port_server_security	Yes
vcli_security	Yes
vcli_password	<unset>
password	<unset>
cli_security	Yes (on each serial port)
port_password	<unset> (on each port)

Configuring the RAC for Use with SecurID

To use the SecurID card, security must be enabled on the RAC:

1. **Set the following RAC parameters to** Y**:**
   - **enable_security**
   - **vcli_security**
2. **Set the following RAC port parameters to** Y **on the global port:**
   - **cli_security**
   - **port_server_security**
3. **Set the RAC port parameter** ppp_security_protocol **to** none **on each port.**

   If **ppp_security_protocol** is set to **none**, the user is prompted again for user name and passcode when tusing the CLI **ppp** command. The user must enter the PIN and SecurID card code for the passcode.

   If you do not want to be prompted a second time, set **ppp_sec_auto** to **Y**.

4. **Set the RAC parameters** password **and** vcli_password **and the port parameter** port_password **to the null string ("") if you want the ACE/Server system to authenticate all login attempts before allowing access to the RAC. Also, do not set a port password in the** acp_passwd **file when using SecurID.**
5. **Enter a host name or IP address for the** pref_secure1_host **and** pref_secure2_host **parameters for each RAC using a SecurID card. The host addresses where each ACP process runs must be activated in the ACE/Server database as clients.**
6. **Set the RAC parameter** security_broadcast **to** N **so that the RAC does not inadvertently contact an ACP process that does user authentication via the** acp_passwd **file unless all the ACP server processes in your network are configured and installed to do user authentication by calling the ACE/Server.**
7. **Set the RAC parameter** acp_key **to its assigned value and enter this value into the** acp_keys **file on the host. Then ACP and the RAC exchange user names and passcodes encrypted with the key.**
8. **Activate valid RAC users in the ACE/Server database with permissions (individual or group) to access the ACP servers. If two ACP servers are used, each user must be allowed access to both servers since either of them can authenticate a user by calling the ACE/Server host.**

Integrating SecurID into ACP

Integrating the ACE/Server software into ACP requires changes to the **erpcd** utility. The following instructions assume that the ACE/Server software is installed in a directory called **/usr/ace** and the RAC software is installed in **/usr/annex**; if your code is installed in different directories, substitute the appropriate pathnames where applicable.

These instructions assume that the software is installed on a UNIX system and that the host tools have been compiled (as opposed to using the binaries from the RAC distribution tape). Also, the target UNIX system requires a development environment (*C* compiler, libraries, etc.).

Make sure the host clock is set correctly.

1. **As a superuser, change into the** /usr/annex/src **directory:**

   ```
   # cd /usr/annex/src
   ```

2. **Create a directory called** sdclient**:**

   ```
   # mkdir sdclient
   ```

3. **Copy the required header files and libraries from the ACE/Server directories:**

   If you have ACE/Server Release 2.1.1 or 2.2:

   ```
   # cp /usr/ace/sdiclient.a sdclient
   # cp /usr/ace/prog/*.h sdclient
   ```

   

   This sequence requires that these files are installed on the slave/client system from the ACE/Server host.

   Make sure the ACE/Server UNIX Client is installed on the system that is running **erpcd**.

4. **Edit the** Makefile **file in the** /usr/annex/src/erpcd **directory:**

   ```
   # vi Makefile
   ```

5. **Kill the existing** erpcd **process (your process number will vary):**

   ```
   # ps -ax | grep erpcd
   25493 IW 0:00 ./erpcd
   25494 IW 0:00 ./erpcd

   25797 p1 S 0:00 grep erpcd
   # kill 25493
   ```

6. **Rebuild** erpcd **(see *Re-Compiling erpcd on page 6-110*).**

7. **If you have linker errors, try running the** ranlib **utility on the** sdiclient.a **library:**

   ```
   # ranlib sdclient/sdiclient.a
   ```

   Then rebuild **erpcd** (see *Re-Compiling erpcd on page 6-110*).

8. **Make sure that ACP is enabled in the** eservices **file (the default is ACP disabled). The default file looks like this:**

```
# erpc  remote programs
#
# prog noverlo verhi  name
#
  1       0     0     bfs
# 3       0     99    acp
```

Enable ACP by removing the pound sign (#) from its entry. The edited file looks like this:

```
# erpc  remote programs
#
# prog noverlo verhi  name
#
  1       0     0     bfs
  3       0     99    acp
```

9. **Run** erpcd **from the current directory or install the newly built** erpcd **in the** /usr/annex **directory by entering:**

```
# ./erpcd
```

or

```
# mv /usr/annex/erpcd /usr/annex/erpcd.old
# make install
# /usr/annex/erpcd
```

10. **Now follow the procedures in the ACE/Server documentation for registering clients and users. The hosts where** erpcd **is running must be registered as clients, and all users with SecurID cards that will log into the RAC(s) must be allowed to access the host clients.**

11. **On the RAC, enable security, configure the preferred security server, and enable CLI security on the ports to be protected by SecurID. If you have a secondary server, the new** erpcd **must be installed on that host and that host must be registered as a client in the ACE/Server database.**

    A sample **admin** session looks like this:

```
admin: set annex enable_security Y security_broadcast N
admin: set annex pref_secure1_host calvin
admin: set port cli_security Y
admin: reset port
```

SecurID Backup Security

The RAC uses the following procedures if the server running SecurID and **erpcd** is down:

- If the RAC finds another server running **erpcd** but not SecurID, ACP will control RAC security.
- If the RAC cannot find another server running **erpcd**, the RAC uses local security.

Using SafeWord AS Security

Enigma Logic's SafeWord AS software verifies the identity of CLI users to permit access to protected systems. When you install SafeWord version 4.x on a central network server and link SafeWord to **erpcd**, SafeWord provides and authenticates fixed or dynamic passwords. SafeWord also supports the RADIUS server.

The difference in the application of SafeWord AS for this release is that the client/server approach now allows **erpcd** to communicate only with the SafeWord server through a client API. The server then interfaces with the database. Another difference is that clients are allowed to be on different hosts.

ACP hosts serve as clients to SafeWord AS.

You can use SafeWord software for:

- SLIP, PPP, IPX, and ARAP sessions only when you start a session from a CLI port. IPX users must connect from Fastlink II or Windows 95 Dialup Networking in terminal mode.
- The ARAP Remote Access client's CCL scripts in versions 1.0 and 2.0 as long as you do not use a SafeWord challenge as part of a dynamic password.

  

  ARAP does not use the **acp_regime** file.
- Macintosh or dial-back users only when the SafeWord user name matches the user name listed in the **acp_userinfo** file.

The RAC supports SafeWord for user authentication only; therefore, authorization is not supported. When you dial in to the network through a RAC, or dial out from a RAC (e.g., if you **telnet** to a port in slave mode), the RAC does not display the SafeWord Failed Access Report. In addition, the RAC does not run the user's SafeWord execute program at the end of the authentication process.

Installing SafeWord AS

To integrate SafeWord into ACP, you must make changes in the **erpcd** utility. You must install SafeWord:

- On a host running **erpcd**.
- On a UNIX system that has a development environment. You compile the host tools on this system (as opposed to using the binaries from the RAC distribution tape).
- In a directory named **/safelog,** and RAC software in the **/usr/annex** directory. If you do not use these directory names, you must substitute pathnames.

Copy the following files during installation from the SafeWord AS installation directory into your **src/enigma** directory:

- NETWORKAPI/swecapi.h
- NETWORKAPI/swecapi.a
- LOCALAPI/custfail.h
- LOCALAPI/custpb.h

For a successful NETWORKAPI installation, you must also install the SafeWord AS client for UNIX machine (option 2 in the installation script) during the installation.

Makefile Switches

Define **Makefile** Switches

Define a new set of switches in the **Makefile** by uncommenting the following lines in **erpcd/Makefile**:

```
#ENIGMAFLAG=-DENIGMA_SAFEWORD -DNET_ENIGMA_ACP
#ENIGMAFILES=../enigma/swecapi.a
#ENIGMACFILES=acp_safeword.c
#ENIGMAOFILES=acp_safeword.o
```

“__assert” comes up undefined (the default). You must uncomment the following line as well:

```
#ENIGMAFLAG = -DENIGMA_SAFEWORD -DNET_ENIGMA_ACP
-DNEED_ENIGMA_ASSERT_PATCH
```

Configuration Management

Place a new file, called **safeword.cfg**, in the installation directory. This file is created as **sid.cfg** when you install the SafeWord AS client.

Move and Rename the **sid.cfg** File

To place the new file in the installation directory:

12. **Copy the** sid.cfg **file into the installation directory.**
13. **Rename the** sid.cfg **file to** safeword.cfg**.**

Create the **safeword.cfg** File

To create the **safeword.cfg** file, use the following example and replace *yourservername* with the name of your SafeWord server:

```
02 Authen. Server (host weight connects port):
     yourservername 0 0 7482

09 User ID Source (USER/SYSTEM): USER

10 Server's System Name: STANDARD

15 Send Status Messages to User: NONE

16 Send Status Messages to Console: ERROR

17 Send Status Messages to log File: NONE

18 Status Message Log Filename: sid.log

23 Status Message Label: sid-7482
```

Integrating SafeWord into ACP

Before using SafeWord, you must integrate it into ACP:

1. **As a superuser, change to the** /usr/annex/src **directory:**

   # **cd /usr/annex/src**

2. **Create a directory called** enigma**:**

   # **mkdir enigma**

3. **Copy the** libidpb.a**,** custpb.h**, and** custfail.h **files into the enigma directory:**

   ```
   # cp /safelog/swecapi.a enigma/swecapi.h
   # cp /safelog/custpb.h enigma/custpb.h
   # cp /safelog/custfail.h enigma/custfail.h
   ```

SafeWord's standard installation provides the **libidpb.a**, **custpb.h**, and **custfail.h** files in the **/safelog** directory.

4. **Edit the** make.config **file in the /annex/root/src directory:**

   ```
   # vi make.config
   ```

5. **Locate the following line, near the bottom of** make.config**:**

   ```
   CFG_STUBLINKING = -L. -lstubs
   ```

   Change the line to include the SafeWord library, as follows:

   ```
   CFG_STUBLINKING = ../enigma/libidpb.a -L. -lstubs
   ```

6. **If** erpcd **is running on the host, kill the existing** erpcd **process (your process number will vary):**

   ```
   # ps -ax | grep erpcd
   25493 ? IW 0:00 ./erpcd
   25797 p1 S 0:00 grep erpcd
   # kill -9 25493
   ```

7. **Rebuild** erpcd**:**

   ```
   # make erpcd.
   ```

   If you have linkage errors, try running the ranlib utility on the sdclient.a library:

   ```
   # ranlib enigma/lipidpb.a
   # make erpcd
   ```

8. **Install** erpcd **into the** usr/annex **directory:**

   ```
   # make install
   ```

9. **Restart** erpcd**:**

   ```
   # /usr/annex/erpcd
   ```

10. **On the RAC, use** admin **or** na **to set** pref_secure1_host **to the Internet address of the host running SafeWord and** erpcd**.**

    You can enter the backup host's address in the **pref_secure2_host** parameter.

SafeWord Passwords

SafeWord provides fixed and dynamic passwords to verify user access to protected systems.

While SafeWord's IDUTIL program allows administrators to create up to three levels of authentication for each user, RAC access allows you to combine one dynamic and one fixed password: you cannot use two dynamic or two fixed passwords for a single authentication process.

If you configure a SafeWord startup file, it will not run when a user accesses a RAC.

Fixed Passwords

System administrators can generate a user's initial fixed password and can set the password's expiration date. When an existing password expires, RAC users can choose a new fixed password:

1. **If the expiration message appears after you enter your username and password, press the** Escape **key and then press** Return.

   The *Old Fixed Password* message appears.

2. **Enter your old password and press** Return.

   The *New Fixed Password* message appears.

3. **Type your new password and press** Return.

   The *Repeat New Fixed Password* message appears.

4. **Type your new password again and press** Return.

   If you completed these steps correctly, the *Permission Granted* message appears. If you did not, SafeWord displays an error message.

Dynamic Passwords

SafeWord generates dynamic passwords using a hand-held password generator called a “token.” The token generates new passwords each time a user wants to access protected systems. Network administrators can configure SafeWord’s dynamic passwords in Synchronous, Semi-synchronous, and Asynchronous modes:

- In Synchronous mode authentication, the token generates a dynamic password that you enter at your terminal.
- In Semi-synchronous mode:
  - Enter the password from your previous session into the token, which then displays a new password.
  - Enter the dynamic password at your terminal.
- In Asynchronous mode:
  - The token displays a string, called a “challenge” before you enter a dynamic password.
  - Enter the challenge into the token, which generates a dynamic password.
  - Enter the dynamic password at your terminal.

You cannot use this (or any other authentication technique that uses a challenge) with PAP or IPX security, because neither PAP nor IPX allows challenges.

For detailed information about configuring and generating fixed and dynamic passwords, refer to Enigma Logic’s SafeWord documentation.

SafeWord Backup Security

The RAC uses the following procedures if the server running SafeWord and **erpcd** is down:

- If the RAC finds another server running **erpcd** but not SafeWord, ACP will control RAC security.
- If the RAC cannot find another server running **erpcd**, the RAC uses local security.

Configuring the IP Basic Security Option (IPSO)

The Department of Defense Basic Security Option for IP identifies the U.S. classification level at which an IP datagram is to be protected and the authorities whose protection rules apply to each datagram, as defined in RFC 1108. The RAC partially implements this security option by adding the IPSO classification level to packets generated by **telnet** or **rlogin** running on a RAC dedicated, adaptive, or CLI port. (The CLI port can be an **auto_detect** or **auto_adapt** port that the user has put into **cli mode** by pressing **Return** when first connected to the port.) The RAC does not add the option to locally generated system packets, such as ICMP messages and RIP updates. Nor does the RAC check incoming packets for the presence of the IP Security Option.

To set the IPSO for packets generated on a port:

1. **Use the** na **utility, the superuser CLI** admin **command, or SNMP to set the RAC parameter** enable_security **to** Y **(the default is** N**).**
2. **Use** na**,** admin**, or SNMP to set the serial line port parameter** ipso_class **to one of the following values:** topsecret**,** secret**,** confidential**,** unclassified**, or** none**. If you specify** none **(the default), the RAC does not add the option to packets.**

The **ipso_class** parameter is also an object in the private-enterprise MIB and can be set via SNMP (for more details, see *The Remote Access Concentrator SNMP MIB Reference*).

The following sample **su** session causes a basic security option of secret to be included in all packets generated by ports 1 and 2.

```
annex: su
Password:
annex# admin
Annex administration Remote Annex Rx.x, 72 ports
admin: set port=1,2 ipso_class secret
admin: set port mode cli
admin:
```

When a router that fully implements IPSO receives a packet with an unacceptable classification level, it sends an ICMP security discard message to the packet's originator. If the RAC receives a discard message, it passes it to the application running on the port that generated the IPSO packet.

Using the ACE/Server

The ACE/Server token is an access control security token which is used to positively identify users of computer systems and secure TCP/IP networks. Used in conjunction with the SecurID card hardware or software access control modules (ACMs), the ACE/Server token automatically generates a unique, unpredictable access code every 60 seconds. The ACE/Server, a daemon that interfaces with the user database, also allows the system administrator to monitor access by running reports of all attempted logins.

Supported ACE/ Server Releases

Remote Annex R4.1 and later releases offer support for ACE/Server Releases 2.1.1 and 2.2.

ACE/Server is supported using ACP and is limited to those UNIX platforms for which the vendor provides client libraries.

Using the SecurID Card

To use the SecurID card feature, you must purchase the ACE/Server software from Security Dynamics. The ACE/Server software includes client software and the SecurID card. The ACE/Server system is designed to prevent any unauthorized access to your network.

SecurID Card Description

The SecurID card is a credit-card sized card containing a microprocessor and an LCD display. This card generates, at a designated interval, a one-time-only, unpredictable code on the LCD display. At the usual system prompt from your RAC, SecurID card users enter a passcode in order to access your protected system.

ACE/Server Authentication

The ACE/Server system provides a unique code, such as the user's PIN number.

Each SecurID card has a unique serial number that identifies it to the ACE/Server.

Assigning a SecurID Card to a User

When you receive the ACE/Server software and SecurID cards, one of the cards is already assigned to the login name *adm* and is enabled for your system administrator. When you become *adm* and execute **sdadmin**, it determines that you have assumed that login name and uses it to find the **adm** card and your correct authority level. The **sdadmin** does not require a passcode entry. By using the one card with administrator authority, at least one person in your SecurID system has the authority to manage the ACE/Server system and all its databases, including changing any relevant information.

Clients

An ACE/Server UNIX Client is a TCP/IP machine connected via a network to the ACE/Server. Whenever a client sends a user-authentication request, the ACE/Server looks up the client's name. For this name to be found, all client's network addresses must be entered into the database, and all the network addresses must be known to the server via the **/etc/hosts** file or your NIS name server.

The SecurID Card User Interface

When a user tries to log into your system, the ACE/Server prompts for the user name and passcode. The user enters the PIN number followed by the current SecurID card code displayed on the SecurID card.

Access Types

The ACE/Server utility authenticates two access types:

- Port-to-port RAC
- Network-to-port

Authenticate SLIP, PPP, and IPX

To authenticate SLIP, PPP, and IPX users:

1. **Log into a CLI port.**
2. **Issue the CLI command** slip, ppp, **or** ipx**.**

   Or

3. **Log into** auto_detect **and** auto_adapt **ports.**
4. **Press** Return **to enter CLI mode, and then issue the** slip, ppp, **or** ipx **command.**

Generating PINs

When...	The...
a SecurID card is assigned to a user	card is set to New-PIN mode in the ACE/Server database.
a user attempts to log on for the first time to a network via a RAC	user enters only the code on the SecurID card (if the PIN has been cleared).
SecurID requires a unique PIN	user must enter a new PIN (user- or system-generated).

The ACE/Server software provides three options related to generating a new PIN:

- CANNOT_CHOOSE_PIN
- MUST_CHOOSE_PIN
- USER_SELECTABLE

Before installing the ACE/Server software, you must determine which of the above options your site will use. The following is an overview of the available options. See the *ACE/Server Manual* for more information.

CANNOT_CHOOSE_PIN

The new PIN is generated by the system and does not give the user the option to select a new PIN. The user is prompted to allow the system to generate and display the new PIN or exit and leave the SecurID card in New PIN mode.

MUST_CHOOSE_PIN

The user must select a new PIN and is not given the option of having the system generate the new PIN. The user is prompted to enter a new PIN containing 4 to 8 alphanumeric characters or exit and leave the SecurID card in New PIN mode.

USER_SELECTABLE

The user is given the option to select a PIN or have the system generate and display a new PIN. The user is prompted to enter a new PIN containing 4 to 8 alphanumeric characters or have the system generate a new PIN and display it or leave the SecurID card in New-PIN mode.

Installation

Copy Files to src/sdclient

During the RAC software installation, you must copy the following library and files from your ACE/Server distribution media to the **src/sdclient** directory:

- **sdclient.a** library
- ***.h** files

The ACE/Server UNIX Client must be installed on each host running **erpcd**. For more detailed information, see the *ACE/Server Installation Guide*.

Each UNIX host running a SecurID-enabled **erpcd** must be enabled as a client in the ACE/Server. For more detailed information, see the *ACE/Server Administration Manual*.

Makefile Switches

Define Makefile Switches

Define a new set of switches in the **Makefile** by uncommenting the two lines that define ACE1_2 or ACE2_0 in **erpcd/Makefile** for ACE/Server V2.1.1 or V2.2. Also, comment out the flag (**PASSFLAG**) that causes the RAC password prompt to appear:

Example

```
#SECURIDFLAG=-DSECURID_CARD -DACE2_0
#SECURIDFILES=../sdclient/sdiclient.a
...
PASSFLAG = -DPASS_SEC
```

to

```
SECURIDFLAG=-DSECURID_CARD -DACE2_0
SECURIDFILES=../sdclient/sdiclient.a
...
#PASSFLAG = -DPASS_SEC
```

To integrate SecurID into ACP, you must make changes in the **erpcd** utility. When you have made the necessary changes to the **Makefile**, rebuild the RAC software. See *Re-Compiling erpcd on page 6-110*.

Define Makefile Switches for non-ANSI Standard Compiler

With a non-ANSI standard C compiler, uncomment the following lines in the **Makefile**:

```
#SECURIDCFILES=fflush.c

#SECURIDOFILES=fflush.o
```

to

```
SECURIDCFILES=fflush.c

SECURIDOFILES=fflush.o
```

New-PIN Mode

If the site allows a user to select a PIN, ACP displays the following text:

```
Enter your new PIN containing 4 to 8 digits,
     or
Press <Return> to generate a new PIN and display it,
     or
<Ctrl-D> <Return> to leave your card in New-Pin mode.
```

The minimum and maximum PIN lengths and the choice between digits only or alphanumeric characters is determined by the system administrator when installing the ACE/Server.

If the user enters a PIN, ACP prompts for the code's re-entry (the typed characters are not echoed back to the terminal). The re-entry prompt displays as follows:

```
Please re-enter PIN:
```

If the user is not allowed to choose the PIN, the following text is displayed:

```
Press <Return> to generate a new PIN and display it,
     or
<Ctrl-D> <Return> to leave your card in New-Pin mode.
```

If the user presses **Return**, the terminal displays the assigned PIN. If the user incorrectly re-enters the PIN or chooses to leave the card in New-Pin mode, the login attempt is terminated.

Using AppleTalk Security

The RAC implementation of ARA provides three areas of security:

- ARA security
- Zone security
- NVE filtering
- Logging

ARA Security

The basic ARA security features are:

- **Username and password authentication**

  The RAC authenticates the client using Apple's DES encryption algorithm. To define a user name and password for a registered (as opposed to guest) user, see *Creating the acp_userinfo File on page 6-74*.

- **Guest access**

  The RAC allows anonymous access to the network. Restrictions can be applied to guests by setting up an ACP *guest* profile with limitations. For more details, see *at_zone on page 6-88*.

- **Connection timer**

  The connection timer is stored in the **acp_userinfo** file. For more details, see *Creating the acp_userinfo File on page 6-74*.

Zone Security

Every user can have a zone list assigned via remote ACP. If a list is not available via ACP, the RAC provides all the zones it has learned from the network. If local security is used, use the per RAC parameter **default_zone_list.** For more details, see *at_zone on page 6-88*.

NVE Filtering

NVE filtering controls a remote access Apple user's view of network resources: when using *Chooser* to select resources, only the resource set defined for the user by the administrator is visible. The administrator can specify the NVE filter on a per-user basis. This feature complements the existing zone list, described above, by offering a higher level of control.

The **nve_filter** entry in the **acp_userinfo** file specifies a list of filters on a per-user basis. For detailed information on creating **nve_filter** entries, see *at_nve_filter on page 6-90*.

This method of limiting NBP traffic is not secure, and can be circumvented by a person willing to write some code to probe the network without using NBP. Also, this feature has no local RAC security equivalent.

Logging

The RAC logs activity and errors from the ARA session. The log is accessed via remote ACP (for more details, see *Using Event Logging on page 4-3*).

Using IPXCP Security

The Internet Packet Exchange Control Protocol (IPXCP) uses PPP security.

Windows ‘95 IPXCP clients must make sure that SPAP security is not enabled on their PCs. SPAP is a proprietary Microsoft security mechanism not available to other systems, such as the RAC.

Using PPP Security

The RAC supports two authentication protocols for PPP:

- Password Authentication Protocol (PAP).
- Challenge-Handshake Protocol (CHAP).

Both of these protocols are run over the PPP link after the LCP negotiations are complete (for more details on using a PPP link, see *Point-to-Point Protocol on page 8-1*).

Password Authentication Protocol (PAP)

PAP is a two-way handshake in which an ID/password pair are exchanged in clear text. Each half of the connection can require security.

If one side of the link agrees to use PAP, after the LCP negotiations are complete, that side will send a user name/password combination to its peer. Upon receipt, the peer authenticates that combination.

When the RAC requests PAP and the peer ACKs the request, the RAC handles the incoming PAP user name/password combination as follows:

- If the **enable_security** and **slip_ppp_security** parameters are set to **Y**, the RAC first tries to authenticate the user name/password combination using ACP. ACP checks the regime file to determine which regime and password file to use (see *Configuring the acp_regime File on page 6-70*). If the ACP server is unavailable, the RAC falls back to local security (i.e., it compares the remote end's user name/password against the global port parameters **user_name** and **port_password**).
- If the **enable_security** parameter is set to **Y** and the **slip_ppp_security** parameter is set to **N**, the RAC uses local security (i.e., it compares the remote end's user name/password against the global port parameters **user_name** and **port_password**).
- If the user name/password combination is valid, the RAC sends a *PAP Authenticate-ACK* message. If the combination is not valid, the RAC sends a *PAP Authenticate-NAK* message.

When the RAC agrees to PAP, it sends the PAP user name/password combination as follows:

- It uses the global port parameter **ppp_username_remote** as the user name.
- It uses the global port parameter **ppp_password_remote** as the password.
- If the user name/password combination is valid, the peer sends a *PAP Authenticate-ACK* message. If the combination is not valid, the peer sends a *PAP Authenticate-NAK* message.

Challenge-Handshake Protocol (CHAP)

CHAP is a three-way handshake that depends on a secret token. The secret token is known to both sides of the peer-to-peer link.

When the challenge is sent by the RAC, the peer responds with a one-way encrypted value. The authenticator then runs the same encryption on the challenge message using the peer's secret token. It then compares the result to the received value. If they match, the authenticator sends a *success* message; otherwise, it sends a *failure* message. Currently, the RAC supports only the MD5 encryption algorithm.

An external mechanism must distribute the secret token to both sides of the link.

ACP is used only when the RAC is authenticating a peer.

CHAP does not use the **acp_regime** file.

The secret token is defined within an *entry* option called **chap_secret** in the **acp_userinfo** file (for more details, see *Creating the acp_userinfo File on page 6-74* and *chap_secret on page 6-93*).

In the following example, user *smith*, when logging into a RAC running CHAP, will have the secret token, *achapsecrettoken*, used in verifying the *response*. The mechanism of receiving a challenge, determining a secret based on the user, and sending the result back to the challenger, is analogous to the user name/password paradigm.

```
user username=smith
     chap_secret achapsecrettoken
end
```

If the **slip_ppp_security** parameter is set to **Y**, the RAC sends the username, challenge message, and challenge response to ACP for authentication. The RAC uses local security when ACP is unavailable and the **port_password** parameter is set; local security ignores the user name and checks the *response* against **port_password** using the **port_password** to encrypt the challenge message. If the **port_password** parameter is not set, the link fails.

Receiving a CHAP Challenge

When the RAC receives a *challenge*, the challenge and the secret token (the **ppp_password_remote** parameter value) are used to generate a *response* message (the *name* field is set to the **ppp_username_remote** parameter value). The value in the *response* message is a result of running MD5 encryption on the secret token and the value in the *challenge* message. If the RAC receives a *success* message, the link enters (or remains in) NCP negotiation; otherwise, the link is terminated.

The RAC negotiates an authentication *challenge* from a peer only if the **ppp_password_remote** and **ppp_username_remote** parameters are set for this session.

CHAP does not use the **acp_regime** file.

Sending a CHAP Challenge

When the RAC sends a *challenge* to the peer (remote node requesting a link), it includes the **chap_auth_name** parameter value as the *name* field and a randomly generated number as the *value* field.

If ACP is used, after receiving the peer's *response*, the RAC passes the following items to the ACP server: CHAP username, challenge, and the peer's response (ID and challenge response).

The ACP server combines the secret, challenge, and ID to create an expected response. The ACP server then compares the response it created with the one it received from the RAC.

If the responses are identical, the ACP server sends a success code to the RAC. If not, it sends a failure code.

Upon receiving a success code, the RAC allows the link to be established. When receiving a failure code, the RAC prevents it from being established.

The RAC sends a *challenge* only if the **enable_security** and **slip_ppp_security** parameters are set to **Y**, the **ppp_security_protocol** parameter is set to **chap**, and CHAP is ACKed during LCP. If the RAC is ACKed for CHAP, it will seek only one valid *response*.

The RAC terminates a link if it cannot authenticate a *challenge*. If the RAC does not receive a *response* to a *challenge* within the allotted time-out, it re-issues the *challenge* for the defined number of retries.

ACP logging for CHAP includes the standard PPP login and reject. It also logs whether or not a chap secret was found in the **acp_userinfo** file.

Re-issuing a CHAP Challenge

By default, the RAC sends a *challenge* only once, at the time the link is established. Optionally, you can configure the RAC to re-issue a challenge at random intervals ranging from one second to the maximum number of seconds you specify. To do so, set the RAC security parameter **max_chap_chall_int** to a value between 1 and 65535 (approximately 18.2 hours). The following example sets the maximum interval to 3600 (two hours). The RAC sends a *challenge* at random intervals between 1 second and two hours over the course of the connection.

```
admin: set annex max_chap_chall_int 3600
```

The **max_chap_chall_int** default, which is 0, disables the re-issuing of challenges.

Using the PPP Security Parameters

There are a variety of settings one can choose when configuring the RAC for PPP security. Table 6-22 lists the possible combinations of PPP security parameter settings and their effect on RAC activity.

The following two statements are true for all cases listed in Table 6-22. First, if a remote side of a link demands PAP, the RAC uses **ppp_username_remote** and **ppp_password_remote** for the username and password. Second, if **ppp_username_remote** and **ppp_password_remote** are not set, the connection fails.

Table 6-22. PPP Security Parameters and Their Effect on RAC Activity

If:	Then:
enable_security = N **ppp_security_protocol** = n/a **slip_ppp_security** = n/a	Request no PPP security incoming. Do not log accesses in the ACP log file.
enable_security = Y **ppp_security_protocol** = none **slip_ppp_security** = Y	Request no PPP security incoming. Log accesses in the ACP log file.
enable_security = Y **ppp_security_protocol** = none **slip_ppp_security** = N	Request no PPP security incoming. Do not log accesses in the ACP log file.
enable_security = Y **ppp_security_protocol** = pap **slip_ppp_security** = Y	Use ACP for incoming user name and password. Log accesses in the ACP log file.

(continued on next page)

Table 6-22 . PPP Security Parameters and Their Effect on RAC Activity (continued)

If:	Then:
enable_security = Y **ppp_security_protocol** = pap **slip_ppp_security** = N	Use **port_password** for incoming password. Do not log accesses in the ACP log file.
enable_security = Y **ppp_security_protocol** = chap **slip_ppp_security** = Y	Use ACP for authentication, sending username challenge, and challenge response.
enable_security = Y **ppp_security_protocol** = chap **slip_ppp_security** = N	Use **port_password** for incoming secret token. Do not log accesses in the ACP log file.
enable_security = Y **ppp_security_protocol** = chap-pap **slip_ppp_security** = Y	Request CHAP in negotiation; if it is NAKed by peer, request PAP. If using PAP, use ACP for incoming user name and password. If using CHAP, use ACP for authentication, sending username, and challenge response. Log accesses in the ACP log file.
enable_security = Y **ppp_security_protocol** = chap-pap **slip_ppp_security** = N	Request CHAP in negotiation; if it is NAKed by peer, request PAP. Use **port_password** for incoming password/secret token and ignore incoming user name. Do not log accesses in the ACP log file.

Dynamic Allocation of Network Addresses

Introduction to DHCP

The Dynamic Host Configuration Protocol (DHCP) enables dynamic IP addressing for remote access clients establishing a PPP connection to a RAC. This eliminates the need to assign an IP address manually (and the subsequent need to reconfigure and reboot) each time that a host is added or moved to a new subnet location.

The RAC acts as a DHCP "client-by-proxy," requesting and accepting a dynamic IP address from a DHCP server on behalf of a dial-in client. The term *DHCP client* always refers to a RAC acting as a DHCP client-by-proxy.

DHCP is enabled by setting the RAC parameter **address_origin** (which has replaced the previously existing parameter, **dialup_addresses**) to **dhcp**, or by setting the **Remote Address** field in the **acp_dialup** file to **dhcp** (see *Creating the acp_dialup File on page 6-143*).

When DHCP is enabled, a DHCP client seeks to discover a DHCP server by requesting an IP address first from the DHCP server specified by **pref_dhcp1_host**, then, if that server does not respond, from the DHCP server specified by **pref_dhcp2_host**. The DHCP server checks the subnet of the requesting DHCP client, allocates an IP address from a pool of IP addresses made available for that subnet, and offers it to the requesting DHCP client. The DHCP client uses the allocated IP address for an interval of time called a "lease," which is maintained for as long as the remote client connection is active, or until the DHCP client terminates the serial connection. When the lease expires, the DHCP client returns the address to the pool of dynamic addresses maintained by the DHCP server. The DHCP server can then reuse that IP address, allocating it to another DHCP client which requests an IP address.

Unsupported Features of DHCP

Some aspects of DHCP are not relevant to its use on a RAC:

- A RAC does not implement the BOOTP Relay function. A RAC does not support host-based DHCP requests, e.g., a DHCP client operating on a PC, expecting the RAC to relay host-generated DHCP protocol messages to the DHCP server.
- The DHCP client cannot configure the dial-in host.
- DHCP clients do not use automatic address allocation, which assigns permanent IP addresses, nor do they retrieve statically configured IP addresses from the DHCP server.

Cautions

- If the DHCP client is invoked, but is unable to obtain an address from a DHCP server, it syslogs the condition *Client did not receive a DHCPOFFER*; the DHCP client cannot supply an address to the IPCP, and the remote connection is terminated.
- It is possible that the DHCP client will be unable to discover a DHCP server and obtain an IP address from it before the PPP connection establishment times out and terminates.

Creating the acp_dialup File

The **acp_dialup** file resides in the RAC install directory. Any ACP dial-up address request that comes from the RAC includes the RAC address and port number, and an associated user name, which are used as keys in this file. Once the keys are matched, the corresponding dial-up addresses are returned to the caller on the RAC. If no match is found, the RAC uses the port's **remote_address** and **local_address** (for more details, see *Determining Dial-up Addresses Using the acp_dialup File on page 6-145*).

The **acp_dialup** file contains the following fields: *User*, *RAC*, *Remote Address*, and *Local Address* (optional). If a local address is not specified, the local address sent back to the RAC is that RAC's IP address. For example:

```
#User   RAC                  Remote address Local address
smith   100.30.200.39        100.30.200.45  100.30.200.46
green   *                    100.30.200.48
harris  mars                 100.30.200.55  100.30.200.40
frank   *                    dhcp
```

You can specify the RAC by name, IP address, or wild card (*); the wild card means that any incoming address request with that user name will match. The file format allows one entry per line; the RAC ignores any data following the comment character (#); a newline character terminates an entry.

In the previous example:

- User *smith* can make a dial-up address request from RAC 100.30.200.39. The remote address is 100.30.200.45; the local address is 100.30.200.46.
- User *green* can make a dial-up address request from any RAC. The remote address is 100.30.200.48; the local address is the address of the RAC from which the request originates.
- User *harris* can make a dial-up address request from the RAC named *mars*. The remote address is 100.30.200.55; the local address 100.30.200.40.
- User *frank* will obtain a remote address from a DHCP server.

Determining Dial-up Addresses Using the acp_dialup File

When the port parameter **address_origin** is set to **acp**, the *local* and *remote* field settings in the **acp_dialup** file supersede the values set in the **local_address** and **remote_address** port parameters.

When **address_origin** is set to **acp**, the RAC searches for the remote client's user name in the **acp_dialup** file. RAC behavior at this point depends on whether or not the RAC finds a matching user name in **acp_dialup**:

- If the RAC does find a matching user name in **acp_dialup**, it looks at the corresponding *local* and *remote* address fields.
  - If both of these addresses are set in the **acp_dialup** file, the RAC forces the use of these values over the settings in the **local_address** and **remote_address** port parameters.
  - If the *local* address field is not set, but the *remote* address field is set, the RAC forces the use of the *remote* address field setting for the remote address and forces the local address setting to be the RAC's IP address.
  - If the *remote* address field is set to **dhcp**, a remote address is allocated dynamically by a DHCP server (see *Dynamic Allocation of Network Addresses on page 6-142* for a complete description).
- If the RAC does not find a matching user name in the **acp_dialup** file, it looks at the **local_address** and **remote_address** port parameters.
  - If the **local_address** and **remote_address** parameters are set, the RAC uses these values for the local address and remote address.

- If the **local_address** and **remote_address** parameters are not set, the RAC negotiates for both the local and remote address values with the remote PPP client. (If these conditions are true for a remote SLIP client, the connection is denied.)
- If the **local_address** parameter is set but the **remote_address** parameter is not set, the RAC forces the use of the value in the **local_address** parameter and negotiates for the remote address value with the remote PPP client. (If these conditions are true for a remote SLIP client, the connection is denied.)

Using Filters for Security

The RAC's implementation of filtering allows you to improve the security of an internal network by preventing potentially dangerous traffic from crossing it. For example, you might want to prevent an outside host from using the Network File System (NFS) protocol or the Trivial File Transfer Protocol (TFTP) to access an internal network, since these protocols have no built-in security and can alter local data. Or, you might want to use filtering to prevent users on your internal network from accessing external hosts and services.

An effective way to provide this kind of protection is to pick one RAC on the internal network to be the network's *chokepoint* or *firewall* through which all traffic to and from external networks must pass. Then, configure filters on that RAC to block undesirable packets (see *Configuring Security on page 6-1*). You can also use filtering to log (in the **syslog** file) traffic for security or network-management purposes. Finally, you can use filters to determine what constitutes traffic on a dial-out serial port.

You can also use filtering to log (in the **syslog** file) traffic for security or network-management purposes.

For the RAC, filtering applies only to the following interfaces:

- lo0
- en0

The only option available when enabling or disabling filters for the RAC through the **cli** command is **all (*),** which means that you can either enable or disable all filters for the specified interface.

You can apply filters to individual users on a session-specific basis through the **acp_userinfo** file. See *filter on page 6-84* for complete details.

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.

Include and Exclude

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 - i.e., 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, since the interactions are subtle and can be confusing even for a networking expert.

Accessing the Filter Subcommands

After enabling filtering, access the filter subcommands using the CLI superuser **filter** command. Invoking this command with no arguments puts you in the filtering subsystem, where you can issue any of the subcommands summarized in Table 6-23.

The **add** and **delete** subcommands affect both the currently running RAC configuration and the configuration stored in non-volatile memory. Rebooting the RAC or issuing a **reset** command is not necessary.

You cannot access the **filter** command from **na**, and you cannot save the filter parameters to a file using the **na** command **write**.

The following example shows entering the filtering subsystem and issuing the **list** subcommand to display the current filters:

```
annex# filter
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
filter:
```

To return to the superuser CLI from the filter subsystem, use the filter subcommand **quit**:

```
filter: quit
annex#
```

You can also issue **filter** subcommands directly at the CLI superuser prompt by using the syntax:

filter *subcommand*

The following shows the **list** subcommand issued from the CLI superuser prompt. When **list** completes, you return to the CLI superuser prompt.

```
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#
```

Table 6-23. Summary of filter Subcommands

Subcommand	Description
add	Adds filters and automatically enables them.
delete	Deletes filters.
disable	Disables filters but does not delete them.
enable	Enables filters.
help	Displays a one-line description of one or all **filter** subcommands.
list	Displays filters.
quit	Exits the filtering subsystem, returning control to the CLI.
usage	Displays the syntax for one or all **filter** subcommands.

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 is 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 6-24 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 6-24. 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.

Configuring Security for the RAC FTP Daemon

When a new FTP session is initiated, the FTP daemon registers the source host with the **who** database. A subsequent **who** displays:

```
annex: who

Port   What   User    Location    When    Idle Address
1      PSVR   ---     jdcm        4:06am 3:13 192.9.200.60
                      console
v1     CLI    hobbes  ---         4:07am       192.9.200.60
v2     FTPD   ---     ---         ---     :01  bryce
annex:
```

In the above sample command display, since the user has not yet logged into the **ftp** session, no user name appears in the *User* field. If the **enable_security** parameter is set to **Y** but a preferred security server is not configured, or if **enable_security** is set to **N**, the user is prompted for a user name and a password. The RAC accepts any user name, but grants FTP access only after checking the password against its administrative password. If the RAC grants access, the user's name appears in the **who** command display.

If the **enable_security** parameter is set to **Y** and a preferred security server is configured, the RAC calls the **ppp_security** function in the **acp_policy.c** file with the user's name and password as entered and the service set to SERVICE_FTP. If ACP grants access, the FTP daemon asks for an "account." The RAC compares the text entered at this prompt against its administrative password for an added level of security.

If the **enable_security** parameter is set to **Y** and the preferred security server is not reachable, the RAC denies access to the FTP daemon.

When the validation process is complete, the RAC logs FTP access in the ACP logfile and updates the **who** command display to look something like this:

```
annex: who

Port   What   User    Location   When    Idle Address
1      PSVR   ---     jdcm       4:06am 3:37 192.9.200.60
                      console
v1     CLI    hobbes  ---        4:07am      192.9.200.60
v2     FTPD   hobbes  ---        2:43pm      bryce
annex:
```

The RAC FTP daemon is compatible with all versions of UNIX **ftp.**

You can completely disable the RAC FTP daemon by setting **ftpd** in the **disabled_modules** parameter.

Logging Security Events

Host-based security can generate audit trails of user activity. Each time the security server grants or denies a request for user access, the security server logs it. Each event is logged as a message in an ACP log file.

The ACP log file can be the default **acp_logfile** located in the **/usr/annex** directory or a RAC-specific log file. A RAC-specific log file is created by uncommenting the following statement in the **acp_policy.h** file:

```
#define SEPARATE_LOGS
```

Once this statement is uncommented, a RAC-specific log file is created with the name **acp_logfile.*Annex_IPaddress*** in the **/usr/annex** directory.

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

- *IP address of the RAC*
- *Sequence number*
- *Port type*
- *Date*
- *Time*

- *Module*
- *Event*
- Packets in
- Packets out
- Bytes in
- Bytes out
- *Protocol-dependent information*
- *Username*

For the RAC, the following additional fields are added:

- *Calling number*
- *Called number*
- Called subaddress
- *Bearer*

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

The parser of the **acp_userinfo** file generates log messages if an error is detected when processing a user's profile.

Chapter 7
Digital Modems

This chapter discusses digital modem support for the RAC and covers the following topics:

- *Digital Modem Support*
- *Busying-Out Modems Manually*
- *Busying-Out DS0 Channels Automatically*
- *Customizing Modem Configuration*
- *Configuring Modem and Channel Error Detection*
- *Displaying and Changing Modem Status*
- *Modem Statistics*

Digital Modem Support

The RAC supports up to 62 digital modems by employing up to two modem cards with 24 or 31 modems per card. This means that each RAC can have a possible configuration of 0, 24, 31, 48, 55, or 62 digital modems. Configurations in multiples of 24 are available to serve the needs of T1 users, while configurations in multiples of 31 are available to serve the needs of E1 users. Up to 2304 digital modems may be contained in a standard eight-foot telco cabinet (48 modem cards, each with 48 digital modems).

All call setup is performed by the WAN modules, not by the digital modems themselves. This includes signaling, detecting dial tones and busy signals, and interpreting and generating DTMF (Dial Tone Multiple Frequency).

Although the digital modems have no physical serial ports associated with them, the serial line card (SLC) emulates serial ports. Each WAN interface has a corresponding SLC which handles low level input and output for the interface.

Digital Modem Assignments

Digital modems are assigned to calls from a pool of available modems.

Modem Assignments

Any available modem can be assigned to any call, regardless of the channel or WAN on which the call arrives. Each call is assigned a modem at random from a pool of modems that is available across both WAN interfaces.

If no modem is available for an incoming call, the RAC rejects the call; PRI users will hear a busy tone, while CAS users will have the DS0 channel they're using timed out.

Spare Modems

If a modem in a PRI/T1 environment fails, it is removed from the pool and the RAC continues to support a full PRI of 23 B channels. If two modems fail in a PRI/E1 environment, the RAC continues to support a full PRI/E1 of 30 channels.

DNIS and ANI

DNIS and ANI are used by the RAC's SPBs to determine how to handle an incoming call.

Dialed Number Information Service (DNIS) provides information about the called number for channelized T1 that is carried by the D channel in PRI.

Additional Number Information (ANI) provides information about the calling number when the R2 protocol is in use. Like the information furnished by DNIS, this information is carried by the D channel in PRI.

Busying-Out Modems Manually

Issuing a request to busy-out a modem causes the modem to stop accepting additional calls. If you attempt to busy-out a modem that is currently handling a call, the request will not be honored until the call terminates. If required, you can terminate the call immediately by issuing a **reset modem** command after issuing the busy-out request.

To configure a modem to be available or unavailable, follow the procedure below, using the indicated **admin** or **na** commands:

To busy-out a modem:

1. **Start** na **or** admin **mode.**
2. **Enter** modem **<**range**> .**
3. **Enter** set modem busy_out **[**y**|**n**] .**
4. **Enter** show modem **.**

After completing this procedure, a message indicates that you should reset the modem. You can ignore this message; the configuration will be performed the next time a call comes in.

For example:

```
admin#: modem modem_list
admin#: set modem unavailable [y|n]
admin#: show modem
```

The first command defines a set of one or more modems by number, e.g., 1-5 or 1,2,3,4,5.

Depending on its value, the second command makes available or unavailable the specified modem set, either removing or restoring them to the pool of modems available for allocation.

The third command displays the status of the defined modem set.

Busying-Out DS0 Channels Automatically

Since modems are assigned to incoming calls at random from the pool of available modems, PRI B channels and CAS DS0 channels are not associated with specific modems on the RAC. Any call that is carried to the RAC on one of these channels can use any available modem. Since there's no way of knowing which modem a channel will be assigned to, it is not an option to busy-out individual modems. This is an issue in a case where all of the modems in the pool are handling calls while one or more incoming channels are still available to carry calls. In such a case, PRI B channels transmit a busy signal to a caller attempting to make a connection. The CAS protocol, however, does not transmit a busy signal in response to an attempted connection, and, as a result, the call is not answered. The solution to this problem is to busy-out the affected DS0 channels.

Automatic Busy-Out Parameter

The WAN parameter **auto_busyout_enable** indicates whether available DS0 channels will be busied out when the last available modem in the pool is used. Once busied out, in order for the DS0s to be usable again, a network administrator must set the DS0s to unbusy manually, using the **wan busy** command, or must reboot the system in order for the DS0s to be usable again. You can set this parameter to **Y** or **N**. (Refer to the *Remote Access Concentrator Software Reference Guide* for complete details.)

Busy-Out Command and Status

You can use the following command to busy-out specific DS0 channels on a particular WAN module:

```
wan interface=<interface number>[busyout | unbusyout]\
     channel= <channel range>
```

where:

interface is the keyword that indicates which WAN module to affect.

<interface number> indicates the WAN module to affect; can be **1**, **2**, or **all**.

[busyout | unbusyout] is the command used to affect the WAN module.

channel indicates which DS0(s) to affect.

<channel range> indicates which DS0s to affect; **all** is a valid range.

When used without the *busyout*/*unbusyout* keywords, this command sets the default WAN module(s) on which to operate. The interface=<interface number> option is also added to the WAN commands to indicate the WAN module on which to operate.

Special Considerations for Hunt Groups

Since CAS does not return busy signals to CO switches, a hunt group that spans multiple T1/E1 lines (whether or not it spans multiple RACs) might fail if it is not configured with enough modems to handle the number of channels in use. If enough modems are available for the channels in the hunt group, no problem occurs because trunk capacity is reached before all the modems are occupied.

If Multilink PPP (MP) is in use, and the MP system dialing in to the RAC can direct subsequent calls to the same unit, a problem might occur if the RAC runs out of available lines. If you do not have control over where subsequent MP links are directed, whether or not MP works correctly depends on the RAC to which the links are directed. In any case, the MP link will not fail; if the link is directed to a different RAC, the remote unit is informed that the link was not created correctly, and the link is deactivated.

Customizing Modem Configuration

For normal operation, digital modems do not require custom configuration. These configuration instructions are provided for users who have a compelling need to alter the digital modems' behavior at a basic level.

The **%digital_modem** section of the **config.annex** file establishes user-defined modem types and modifies the digital modem configuration parameters described in *Digital Modem Configuration Parameters on page A-1*.

An example of the **%digital_modem** section follows below:

```
%wan

begin_session credit_card
called_no 103
call_action modem
set type_of_modem credit_card
end_session

%digital_modem

type_of_modem credit_card
config_bytes 12=27,99,4
end
```

When the RAC receives a call on a number that ends with 103, it is treated as a modem call, with a user-defined modem type named *credit_card*. Before the modem answers the call, its configuration parameters 12, 13, and 14 are set to 27, 99, and 4, respectively.

Note that configuration parameters are modified in the **%digital_modem** section by specifying the number of the parameter, then specifying a value that corresponds to it. You can modify a series of parameters by specifying the number of the first parameter in the series, then specifying the corresponding values for all of the parameters in that series.

You can set parameter numbers and parameter values in decimal base (the default), octal base (values beginning with 0), or hexadecimal base (values beginning with 0x) numbers. Binary numbers are not used with the **config_bytes** keyword. Each parameter value in a series can be set separately, so it is not necessary to use the same base for a single series of configuration parameters.

Configuring Modem and Channel Error Detection

The RAC detects call errors by comparing counts of different events in the progress of calls coming in through channels (for CAS only) and modems. The error counts are kept separately because modems are assigned to channels dynamically, and are not mapped directly. Mismatches between the event counts for a call on a channel or modem indicate errors in the call. A call fails if it does not reach DCD assertion, or, in the case of a non-modem call, if it is not assigned to a device successfully. Failed calls for each channel or modem are counted as well, and when the count of consecutive failures reaches a specified threshold value, three events result: a syslog message is sent, an SNMP trap is sent, and the corresponding channel or modem is taken out of service (busied-out). The intent of this is to notify a network administrator who can examine and correct the problem that is causing the errors.

Busied-out channels and modems can be returned to service with the appropriate CLI commands, or by rebooting the RAC.

Counting the Call Events

The following events are counted for each incoming call on a channel:

Event	Description
SEIZE	This message from a WAN module to a WAN manager process on the main processor indicates that the central office switch has sent a channelized T1/E1 seize event.
RING	This message from a WAN module to a WAN manager process on the main processor indicates that a call has arrived.
Modem/device assignment (start of session)	
CONNECT	This message from the WAN module to the WAN manager process means that the has accepted the call and the WAN module and the CO have agreed that the call has been established.
Chat process	This process tells the modem to answer and waits for DCD.
DCD assertion	The modem has connected successfully with the remote modem.

Modems count only three of these events: modem assignment, the chat process, and DCD assertion.

For non-modem calls, the chat process and DCD assertion are not counted or don’t occur at all. The call is considered to be successful once a device has been assigned.

Some possible causes for counts which are less than that of their preceding event are shown below:

Missing Event	Description	Cause
RING	Call aborted, “short seize.”	The T1/E1 is misconfigured, or the caller hung up.
Modem assignment	No modem available.	Not enough modems, or some modems have been removed from the pool
CONNECT	Call aborted, “short seize.”	The T1/E1 is misconfigured, or the caller hung up.
Chat process finished	Modem error or hangup.	This may be caused by the modem.
DCD assertion	Modem did not establish a connection with its peer.	There are many possible causes for this, among them: • Modem error. • Bad data from a bad channel. • Low end-to-end line quality. • Noise. • The caller hung up. • The calling device wasn’t a modem.

Specifying the Consecutive Failure Threshold

You can configure error reporting using the SNMP Annex parameters **ds0_error_threshold** for channel event counts and **modem_error_threshold** for modem event counts. Each parameter specifies the number of consecutive failures to reach DCD assertion that is allowed before busying out the channel or modem and sending a syslog message and SNMP trap. Each value is a 16-bit unsigned number, allowing a range of values from 0 to 65535. The default value for both parameters is 0, meaning that no action is taken for failures.

Displaying the Call Event Counts

You can view call event counts using the superuser CLI commands **inchanerr** and **inmoderr**.

The **inchanerr** command shows the progress of incoming calls for each CAS channel on a WAN interface, displaying counts for the seize, ring, assignment, connect, answer (chat process), and DCD events, as well as the consecutive failure count. The syntax for the command is:

```
inchanerr wan=<range> channel=<range>
```

The channel event counts are displayed by WAN number and channel number. Event counts are not displayed if the WAN uses PRI.

The **inmoderr** command shows the progress of incoming calls for each digital modem on the RAC, displaying counts for the assignment, answer (chat process), and DCD events, as well as the consecutive failure count. The syntax for the command is:

```
inmoderr modem=<range>
```

Issuing either command without arguments displays the event counts for all channels or all modems, respectively. Issuing either command with the flag **-f** displays only those channels or modems that have consecutive failure counts greater than 0.

Resetting the Consecutive Failure Counts to Zero

You can reset t he consecutive failure counts for a range of channels or modems by issuing the **inchanerr** command or the **inmoderr** command with the **-c** flag. The first example below clears the consecutive failure counts for channels 1 through 23 on WAN 1; the second example clears the consecutive failure counts for modems 1 through 48:

```
inchanerr -c wan=1 channel=1-23
inmoderr -c modem=1-48
```

Consecutive failure counts are reset in the following cases as well:

- A successful modem call clears the consecutive failure count for the corresponding channel and modem.
- A successful non-modem call clears the consecutive failure count only for the corresponding channel.

Displaying and Changing Modem Status

Displaying Internal Modem Information

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.

To display modem status:

1. **At the prompt, enter** modem**. This shows the value of the type_of_modem parameter, which is BAY_5399_DEFAULT by default. The** modem **command is a superuser CLI command.**

modem Command Syntax

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

Don’t enter a space between the argument and the value for the range of affected modems.

Table 7-1 describes the modem command used with and without its arguments.

Table 7-1. Arguments for the modem Command

Argument	Description
none	When no arguments are given, the command lists the names of the internal modem types defined in the RAC configuration file, excluding those not in use for this machine. (Modem types are not in use unless they are also 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*.
-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 (i.e., 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 the 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, displays *BAY_5399_DEFAULT*.
-m*number-range*	Displays whether or not 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, then this information is displayed for all the internal modems.

(continued on next page)

Table 7-1 Arguments for the modem Command (continued)

Argument	Description
-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, then this information is displayed for all the 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

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

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

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

The following **modem -m** command 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#
```

The final 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.
```

Modem Statistics

Modem statistics are a subset of call statistics. Refer to the *Remote Access Concentrator SNMP MIB Reference* for more information.

The modem statistics that are tracked are:

- modem number
- modem state

- begin connect speed
- current speed
- begin signal quality
- current signal quality
- begin rx (receive) line level
- current rx (receive) line level
- connection type
- blocks sent
- blocks received

Chapter 8
Point-to-Point Protocol

This chapter describes how to configure the Remote Access Concentrator (RAC) for access by remote nodes using Point-to-Point Protocol (PPP). PPP lets you use the RAC to provide access for remote nodes through the telephone network, using the RAC channels or WAN/CAS DS0 channels in combination with:

- Analog modems (V.34, V.32)
- ISDN BRI lines with terminal adapters using a rate adaptation protocol
- ISDN BRI lines using ISDN adapter cards or personal ISDN routers (Nautica Series CLAM)
- ISDN PRI lines using ISDN routers (Nautica Series Marlin)
- Switched 56 and switched 64 synchronous lines

Point-to-Point Protocol

PPP is a standardized method for transmitting data from multiple protocols over asynchronous and synchronous point-to-point links. Data transmission and reception takes place only between the nodes at the ends of the link. PPP provides four functions:

- Asynchronous High-level Data Link Control (AHDLC) or standard HDLC to encapsulate the packets
- Link Control Protocol (LCP) to establish the connection between peers
- PAP and CHAP security
- A family of Network Control Protocols (NCPs) to configure network interfaces

PPP features include:

- FCS (CRC-16) error checking
- Dynamic negotiation by each end of the connection for a mutually acceptable set of features for that connection
- Large Maximum Receive Unit (MRU) size, compared to SLIP negotiations start at 1500

PPP Configuration Overview

To configure the RAC for PPP sessions:

1. **Assign IP addressing.**
2. **Edit the** config.annex **file to define Session Parameter Blocks (SPB).**
3. **Review the default global port parameters, then reset the parameters required for the PPP configuration.**

Step 1: Decide How to Assign IP Addressing

All IP addressing for PPP links is based on the value of the **address_origin** parameter, which determines the method that the RAC uses to assign IP addresses. The addressing methods and their corresponding **address_origin** values are as follows:

- Setting the **address_origin** parameter to **dhcp**. This enables dynamic IP addressing, using the Dynamic Host Configuration Protocol (DHCP). Refer to *Dynamic Allocation of Network Addresses on page 6-142* for a complete description of dynamic addressing.

- Using the **acp_dialup** file. Setting the **address_origin** parameter to **acp** causes the IP addressing for individual users to be determined by the **acp_dialup** file. This method also enables dynamic IP addressing via DHCP. Refer to *Creating the acp_dialup File on page 6-143* for a complete description of the **acp_dialup** file. For users that are not in the **acp_dialup** file, the local EEROM addresses are used as described below.

  

  In this book, this method is also referred to as *dial-up addressing*.

- Using the per-B-channel port parameters **local_address** and **remote_address**. Setting the **address_origin** parameter to **local** (the default) causes IP addresses to be assigned according to the values of the port parameters. Refer to *The Remote Access Concentrator Software Reference* for a description of these parameters' use.

  

  In this book, this method is also referred to as *fixed addressing*.

You can choose to configure the RAC for any one of the methods, but setting **address_origin** to **dhcp** has priority over addressing using the **acp_dialup** file, which has priority over addressing using the asynchronous port parameters.

For information about how the RAC operates when both dial-up and fixed addressing are enabled, see *Determining Dial-up Addresses Using the acp_dialup File on page 6-145*.

Setting address_origin to dhcp

Setting the **address_origin** parameter to **dhcp** enables dynamic IP addressing. DHCP allows the central site to maintain a common pool of addresses for dialup IP users, and elimiates the need to statically configure these addresses on each new RAC installed.

DHCP also permits dialup users to reclaim previously used IP addresses if not reassigned, which makes applications more resilient to link drops.

When DHCP is enabled, the RAC acts as a DHCP client-by-proxy for IPCP, requesting a remote PPP client address first from the DHCP server specified by the **pref_dhcp1_addr** parameter, then, if that server does not respond, from the DHCP server specified by the **pref_dhcp2_addr** parameter. For more information on DHCP, see *Dynamic Allocation of Network Addresses on page 6-142*.

About Addressing via the acp_dialup File

Addressing using the **acp_dialup** file offers the ability to assign IP addresses to individual users. When the **address_origin** parameter is set to **acp**, the RAC uses the host-resident **acp_dialup** file to handle IP addressing. The file resides in the RAC install directory. For information on making entries into the **acp_dialup** file, see *Creating the acp_dialup File on page 6-143*.

Any ACP address request that comes from the RAC includes the RAC address and an associated user name, which are used as keys in this file. Other keys include:

- Time of Day
- Port Number
- Port Type

Once the keys are matched, the corresponding user addresses are returned to the caller on the RAC. If the keys are not matched, the EEROM addresses are used if set to non-zero.

You can also use the **acp_dialup** file to enable dynamic IP addressing for individual users, by setting the remote address field of the file to **dhcp**. For complete information, see *Dynamic Allocation of Network Addresses on page 6-142*.

About Addressing Using Port Parameters

Setting the **address_origin** parameter to **local** causes IP addressing for the RAC to be controlled by the values of two parameters, port **local_address** and WAN b **remote_address**. This method of fixed IP addressing associates IP addresses with specific ports.

Step 2: Edit the Configuration File

Session Parameter Blocks (SPBs) are structures within the configuration file. SPBs enable a RAC to handle one or more types of calls differently from the default call handler. If you need to create any SPBs to handle special PPP details, do it now.

See *Configuring the WAN Interfaces, Global Ports, and Sessions on page 5-1*, for detailed information on SPBs.

To edit the configuration file:

1. **Open the configuration file.**

   The configuration file is located in the install directory on the UNIX load host (default filename is **/usr/spool/erpcd/bfs/config.annex**). Use any system editor (e.g., **vi**, **textedit**) to edit the file.

2. **Go to the** wan **or** pri **section in the file. This section contains examples. Do the following:**

   - Read the information that precedes each sample SPB provided in the configuration file. You should be familiar with this information before you enable or disable a default SPB.
   - Remove the comment character (#) from the beginning of each line of any sample SPB you want to enable.
   - Enter a comment character (#) at the beginning of each line of each SPB you want to disable.
   - Enter any new SPBs.

3. **Save the file.**
4. **Issue a** reset annex session **command from** na **or** admin**.**

Step 3: Review and Reset Global Port Parameters

The RAC ships with a set of default global port parameters already stored in non-volatile RAM. Review the defaults to determine which ones you need to change to satisfy your configuration requirements for PPP or security.

The remainder of this section provides the following information:

- A list of the default settings for the Serial Networking and PPP global port parameter groups.
- Instructions for changing a global port parameter setting. Instructions for using the **set wan b** command to associate IP addresses with RAC WAN B channels.

To view the entire set of default global port parameters, use **na** or **admin** to issue the **show port all** command.

Default PPP-Related Global Port Parameters

Table 8-1 lists the default parameters related to the PPP protocol stored in the RAC nonvolatile memory when shipped. You can view these PPP-specific parameters through the **show port ppp** command issued from the **na** or **admin** utility.

Table 8-1. Default PPP-related Global Parameter Settings

Parameter	Default Setting
local_address	0.0.0.0
metric	1
net_inactivity	off

(continued on next page)

Table 8-1. Default PPP-related Global Parameters Settings (continued)

Parameter	Default Setting
allow_compression	N
address_origin	local
slip_ppp_security	N
net_inactivity_units	minutes
ppp_mru	1500
ppp_security_protocol	none
ppp_password_remote	“<unset>”
ipx_network	00000000
ipx_node	00-00-00-00-00-00
ppp_acm	0x0
ppp_username_remote	““
ppp_sec_auto	N
ppp_ncp	all

The following table lists the wan b parameters related to the PPP protocol stored in the RAC nonvolatile memory when shipped.

Table 8-2. Default wan b Parameter Settings

Parameter	Default Setting
ipx_network	00000000
ipx_node	00-00-00-00-00-00
remote_address	0.0.0.0

How to Change a Global Port Parameter Setting

To change a global port parameter setting using **na**:

1. **At a terminal connected to a UNIX host, enter:**

   % **na**

The following prompt displays on the screen:

```
Annex network administrator Rx.x January 1997
COMMAND:
```

2. **Specify the RAC on which you intend to change global port parameter settings at the** COMMAND: **prompt. Specify the administrative password for host at the** password: **prompt.**

   You can specify the RAC by its IP address or name. If you intend to change global port parameter settings on more than one RAC, separate their IP addresses or names using a comma (,). If prompted for a password, the password is the administrative password for the RAC.

   For example:

   ```
   COMMAND:annex 132.245.6.40 or
           annex 132.245.6.40,132.245.6.45
           password:
   ```

3. **Specify a new setting for the global port parameter at the** COMMAND: **prompt.**

   For example, to change the default setting of the **address_origin** parameter (**local**) to enable IP addressing through the **acp_dialup** file, enter the following:

   ```
   COMMAND:set port address_origin acp
   ```

   

   The new parameter setting is stored automatically in non-volatile RAM.

4. **To review your changes, issue the** show port all **command at the** COMMAND: **prompt.**

   This command displays all of the global port parameter settings. To locate the parameters you changed, press the **Return** key (only if pager is set), which allows you to scroll down through the file.

   ```
   COMMAND: show port all
   ```

5. **Enter** quit **at the** COMMAND: **prompt to exit** na**.**

   ```
   COMMAND: quit
   ```

How to Assign IP Addresses to PRI B Channels

Use the **set wan b** command issued with the **remote_address** global port parameter to associate IP addresses with channels. This lets you configure fixed IP addressing for RAC sessions. You can use **na** or **admin** to use this command.

Command Syntax

Use the following command syntax when creating B channel IP address assignments:

`set wan b=<`*`ch-range`*`> remote_address <`*`ip-addr`*`> [<`*`increment`*`>]`

where

<*ch-range*> is a single B channel number or the entire set of B channels specified by a range.

<*ip-addr*> is the IP address you want to assign to a single B channel or the first channel of the entire set.

<*increment*> is the value (number) by which you want to increment automatic IP address assignment to B channels in a range or list after the first assignment. The value can be either an integer or a dotted quad.

Usage Rules

You can associate IP addresses with channels in the following ways:

- For a single B channel (e.g., 4).
- For the entire set of channels using one of the following choices:
  - Specifying channels as a range of two numbers separated by a dash (1-23 for T1; 1-30 for E-1).
  - Using the keyword **all.**

When assigning IP addresses to the entire set, you can specify an increment by which IP addresses are assigned in sequence based on the increment value. For example, if you specify the entire set of B channels available with a T1-based RAC PRI module (23), an IP address of 132.245.66.230, and an increment of 2, the following assignment results:

```
set wan b=1-23 remote_address 132.245.66.230 2
```

B channel #1 is assigned the IP address of 132.245.66.230, B channel #2 is assigned 132.245.66.232, B channel #3 is assigned 132.245.66.234,....., and B channel #23 is assigned 132.245.10.7.

When you do not specify any B channels, the command assigns 23 or 30 IP address/B channel assignments based on the increment value.

When an increment is not specified, the command assumes a default increment of zero (0).

Configuration Samples

The following samples illustrate how to set global port parameters to enable PPP configurations.

Sample Configuration for Addressing Using the acp_dialup File

Figure 8-1 shows a configuration in which a remote PC (i.e., user *green*) is connected to a RAC through a PPP link. The PC appears to the network as directly attached device. This configuration uses addressing enabled through the **acp_dialup** file. The PC is connected through a BRI line with a V.120 terminal adapter and the PRI line to the RAC.

Based on the entries in the **acp_dialup** file, user *green* has access from all RACs and Remote Annexes since the **acp_dialup** file entry is a wildcard (*). User *green*'s **acp_dialup** file remote address is 132.245.5.18.

Following Figure 8-1 are the steps that implement this configuration.

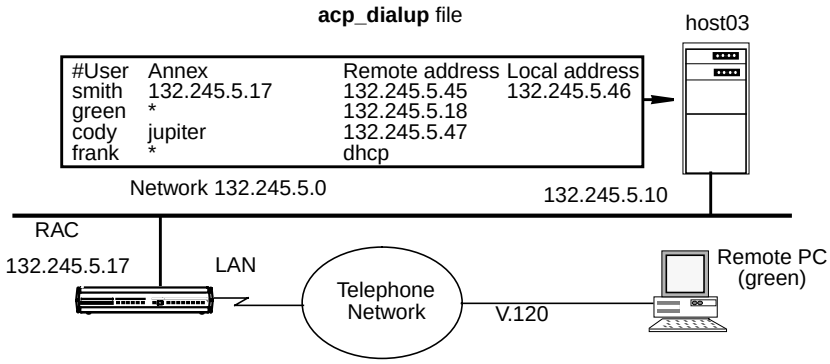

Figure 8-1. Connecting a Single Host Using PPP

To enable this configuration:

1. **Edit the** acp_dialup **file. Provide user *green* with access from all RACs and Remote Annexes by specifying a wildcard** (*) **and a remote address of 132.245.5.18.**
2. **Use the** set wan b **command with the** remote_address **global port parameter to associate a set of IP addresses with the PRI B channels.**

   Step 2 is optional since the RAC ignores the IP address/B channel assignments created using the **set** command when it uses dial-up addressing. However, if the host where the ACP server resides is unreachable by the RAC, or there is no entry in the **acp_dialup** file for a particular user, the RAC relies on the IP addresses assigned to the B channel to provide a remote address for the link.

3. **Edit the configuration file to define an SPB.**

   You can use the default SPBs provided as part of the configuration file or create them specifically for your requirements. For more details, see *Step 2: Edit the Configuration File on page 8-5*.

4. **Reset the default global port parameters as required to the following settings:**

   - Enable CLI and/or connection security using the security parameters: **cli_security** and **connect_security**.
   - The **slip_ppp_security** parameter controls dial-in PPP access and use of ACP for PPP and protocol security such as **pap**, **chap** and **chap-pap**. If **enable_security** and **slip_ppp_security** are enabled, access to the PPP command is restricted via ACP and call access is logged in the ACP log file.
   - Set the **ppp_security_protocol** parameter to **pap**, **chap**, **chap-pap**, or, for CLI users (in some instances), **none**.
   - Set the **ppp_username_remote** and **ppp_password_remote** parameters to the values expected by the remote node (the PC in Figure 8-1).
   - Set the **allow_compression** parameter to **Y** if you want the RAC to negotiate for VJ TCP header compression.
   - Set the **address_origin** parameter to **acp** so that the RAC requests the endpoint addresses, based on the user's login, from ACP.
   - You can leave **ppp_mru** parameter set to its default.

Sample Configuration for Addressing Using Asynchronous Port Parameters

Figure 8-2 illustrates a configuration in which a single remote PC is connected to a RAC through an asynchronous PPP session. The session is running via a BRI line with a V.120 terminal adapter and the PRI line to the RAC.

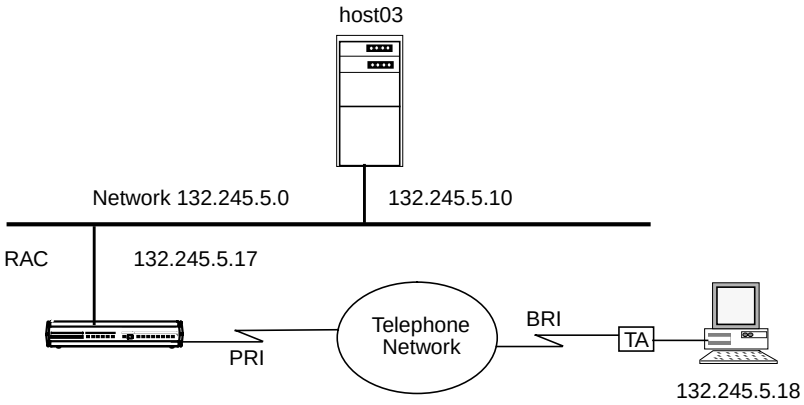

Figure 8-2. Connecting a Single Host Using PPP with Fixed Addresses

To enable this configuration:

1. **Use the** set wan b **command with the** remote_address **global port parameter to associate a set of IP addresses with the channels.**

   See *How to Assign IP Addresses to PRI B Channels on page 8-9* for instructions to perform this step.

2. **Edit the configuration file to define an SPB.**

   You can use the default SPBs provided as part of the configuration file or create them specifically for your requirements (for more details, see *Step 2: Edit the Configuration File on page 8-5*).

3. **Reset default global port parameters as required to the following settings:**

   - Enable CLI and/or connection security using the security parameters **cli_security** and **connect_security**.
   - The **slip_ppp_security** parameter controls dial-in PPP access and use of ACP for PPP and protocol security such as **pap**, **chap**, and **chap-pap**. If **enable_security** and **slip_ppp_security** are enabled, access to the **ppp** command is restricted via ACP and RAC access is logged in the ACP log file.

- Set the **ppp_security_protocol** parameter to **pap**, **chap-pap**, or, for CLI users, **none**.
- Set the **allow_compression** parameter to **Y** if you want the RAC.
- Set the **local_address** parameter to the RAC **annex inet_addr** address.
- Set the **address_origin** parameter to **local**.
- Leave the **ppp_acm** and **ppp_mru** parameters set to their defaults.

Sample Configuration for Connecting Two Subnets

Figure 8-3 illustrates two Ethernet subnets interconnected via synchronous PPP over ISDN using an ISDN router (e.g., Nautica Series Marlin) and the Remote Access Concentrator.

The RAC has security enabled for this PPP session. Following the figure are the global port parameter settings required for this configuration.

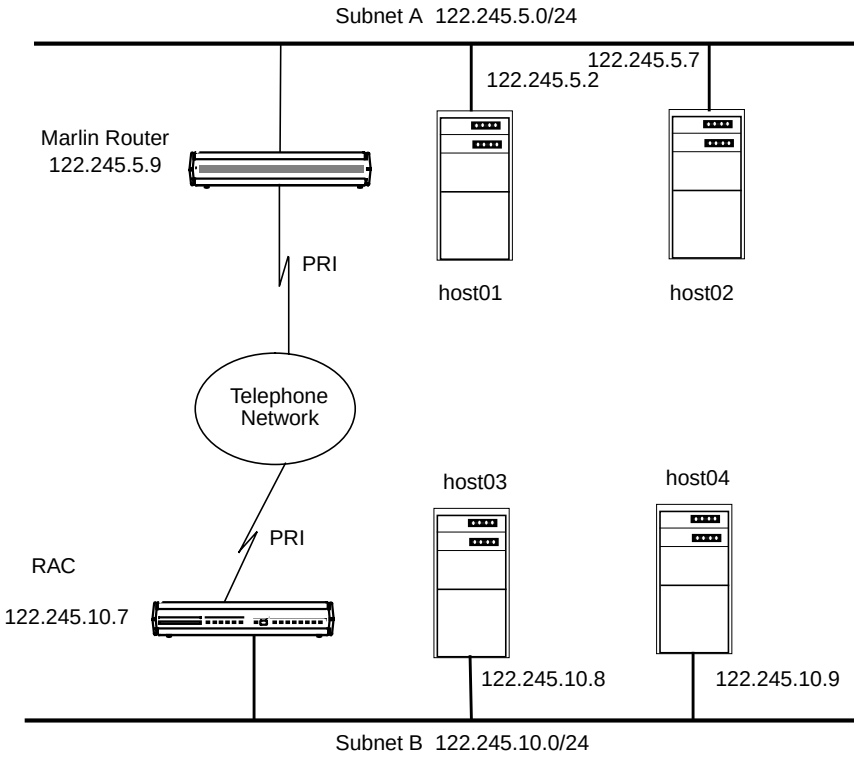

Figure 8-3. PPP Link Connecting Two Ethernet Subnets

To enable this configuration:

1. **Use the** set wan b **command with the** remote_address **global port parameter to associate a set of IP addresses with the channels.**

   Specify an IP address of zero (0).

   When you specify an IP address of zero (0), the peer (the Marlin router in this sample configuration) must provide its IP address.

2. **Edit the configuration file to define an SPB.**

   You can use the default SPBs provided as part of the configuration file or create them specifically for your requirements. For more details, see *Step 2: Edit the Configuration File on page 8-5*.

3. **Reset default global port parameters to the following settings as required:**

   - Use the defaults for the **data_bits** (8), **stop_bits** (1), and **parity** (none) parameters.

     PPP is an 8-bit protocol. If **data_bits** is set to **7**, and **parity** is not set to **none**, the RAC forces the **data_bits** setting to **8** and the **parity** setting to **none**. Otherwise, the RAC *syslogs* an error message for the port.

   - Set the **address_origin** parameter to **local**.
   - Set the **local_address** parameter to 122.245.10.7.
   - Set the **metric** parameter to one.
   - Set the **ppp_username_remote** parameter to the string "".
   - Set the **ppp_password_remote** parameter to the string "".
   - Set the **slip_ppp_security** parameter to **Y.**
   - Set the **ppp_security_protocol** parameter to **pap** (password authentication protocol).

     Setting **ppp_security_protocol** parameter to **pap** assumes that the Marlin router is configured to use PAP.

   - Set the **allow_compression** parameter to **Y** if you want the RAC.

4. **Edit the** acp_passwd **file to add the Marlin router's username and password to the file.**

   For more details, see *Creating User Password Files on page 6-72*.

5. **Enter the routing information into the** gateway **section of the configuration file. For example:**

```
%gateway
# PPP link to the 122.245.5.0 net
annex 122.245.10.7

     #122.245.5.9 is a gateway to the entire
     #122.245.5.0 net with a metric of 1
     route add 122.245.5.0 255.255.255.0 122.245.5.9 1

else

     #other Annexes will route to 122.245.5.0 via
     #122.245.10.7 with a metric of 2
     route add 122.245.5.0 255.255.255.0\
     122.245.10.7 2

end
```

Routing across a PPP Link (Basic Passive RIP)

Both active and passive routing are available via the Routing Information Protocol (RIP) on the RAC. The following sections deal with using only the most basic features of passive RIP and are intended for administrators who need minimal routing features.

Both active and passive RIP are enabled by default. To turn off active RIP, set the interface parameter **rip_advertise** to **none** for all RAC interfaces. See *rip_advertise on page 11-55*.

The RAC bases its routing table on the information you specify in the **gateway** section of the configuration file. As a passive gateway, the RAC updates the table according to RIP information it receives from other routers, but does not broadcast routing information itself, as an active gateway would. This means that the RAC with the PPP interface forwards packets addressed to the host at the remote end of the connection, but does not inform other hosts or RACs that it has this capability. Other hosts, routers, and RACs on the same network must be told about the route before they can use it.

Route Cache

The route cache is a list of routing entries stored by the RAC. When the RAC boots, the route cache is created from the **annex...end** and **subnet...end** blocks in the **gateway** section of the configuration file. When **routed** starts, entries in the route cache are added to the routing table if their next hop addresses (i.e., destinations) are on a network or link directly connected to the RAC. The RAC examines once and discards routes outside the **annex...end** blocks if the destination is not a directly connected network or is an inactive link such as a PPP link at boot time. Do not enter cache routes whose address is a proxy-ARP interface, because the next hub address would be ambiguous and appear to be reachable at boot time over the ethernets.

Protocol Stack

Bringing up a PPP link includes three stages: link control protocol (LCP) negotiation, security, and NCP negotiation. The LCP establishes and negotiates the data link with the peer system. Next, an optional security phase authenticates the peers to each other. Finally, NCP establishes and negotiates the network details and informs the RAC that the interface (i.e., PPP link) is available.

Negotiating the LCP Options

The following subsections describe how the RAC negotiates the LCP options.

Maximum Receive Unit (MRU)

The **ppp_mru** parameter sets the maximum receive unit (MRU). Acceptable values range from **64** to **1600**. The RAC default is **1500**. The RAC informs the peer that the **ppp_mru** parameter is its local Maximum Receive Unit (the largest packet it can receive). The parameters are not sent in the configure request if the value is set to 1500.

Asynchronous Control Character Mask (ACCM)

The port parameter **ppp_acm** (asynchronous control mask) specifies which of theASCII control characters (0x0 to 0x1F) can be sent as clear text, and which should be protocol-escaped before being sent to the serial port.

The RAC requests the **ppp_acm** parameter as its local mask. If the peer NAKs **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** to **0xffffffff**. The RAC default is **0x00000000**.

Setting the **ppp_acm** mask avoids sending characters that may bother the modems or devices through which the peers are connected.

For example, if the modem uses CTRL-A (^A/0x01) as its attention character, it must be escaped before the RAC sends it.

- **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, or 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 0-31 decimal, the bit indexed by this parameter is set in the ACCM.

If **input_stop_char** is 0-31 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 0-31 decimal, the bit indexed by this parameter is set in the ACCM.

  If **output_stop_char** is 0-31 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 (i.e., 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**

Since the output flow control parameters are outside the range 0-31 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, i.e., the value negotiated between the two PPP processes.

Magic Numbers

The Magic Number option detects data-link anomalies, namely loopback. The RAC always requests this option by sending a random 4-byte word out as its Magic Number in an LCP *Configure Request*.

Link Quality Monitoring (LQM)

The RAC will not request LQM. It rejects any attempts by the remote peer for LQM and hints for the PPP default of none.

Protocol Field Compression (PFC)

PFC compresses the two-byte protocol field to one byte for some PPP protocols.

The RAC always requests and accepts PFC from the peer. If NAKed, it accepts the PPP default of off. If the peer does not request PFC, the RAC hints for PFC on. If the peer NAKs this hint, the RAC accepts PFC off.

Address and Control Field Compression (ACFC)

ACFC deletes the constant address and control fields in the HDLC headers. The RAC always requests, and accepts, ACFC. If NAKed, it accepts the PPP default of off. If the peer requests ACFC off, the RAC hints for ACFC on. If the peer NAKs this hint, the RAC accepts ACFC off.

Negotiating the Network Control Protocol

The RAC supports the following NCPs: AppleTalk Control Protocol (ATCP), Internet Packet Exchange Protocol Control Protocol (IPXCP), Internet Protocol Control Protocol (IPCP), CCP (Compression Control Protocol), and MP (Multilink PPP). NCP options are negotiated in the same way as LCP options. An NCP peer opens the link and the interface is available to the RAC.

To specify one or more NCPs, set the **ppp_ncp** port parameter to any combination of **ipxcp**, **ipcp**, **atcp**, **mp**, and **ccp**. Separate multiple values with a commas. You can also specify **all** to indicate all of the protocols, which is the default.

Negotiating Data Compression

If you specify **ccp** as an NCP, the RAC automatically requests data compression for a PPP link. Three types of compression are negotiated:

- Predictor-1, a public-domain algorithm
- BSD-Compress, a freely available portion of the BSD UNIX sources
- STAC (with Check Modes 1, 3, and 4), a licensed, standard compression scheme. Check Mode 4 is one of the compression types used by the Windows 95 Dial-up Networking feature.

These three compression types have higher compression ratios than that provided by V.42 bis in standard modems.

Authentication Type

The authentication type specifies the style of authentication. The RAC supports two authentication protocols for PPP:

- Password Authentication Protocol (PAP).
- Challenge-Handshake Authentication Protocol (CHAP).

Both of these protocols are run over the PPP link after the LCP negotiations are complete.

The RAC can require the peer to pass a security check before starting NCP, and can also authenticate itself to the peer. The RAC negotiates for the security specified by the **ppp_security_protocol** parameter. Valid arguments for this parameter are:

- **pap** (password authentication protocol [PAP]).
- **chap** (challenge-handshake authentication protocol [CHAP]).
- **chap-pap** (first negotiate for CHAP; if peer NAKs, negotiate for PAP).
- **none** (do not negotiate; the default).

The RAC responds to an authentication request from a peer only if **ppp_password_remote** and **ppp_username_remote** are set for this port.

If the peer refuses a negotiation request from the RAC, the RAC closes the link.

For a complete description of the RAC's implementation of these protocols, see *Using PPP Security on page 6-135*.

Negotiating the IP Address

The RAC and the peer negotiate the IP address to be used on both sides of the link. Any address sent as zero requests that the peer set the address. Four parameters control the RAC IP address negotiation: **address_origin**, **local_address**, **remote_address**, and **enable_security**.

If **address_origin** is set to **acp**, the RAC makes an ACP **dialup_address()** call for the addresses to be used from the **acp_dialup file**.

If **address_origin** is set to **dhcp**, or if it is set to **acp** and the remote address field of the **acp_dialup** file is set to **dhcp**, the RAC receives a dynamically assigned IP address via DHCP.

If **address_origin** is set to **local** (its default value), or DHCP and ACP are not available, the RAC defaults to using the **local_address** and **remote_address** as the addresses. The RAC allows the other side of the link to select addresses only if these addresses are zero and the **address_origin** is set to local.

The RAC uses two methods to negotiate the IP addresses. The preferred technique is to use the IPCP type 3 *IP-Address* option.

If the peer rejects this style of address negotiation, the RAC falls back to using the deprecated IPCP type 1 *IP-Address* option.

In either case, the RAC requires the peer to use both the local and remote address of the RAC. To allow the peer to select addresses, the RAC addresses must be set to zero and the **address_origin** is set to local.

If each end has a zero address and the peer cannot provide both, or the RAC has a non-negotiable address, the RAC and the peer will never agree upon an address, and the link will fail to come up.

Negotiating the Header Compression Type

The RAC and the peer can negotiate a specific protocol compression for TCP/IP Headers. The options are **VJ**, **TCP/IP**, and **none**. If the **allow_compression** parameter is set to **Y**, the RAC always negotiates for compression on its side of the link and allows the peer to determine whether to compress data. If **allow_compression** is set to **N**, the RAC never requests, and always rejects, TCP/IP header compression; the default is **N**.

BOOTP Requests

BOOTP is a bootstrap protocol that allows a diskless client to determine its Internet address, the Internet address of the server, and the name of the file to be loaded into memory.

- The RAC ROMs use BOOTP to obtain boot information without requiring any manual set up on the RAC.
- If a diskless client sends a BOOTP request to the RAC over a PPP line, the RAC responds with its current local address, remote address, and boot host, or forwards the request to a named server, if a server name is present in the BOOTP request.

The *Remote Access Concentrator Hardware Installation Guide* for your RAC discusses BOOTP in greater detail.

Using the CLI netstat -ip Command

The **netstat -ip** *device_id* command displays configuration and statistical data for serial interfaces. The *device_id* argument specifies a serial port.

Displaying Data for Ports

Ports are specified by port number alone, or the string *asy, ta,* or *syn* followed by the port number (with no intervening white space). Each of the following sample commands specify PPP port 1:

```
netstat -ipl or netstat -ipasy1
```

```
netstat -ipsyn1
```

PPP Over Sync Port

The model 8000 RAC includes a general sync port, which has a physical connector associated with it. This general sync port constitutes its own port class (GSY).

The GSY is a dedicated PPP connection, and is not allocated dynamically like other port classes on the RAC. Like other sync ports (class SYN), the GSY is configured as a route in the **%dialout** section of the configuration file. The GSY port does not require any dialout activity to instantiate the route, however; the route is brought up as soon as it receives valid PPP frames from the associated port. The route remains active until the net_inactivity timer expires. An example of a dialout route for the GSY port follows:

```
%dialout
annex 132.245.66.74
begin_route 1
ports gsy1
remote 99.99.99.1
set local_address 99.99.99.101
set net_inactivity 10
set net_inactivity_units minutes
aet rip_advertise none
set rip_sub_advertise n
set rip_sub_accept none
set rip_accept none
advertise n
mode ppp
end_route
end
```

The status of the GSY port is displayed along with that of the other port classes when you issue the **stats** command.

Chapter 9
Multilink Point-to-Point Protocol (MP)

This chapter covers the following topics:

- Overview of MP
- Supported and unsupported MP features
- MP Function and Process Information
- MP Configuration
- The **acp_userinfo** File
- Administration of Multilink PPP
- MP Parameters
- PPP Parameters
- Multi-system Multilink PPP

Overview

Multilink PPP (MP) is a protocol that adjusts the bandwidth of a connection between two network devices to accommodate dynamically changing network loads. This implementation of MP is based on the RFC1990 technical specification.

Member Links and Bundle Links

To support this protocol, the RAC PPP protocol stack has been extended to allow one or more PPP links to form a single virtual PPP interface for the network layer protocols, such as IP, IPX and AppleTalk over PPP. In this document, the virtual PPP interface is referred to as the bundle and the PPP links are referred to as member links.

When a member link is established, the endpoints of the connection negotiate to assign the new link to a bundle. Refer to Figure 9-1 for a representation of the functional flow.

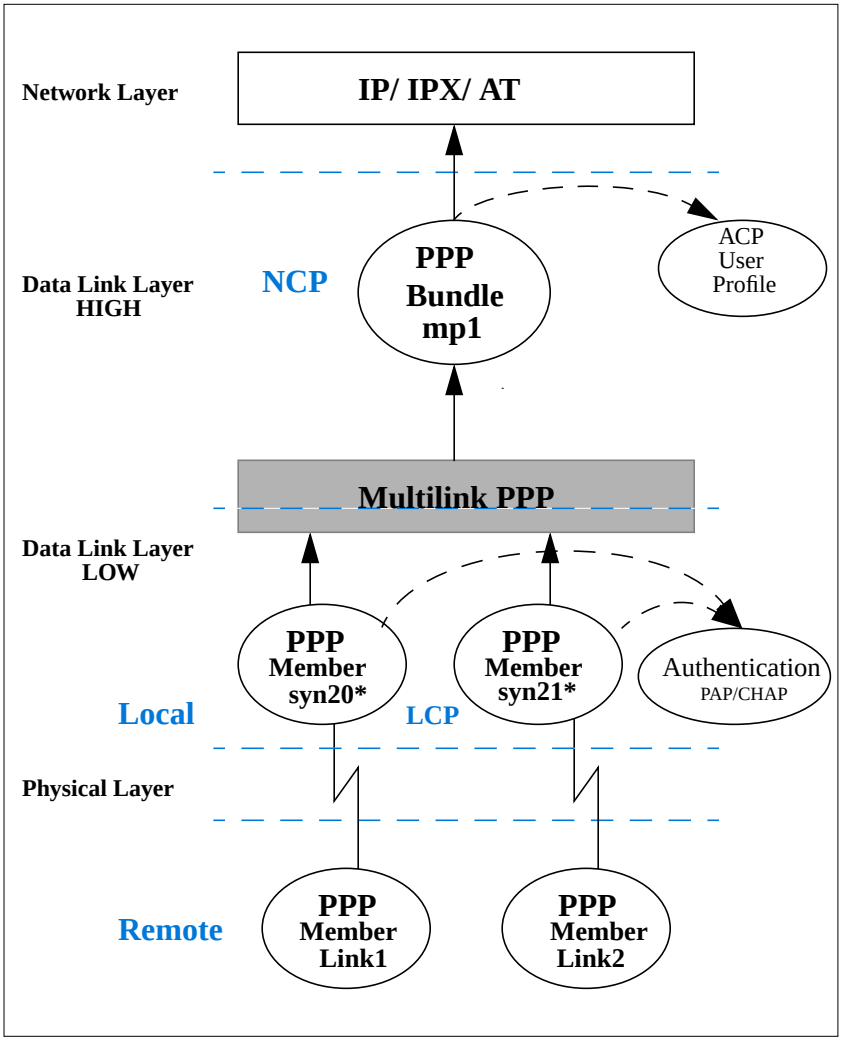

Figure 9-1. MP Functional Flow

The RAC will negotiate MP but cannot initiate additional member links based on load (bandwidth on demand). It is up to the remote side to initiate additional member links.

The number of member links can be adjusted to increase bandwidth in response to network traffic over the bundle. Active bandwidth control protocols, such as BACP/BAP, are not supported in this release. However, the RAC, using the security services of RACP, can restrict the maximum number of member links permitted to join a bundle, if this control is required by the network administrator. For more information, see *mp_max_links* on page 11.

Supported MP Features

The RAC supports the following MP features:

- Short and Long Sequence Header
- Fragmentation
- Local Endpoint Discriminators:
  - NULL
  - IP
  - MAC (alias is DEFAULT)
  - PSNDN
- All Remote Endpoint Discriminators
- Maximum Reconstructed Receive Unit

Unsupported MP Features

The following MP features are not supported for this software release:

- Locally Assigned Address
- Magic Number Blocks

The MP Process

The following table outlines the processing of a typical MP connection:

When...	Then...	For More Information See...
a remote user opens PPP link1,	link control protocol (LCP) establishes and negotiates the data link options for PPP link1.	*Point-to-Point Protocol on page 8-1*.
the LCP negotiation is successful,	PPP link1 requests (optional) authentication by CHAP or PAP security protocols.	*Point-to-Point Protocol on page 8-1 and Configuring Security on page 6-1*.
the PPP link1 is authenticated (optional) by CHAP or PAP security protocols,	the member link attempts to join a bundle.	*Configuring Security on page 6-1*.

(continued on next page)

When...	Then...	For More Information See...
the member link's MP LCP options and authentication information match an existing bundle,	the bundle checks that the maximum number of member links will not be exceeded and adds the member link to the bundle.	RFC1990/ **mp_max_links**
the member link's MP LCP options and authentication information do not match an existing bundle,	the bundle checks that the maximum number of member links has not been reached and establishes a new bundle.	RFC1990/ **mp_max_links**
a bundle determines that a network layer protocol needs to be transported,	the network control protocol (IPCP, IPXCP, ATCP) establishes and negotiates the network protocol options and attaches the bundle to a proper network route.	*Remote Access Concentrator Software Reference* .

Bundling Scenarios

The following four scenarios are used as guidelines for bundling member links.

If there is...	Then...
no authentication and no discriminator,	all new links must be joined to one bundle.
a discriminator but no authentication,	if the discriminator matches, the new member link must join a matching bundle. if the discriminator does not match, a new bundle must be established.
no discriminator but authentication is successful,	the authenticated match must join a matching bundle. the authenticated mismatch must establish a new bundle.
a discriminator and authentication is successful,	if the discriminator and authentication match, the new member link must join a matching bundle. if the discriminator and/or authentication is mismatched, a new bundle must be established.

Closing Member Links

Terminate-RequestTerminate-Ack

Member links can be terminated according to normal PPP LCP procedures using LCP Terminate-Request and Terminate-Ack packets on the member link. Receipt of a Terminate-Ack is a sufficient indicator that any MP packets ahead of it are at no special risk of loss.

Security Considerations

It is important to understand that LCP and authentication negotiations do not occur on the bundle itself. These phases occur on the member links. Refer to Figure 9-1.

MP Configuration

All PPP port parameters still apply to MP bundled links. See *Point-to-Point Protocol on page 8-1* and *Configuring Security on page 6-1* for more information.

MP is enabled by default.

Option	Description	MP Parameter
ppp_ncp	This option activates MP and is required to make MP operational. In order for NCP to recognize MP as a valid protocol, you must set **ppp_ncp** to **all** or to include **mp**. If you do not set this option, NCP will ignore all MP bundling implementation.	**mp**
Short Sequence Number Header Format (**SSNHF**), LCP option 18	This option advises the peer that the implementation expects to receive fragments with short, 12-bit sequence numbers. By default, sequence numbers are 24-bits long.	The RAC always requests this option.

(continued on next page)

Option	Description	MP Parameter
Multilink Maximum Reconstructed Receive Unit (**MRRU**), LCP option 17	This option initiates MP and is required to make MP operational. The RAC supports an MRRU of 1600 octets or less for the local value (accepts any reasonable value for the remote MRRU). This option also advises the peer that the implementation can reconstruct a PPP packet whose information contains the number of bytes as Max-Receive-Reconstructed-Units (**MRRU**).	**mp_mrru**
Endpoint Discriminator, LCP option 19	This option identifies the system transmitting the packet. It advises a system that the peer on this link could be the same as the peer on another existing link. If this option distinguishes a peer from all others, a new bundle is created. The RAC accepts all remote class requests, but only supports NULL, MAC, IP, and PSNDN locally.	**mp_endpoint_class, mp_endpoint_address**

MP Operational Characteristics

MP Load Balancing

The bundle distributes fragmented packets over all active member links.

MP Fragmentation Bypass

Packets smaller than a fixed size are not fragmented in order to reduce processing load caused by reassembly at the remote endpoint.

MP Parameters

ppp_ncp

Allows you to set the mode for NCP negotiations for MP. You must set this parameter to **all** or include **mp** in order to negotiate MP over NCP.

- Default = all

mp_mrru

Sets the upper limit of the MRRU LCP negotiation. All LCP negotiations start with this value. Downward negotiation is allowed. You must set this parameter to a non-zero value in order to negotiate MP.

- Default = 1500
- Minimum value = 64
- Maximum value = 1500

mp_endpoint_class

Sets the value of the local Endpoint Discriminator Class to one of the following allowed classes. (The remote endpoint is allowed to use all classes specified in RFC 1990.)

- Default = MAC (class 3)
- Accepted values:

  NULL, IP, MAC, PSNDN, DEFAULT (alias for MAC)

Selecting...	Causes the RAC...
NULL	to use the NULL Endpoint Discriminator (no address).
IP	to use the Ethernet interface IP address of the RAC for the Endpoint Discriminator Address.
MAC	to use the MAC address of the ethernet interface of the RAC for the Endpoint Discriminator Address.
PSNDN	to use the value stored in the **mp_endpoint_address** parameter for the Endpoint Discriminator Address.
DEFAULT	to use the default Endpoint Discriminator Class (MAC).

mp_endpoint_address

Sets the value of the Endpoint Discriminator Address for Endpoint Discriminator Classes that allow user-configured information. This parameter is accessed only when set to PSNDN (Public Switched Network Directory Number).

- Default = not set or empty string
- Minimum length = 0 characters
- Maximum length = 16 characters

mp_endpoint Class and Address Example

PSNDN - Public Switched Network Directory Number

A telephone number, up to 16 numeric characters long.

Example: 16175558140

The acp_userinfo File

The identity of the bundle is not fully qualified until the LCP Endpoint-Discriminator and the (optional) authentication processing is performed. After this process is complete, the bundle policy can be determined and enforced.

mp_max_links

Controls the total number of simultaneously active PPP links that can be joined in the identified bundle. If this number is exceeded, allowing the link to become fully established causes the RAC to abort the PPP link attempting to join the bundle.

- Default value = 1
- Minimum value = 1
- Maximum value = 255 (RAC supports a maximum of 60 (2) E1 channels , a maximum of 48 (2) T1 channels, and a maximum of 46 (2) PRI channels)

mp_max_links Example

```
user username=myname;group=developers
     climask telnet end
     mp_max_links 3
end
```

Using Multi-System Multilink PPP

Multi-system Multilink PPP (MMP), a superset of Multilink PPP, allows MP links belonging to the same MP bundle to terminate on multiple RACs. The RACs are combined together in an MMP group to use all of the incoming channels in the group, increasing the potential bandwidth of an MP bundle. Bay Networks RACs support MMP for incoming calls only.

The locations of the MP links are completely transparent to remote users; users need (and receive) no information about which RAC terminates a given MP link. The location of the MP bundle head is determined by the bundle discovery protocol. Layer 2 Tunneling Protocol (L2TP) is used to tunnel MP links to remote MP bundle heads, ensuring that successive links on one RAC in an MMP group are combined into the same bundle as the primary link on another RAC.

MMP is disabled by default, through the annex parameter **mmp_enabled** set to **n**. To enable MMP, you must set **mmp_enabled** to **y** and configure the global port or inbound channels for MP, if you have not done so already. MMP relies on three other features, which are enabled when MMP is enabled:

- MP bundle discovery
- Virtual MP links
- Layer 2 Tunneling Protocol (L2TP)

A graphical description of MMP for a hunt group is shown in Figure 9-2.

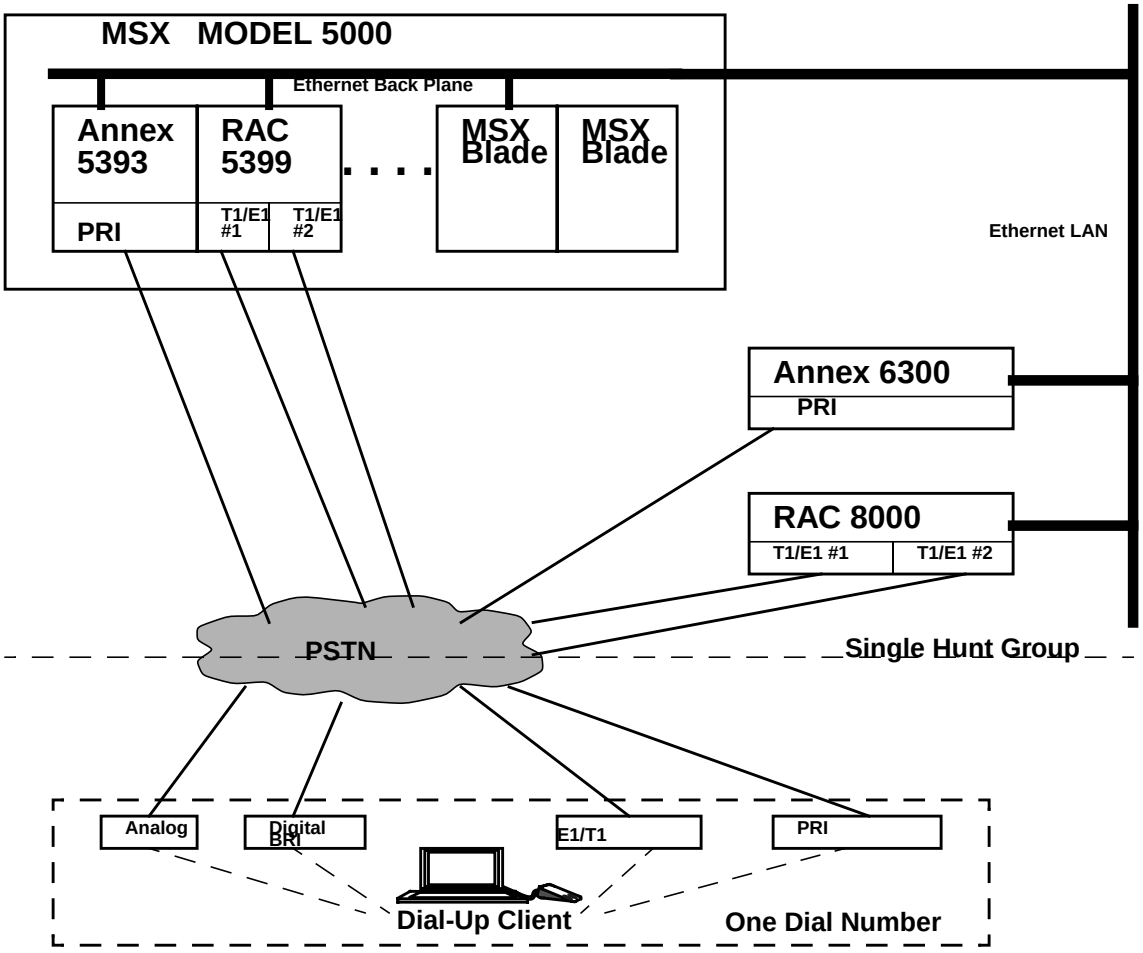

Figure 9-2. Illustration of Single Hunt Group Configuration

MMP Groups

An MMP group is a set of one or more RACs that act as a single entity for any MP links that terminate on any of the RACs in the group. MMP groups usually are organized to correspond to telco hunt groups. All RACs in an MMP group must reside on the same Ethernet segment, and, where applicable, on the same IP subnet. A RAC can belong to a single MMP group only. Otherwise, RACs can be grouped any way, with multiple groups allowed on the same Ethernet segment.

MMP groups are identified by their endpoint discriminator, which comprises the port parameters **mp_endpoint_class** and **mp_endpoint_address**. All the RACs in an MMP group must have the same **mp_endpoint_class** and the same **mp_endpoint_address**. When MMP is enabled and **mp_endpoint_class** is set to **loc** or **psndn**, the **mp_endpoint_address** parameter indicates an endpoint discriminator address, which is the number of the hunt group or the name of the rotary served by the MMP group. *Refer to the Remote Access Concentrator Software Reference for complete descriptions of these parameters.*

Establishing MMP Connections

When a remote user dials the number for a hunt group, a RAC belonging to the MMP group assigned to that hunt group answers the call. The remote user can use any type of line: BRI, PRI, CAS, or asynchronous; the answering RAC detects the type of call automatically and answers it appropriately, creating an MP link.

While the MP link is in the authentication phase, the RAC initiates the Bundle Discovery Protocol to determine if an MP bundle head for the user exists already. The MP bundle head is the RAC that contains the first (primary) link of an MP bundle. The Bundle Discovery Protocol searches first the current RAC for an MP bundle, then the other RACs in the MMP group. If the Bundle Discovery Protocol does not find an existing MP bundle, the current MP link becomes the primary MP link and creates an MP bundle, and the current RAC becomes the MP bundle head. If the Bundle Discovery Protocol locates an existing MP bundle for the remote user, the current MP link becomes a secondary MP link. If the MP bundle head is a different RAC, the current link is tunnelled to that MP bundle head via L2TP and becomes a virtual link; virtual links are always secondary links. Secondary MP links are combined with the primary MP link to form an MP bundle; all MP fragments are reassembled by the MP bundle head.

MP bundles handle links terminating on the MP bundle head and virtual links equally, without preference. A primary link always resides on the MP bundle head and is never a virtual link.

The maximum possible number of MP links permitted on any RAC is the maximum number of actual channels plus an equal number of virtual links.

Configuring MMP

To configure a RAC for MMP:

1. **Enter** admin **mode.**
2. **If you have not done so already, enable MP by setting the port parameter** ppp_ncp **to include** mp**, as well as** ipcp**,** ipxcp**, or** atcp**, whichever is in use.**
   [`set port ppp_ncp mp,ipcp|ipxcp|atcp`**]**
3. **Set the annex parameter** mmp_enabled **to** y**.**
   [`set annex mmp_enabled y`**]**

4. **Set the port parameter** mp_endpoint_class **to** loc **or** psndn**.**
   [`set port mp_endpoint_class loc | psndn`**]**
   (When MMP is enabled on the RAC, the default value of this parameter is null**; the default value for regular MP is** mac**.)**
5. **Set the port parameter** mp_endpoint_address **to the string that indicates the correct address. (This parameter is unset by default.)**
   [`set port mp_endpoint_address <string>`**]**
   When mp_endpoint_class **is** loc**, the string can be up to 16 alphanumeric characters; when** mp_endpoint_class **is** psndn**, the string can be a phone number of up to 15 characters.**

   

   Refer to the *Remote Access Concentrator Software Reference* for a complete description of the **mp_endpoint_address** parameter.
6. **Reboot the RAC.**

Resetting Ports for MP Links

RAC ports through which MP links are connected can be reset using the **admin** command **reset vpn=**<range>, where <range> is the range of ports to be reset.

Administration of Multilink PPP (MP Statistics)

Several extensions are added to allow management of MP from the network and from the administrative utilities (host-based **na** and RAC-based **admin**). Additional status information is also available through the CLI **netstat** command and SNMP proprietary MIB objects.

netstat -i

This command lists bundles as devices named "**mp<n>**" where <**n**> is a number assigned when the bundle is created. Member links appear but do not have associated addresses, since they are represented by the bundle as a single interface to the network layer. A bundle has an address after it completes the NCP negotiations selected for that connection.

When B-channel assigned addresses are used by MP, only the address of the first channel called is used. All future calls use the same IP, IPX, and IPX-net as the first call.

netstat -i Example

The following example displays a bundle with two links:

```
annex# net -i
```

Name	Mtu	Network	Address	Ipkts	Ierrs	Opkts	Oerrs	Collis
en0	1500	132.245.66	132.245.66	85422	0	30832	1	0
en0	1500	32004-32005	32005.243	85422	0	30832	1	0
WAN1*	1500	none	none	0	0	0	0	0
WAN2*	1500	none	none	0	0	0	0	0
lo0	1536	127	127	0	0	0	0	0
syn20*	1500	none	none	73	0	105	0	0
mp1	1524	132.245.252	132.245.252	129	0	123	0	0
syn21*	1500	none	none	65	0	100	0	0
en0	576	80230066	00802d02cea1					

You may notice that there is an * after the two member links, **syn20*** and **syn21***. You can use this as a visual cue of the **netstat -i** command. It is an indicator that the interface is not completely configured or that it might be a member link. Use the **netstat -ip** command on a member link, **syn20*** or **syn21***, to identify the bundle to which it belongs.

netstat -ip

This command has been extended to provide MP information, including negotiated MP LCP options. If the command is issued for a member link, the MP status block identifies the bundle device name to which the link has been attached.

netstat -ip
Bundle Example

```
annex# net -ip mp1
                        *** PPP Layering Status ***
State               Current:  NCP Open       Prior:    NCP Open
                    *** LCP Status ***
State               Current:  Open           Prior:    Ack sent
Options             Local:                   Remote:
MRU                 1500                     1500
MRRU                1500                     1500
Short Sequence      Off                      Off
Endpoint Disc       3:00-80-2d-05-58-29      1:0x14f3
                    *** NCP (IPCP) Status ***
State               Current:  Open           Prior:    Ack sent
Options             Local:                   Remote:
IP addresses        132.245.44.98  [REM]     132.245.88.115 [REM]
Compression         None                     None
                    *** NCP (ATCP) Status ***
State               Current:  Closed         Prior:    Closed
                    *** NCP (IPXCP) Status ***
State               Current:  Open           Prior:    Ack sent
Options             Local:                   Remote:
Network No          00000001                 00000001
Node No             00802d055829             00802d01804f
Compression         None                     None
Routing Prot        RIP/SAP                  RIP/SAP
Router Name         LM055829                 LM01804F
```

(continued on next page)

(continued)

```
                        *** NCP (CCP) Status ***
State              Current:  Open          Prior:    Ack sent
Options            Transmit:               Receive:
Protocol            BSD Compress/15          BSD Compress/12
```

NCP states not negotiated are displayed.

netstat -ip
Member Link
Example

```
netstat -ip syn28
                    *** PPP Layering Status ***
State              Current:  NCP Open       Prior:    NCP Opening
                    *** LCP Status ***
State              Current:  Open          Prior:    Ack received
Options            Local:                  Remote:
MRU                1500                    1500
Auth type          None                    None
LQM                None                    None
Magic               0x7641f607               0xee82361e
MRRU               1500                    1500
Short Sequence      Off                     Off
Endpoint Disc       3:00-80-2d-05-58-29      1:0x14f3
                    *** NCP (MP) Status ***
State              Current:  Open          Prior:    Closed
                    Attached to bundle mp1
```

netstat -b

The **-b** option displays MP bundle information for currently active bundles.

The syntax for **netstat -b** is:

netstat -b [bundle]

If the bundle ID is not specified, then all active bundle information is displayed. If the bundle ID is specified (e.g., **netstat -b mp1**), then the specified bundle information is displayed.

netstat -b Example

The following example displays a bundle with two links. The member link PPP statistics details are repeated for each link in the bundle:

```
annex# netstat -b
[Bundle: mp1]
     MP packets sent: 128        MP packets accepted: 87
          Bytes sent: 4191            Bytes received: 2875
      Tx speed (Bps): 2               Rx speed (Bps): 0
      Capacity (bps): 128000          Capacity (Bps): 16000
  Packets fragmented: 128        Fragments discarded: 0
 Fragments generated: 128        Fragments assembled: 87
     Peak links used: 2           Current links used: 2
         Frames sent: 128            Frames received: 177
    Frames discarded: 0                      Dropped: 0
          Mbuf drops: 0                    Net down: 0
       Truncated hdr: 0          Bad hdr or LCP code: 0
[Bundle: mp1][Member link: syn32]
         Frames sent: 84             Frames received: 84
    Frames discarded: 0                      Dropped: 0
          Mbuf drops: 0                    Net down: 0
       Truncated hdr: 0          Bad hdr or LCP code: 0
[Bundle: mp1][Member link: syn28]
         Frames sent: 44             Frames received: 47
    Frames discarded: 0                      Dropped: 0
          Mbuf drops: 0                    Net down: 0
       Truncated hdr: 0          Bad hdr or LCP code: 0
```

MP Statistics

The following table describes the MP data displayed when entering a **netstat-b** command:

MP Statistics	**Description**
MP packets sent:	Packets sent to all NCPs.
MP packets accepted:	Packets received from all NCPs.
Bytes sent	Number of bytes sent by the .
Bytes received	Number of bytes received by the .
Tx speed (BPS)	Speed in bytes per second that a modem/TA has negotiated (outbound).
Rx speed (BPS)	Speed in bytes per second that an external modem receives (inbound).
Capacity (bps)	The total aggregate bandwidth of each leg of the mp bundle in bits per second.
Capacity (Bps)	The total aggregate bandwidth of each leg of the mp bundle in bytes per second.
Packets fragmented:	Packets that were fragmented.
Fragments discarded:	Fragments lost.
Fragments generated:	Fragments that were generated.
Fragments assembled:	Fragments successfully assembled.
Peak Links used:	Peak links used during the lifetime of the bundle.
Current Links used:	Current links in use.

PPP Statistics

The following table describes the PPP data displayed when entering a **netstat-b** command:

PPP Statistics	Description
Frames sent:	The number of frames successfully sent.
Frames received:	The total number of frames received.
Frames discarded:	The total number of frames that were discarded for one of the following reasons:
Dropped:	Queuing slots were not available.
Mbuf drops:	The output routines were called without a packet.
Net down:	The interface wasn't ready.
Truncated hdr:	Received frame was missing data.
Bad hdr or LCP code:	Frames received with invalid or unsupported protocol types.

Chapter 10
Serial Line Internet Protocol

This chapter describes how to configure the RAC for Serial Line Internet Protocol (SLIP) applications. Applications include:

- Connecting a single remote node (PC) to a TCP/IP network.
- Connecting two TCP/IP networks together.

SLIP lets the RAC establish sessions over its ISDN B channels or WAN DSOs. Once established, these sessions carry TCP/IP data from remote nodes and other types of RACs that originated as serial traffic. SLIP sessions operate over RAC ISDN B channels in combination with the following:

- Analog modems (V.34).
- ISDN BRI lines with terminal adapters using a rate adaptation protocol.

The RAC implementation of SLIP is compatible with the 4.3BSD implementation.

SLIP and Compressed SLIP

A SLIP link is a point-to-point connection between two hosts or devices that enables the transmission of TCP/IP packets over a serial line. Data transmission and reception is possible only between the nodes at the ends of the link.

For example, using a SLIP link, you can connect a remote PC to a network without requiring special interface hardware.

The Compressed Serial Line Internet Protocol (CSLIP), which is basically SLIP with VJ header compression improves interactive latency. This means that when you type characters on the keyboard, they show up immediately. You can choose either a configuration that uses compressed SLIP always, or one that uses compressed SLIP only when the remote end sends compressed SLIP packets.

The RAC's implementation of SLIP offers four options:

- Perform compressed SLIP
- Allow compressed SLIP
- Discard ICMP requests over the SLIP link
- Give interactive traffic priority over other traffic

SLIP Configuration Overview

To configure the RAC for SLIP sessions, follow these steps:

1. **Decide how IP addressing will be handled. Addresses can be handled by one of the following methods:**
   - Dial-up addressing (dynamic addressing)
   - Fixed addressing (static addressing)
2. **Edit the configuration file to define SPBs.**
3. **Review the default global port parameters, then reset the parameters you require to implement your SLIP configuration.**

Step 1: Decide How to Handle IP Addressing

The RAC handles IP addresses using one of the following methods:

- Dial-up addressing
- Fixed Per B-Channel addressing
- SPB addressing

You can configure the RAC for both methods, but dial-up addressing has priority over fixed addressing. For information about how the RAC operates when both dial-up and fixed addressing are enabled, see *Determining Dial-up Addresses Using the acp_dialup File on page 6-145*.

If you start SLIP via the CLI, you are prompted for the username and password if CLI security is not used and **ppp_slip_security** is set. If CLI security is used, then the name is acquired at log-in time. In either case, the name is used for dial-up addressing. If neither security is set or the SPB specifies SLIP mode, then no username is available and dial-up addressing cannot be used.

About Dial-up Addressing

Dial-up addressing is controlled through the global port parameter **address_origin**. When this is set to **acp**, the RAC uses the host-resident **acp_dialup** file to handle IP addressing. The file resides in the install directory. For more details on using the **acp_dialup** file (done via CLI), see *Dynamic Allocation of Network Addresses on page 6-142*.

Any ACP dial-up address request that comes from the RAC includes the RAC address and an associated user name, which are used as keys in this file. Once the keys are matched, the corresponding dial-up addresses are returned to SLIP on the RAC. The addresses are displayed to the user in **The Annex Address is... Your address is...** message before SLIP starts. If the addresses are not found in the dial-up file, then the local addresses are used. Dial-up addressing offers the ability to assign IP addresses to individual users.

About Fixed Per B-Channel Addressing

Fixed IP addressing for the RAC is controlled through the **remote_address** parameter when used with the **set wan b** command. This parameter and command associates IP addresses with PRI B channels.

When needed, if the **address_origin** parameter is set to **local**, the IP address set using this command/parameter combination is automatically used for calls over the B channels. Unlike dial-up addressing, fixed IP addressing associates IP addresses with B-channels, not with specific users.

About SPB Addressing

You can set the local and the remote addresses in the SPB. You can use this method to set addresses based on calling number or other call setup parameters.

Setting addresses by using SPB addressing takes precedence over Fixed Per B-Channel addressing.

Step 2: Edit the Configuration File

Session Parameter Blocks (SPBs) are structures within the configuration file. SPBs enable a RAC to handle one or more types of calls differently from the default call handler. For details, refer to *Using the Default Call Configuration* section. If you need to create any SPBs to handle special PPP details, do it now.

See *Configuring the WAN Interfaces, Global Ports, and Sessions on page 5-1*, for detailed information on SPBs.

To edit the configuration file:

1. **Open the configuration file.**

   The configuration file is located in the install directory on the UNIX load host (default filename is **/usr/annex/config.annex**).
   Use any system editor (e.g., **vi**, **textedit**) to edit the file.

2. **Go to the** wan **or** pri **section in the file. This section contains examples. Do the following:**

   - Read the information that precedes each sample SPB provided in the configuration file. You should be familiar with this information before you enable or disable a default SPB.
   - Remove the comment character (#) from the beginning of each line of any sample SPB you want to enable.
   - Enter a comment character (#) at the beginning of each line of each SPB you want to disable.

- Enter any new SPBs.

3. **Save the file.**
4. **Issue a** reset annex session **command from** na **or** admin**.**

Step 3: Review and Reset Global Port Parameters

The RAC ships with a set of default global port parameters already stored in non-volatile RAM. Review the defaults to determine which ones you need to change to satisfy your configuration requirements for SLIP, security, etc.

The remainder of this section provides the following information:

- A list of the default settings for the Serial Networking and SLIP global port parameter groups.
- Instructions for changing a global port parameter setting.
- Instructions for using the **set wan b** command to associate IP addresses with RAC PRI B channels.

To view the entire set of default global port parameters use **na** or **admin** to issue the **show port all** command.

Default SLIP-Related Global Port Parameters

Table 10-1 lists the default parameters related to the SLIP protocol stored in the RAC nonvolatile memory when shipped. You can view these SLIP-specific parameters through the **show port slip** command issued from **na** or **admin**.

Table 10-1. Default Serial Networking Parameter Settings

Parameter	Default Setting
local_address	0.0.0.0
metric	1
net_inactivity	off
allow_compression	N
address_origin	local
slip_ppp_security	N
do_compression	N
net_inactivity_units	minutes
subnet_mask	0.0.0.0
slip_no_icmp	N
slip_mtu_size	small
slip_tos	N
wan b remote_address	0.0.0.0

Changing a Global Port Parameter Setting

To change a global port parameter setting using **na**:

1. **At a terminal connected to a UNIX host, enter:**

   `%` **na**

   The following prompt displays on the screen:

   ```
   Annex network administrator Rx.x January 1, 1997
   COMMAND:
   ```

2. **Specify the RAC on which you intend to change global port parameter settings at the** COMMAND: **prompt. Specify the administrative password for host at the** password: **prompt.**

   You can specify the RAC by its IP address or name. If you intend to change global port parameter settings on more than one RAC, separate their IP addresses or names using a comma (,). The password is the administrative password for this host.

The following is an example:

```
COMMAND:annex 132.245.6.40 or
        annex 132.245.6.40,132.245.6.45
        password:
```

3. **Specify a new setting for the global port parameter at the** COMMAND: **prompt.**

   For example, to change the default setting of the **address_origin** parameter (**local**) to enable dial-up IP addressing through the **acp_dialup** file, enter the following:

   ```
   COMMAND:set port address_origin acp
   ```

   

   The new parameter setting is automatically stored in nonvolatile RAM.

4. **To review your changes, issue the** show port all **command at the** COMMAND: **prompt.**

   This command displays all of the global port parameter settings. To locate the parameters you changed, press the return key which allows you to scroll down through the file.

   ```
   COMMAND: show port all
   ```

5. **Enter** quit **at the** COMMAND: **prompt to exit** na**.**

   ```
   COMMAND: quit
   ```

Assigning IP Addresses to PRI B Channels

Use the **set wan b** command issued with the **remote_address** global port parameter to associate IP addresses with PRI B channels. This lets you configure fixed IP addressing for RAC sessions. You can use **na** or **admin** to use this command.

Command Syntax

Use the following command syntax when creating B channel IP address assignments:

```
set wan b=<ch-range> remote_address <ip-addr> [<increment>]
```

where

<*ch-range*> is a single B channel number or the entire set of B channels specified by a range.

<*ip-addr*> is the IP address you want to assign to a single B channel or the first channel of the entire set.

<*increment*> is the value (integer or dotted quad) by which you want to increment automatic IP address assignment to B channels in a range or list after the first assignment.

Usage Rules

You can associate IP addresses with PRI B channels:

- For a single B channel (e.g., 4).
- For the entire set of PRI B channels by:
  - Specifying channels as a range of two numbers separated by a dash (1-23 for T1; 1-30 for E-1).
  - Using the keyword **all**.

When assigning IP addresses to the entire set, you can specify an increment by which IP addresses are assigned in sequence based on the increment value. For example, if you specify the entire set of B channels available with a T1-based RAC PRI module (23), an IP address of 132.245.66.230, and increment of 2, the following assignments occur:

```
set wan b=1-23 remote_address 132.245.66.230 2
```

B channel #1 is assigned the IP address of 132.245.66.230, B channel #2 is assigned 132.245.66.232, B channel #3 is assigned 132.245.66.234,...., and B channel #23 is assigned 132.245.67.18.

When you do not specify any B channels, the command makes 23 or 30 IP address/B channel assignments based on the increment value.

When an increment is not specified, the command assumes a default increment of zero (0).

Sample Configuration for a Single Remote Node

In Figure 10-1, a single remote PC (user *green*'s) is connected to a RAC through a SLIP session. The session is running via a modem and the PRI line to the RAC. Once the SLIP session is established, the remote PC appears to the LAN as a directly attached device.

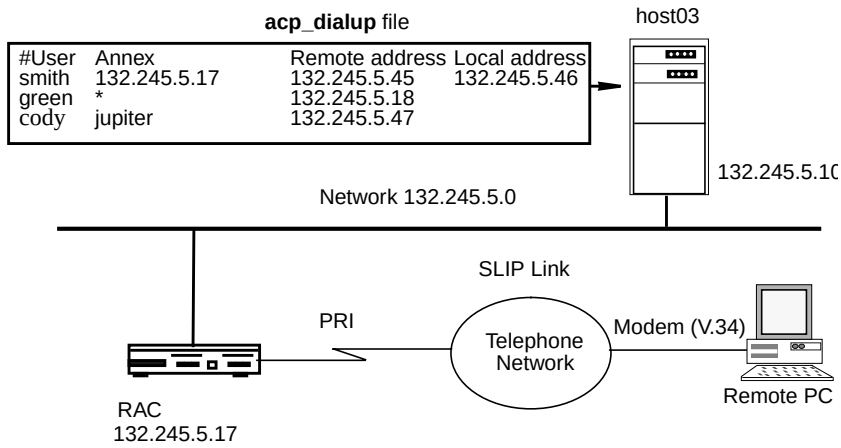

Figure 10-1. Connecting a Single Host Using SLIP

To enable this configuration:

1. **Choose whether you want to use dial-up IP addressing or fixed IP addressing.**

   If you choose dialup addressing, edit the **acp_dialup** file. Provide user *green* with access from all RACs and Remote Annexes by specifying a wildcard (*) and a specific remote address for user *green* (e.g., 132.245.5.18). Also, set the **address_origin** parameter to **acp**.

   For more details, see *Dynamic Allocation of Network Addresses on page 6-142*.

If you choose fixed IP addressing, use the **set wan b** command with the **remote_addresses** global port parameter to associate a set of IP addresses with the PRI B channels. Also, set the **address_origin** parameter to **local**.

You can configure the RAC for both methods, but dialup addressing has priority over fixed addressing. However, if the host where the ACP server resides becomes unreachable, or there is no entry in the **acp_dialup** file for a particular user, the RAC relies on fixed addresses to provide a remote address for the link.

2. **Edit the configuration file to define an SPB.**

   You can use the default SPBs provided as part of the **config.annex** file or create them specifically for your requirements. For more details, see *Step 2: Edit the Configuration File on page 10-4*.

3. **Review the global port parameter settings, and if required, reset these parameter settings.**

   - Set the **cli_security** parameter to Y enabling **cli_security** which implements user authentication by the host-based ACP server for all CLI connections.
   - Set the **allow_compression** parameter to **Y** if you want the RAC to accept compressed packets.
   - Set the **local_address** parameter to the RAC **annex inet_addr** address.

Sample Configuration for Connecting Two Subnets

Figure 10-2 illustrates two Class C subnets connected through a SLIP link. The IP addresses assigned to the end-points of the SLIP link are the hosts' primary network IP addresses.

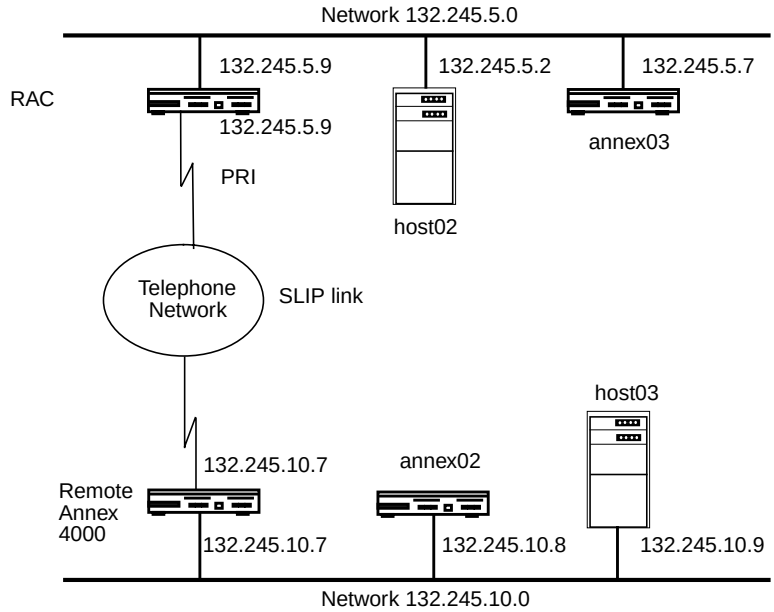

Figure 10-2. SLIP Link with Two IP Addresses

To enable this configuration:

1. **Edit the configuration file to add the IP address of the RAC.**

   For more details, see *Dynamic Allocation of Network Addresses on page 6-142*.

2. **Edit the configuration file to define an SPB.**

   You can use the default SPBs provided as part of the **config.annex** file or create them specifically for your requirements. For more details, see *Step 2: Edit the Configuration File on page 10-4*.

3. **Review the global port parameter settings, and if required, reset these parameter settings.**

   - Set the **cli_security** parameter to **Y** enabling **cli_security** which implements user authentication by the host-based ACP server for all CLI connections.

- Set the **allow_compression** parameter to **Y** if you want the RAC to accept compressed packets.
- Set the **local_address** parameter to the RAC **annex inet_addr** address.

Routing Across a SLIP Link (Basic Passive RIP)

Both active and passive routing are available via the Routing Information Protocol (RIP) on the RAC. The following sections deal with using only the most basic features of passive RIP and are intended for administrators who need minimal routing features. For complete information on both passive and active RIP, see *Routing Information Protocol (RIP) on page 11-1*.

Both active and passive RIP are enabled by default. To turn off active RIP, set the interface parameter **rip_advertise** to **none** for all RAC interfaces.

The RAC bases its routing table on the information you specify in the **gateway** section of the configuration file. As a passive gateway, the RAC then updates the table according to information it receives from other routers but does not broadcast routing information itself. This means that a RAC with a SLIP interface forwards packets addressed to the host at the remote end of the connection, but does not inform other hosts, routers, or RACs that it has this capability. Other hosts and routers on the same network must be told about the route before they can use it.

To guarantee that a route in the **gateway** section of the **config.annex** file (or in the gateway entry in **/etc/gateways** on UNIX hosts) uses a particular SLIP interface, the next hop in the route must match the remote address of the SLIP link.

Routing Between Two Networks

To make other hosts aware of a route over a SLIP link, you could use static routing in which a host running **routed** advertises a route for the RAC. Create an entry in a host's **/etc/gateways** file. Using the example in Figure 10-2 on page -11, *host03*, whose Internet address is 132.245.10.9, has the following **/etc/gateways** file entry:

```
host 132.245.5.9 gateway 132.245.10.7 metric 1 passive
```

This entry advertises a route for the host with the Internet address 132.245.5.9 through the RAC at 132.245.10.7. A host running **gated** can accomplish the same thing.

Having a host advertise a route is generally difficult to do correctly, and results in an *extra-hop* situation. Hosts must direct their traffic destined for host 132.245.5.9 to host 132.245.10.9, which then routes the traffic to the RAC at 132.245.10.7. To avoid this extra hop, the host at 132.245.10.9 needs to send out an ICMP redirect message.

A RAC can advertise SLIP links via RIP or RIP-2. To make RACs aware of a route using a SLIP link, create a **gateway** entry in the configuration file. Using Figure 10-2 on page -11, the entries for the RACs on network 132.245.10.0 are:

```
annex 132.245.10.7

     route add 132.245.5.0 255.255.255.0 132.245.99.2 1

else

     route add 132.245.5.0 255.255.255.0 132.245.10.7 2
     route add 132.245.99.2 255.255.255.0 132.245.10.7 1

end
```

These entries inform *RAC01* that *host01* is a gateway to network 132.245.5.0 (with a metric of 1) and inform other RACs on network 132.245.10.0 that *RAC01* is a gateway to either *host01*(with a metric of 1) or network 132.245.5.0 (with a metric of 2).

Route Cache

The route cache is a list of routing entries stored by the RAC. When the RAC boots, the route cache is created from the **annex...end** and **subnet...end** blocks in the gateway section of the configuration file. When **routed** starts, entries in the route cache are added to the routing table if their next hops are on a network directly connected to the RAC. The RAC examines once and discards routes outside the **annex...end** blocks if the destination is not a directly connected network or is an inactive link (such as a SLIP link at boot time).

Extending a Single Host onto the Network

The RAC can use Proxy-ARP to attach a single host and remote RACs onto the network transparently. Using Proxy-ARP, the RAC answers ARP requests for the destination address of a SLIP link with its own hardware address. The following is an example of the type of ARP entry that would appear on the RAC for the SLIP interface in Figure 10-1:

```
bunky (132.245.5.18) at 00-80-2d-00-26-cd permanent published
```

Typically, a Proxy-ARP is used when the RAC's SLIP link is to a single device; i.e., both the device and the RAC use the same Internet network address. No other routing information is required, nor can it be used with this configuration. Do not attempt to use Proxy-ARP with routing commands in the **%gateway** section of the configuration file. If necessary, only **acp_userinfo** can reliably add routes on such links.The destination address of the SLIP link must be on the same network as the RAC.

BOOTP Requests

BOOTP is a bootstrap protocol that allows a diskless client to determine its Internet address, the Internet address of the server, and the name of the file to be loaded into memory.

- The RAC ROMs use BOOTP to obtain boot information without requiring any manual setup on the RAC.
- If a client sends a BOOTP request to the RAC over a SLIP line, the RAC responds with its current local address, remote address, subnet mask, and boot host.

Chapter 11
Routing Information Protocol (RIP)

This chapter describes the RAC implementation of Internet Protocol (IP) routing and the Routing Information Protocol (RIP) for SLIP, PPP, and Ethernet ports. The following topics are covered in this chapter.

- Prerequisites.
- Understanding RAC IP routing and RIP. This includes sections on:
  - RIP versions.
  - Routing tables.
  - The difference between passive and active RIP.
  - Routing interfaces.
  - IP addressing.
  - Proxy ARP.
- Overview of routing configuration parameters.
- Enabling, disabling, and configuring passive RIP without active RIP.
- Prerequisites for using active RIP.
- Configuring active RIP.
- Reference descriptions of all RIP configuration parameters.
- Displaying routing information.
- Troubleshooting.
- Other documentation.

Prerequisites

This chapter assumes that:

- Your RAC is attached to the network and both are operational.
- Any hosts you wish to reach are attached to the Ethernet or the RAC and links to them have been proven to work.
- Any modems you intend to use have been tested in the configuration in which you intend to use them. For example, testing a modem for dial-in and thereby assuming dial-out works is not sufficient.
- RIP is the primary routing application on your network and you know which routers are running it; you also know which RIP version (1 or 2) the routers are running.
- You understand RIP. Read the remainder of this chapter, and if you need more information, consult the sources cited at the end of the chapter.

Understanding IP Routing and RIP

IP routing is the process of determining the best path to follow to deliver a piece of data, contained in an IP *datagram*, to its destination on a TCP/IP network. Only a simple network, in which all systems directly attach to a single LAN, does not require routing. But a simple network can easily grow into an internet - a collection of multiple, interconnected networks - where routing is required to reach hosts on distant networks. Special machines called routers connect two or more networks to each other and route packets from one to the other.

Routers use routing applications to learn and generate routing information. The RAC uses RIP, an application that runs on top of the transport-layer protocol UDP (User Datagram Protocol), for this purpose. Through information broadcast by other RIP routers, the RAC maintains a routing table containing up-to-date routes to various destinations on local and remote networks. (This table also contains AppleTalk routes.)

Definition of a Route

The information a RIP router broadcasts contains the network addresses in the router's routing table and the number of routers (including itself) that must be crossed to reach these addresses. Each router is a *hop* in the path to the address; the next router in the path is called the *next hop*; and the total number of routers that must be crossed is called the *hop count* (also known as the *metric*). The next hop in a routing table is always on a network directly attached to the router containing the table. The RIP metric is an integer from 1 through 15, with 1 representing the best (shortest) route. Together, the IP address of a destination host or network, the next hop, and the metric constitute a *route*.

If one of the routers or networks in a particular RIP route fails, a time-out mechanism prevents other routers from keeping the route in their routing tables and advertising it as viable. If Router A's table contains a route learned from Router B, and three minutes elapse between Router A hearing from Router B, Router A marks the route in question as invalid by setting its metric to 16; this value is referred to as *infinity*. Router A advertises the route, with a metric of 16, for two more minutes, so that other routers can learn the route is unusable. At the end of that time, Router A deletes the route from its routing table.

Routing versus Forwarding

Do not confuse IP routing with IP forwarding. IP forwarding is the process of sending an IP datagram to the next hop on the way to its destination. IP routing is the algorithm that determines the next hop to use. The RAC kernel performs IP forwarding; RAC RIP performs routing.

Choosing Passive or Active RIP

The RAC can run passive RIP alone, or both passive and active RIP. The default is both, but you can disable active RIP by changing the value of the **rip_advertise** parameter (see *Enabling Passive RIP Alone on page 11-26*). When running only passive RIP, the RAC revises its routing table based on the routing updates it receives, but does not broadcast updates itself. When configured for active RIP (which is the default), the RAC acts as a full router: it not only listens for updates but also broadcasts them every 30 seconds.

Using only passive RIP is appropriate when:

- Your network is small and does not connect to other networks. In this case, you can configure any necessary routing information by defining static routes. Static routes are routes you enter yourself, as opposed to those RIP learns over the network.
- Your network is large but has only one gateway to the other networks you need to access. In this case, you can save the overhead of broadcasting updates over all nodes. Instead, you can use passive RIP, configure static routes where needed, and define the gateway to other networks as the default route. The RAC will use the default route if it knows no other route to a given destination.

In most other situations, active RIP is more useful than passive RIP.

The **routed** parameter controls whether any kind of RIP is enabled. Initially, **routed** is set to **Y** and both active and passive RIP are enabled. See *Enabling Passive RIP Alone on page 11-26* for information on setting **routed** and disabling active RIP; see *Active RIP Prerequisites on page 11-46* for information about active RIP requirements.

RIP Versions

The RAC supports both RIP version 1 (STD 34, RFC 1058) and RIP version 2 (RFC 1388). RIP 2 is a backward-compatible extension of RIP 1 that expands the amount of information carried in RIP updates and authenticates them using passwords. RIP 2 broadcasts or multicasts updates, depending on whether or not updates are to be sent to RIP 1 as well as RIP 2 systems.

Not all routers support RIP version 2. Check the RIP versions running on the routers on your network before configuring RIP on the RAC.

Route Cache and Routing Table

The RAC stores IP routes in two places: the route cache and the routing table.

Route Cache

The route cache contains user-configured routing information - static routes, and sometimes a default route. The RAC can learn default routes through RIP updates as well as from you, which is why all default routes are not necessarily cached.

The RAC copies static and default routes to the route cache from the **gateway** section of the RAC configuration file (see *Entering Routes in the Configuration File on page 11-28*) and/or from routes you define via the CLI superuser **route** command (see *Entering Routes Using the route Command on page 11-39*).

Routes in the cache include those whose next hops are directly reachable - that is, up and running on a network directly connected to the RAC - and those that are not yet reachable. Routes with reachable next hops are immediately copied to the RIP routing table. Routes whose next hops are not yet directly reachable are copied to the routing table as soon as their next hops become reachable. The latter technique saves the RAC the trouble of consulting the configuration file, which typically is not stored on the RAC, each time a route's status changes. A copy of the route remains in the routing cache, in case its next hop becomes unreachable again.

The route cache also contains back-up routes if two routes in the routing table have the same destination interface. In this case, one of the routes is removed from the routing table and stored in the cache, as follows:

- If one of the routes is for dial-out and the other is not, the dial-out route is moved to the cache.
- If the routes are both dial-out or both non-dial-out, the route with the highest metric is moved to the cache.
- If the routes are both dial-out or both non-dial-out and have the same metric, the route with the preferred interface type is stored in the routing table and the other in the cache. For example, en0 routes are preferred over serial interface routes.

To display the route cache, use the **netstat -C** command described in *Displaying the route cache on page 11-66*.

Routing Table

The RIP portion of the routing table contains routes for all IP destinations the RAC can reach currently. This includes the local loopback route and the user-configured default route (if one exists), as well as routes copied from the route cache, routes learned by RIP, and routes to directly connected networks. The latter are called *interface* routes and are discussed in *Interface Routes on page 11-10*.

If a route becomes invalid (because its hop count reaches 16 or the interface to its next hop goes down), it is kept for two minutes in the routing table, so that the route's invalidity is advertised to other routers (when active RIP is enabled).

RIP uses the routing table for the updates it advertises (when active RIP is enabled).

To display the routing table, use the **netstat -r** command described in *Displaying Routing Information on page 11-60*.

How Hosts Learn Routing Information

Unlike routers, hosts do not run routing applications and do not maintain extensive routing tables. Host table entries tend to be restricted to information learned via ICMP messages generated by routers on the network. When the RAC is acting as a router, it generates messages to provide hosts with the following information:

- Whether or not the RAC is available as a router on the host's local network.
- Whether or not the RAC knows of a better first hop than itself to a given destination.

- The size of the biggest packet the RAC can forward without fragmentation. Fragmentation involves breaking a packet into pieces for transmission; fragmented packets must be reassembled when received.

The following sections describe the ICMP techniques the RAC uses to transmit this information.

Not all TCP/IP routers support these features, nor are all hosts configured to take advantage of them. Check the hosts and routers on your network.

RAC Availability - Router Discovery Mechanism

Router Discovery (RD) is an ICMP-based mechanism that helps hosts locate operational routers on directly connected networks. The RAC uses this mechanism if IP routing is enabled (i.e., if the **routed** parameter is set to **Y**; see *Choosing Passive or Active RIP on page 11-4*).

Using RD, the RAC responds to solicitations from hosts by sending them ICMP messages advertising the RAC as a router. The RAC also broadcasts unsolicited RD advertisements every 10 minutes to all hosts listed on the LAN. If a host does not seem to be learning about RAC routers on directly connected networks, make sure RD is implemented on that host.

RD is not a routing protocol, since it does not help a host decide the best next hop for a particular destination address.

Better First Hops - Redirect Messages

If the RAC determines that there is a better first hop than itself for a datagram it has received, it forwards the datagram and sends an ICMP *redirect* message to the host that originated the datagram. The redirect message contains the address of the next hop the originator should use for subsequent datagrams to that destination.

The RAC itself does not listen for redirects if routing is enabled (i.e., **routed** is set to **Y**). If a redirect is received while routing is enabled, the RAC syslogs the event at WARNING level. If routing is disabled (i.e., **routed** is set to **N**), the RAC listens for ICMP redirects but does not transmit them.

Avoiding Fragmentation - Path MTU Discovery

The Maximum Transmission Unit (MTU) is the largest packet a network can transmit, and it varies depending on the type of network. One way IP accommodates different MTU sizes is by allowing datagrams to be fragmented. Another, more efficient, IP technique avoids fragmentation by discovering the Path MTU, which is the smallest MTU of any hop in the path from a datagram's source to its destination.

The RAC fragments datagrams when necessary, but also implements Path MTU discovery for datagrams that call for it. If the Don't Fragment (DF) bit in a datagram's header indicates the datagram can be fragmented, and the datagram's size exceeds the MTU of the next hop in the datagram's route, the RAC fragments the datagram. If the next hop's MTU is exceeded by a datagram in which the DF bit specifies *not* to fragment, the RAC returns an ICMP *Datagram Too Big* message to the source host. This message includes the next-hop MTU and indicates that fragmentation was needed and the DF bit was 1.

When a host receives a *Datagram Too Big* message, it replaces its estimated Path MTU with the value returned in the message. Typically, the Path MTU discovery process continues until the host succeeds in sending datagrams to a given destination without triggering any *Datagram Too Big* messages.

Routing Interfaces

Initially, RIP runs on all operational IP *interfaces*. RAC interfaces are the two WAN interfaces and the Ethernet interface. The IP address of a RAC interface is the local address you configure for the WAN interface. Typically, the **en0** address is also the IP address of the entire RAC (as defined by the ROM monitor command **addr** or the **inet_addr** parameter).

Interface Routes

For each operational link to the RAC that is not on the same subnet (or network, if subnetting is not used) as the RAC, the RAC creates an interface route to the link's remote destination, with the RAC as the next hop. The RAC stores interface routes in the routing table. *These routes are never replaced by routes RIP learns, and you cannot delete them yourself.* (For information on subnetting, see *Subnetting Using Subnet Masks on page 11-15*.)

Non-Operational Interfaces

Except in the case of dial-out interfaces, when an interface goes down, the route for the network attached to that interface is marked as unreachable in the routing table, as are any routes whose next hops are on that network. After two minutes, the RAC deletes the unreachable routes from the table. The CLI command **netstat -i** displays the status of RAC interfaces.

Dial-out interfaces are treated differently from other interfaces. When a demand-dial or dial-out link has been established and the RAC has received the pertinent RIP routing information, the RAC retains that information even when the link becomes quiescent. After a link becomes quiescent, if an alternate path with an equal metric becomes available for any of the destinations previously reachable over that link, the routes for those destinations are updated to use the new path. However, if you **reset** the link using **na** or **admin**, the routes expire immediately.

IP Addressing

Assigning appropriate IP addresses and subnet masks to your network and the interfaces of attached nodes is essential for routing to work properly. The following sections explain basic IP addressing, subnetting, and supernetting.

An IP address contains four bytes (octets), expressed in decimal, with a period (dot) separating the octets. This is referred to as *dotted decimal notation.* 132.254.1.2 is a sample IP address.

Address Classes

Each IP address contains a network portion and a host portion. To allow for different network-to-host ratios, the Internet supports three classes of IP addresses - A, B, and C - that vary in the number of bytes they allot to the network and host portions. The leftmost byte of an address indicates its class. Table 11-1 lists the network classes, the decimal number that appears in the first octet, and the sections of the Internet address that are assigned to the network and to the host. The *nnn* represents all or part of the network number and the *hhh* represents all or part of the host address.

Table 11-1. Network Classes

Class	First Octet	Internet Address
A	1-126	*nnn.hhh.hhh.hhh*
B	128-191	*nnn.nnn.hhh.hhh*
C	192-223	*nnn.nnn.nnn.hhh*

The following values for the first octet are illegal; do not use them: 0, 127 (reserved for loopback), and 224-255.

Sample Addresses for Different Classes

According to Table 11-1:

- 125.17.5.2 is a class A address, indicating host 17.5.2 on network 125. Class A addresses allow the most hosts and the fewest networks.
- 132.254.1.1 is a Class B address, indicating host 1.1 on network 132.254. Class B addresses provide the most balanced network/host ratio and are in widespread use on the Internet.
- 192.254.230.6 is a Class C address, indicating host 6 on network 192.254.230. Class C addresses allow the most networks and the fewest hosts.

Example of Class C Address Configuration

Figure 11-1 shows a configuration using four Class C node addresses: 194.254.230.1 (*host01)*, 194.254.230.2 (the RAC), 192.252.230.1 (*host02*), and 191.250.230.55 (the PC).

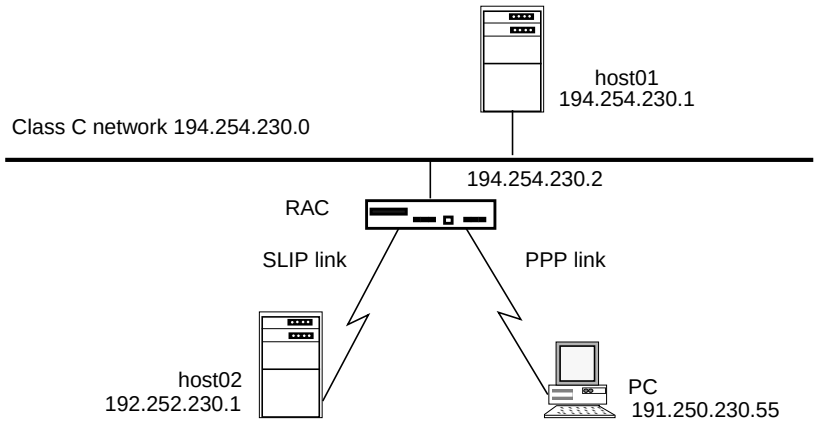

Figure 11-1. Configuration Using Four Class C Node Addresses

Obtaining Network Addresses

Before you can assign host addresses, you need one or more network addresses. It is recommended that you use network addresses that can attach to the Internet. Your Internet service provider should supply you with the network address(es) you can use. If you do not have a provider, contact the Registration Services of the InterNIC by sending electronic mail to **hostmaster@rs.internic.net** or by sending U.S. mail to:

Network Solutions
Attn: InterNIC Registration Services
505 Huntmar Park Drive
Herndon, VA 22070

If you have questions concerning registration policy or status, telephone service is available Monday through Friday, 7 a.m. to 7 p.m., Eastern Standard Time. Call 703-742-4777.

Setting the RAC IP Address

Set the RAC address by using the ROM Monitor **addr** command or by setting the configuration parameter **inet_addr**. There are the following special cases:

Turning off IP Services

- Setting **addr** or **inet_addr** to 255.255.255.255 turns off all IP services, including SLIP, PPP, and IP routing. The RAC will continue to support non-IP services, such as ARAP and LAT, provided that they are configured properly. If IP is not being used, turning it off saves overhead and can enhance security.

Enabling IP without an Ethernet Interface

- Setting **addr** or **inet_addr** to a valid IP address and RAC **subnet_mask** to all ones (255.255.255.255) installs IP but leaves the IP address of the Ethernet interface (en0) undefined. IP services, including SLIP, PPP, and IP routing, will be available (if configured properly) but no routes will use the RAC Ethernet interface, if one exists. (If there is no Ethernet interface, you must set the subnet mask to all ones.)

Obtaining the IP Address via BOOTP and RARP

- Setting **addr** or **inet_addr** to 0.0.0.0 causes the RAC ROMs to broadcast for an IP address (at boot time) via BOOTP and RARP. If no IP address is found on the network, the RAC will not boot.

Subnetting Using Subnet Masks

Every IP address on a RAC interface or on a node connected to the RAC has a subnet mask associated with it. You can define the mask yourself or let the RAC assume the default.

If the RAC uses the default mask, your network will not be divided into subnets. The advantages to subnetting a network include:

- Creating a hierarchically organized addressing environment that is logical and easy to remember.
- Reducing the number of entries required in individual routing tables as well as the number of network addresses you must use (and Internet administrators must assign).

A subnet mask lets you designate part of the host portion of an IP address as the subnet. The mask:

- Contains ones in every position that corresponds to the network and subnet part of the address.
- Contains zeros in every position that corresponds to the host address.

For example, used with a Class B address, a subnet mask containing the following bits identifies the first eight bits of the host portion (third octet) as a subnet:

```
11111111 11111111 11111111 00000000
```

For convenience, enter a mask in dotted decimal notation, not in binary. The following is the dotted decimal equivalent of the bits shown above:

```
255.255.255.0
```

You assign a subnet mask to the RAC and to its ports by using the RAC **subnet_mask** parameter and the port **subnet_mask** parameter, respectively. By default, the subnet mask is set to the intrinsic mask for the class of address: 255.0.0.0 for Class A addresses, 255.255.0.0 for Class B addresses, and 255.255.255.0 for Class C addresses. The default mask for a SLIP or PPP port is 0.0.0.0, which is interpreted as 255.255.255.255 (denoting a host route). To use subnetting, you must change these masks.

Incorrectly configuring or not setting subnet masks can cause unrecoverable corruption of the routing table. To detect potential problems, RIP generates a syslog LOG_WARN message if the RAC subnet mask or a port subnet mask is left unset.

The RAC does not accept subnet masks containing non-contiguous one bits.

Figure 11-2 shows a simple configuration using subnet addressing. Given a network address of 132.254.0.0, you assign a subnet mask of 255.255.255.0 to *host01* and to the RAC. Assigning this mask defines your network as a subnet whose address is 132.254.1.0 and indicates that the final octet of each remote node defines the host portion of the node's address. In Figure 11-2, the host portions are 1, 2, 3, and 4.

For more information on configuring SLIP and PPP ports, see *Serial Line Internet Protocol on page 10-1* and *Point-to-Point Protocol on page 8-1*.

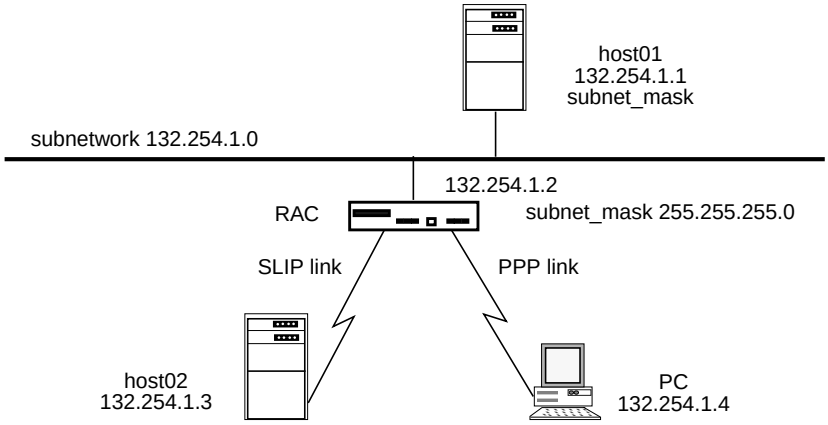

Figure 11-2. Subnetting with Passive RIP

Supernetting Using Subnet Masks

The RAC also supports supernetting for Class C addresses. Supernetting allows you to use a subnet mask that is shorter than the intrinsic subnet mask derived from the Internet address' class. Permissible Class C subnet masks range from 255.255.0.0 to 255.255.255.252, excluding 255.255.255.128, which is illegal.

Supernetting Example

Suppose that your Internet service provider (ISP) has a block of Class C addresses ranging from 192.24.0.0 to 192.31.255.0. Suppose that from this block, your ISP assigns you 4 Class C network addresses, 192.24.8.0 through 192.24.11.0. To the world outside, the route to your 4-network domain has a destination address of 192.24.8.0 and a mask of 255.255.252.0. Your 4 Class C networks provide you with up to 1024 subnet/host addresses, which you can configure as you wish. For more information on supernetting, see RFC 1519.

Proxy ARP for Interfaces on the Same Network

The section *Interface Routes on page 11-10* explained that when remote nodes connected directly to the RAC via SLIP or PPP links are not on the same subnet as the RAC, the RAC creates routes for them in its routing table, with the RAC as the next hop. However, when directly attached nodes *are* on the same or subnet as the RAC (see Figure 11-2), the RAC behaves differently. It ignores the interface routes in its tables and uses the Proxy ARP mechanism to forward packets across those links.

Proxy ARP is a variation of the Address Resolution Protocol (ARP), which dynamically maps IP addresses to their physical (Ethernet) addresses. When a network node must resolve an IP address, it broadcasts an ARP request on the network. The node whose IP address matches the one that was broadcast responds with its own Ethernet address. (To display the RAC ARP table, issue the CLI superuser **arp** command.)

The Proxy ARP scheme allows the RAC to answer an ARP request on behalf of any directly attached nodes on its network. In this way, the RAC becomes a Proxy ARP interface for the nodes and installs a static ARP entry for them in the RAC ARP table. When the RAC receives an ARP request for the address of any of these nodes, the RAC automatically responds by providing its own Ethernet address, thereby becoming responsible for forwarding packets to that node. Thus, when the RAC at 132.254.1.2 in Figure 11-2 receives a packet for 132.254.1.4, it forwards the packet to the PC at that address.

If one directly attached node is on the same subnet as the RAC and another is not, the RAC uses Proxy ARP for the former and routing for the latter, as shown in Figure 11-3. In this figure, the RAC uses Proxy ARP for the SLIP link and routing for the PPP link.

Do not attempt to configure a static route whose next-hop address is a Proxy ARP interface. Doing so causes packets to be routed improperly or not routed at all.

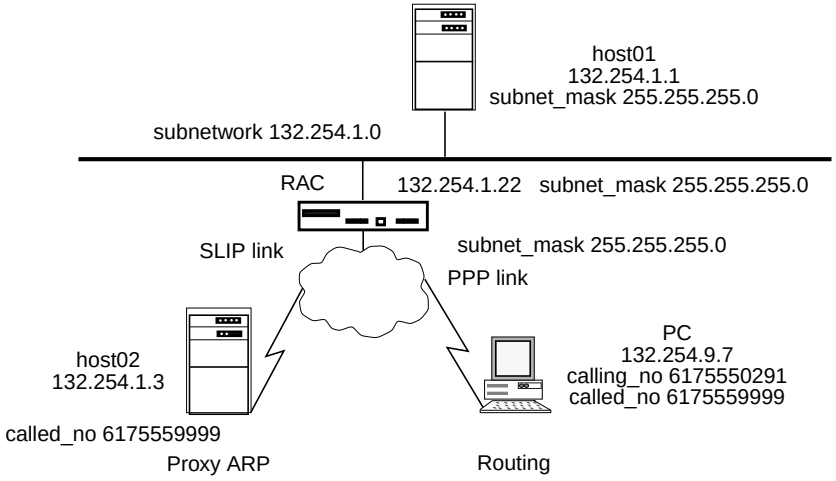

Figure 11-3. Proxy ARP versus Routing

To configure *host02* and the PC as in Figure 11-3 you use a Session Parameter Block (SPB):

```
%wan

begin_session routing
calling_no 6173335555
called_no 6175559999
call_action modem
set mode slip
set remote_address 132.254.9.7
set subnet_mask 255.255.255.0
end_session

begin_session proxy_ARP
called_no 6175559999
call_action modem
set mode ppp
end session
```

Once you have defined the remote end of the link and the subnet mask, the RAC applies the link's **subnet_mask** to this remote address to determine the network route to the remote link.

You can also set the remote address in the **acp_dialup** file, or, in the case of a PPP link, the RAC can learn the remote address from the remote node itself.

Setting the Broadcast Address

In order for RIP to either receive or advertise routing updates properly, you must set the RAC broadcast address correctly. Use the ROM monitor **addr** command (which prompts you for the broadcast address) or set the RAC **broadcast_addr** configuration parameter.

The IP routing code checks to make sure that the broadcast address is one of the following:

- A subnet broadcast address. If your network is subnetted, this is the recommended broadcast address. To specify this address, 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 one-bits. For example, if the RAC subnet address is 132.254.9.0, and the RAC subnet mask is 255.255.255.252, set the broadcast address to 132.254.9.3. To arrive at this result, 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 and then add this result to the subnet address.
- A network broadcast address. If your network is not subnetted, you can specify this type of broadcast address. To do so, set the network portion of the broadcast address to match the RAC network address, as determined by the intrinsic mask for the network class. And set the host portion of the broadcast address to all 1-bits.
- A limited broadcast address of 255.255.255.255. This reaches all nodes on the subnet. However, if you have more than one subnet on the same physical cable, the RAC will broadcast to all nodes on all of the subnets. This can be troublesome if some of the subnets or nodes do not recognize the broadcast.

If you do not specify one of the broadcast addresses listed above, RAC RIP generates a syslog message. It should be noted that the default for **broadcast _addr** is **0.0.0.0**, which RAC RIP routing does not support (because most hosts do not recognize it).

Overview of Configuration Parameters

This section provides an overview of the RIP-related configuration parameters and the commands for setting them. Subsequent sections describe using these commands. *RIP Configuration Parameters - Reference on page 11-53* provides formal descriptions of the parameters.

RIP configuration parameters fall into two groups - those that apply to the RAC as a whole, and those that apply to particular RAC interfaces (see Table 11-2 and Table 11-3). Within these two groups, some parameters apply to updates the RAC accepts, and others apply to updates the RAC generates. If only passive RIP is enabled, the parameters you can view and set are limited to the ones controlling the reception of updates. If active RIP is enabled, you can view and set all RIP parameters; the active-only parameters are so noted in the tables.

Table 11-2. RIP-specific RAC Parameters

Parameter	Description
rip_auth	Enables and disables authentication of RIP 2 datagrams.
rip_routers	Directs periodic RIP updates to a list of routers rather than broadcasting the updates. *Available with active RIP only.*
routed	Enables and disables RIP.

Table 11-3. RIP-specific Interface Parameters

Parameter	Description
rip_accept	Controls the networks for which RIP accepts routes and queries.
rip_advertise	Controls the networks for which RIP advertises routes. *Available with active RIP only.*
rip_default_route	Controls whether or not the RAC advertises itself as a default router. *Available with active RIP only.*
rip_horizon	Controls the split horizon and poison reverse mechanisms. *Available with active RIP only.*
rip_next_hop	Specifies whether or not the next hop value is included in RIP version 2 advertisements. *Available with active RIP only.*
rip_accept	Controls the networks for which RIP accepts routes and queries.
rip_advertise	Controls the networks for which RIP advertises routes. *Available with active RIP only.*
rip_horizon	Controls the split horizon and poison reverse mechanisms. *Available with active RIP only.*
rip_next_hop	Specifies whether or not the next hop value is included in RIP version 2 advertisements. *Available with active RIP only.*
rip_recv_version	Controls the version(s) of RIP updates accepted.
rip_send_version	Controls the version(s) of RIP updates advertised. *Available with active RIP only.*
rip_sub_accept	Controls whether or not subnet routes are accepted in updates.
rip_sub_advertise	Controls whether or not RIP advertises subnets. *Available with active RIP only.*

To display the values of RAC parameters, use the **na** or **admin** command **show annex all**. You can also use SNMP.

The **admin** command **show annex** works the same way as the **na** command **show annex**. This is true for all the **na** and **admin** commands in this subsection, except for **boot**, an **na** command you cannot issue from **admin.** When using **admin**, return to the CLI superuser command level to invoke **boot**.

To set a RAC parameter, use the **na** or **admin** command **set annex**.

To define several RACs for subsequent **set** or **show annex** commands to act on as a group, use the **na** command **annex**.

Setting Parameters for Routing

In general, information from ACP security (**acp_dialup** and **acp_userinfo**) or DHCP take priority over other forms of information. A matching dial-out record, however, takes priority over any matching SPB or other parameter source. SPBs in turn, take priority over EEROM parameters. EEROM parameters are thus used only if no other mechanism specifies the value to be used.

Using SPBs to Set Parameters

SPBs are matched based on information available at call set-up time, which may include dialed number, calling number, bearer type (ISDN only; fixed to “voice” on CAS), WAN interface, DS0 index, and detected protocol (sync PPP, TA, or modem). This information does not include verified user identity, so if you want to set parameters based on user name, tdo not use this mechanism to configure routing. The settable routing parameters are in Table 11-3.

There is no way to include filters or arbitrary routes in an SPB. Setting the **subnet_mask** will cause a subnet route to be advertised (assuming RIP is being used) based on the user's remote address, but no other routes are possible from an SPB.

An example of such usage is:

```
begin_session subnet_users
calling_no 1231000
call_action modem
set mode ppp
set address origin acp
set subnet_mask 255.255.255.0
end_session

begin_session proxy_arp_users
called_no 1231000
call_action modem
set mode ppp
set address_origin local
set subnet_mask 0.0.0.0
end session
```

Activating RAC Parameter Settings

Reboot the RAC to activate RAC parameter settings.

To display the values of interface parameters, use the **na** or **admin** command **show interface**.

To set an interface parameter, use the **na** or **admin** command **set interface**.

To define several interfaces for subsequent **show interface**, use the **na** or **admin** command **interface**.

Activating Interface Parameter Settings

To activate interface settings, issue the **na** or **admin reset** command - unless you set a parameter for the Ethernet interface (**en0**). In the Ethernet case, reboot the RAC.

You can also use SNMP to set and display RAC and interface parameters. The commands vary with SNMP applications. For information on the RAC SNMP implementation, see T*he Remote Access Concentrator SNMP MIB Reference*.

Enabling Passive RIP Alone

The RAC parameter **routed** must be set to **Y** (the default) to enable any type of routing other than static routes. Use **na**, **admin** or SNMP to make sure the setting has not been changed to **N**. If it has been changed, reset it to **Y**, and reboot the RAC (the following example uses **admin**):

```
annex: su
Password:
annex# admin
Annex administration Remote Annex Rx.x, 72 ports
admin: show annex routed

     routed: N

admin: set annex routed Y

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take
     effect

admin: q
annex# boot
```

At this point, both active and passive RIP are enabled for all interfaces. To use only passive RIP, change the value of the **rip_advertise parameter** from **all** to **none**:

```
admin: interface all
admin: set interface rip_advertise none
     You may need to reset the appropriate port, Annex subsystem or
     reboot the Annex for changes to take effect.
admin: reset interface
```

Configuring Passive RIP

The following subsections describe the default passive RIP configuration and how to change it.

Read this section even if you are using active RIP, since active RIP performs passive RIP as well. The configuration parameters and commands discussed in this section are a subset of those available for active RIP, and descriptions of them are not repeated in the active RIP section that appears later in this chapter.

Defining Routes

Typically, your first configuration task is to define a default route and/or static routes that you believe the RAC will not learn. The RAC provides two ways to do this:

- By entering routes in the configuration file. This method allows you to specify routes that remain defined across RAC boots.
- By issuing the CLI superuser **route** command. This lets you define a static route during a RAC session, thereby avoiding the need to change the configuration file and reboot. However, the route only remains defined for the duration of the session. If the RAC is rebooted, the route disappears.

Entering Routes in the Configuration File

The sections that follow describe how to enter routes in **annex...end** and **subnet...end** blocks in the **gateway** section of the RAC configuration file. Usually, the configuration file path name is **/usr/spool/erpcd/bfs/config.annex** on the load host; you can modify this file using an editor such as *vi*.

You can define routes anywhere in the **%gateway** section of the configuration file, but routes not in an **annex...end** or **subnet...end** block are discarded and not cached if they apply to interfaces that are immediately operational at boot time. Typically, the Ethernet interface is operational immediately, but SLIP and PPP interfaces may be slower to come up.

You can also set a default route and/or static routes in the configuration file of one or more hosts on your network. On a Berkeley-style UNIX host, define these routes in the **/etc/gateways** file.

Purpose of a Default Route

The RAC uses its default route when it cannot find a route in the routing table for a particular destination. Initially, no default route is defined. If the RAC receives a default route in a RIP update, it learns and uses it. Or, you can configure a default route using the method described in this section. Once configured, this route is not replaced by any learned routes.

Default Route Entry Format

To define a default route for a RAC, create an entry of the following format in the **gateway** section of the configuration file:

```
%gateway
annex IP_addr
route add [-h] default gateway_address [metric]
end
```

IP_addr is the address of the RAC that will use the default route; *gateway_address* is the IP address the RAC will use as the default gateway; and *metric* is the cost of using this default gateway.

Sample Configuration for Default Routes

Figure 11-4 shows a configuration in which RACs and hosts on the local network must communicate with nodes on a remote network. With only passive RIP enabled on the RACs, *RAC01* cannot reach network *132.254.2.0*, and *host01* and the PC cannot reach either network *132.254.2.0* or 1*32.254.1.0*. Defining a default route to that network enables the communication.

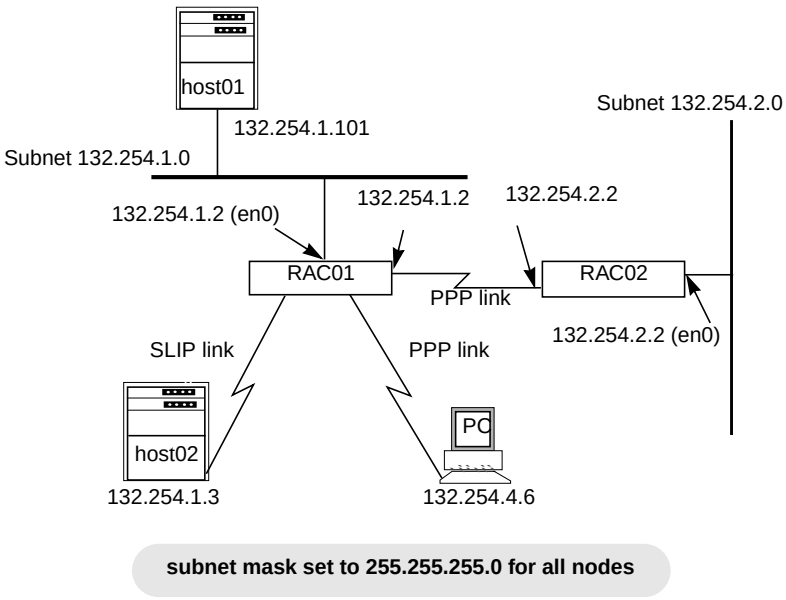

Figure 11-4. Sample Network for Defining Default Routes with Passive RIP

For convenience, on the PPP link connecting *RAC01* and *RAC02*, the local address for each RAC is the same as its **en0** address. You can change this if you want.

Keep in mind that the local address for each RAC is the remote address for the other, e.g., *132.254.1.2* is the remote address of the PPP link from the standpoint of *RAC02*.

Entering Default Routes

Use the following procedure to define default routes for the configuration in Figure 11-4. This procedure includes setting subnet masks to *255.255.255.0*, as the configuration requires.

1. **On *host01*, *host02*, and the PC, define 132.254.1.2 (*RAC01*) as the default route; give it a metric of 1, since it is on a network directly attached to *host1*. The method you use to define the route depends on the host's operating system.**

   Your PC may refer to the default route by another term, such as gateway or router address; check your PC's documentation.

2. **In the configuration file for *RAC02*, define 132.254.1.2 as the default route; again, use a metric of 1.**

   ```
   %gateway
   annex 132.254.2.2
   route add default 132.254.1.21
   end
   ```

   Reboot *RAC02*, so that it copies the default route into the routing table.

3. **On each host and RAC, and on the PC, define a subnet mask of 255.255.255.0. For a host or PC, the command you use depends on the operating system. For a RAC, use the RAC parameter** subnet_mask**. The following is an example using** admin**:**

   ```
   annex: su
   Password:
   annex# admin
   Annex administration Remote Annex Rx.x, 72 ports
   admin: set annex subnet_mask 255.255.255.0
        You may need to reset the appropriate port, Annex
        subsystem or reboot the Annex for changes to
        take effect.
   admin: q
   annex# boot
   ```

4. **Define 255.255.255.0 as a subnet mask for each end of the PPP link between *RAC01* and *RAC02* by setting the port parameter** subnet_mask**. Using** admin**, enter the following to set port *asy4* on the *annex01*:**

```
admin: set port=4 subnet_mask 255.255.255.0
     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take
     effect.
admin: reset 4
reset default asynchronous port set [y]
```

5. **Repeat the procedure for *port 8* on *RAC02.***

When the PPP link comes up, each RAC adds the remote address for its PPP port as a network route in the routing table. For example, *annex01* adds a route for the *132.254.2.0* network, using *132.254.2.2* as the next hop to that network, with a metric of 1.

If a subnet mask is unset for remote address *132.254.2.2*, the RAC assumes a mask of 0.0.0.0, which is interpreted as 255.255.255.255 (this applies to SLIP and PPP ports only). In this case, *annex01* adds to its routing table a host route for *132.254.2.2*, using *132.254.2.2* as the next hop to that host, with a metric of 1. This host route would be appropriate if *annex02* were not connected to an Ethernet.

Purpose of Static Routes

A static route is one you enter manually, as opposed to a route RIP learns from routing updates. For most network configurations, you typically want to configure at least a few static routes. For example, if a device to which your RAC will be routing does not itself advertise routes, you will want to define a static route to that device. To ensure that the RAC uses defined static routes, specify them in an **annex...end** or **subnet...end** block in the **gateway** section of the configuration file. Static routes defined elsewhere in the **gateway** section are discarded if the interface is not up immediately after the RAC boots. This is not a problem for the Ethernet interface, but SLIP and PPP interfaces can take longer to come up.

annex...end Format for Static Routes

A static route entry in an **annex...end** block has the following format:

```
%gateway
annex annex_IP_addr
route add [-h] dest_IP_addr subnet_mask  gateway_address  [metric]
end
```

In the format above:

- *annex_IP_addr* is the address of the RAC that will use the route.
- *dest_IP_addr* is the destination IP address. Do not attempt to give a Proxy ARP host address here; it will not work.
- *subnet_mask* specifies the subnet mask for the destination address. You can enter the mask in dotted decimal notation, e.g., 255.255.255.0, or you can specify the mask by appending a *bits* field to *dest_IP_addr*. Specify the *bits* field as */n*, where *n* is the number of 1 bits in the mask, from left to right. For example, appending **/24** to an address specifies 255.255.255.0 for the subnet mask.

  

  Incorrectly configuring or not setting subnet masks can cause unrecoverable corruption of the routing table. To detect potential problems, RIP generates a syslog LOG_WARN message if the RAC subnet mask or a port subnet mask is left unset.

- *gateway_address* is the IP address of the gateway the RAC will use as the next hop to the destination address.
- *metric* is the cost of using this default gateway.
- The **-h** option specifies the route is hardwired, which means that the RAC will not replace the route with any route it learns from RIP.

Table 11-4 shows the values you can supply for the *bits* field, along with the resultant subnet mask and its hexadecimal value.

Table 11-4. Values for Bits Field with Corresponding Subnet Masks

Bits	Mask	Hex Value	Bits	Mask	Hex Value
8	255.0.0	FF000000	20	255.255.240.0	FFFFF000
10	255.192.0.0	FFC00000	21	255.255.248.0	FFFFF800
11	255.224.0.0	FFE00000	22	255.255.252.0	FFFFFC00
12	255.240.0.0	FFF00000	23	255.255.254.0	FFFFFE00
13	255.248.0.0	FFF80000	24	255.255.255.0	FFFFFF00
14	255.252.0.0	FFFC0000	25	255.255.255.128	FFFFFF80
15	255.254.0.0	FFFE0000	26	255.255.255.192	FFFFFFC0
16	255.255.0.0	FFFF0000	27	255.255.255.224	FFFFFFE0
17	255.255.128.0	FFFF8000	28	255.255.255.240	FFFFFFF0
18	255.255.192.0	FFFFC000	29	255.255.255.248	FFFFFFF8
19	255.255.224.0	FFFFE000	30	255.255.255.252	FFFFFFFC

For each of the valid network classes and subnet bit counts, Table 11-5, Table 11-6, and Table 11-7 show the total number of subnets, and hosts per subnet, that are possible.

Table 11-5. Class A: Total Available Subnets and Hosts

Bits	Subnets	Hosts	Bits	Subnets	Hosts
8	1	16,777,214	20	4,094	4,094
10	2	4,194,302	21	8,190	2,046
11	6	2,097,150	22	16,382	1,022
12	14	1,048,574	23	32,766	510
13	30	524,286	24	65,534	254
14	62	262,142	25	131,070	126
15	126	131,070	26	262,142	62
16	254	65,534	27	524,286	30
17	510	32,766	28	1,048,574	14
18	1,022	16,382	29	2,097,150	6
19	2,046	8,190	30	4,194,302	2

Table 11-6. Class B: Total Available Subnets and Hosts

Bits	Subnets	Hosts	Bits	Subnets	Hosts
16	1	65,534	24	254	254
18	2	16,382	25	510	126
19	6	8,190	26	1,022	62
20	14	4,094	27	2,046	30
21	30	2,046	28	4,094	14
22	62	1,022	29	8,190	6
23	126	510	30	16,382	2

Table 11-7. Class C: Total Available Subnets and Hosts (with no supernetting)

Bits	Subnets	Hosts	Bits	Subnets	Hosts
24	1	254	28	14	14
26	2	62	29	30	6
27	6	30	30	62	2

Sample Configuration for Static Routes

The configuration shown in Figure 11-5 requires static (and default) routes. In this example, note the following:

- For convenience, on the PPP link connecting *annex01* and annex*02*, the local address for each RAC is the same as its **en0** address. You can change this if you want.
- The local address for each RAC is the remote address for the other; e.g., *132.254.1.2* is the remote address of the PPP link from the standpoint of *annex02*.
- It is assumed that the SLIP/PPP interfaces on *annex03* are the following networks: 148.254.0.0, 149.12.0.0, 150.14.0, 151.13.0, and 152.254.0.0.

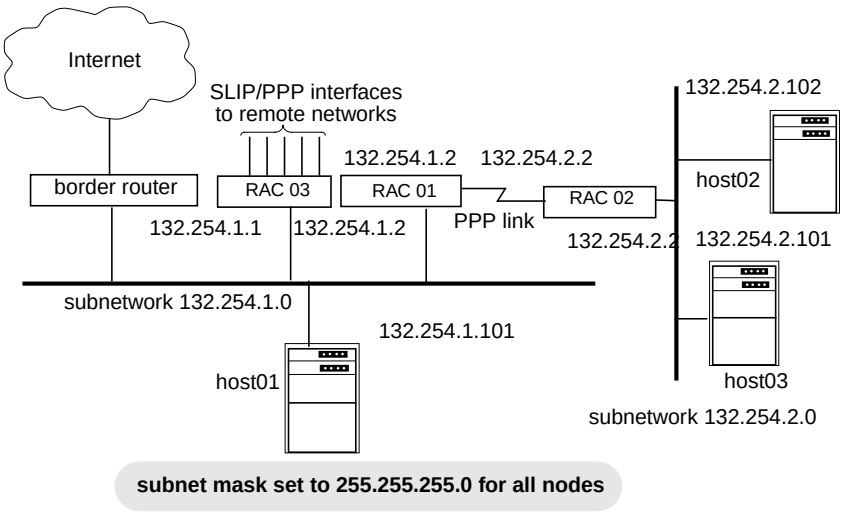

Figure 11-5. Sample Network for Static and Default Routes (Passive RIP)

Entering Static Routes

Do as follows to define routes for the network shown in Figure 11-5 (a sample **%gateway** section appears after this procedure):

1. **On *host02* and *host03*, define *RAC02* (132.254.2.2) as the default route.**
2. **On RAC*02*, define RAC*01* (132.254.1.2) as the default route.**
3. **On RAC01, define *border router* as the default route, and configure five static routes defining *RAC03* (132.254.1.1) as the next hop for each of the remote networks attached to the RAC*03*'s SLIP and PPP interfaces**.
4. **On *host01*, define *border router* as the default route, configure five static routes defining *RAC03* (132.254.1.1*)* as the next hop for each of the remote networks attached to the RAC*03*'s SLIP and PPP interfaces, and configure *RAC01* as the next hop for network 132.254.2.0.**

5. **On RAC*03*, define *border router* as the default route, and define a static route to subnetwork 132.254.2.0 via *RAC01* (132.254.1.2).**
6. **On *border router*, configure five static routes defining *RAC03* (132.254.1.1) as the next hop for each of the remote networks attached to the *annex03*'s SLIP and PPP interfaces. Also, define a static route to network 132.254.2.0 via *RAC01* (132.254.1.2).**

To configure static and default routes for the hosts in Figure 11-5, use the methods appropriate for the hosts' operating systems. To configure routes for the RACs in Figure 11-5, enter the following **gateway** definition in the configuration file. Specify static and default routes as **hardwired** if you do not want RIP to replace them with routes it learns over the network.

```
%gateway

annex 132.254.2.2
     #for annex02, use 132.254.1.2 (annex01) as default route
     route add  -h  default  132.254.1.2  1
end
annex 132.254.1.2
     #for annex01, use 132.254.1.1 (annex03) as gateway to 5
     #SLIP/PPP nets
     route add  -h  148.254.0.0/16  132.254.1.1  1
     route add  -h  149.12.0.0/16   132.254.1.1  1
     route add  -h  150.14.0.0/16  132.254.1.1  1
     route add  -h  151.13.0/16   132.254.1.1  1
     route add  -h  152.254.0.0/16  132.254.1.1  1
     #for all other destinations, use 132.254.1.8 (border
     #router) as default
     route add  -h  default  132.254.1.8  1
end
annex 132.254.1.1
     #for annex03, use 132.254.1.2 (annex01) as gateway to
     #132.254.2.0/24 subnet
     route  add  -h 132.254.2.0  132.254.1.2  1
     #for all other destinations, use 132.254.1.8 (border
     #router) as the hardwired default
     route  add  -h  default  132.254.1.8  1
end
```

Routes are automatically added to RAC01 for the 132.254.2.0 network, based on the PPP **remote_address** and **subnet_mask** parameters, so no static route needs to be defined for that subnet.

Entering Subnet Routes

You can create **subnet...end** in the gateway section of **config.annex**. This allows you to define a static or default route for all RACs on a given network or subnet. The syntax is as follows:

subnet *ip_addr*
...
end

The lines enclosed by the **subnet...end** block are to be used only by RACs on the subnet or network with the IP address *ip_addr*. Any routes enclosed by the **subnet...end** block are cached. An **else** keyword can also be used (alone on a line) to list configuration information for all subnets/networks except the one identified on the **subnet** line. You cannot nest **subnet...end** blocks.

The following are sample **subnet...end** blocks:

```
subnet 132.245.33.0
route add default 132.245.33.22 1
end
subnet 132.245.66.0
route add -h default 132.245.66.22.1
end
```

Entering Routes Using the route Command

The CLI superuser **route** command lets you define static routes during a RAC session, thereby avoiding the need to change the configuration file and reboot. The catch is that the route remains defined only for the duration of the session. If the RAC is rebooted, the route disappears.

You can also use **route** to define a default route, but this is not recommended (see *Risks When Adding or Deleting Default Routes on page 11-42*).

The arguments for **route** are shown in Table 11-8. The syntax is:

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

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

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

route [**-fF**] **expire** [**-h**] {**default** | *dest*}

route [**-fF**] **replace** [**-h**] {**default** | *dest*} *gateway* [*metric*]

Specify routes as hardwired (by using the **-h** argument) if you do not want RIP to replace them with routes it learns over the network.

Changes made using the **route** command take effect immediately; you do not need to reboot.

Table 11-8. Arguments for the Superuser CLI route Command

Argument	Description
-f	Flushes (deletes) all static and learned routes from the routing table and all static routes from the route cache.
-F	Flushes all hardwired routes from the route cache and routing table. It does not flush interface routes.
-fF	Flushes all non-interface routes from the routing table and route cache.
add	Adds a static route to the route cache. It also adds the route to the routing table if the *gateway* argument (see below) specifies an address that is directly reachable on an active interface.
delete	Deletes a route from the route cache and the routing table. No notification is sent to other routers; they must age the route themselves, which takes 3 minutes. Cannot be used to delete interface routes.
expire	Marks a route as expired by setting its usage to +0 and its metric to 16 (infinity). The lifetime of the route is set to two minutes, during which time RIP continues to advertise the route. This allows other routers to learn that the route is invalid. Cannot be used for interface routes.
replace	Replaces the gateway in the route for *dest* with the new *gateway* you specify. Cannot be used to replace interface routes.
-h	Specifies a hardwired static or default route that RIP cannot replace.

(continued on next page)

Table 11-8 Arguments for the Superuser CLI route Command (continued)

Argument	Description
default	Specifies the default route. See *Risks When Adding or Deleting Default Routes on page 11-42*.
dest	Specifies the destination address of the route.
mask	Specifies the subnet mask for the destination address. You can enter the mask in dotted decimal notation, e.g., 255.255.255.0, or you can specify the mask by appending /*n* to the destination address, where *n* is the number of 1 bits in the mask, from left to right. For example, appending /24 specifies 255.255.255.0 as the subnet mask. Table 11-4 on page -33 shows all possible values for the subnet bit count, with the resultant subnet masks.
gateway	Specifies the IP address of the gateway (router) that is the next hop for the route.
metric	Specifies the number of hops to the destination. Values range from **1** through **15**; the default is **1**.

route Command Examples

Both of the following examples add a hardwired route for the destination address 131.108.3. 0 using a subnet mask of 255.255.255.0. This subnet mask indicates that the first two octets are the network address, the third octet (which normally is part of the host portion of this Class B address) is the subnet, and the fourth octet is the host (for more information on subnets, see *Subnetting Using Subnet Masks on page 11-15*). The next hop is 131.254.33.2 and the metric (hop count) for the route is 2. The examples are:

```
annex# route add -h 131.108.3.0 255.255.255.0 131.254.33.2 2
```

and

```
annex# route add -h 131.108.3.0/24 131.254.33.2 2
```

Both of the preceding examples configure the RAC to use the gateway at 131.254.33.2 as the next hop for any host destination whose address is within the range 131.108.3.1 through 131.108.3.254.

Risks When Adding or Deleting Default Routes

Although permissible, using the **route** command to add or delete a default route can have unpredictable results. For example, if you add a non-hardwired default route to the RAC configuration file and then define another one using the superuser CLI **route** command, you have no way of knowing which route the RAC will use. Even if a default route is not in the configuration file, confusion can arise if the RAC learns of a default route through a RIP update. Deleting a default route via the CLI **route** command can cause similar problems. If the RAC knows about two default routes, you cannot know which one you are deleting.

The only time you can safely use the **route** command to add a default route is when one is not defined in the configuration file and the RAC is not receiving RIP updates. In this case, using **route** to define a default route is more convenient than adding the route to the configuration file, since the latter requires rebooting the RAC. Later, you can delete the default route using the **route** command.

Accepting RIP 1 and/or RIP 2 Packets

The RAC's default RIP configuration accepts both version 1 and version 2 packets, making no distinction between version 2 packets that are broadcast and those that are multicast. In most cases, you should not change this default configuration because there may be version 1 RIPs running on your network nodes, and accepting version 2 packets as well does no harm. However, if all nodes on your network run only version 2, you may want to set some or all of the RAC interfaces to accept only version 2 packets. This takes advantage of the fact that RIP 2:

- Supports subnetting better than RIP 1; subnet masks are contained in a RIP 2 route's header, while RIP 1 carries no subnet information.
- When used with the **rip_auth** parameter set to a password, provides full authentication of incoming RIP 2 updates and requests (see *rip_auth on page 11-56*).

To configure some or all of the RAC interfaces to accept only version 2 packets, use **na** or **admin** to set the **rip_recv_version** parameter to **2** for those interfaces. The following example configures all RAC interfaces to accept only version 2 packets:

```
annex: su
Password:
annex# admin
Annex administration Remote Annex Rx.x, 72 ports
admin: set interface=all rip_recv_version 2

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take effect.

admin: q
annex#: boot
```

The **boot** command is required in the preceding example because all interfaces, including *en0*, are being set. If *en0* were not among the interfaces, you could use the **admin** command **reset interface** instead of the superuser **boot** command.

Do not set **rip_recv_version** to **2** unless you are sure all nodes on your network or internet are advertising only RIP 2 updates.

Authenticating Incoming RIP 2 Updates and Requests

To authenticate incoming RIP 2 messages, set the **rip_auth** parameter to a password containing 1 to 16 characters. The following example sets the password to *ps44D6*.

```
admin: set annex rip_auth ps44D6
admin: set interface=all rip_recv_version 2

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take effect.

admin: quit
annex# boot
```

RIP 1 packets do not contain a password field, so they cannot be authenticated. Therefore, if you leave the **rip_recv_version** parameter set to the default **both** (which accepts both RIP 1 and RIP 2 packets), setting **rip_auth** provides only partial security.

Once you have set **rip_auth** to a password, an incoming RIP 2 message is authenticated if both of the following conditions are met:

- The message includes the required authentication information. This means that the *Address Family Identifier* field of the first entry in the message must have a value of 0xFFFF, the *Authentication Type* field of the first entry must have a value of **2** (a value of **1** indicates no authentication), and the *Authentication* field must contain a 16-byte unencrypted password.
- The password in the message matches the value of the **rip_auth** parameter.

The RAC accepts all RIP 2 messages it authenticates, but does not necessarily discard all unauthenticated messages it receives. Table 11-9 shows how the RAC accepts or discards a RIP message depending on whether or not the RAC and/or the message are configured for authentication and whether or not the password in the message matches the **rip_auth** password.

Table 11-9. RAC RIP Version 2 Authentication

	Message Includes Authentication	Message does not Include Authentication
RAC Configured for Authentication	Accepts message if passwords match; otherwise, discards message.	Discards message.
RAC not Configured for Authentication	Discards message.	Accepts message.

Although RIP-2 authentication cannot protect your system against a user who has the physical means or access to diagnostic tools to tap the network, it nevertheless prevents SLIP or PPP users from injecting routes into the system.

Active RIP Prerequisites

The following are prerequisites for using active RIP:

- The **rip_advertise** parameter must be set to **all** for all interfaces and **routed** must be set to **Y**, which are the defaults. To make sure these values are set, issue the following **admin** commands:

```
admin : show interface rip_advertise

interface en0:
     rip_advertise: all

interface asy1:
     rip_advertise: all

interface asy2:
     rip_advertise: all

interface asy3:
     rip_advertise: all

interface asy4:
     rip_advertise: all

     .
     .
     .

admin : show annex routed

     routed: Y
```

- At least two RAC interfaces must be operational.
- Each SLIP, PPP, and Ethernet interface over which routing is to occur must have an IP address.
- All SLIP, PPP, and Ethernet interfaces must have appropriate subnet masks. For more information, see *Subnetting Using Subnet Masks on page 11-15*.

Configuring Active RIP

The following section assumes you have read the earlier sections on passive RIP. Where appropriate, this section refers back to the passive RIP sections to avoid repeating material described there.

Defining Routes

With active RIP running, you do not need to define the default and static routes described for the configurations shown in Figure 11-4 and Figure 11-5. The RACs will learn about the routes to each other and to other networks through RIP updates they exchange, provided that, for subnetted networks, the **rip_sub_advertise** parameter is set to **Y**, which is the default *(*see *Advertising Subnet Routes on page 11-49*) and that you have configured subnet masks correctly.

Although the routes required for passive RIP need not be defined if you are running active RIP as well, you may want to define a default route and one or more static routes for other purposes. For example, a default router can act as a chokepoint through which all traffic to and from a network must pass. And, you can use static routes to reach routers that are not running active RIP.

To define default and static routes that remain across RAC boots, enter them in the **config.annex** file (see *Entering Routes in the Configuration File on page 11-28*). You can define routes anywhere in the configuration file, but routes not defined in an **annex...end** or **subnet...end** block are discarded and not cached if their interfaces are not operational at boot time. Typically, the Ethernet interface is operational immediately, but SLIP and PPP interfaces may take longer to come up.

To define routes for the current session only, see *Entering Routes Using the route Command on page 11-39*.

Advertising RIP 1 and/or RIP 2 Updates

By default, active RIP sends RIP version 2 updates to the IP broadcast address, so that both RIP 1 and RIP 2 systems can receive them. This assumes that **rip_send_version** is set to **compatibility**, which is the default (see *Accepting RIP 1 and/or RIP 2 Packets on page 11-42* and *rip_recv_version on page 11-58*). It also assumes the routers on your network accept both RIP 1 and RIP 2 updates. Although discarding RIP 2 updates violates RIP 1 RFC (1058), some RIP implementations written before the RFC still do so. If you have both RIP 1 and RIP 2 nodes on your network, make sure that there are no RIP 1 implementations that discard RIP 2 packets. If there are, use **na** or **admin** to set the **rip_send_version** parameter to **1** (see *rip_send_version on page 11-59*), as shown in the following example:

```
annex: su
password:
annex# admin
Annex administration Remote Annex Rx.x, 72 ports
admin: set interface=all rip_send_version 1

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take effect.

admin: quit
annex# boot
```

The **boot** command is required in the preceding example because you are setting *en0*. If *en0* were not among the interfaces, you could substitute the **admin** command **reset interface** for the **boot** command.

Advertising Subnet Routes

By default, active RIP advertises subnet routes over RAC interfaces. If your network is subnetted, you usually do not want to change this default, unless you are trying to hide your subnet routes from all nodes on a particular interface. In either case, be sure to set the proper subnet masks for the subnet addresses on the network. Also, be aware that RIP 1 packets do not have fields containing subnet masks. Thus, both ends of a RIP 1 communications link must agree beforehand on the part of an address that contains the subnet specification.

Figure 11-6 shows an example in which *host03* learns routes to the PC at 132.254.93.2 and the host at 132.254.55.222, even though they are on different subnets from that of the RAC. This is because (1) the RAC has the **rip_sub_advertise** parameter set to **Y**, and (2) all nodes have a subnet mask set (correctly) to 255.255.255.0.

For convenience, the local addresses of the links to the PC and *host14* in Figure 11-6 are configured to be the same as the RAC's **en0** interface - *132.254.66.134*.

If *host03* supports host routes, a subnet mask of all 1s (255.255.255.255) would work for the two serial ports on annex01.

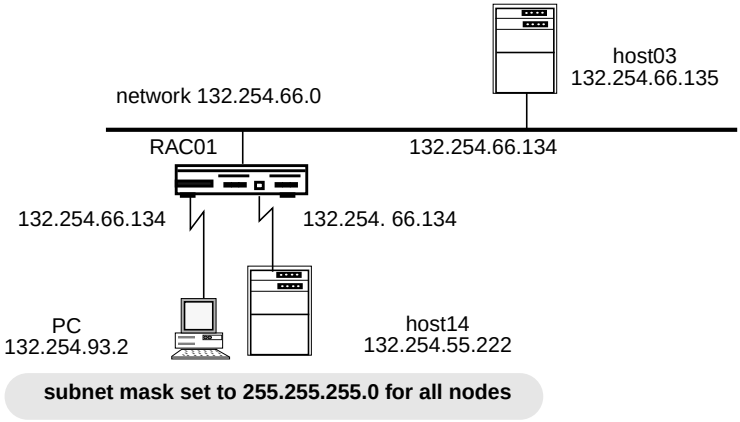

Figure 11-6. Advertising Subnet Routes

Subnets are explained in *Subnetting Using Subnet Masks on page 11-15*. The **rip_sub_advertise** parameter is described in more detail in *rip_sub_advertise on page 11-59*.

Using Split Horizon and Poison Reverse

RIP uses a distance-vector algorithm that removes routes from a routing table by aging them and by determining that their destinations are unreachable (more than 15 hops away). To resolve two problems that can occur with distance-vector algorithms, the default RAC implementation of active RIP uses the *split horizon* and *poison reverse* mechanisms.

The split horizon mechanism prevents a route from being advertised as reachable over the interface on which it was learned. Without this mechanism, two routers could advertise the same route back and forth, each adding to the hop count until it reached 16 and the route's destination was determined unreachable. This process could be lengthy and the information conveyed by the hop count, although eventually correct, might no longer be useful.

Poison reverse optimizes the split horizon mechanism. It re-advertises a route over the interface on which it was learned, but does so with a hop count of 16 (unreachable). In general, split horizon with poison reverse is more effective than split horizon alone. If two routers have routes pointing at each other, advertising them as unreachable breaks the loop immediately, while merely not advertising them requires that they time out before being removed from a routing table.

To disable split horizon and/or poison reverse for one or more interfaces, see *rip_horizon on page 11-57*. It is recommended that you set this parameter to either **poison** (the default) or **split**, but not to **off**.

Authenticating Outgoing Updates and Requests

If a RAC is configured for authentication, as described in *Authenticating Incoming RIP 2 Updates and Requests on page 11-44*, the RIP 2 messages it sends to other routers carry authentication information. When the RAC builds the RIP 2 message (update or request), it sets the *Address Family Identifier* and *Authentication Type* fields in the message's first entry to 0xFFFF and 2, respectively. It also sets the *Authentication* field in the first entry to the value of **rip_auth** (the authentication password configured for the RAC).

RIP 1 updates and requests do not carry authentication information.

Advertising the Default Route

Initially, no default route is defined for the RAC so it cannot advertise one. If RIP receives a default route in an update, the RAC learns, uses, and advertises that route. However, a user-configured default route takes precedence over a learned one, and the RAC does not advertise a user-configured default route.

If you want the RAC to advertise *itself* as a default router over one or more interfaces, set the **rip_default_route** parameter to an integer between 1 and 15 for the specific interfaces. This integer specifies the metric the RAC advertises for itself as a route. Routers on the directly connected network will use the RAC when they have no other route defined for a particular destination.

The RAC need not have a default route itself in order to advertise itself as a default router.

Once the RAC advertises itself as a default route, active RIP no longer advertises a learned default route and instead advertises the route requested by the **rip_default_route** parameter. However, for IP forwarding, the RAC still uses the default route (if any) specified via the **route** command or defined in the configuration file.

Creating or deleting a default route via the **route** command can have unpredictable results and is discouraged except in special circumstances (see *Risks When Adding or Deleting Default Routes on page 11-42*).

Advertising to a Subset of Routers

By default, the RAC sends one update to the broadcast address every 30 seconds so that all routers on the network can receive it. You can restrict updates to only certain routers by specifying them with the RAC parameter **rip_routers**. The routers must be on directly attached networks.

You can specify up to eight routers using the **rip_routers** parameter. The following example specifies two routers.

```
annex: su
password:
annex# admin
MICRO-XL-UX I13.2.21, 16 async, 0 modem ports
admin: set annex rip_routers 132.254.54.2,132.254.54.33

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take effect.

admin: quit
annex# boot
```

Sending updates to a set of routers can create a significant amount of overhead. For example, if you specify eight routers, eight updates (rather than one) are sent every thirty seconds (one to each router).

RIP Configuration Parameters - Reference

Descriptions of the RIP-specific configuration parameters follow.

Do not change the settings unless you have read the earlier sections of this chapter.

rip_accept

The interface parameter **rip_accept** controls the networks for which routes are accepted from RIP updates and requests. The syntax is:

rip_accept {*access_spec* | **none** | **all**}

Specify *access_spec* as:

{**include** | **exclude**} *network_list*

Use **include** to accept RIP updates only for routes whose destination addresses are on the networks in *network_list*. Use **exclude** to accept all RIP updates except those whose destination addresses are on networks in *network_list*. Do not use **include** and **exclude** within the same command.

For *network_list*, specify up to eight IP network addresses, separated by commas (no spaces are allowed anywhere in the list).

When interpreting an address in *network_list*, RIP uses the intrinsic subnet mask derived from the address class, regardless of the port or the RAC **subnet_mask** parameter setting.

Instead of *access_spec*, you can enter **none** or **all**; **none** specifies that no RIP messages (updates or requests) are accepted over the interface, while **all** specifies that RIP updates are accepted for all networks. The default is **all**.

The **na** commands in the following example turn off the acceptance of any RIP messages (updates or requests) over interface *asy4:*

```
command: interface asy4
command: set interface rip_accept none

     Changes will take effect at next annex boot or
     interface reset

command: reset interface
```

The **admin** commands in the following example specify that, on interface *asy3*, RIP does not accept routes to destinations on networks 132.254.0.0 and 192.200.0.0. Routes to all other networks are accepted on *asy3*.

```
admin: set interface=asy3 rip_accept exclude\
132.254.0.0,192.200.0.0

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take effect.

admin: reset interface asy3
```

rip_advertise

The interface parameter **rip_advertise** controls the networks for which routes are advertised. The syntax is:

rip_advertise {*access_spec* | **none** | **all**}

Specify *access_spec* as:

{**include** | **exclude**} *network_list*

Use the keyword **include** to advertise routes for all routes whose destinations are on the networks in *network_list*. Use **exclude** to advertise routes for all whose destination are on networks other than those in *network_list*. Do not use **include** and **exclude** within the same command.

For *network_list*, specify up to eight IP network addresses, separated by commas (no spaces are allowed anywhere in the list).

When interpreting an address in *network_list*, RIP uses the intrinsic subnet mask derived from the address class, regardless of the port or the RAC **subnet_mask** parameter setting.

Instead of *access_spec*, you can enter **none** or **all***;* **none** specifies that no RIP updates are advertised over the interface, while **all** specifies that RIP updates are advertised for all network addresses. The default is **all**.

The following **admin** example causes RIP to advertise routes for all destination addresses except those on networks 10.0.0.0 and 190.9.200.0. The example applies to routes advertised over interfaces *asy3* through *asy5*:

```
annex: su
password:
annex# admin
Annex administration Remote Annex Rx.x, 72 ports
admin: interface asy3-5
admin: set interface rip_advertise exclude\
10.0.0.0,190.9.0.0

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take effect.

admin: reset interface
```

If the above example contained the keyword **include** instead of **exclude**, RIP would advertise *only* those routes whose destination networks were *10.0.0.0* and *190.9.0.0* and no others.

The following **na** example specifies that no RIP messages (updates or requests) are to be sent over interface *asy2*:

```
command: set interface=asy2 rip_advertise none rip_accept\
none

     Changes will take effect at next annex boot or
     interface reset

command: reset interface asy2
```

rip_auth

The RAC parameter **rip_auth** enables and disables RIP 2 authentication. The syntax is:

rip_auth {*password* | ""}

Specifying *password* enables authentication; specifying the null string ("") turns off authentication. The password can be a string of up to 16 characters. The **show annex** command displays this parameter's value as **"<set>"** or **"<unset>"**. The value **"<unset>"** means authentication is turned off; all unauthenticated RIP packets are accepted. The default is the null string (no authentication).

Reboot the RAC to put this parameter setting into effect.

For details on the RAC implementation of authentication, see *Authenticating Incoming RIP 2 Updates and Requests on page 11-44* and *Authenticating Outgoing Updates and Requests on page 11-51*.

rip_default_route

The interface parameter **rip_default_route** controls whether or not the RAC advertises itself as the default route. The argument *metric* is any value from **0** through **15**, or **off**. A value of **1** through **15** indicates the hop count to be advertised for the RAC's route. A value of **0** or **off** turns off advertisement of the RAC as a default router. The default is **off**. For more information, see *Advertising the Default Route on page 11-51*.

rip_horizon

The interface parameter **rip_horizon** controls the split horizon mechanism: **off** disables split horizon, **split** enables split horizon without poison reverse, and **poison** enables split horizon with poison reverse. The default is **poison**. For more information, see *Using Split Horizon and Poison Reverse on page 11-50*.

rip_next_hop

The interface parameter **rip_next_hop** controls whether or not a route's next hop is advertised in RIP Version 2 updates. Valid options are **never**, **needed**, or **always**. The default is **needed**. This parameter has no effect unless the RAC is running more than one IP routing protocol.

rip_recv_version

The interface parameter **rip_recv_version** controls which RIP version the RAC accepts: **1** indicates that version 1 or higher packets are accepted but the non-RIP-1-specific data fields are ignored (i.e., version 2 packets are interpreted as version 1 packets); **2** indicates only version 2 or higher packets are accepted; and, **both** indicates both versions are accepted. The default is **both**. For more information, see *Advertising RIP 1 and/or RIP 2 Updates on page 11-48*.

rip_routers

The RAC parameter **rip_routers** directs periodic RIP updates to a list of routers rather than broadcasting them. The syntax is:

rip_routers *router_list*

The argument *router_list* specifies a list of the IP addresses of up to eight directly reachable routers. Separate the addresses with a comma (and no spaces). The RAC ignores any address not on an attached subnet. Specifying **all** causes RIP updates to be broadcast. The default is **all**.

The following **na** command causes RIP to send updates to the routers at the following addresses: 132.254.33.4, 132.254.1.30, and 132.254.2.2. You must then reboot the RAC.

```
command: set annex rip_routers
132.254.33.4,132.254.1.30,132.254.2.2

     Changes will take effect at next annex boot or
     interface reset

command: boot
```

rip_send_version

The interface parameter **rip_send_version** controls which RIP versions the RAC sends: **1** indicates version 1 packets are sent to the IP broadcast address; **2** indicates version 2 packets are sent to the multicast address; and **compatibility** indicates version 2 packets are sent to the IP broadcast address. The default is **compatibility**. For more information, see *Advertising RIP 1 and/or RIP 2 Updates on page 11-48*.

rip_sub_accept

The interface parameter **rip_sub_accept** controls whether or not subnet routes are accepted. A **Y** accepts subnet routes; an **N** rejects them. The default is **Y**. For more information, see *Advertising Subnet Routes on page 11-49*.

rip_sub_advertise

The interface parameter **rip_sub_advertise** parameter controls whether or not subnet routes are advertised. A **Y** enables subnet advertising; an **N** disables it. The default is **Y**. For more information, see *Advertising Subnet Routes on page 11-49*.

The following **na** command specifies that RIP does not advertise subnet routes over the RAC's Ethernet interface. Note that the RAC must be rebooted to effect the change to **en0**.

```
admin: set interface=en0 rip_sub_advertise n
admin: quit

     You may need to reset the appropriate port, Annex
     subsystem or reboot the Annex for changes to take effect.

annex#: boot
```

routed

The RAC **routed** parameter enables and disables RIP. The syntax is:

routed {**Y** | **N**}

Y enables RIP, **N** disables it. Setting **routed** to **Y** activates both passive and active RIP. The default is **Y**.

Reboot the RAC to put this parameter setting into effect.

Displaying Routing Information

This section describes three diagnostic tools that let you:

- Display RIP interface statistics.
- Display the contents of the RAC routing table.
- Trace the path of a packet as it is routed to its destination.

Displaying RIP Statistics

The **netstat -g** command displays RIP statistics. Table 11-10 describes the field definitions for the command display.

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

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

Sources:

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

Routing Changes: 1  Queries received: 0

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

Table 11-10. Field Definitions for the netstat -g Command

Field	Definition
Intf	Displays the interface.
Bad Pkts	Displays the number of packets the interface dropped due to invalid format or data.
Bad Rtes	Displays the number of routes the interface dropped due to invalid format or data.
Trigg.	Displays the number of triggered updates transmitted over the interface. The RAC sends triggered updates whenever it changes the hop count of a route. It transmits them immediately, even if it is not yet time for one of the regular update messages to be transmitted.

(continued on next page)

Table 11-10 Field Definitions for the netstat -g Command (continued)

Field	Definition
Rec'd	Displays the number of packets (with or without errors) received over the interface.
Sent	Displays the number of output packets the RAC tried to send over the interface. This number includes packets that were dropped because the RAC ran out of buffers or the link's output queue was full.
Disc'd	Displays the number of input packets discarded due to protocol errors or restrictions set by configuration parameters (e.g., rip_accept).
Update	Displays the number of lines in the routing table that were modified due to packets received on that interface.
Queries	Displays the number of routing-table queries received on the interface.

Displaying the RAC Routing Table

To display the contents of the RAC routing table, use the CLI **netstat -r** command, as shown in the example that follows. Note that the command displays AppleTalk routes as well as IP routes. (You can display RIP routes exclusively by issuing the command **netstat -ri**.) The IP routes are displayed together, sorted by IP destination address from the lowest to the highest number. The IP fields that **netstat -r** displays are explained in Table 11-11.

```
annex: netstat -r
Routing Tables
Destination       NextHop         Flags    Usage    UseCount   Mtr  Interface
32004 - 32005     32005.77        UHF      2        10         0    en0
Apple default     32004.22        UGF      0        0          0    en0
IP default        132.245.66.22   USH      fixed    2          1    en0
1.1.1.1           132.245.66.232  UR       -26760   0          2    en0
1.1.1.2           132.245.66.232  UR       -26760   0          2    en0
1.1.1.3           132.245.66.232  UR       -26760   0          2    en0
1.1.1.4           132.245.66.63   UR       -36      0          2    en0
1.1.1.5           132.245.66.63   UR       -36      0          2    en0
4.0.0.2           *               UI       fixed    0          1    asy2 (ppp)
127.0.0.0/8       *               UI       fixed    0          1    lo0
128.128.0.0/16    132.245.66.66   UR       -26760   0          3    en0
128.128.129.204   132.245.66.122  UR       -18      0          2    en0
131.110.0.8/29    132.245.66.22   UR       -26760   0          2    en0
132.245.1.0/24    132.245.66.22   UR       -26760   0          2    en0
132.245.9.0/24    132.245.66.22   UR       -26687   73         2    en0
132.245.10.0/24   132.245.66.22   UR       -26760   0          2    en0
```

Table 11-11. IP Fields in netstat -r Display

Field	Explanation
Destination	The IP address of the route's destination, followed by a slash (/), followed by the number of 1 bits, counting from left to right, in the Destination's subnet mask. For example, the */24* following the IP address *132.254.1.0* indicates a subnet mask of 24 bits (eight octets), or 255.255.255.0. (For more information, see *Entering Routes in the Configuration File on page 11-28*.) If *IP Default* appears in the Destination field, the entry specifies the route the RAC uses if it can find no other route for a destination. If a name appears in the Destination field, the entry is for a host route; name servers do not have names for network routes. (However, the RAC does not always know a host's name.)
NextHop	The next router to which packets with the given Destination are sent. If the Destination is a local interface, this field displays an asterisk (*); interface routes have no next hop.

(continued on next page)

Table 11-11 IP Fields in netstat -r Display (continued)

Field	Explanation
Flags	The following three flags:
First flag (Status) *U* *Q* *D*	 The route is valid (up) and in use. The route is valid but the interface is quiescent; i.e., the interface is not up yet or was brought down by expiration of the timer set by the **net_inactivity** port parameter. The route is invalid (down) and has a metric of 16 (RIP infinity). It will stay in the routing table for two more minutes so that other routers can learn that it is invalid.
Second flag (Source) *C* *I* *R* *S*	 The route was learned via an ICMP redirect. This can occur only when IP routing is disabled (by setting the **routed** parameter to **N**). The route is an interface route. The route was learned via RIP. The route is a static route, learned from a route you defined in the gateway section of the **config.annex** file or a route you entered via the CLI superuser **route** command.
Third flag *H*	 The route is a hardwired static route.

(continued on next page)

Table 11-11 IP Fields in netstat -r Display (continued)

Field	Explanation
Usage	A positive or negative integer indicating a route's usage. When RIP adds a route to the routing table, it sets its usage value to 0. Every time the route is used, RIP adds 1 to this value. And every thirty seconds, RIP subtracts one from the value. When the routing table reaches its maximum size of 256 entries, RIP removes the route with the lowest usage value. If there is a tie, the first route listed is deleted. The range of values is -9999999, for a route that has not been used in 9.5 years, to +9999999, for a very frequently used route. Interface, hardwired, and *extremely* frequently used routes contain the word *fixed* in this field instead of a number.
UseCount	A positive integer indicating the number of times the route has been used to transmit a packet. If you subtract the value in this field from the value of *Usage*, you can determine how long a route has been in the routing table.
Mtr	The metric associated with the route.
Interface	The interface over which the RAC can reach the next hop.

Displaying the route cache

The **netstat -C** command displays the contents of the cache route, including both static routes added from the **gateways** section of the configuration file and routes added by the **route** command. Table 11-12 describes the flags for the command display.

Table 11-12. Flag Descriptions for the netstat -C Command

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

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

```
annex01# netstat -C

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

Using the ping -t (traceroute) Option

The superuser command **ping -t** traces the path of a packet from the local host to the destination host and back, displaying information about each router in the path. This option allows you to see whether a packet arrived at and/or returned from its remote destination and, if not, where it stopped. The option is based on the Traceroute facility described in RFC 1393.

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 zero. 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 11-7 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 zeros 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 zero. 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 11-7).

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.

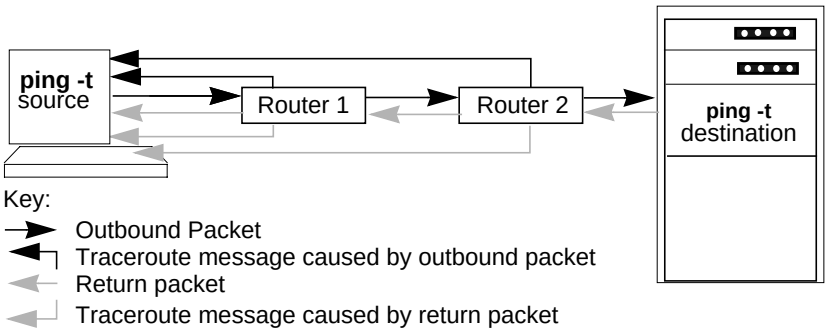

Figure 11-7. Overview of ping -t Actions

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. And, if a packet cannot be forwarded, **ping -t** locates the failure. Table 11-13 describes the fields displayed by **ping -t**.

Table 11-13. Fields Displayed by the ping -t Option

Field	Definition
Dir	The direction in which the ICMP packet is heading. The >>> symbols indicate an outbound packet heading towards the **ping -t** destination. The <<< symbols indicate a return packet heading back towards the **ping -t** source. The *** symbols indicate 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 (e.g., 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 zero 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 zero in this field.

The sample topology shown in Figure 11-8 is assumed by the **ping -t** examples that follow it.

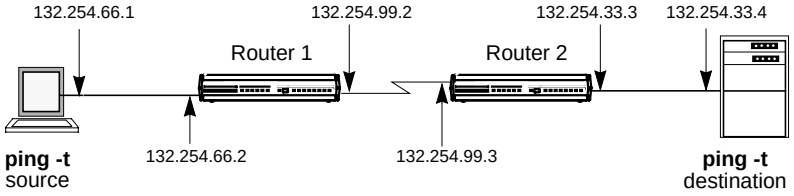

Figure 11-8. Topology for ping -t Examples

Given the topology in Figure 11-8, the **ping -t** command displays output such as the following when a traceroute packet passes successfully to the **ping -t** destination and back.

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
```

In the preceding example:

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 10000000 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 having to fragment 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 2 forwarded the packet has an IP address of 132.254.66.2, a speed of 10000000 bits per second, and can transmit packets of up to 1500 bytes in length without having to fragment 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, Router 2 is unable to forward the outbound packet, as indicated by the asterisks (***) under the *Dir* heading. Note that the hop count remains at *1*, since 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
```

Troubleshooting

This section describes:

- CLI commands for displaying routing information.
- Common configuration errors.
- What to do if the RAC is not advertising updates as expected.
- What to do if the RAC is not receiving updates as expected.

CLI Commands Providing Routing Information

Use the following commands to obtain information about IP routing on your network.

- To display the contents of the RAC routing table, use the CLI command **netstat -r** (see *Displaying the RAC Routing Table on page 11-62*).
- To display the contents of the routing cache (which contains user-configured routes), use the CLI command **netstat -C** (see *Displaying the route cache on page 11-66*).

Common Configuration Errors

In configuring routing, users make four common mistakes:

- Attempting to route through Proxy ARP interfaces. As has been stated several times in this chapter, do not configure a static route whose next-hop address is a Proxy ARP interface. Doing so causes packets to be routed improperly or not routed at all.
- Depending on Proxy ARP when routing is more reliable.
- Configuring network/subnet addresses that overlap.
- Configuring non-contiguous subnets.

The following sections describe in detail the final three of these four mistakes.

Depending on Proxy ARP When Routing Is More Reliable

Figure 11-9 shows a subnet configuration in which a PC can dial into one of two RACs. Since all nodes are on the same subnet, it is tempting to assume that *host01*could reach the PC via Proxy ARP. However, hosts and RACs retain Proxy ARP information long enough so that it is possible for the PC to have dialed into one RAC, disconnected, and then dialed into the other RAC before *host01* updated its ARP table. Therefore, the host may not have the up-to-date Proxy ARP information.

To avoid this kind of confusion, assign the PC a different subnet address - e.g., 132.254.6.*x*. This forces *host01* and the RACs to use routing updates (which are more frequent than ARP requests) to determine the RAC to which the PC is attached. (Make sure that active routing is enabled; see *Configuring Active RIP on page 11-47*. Also, make sure that *host01* supports either subnet or host routes.)

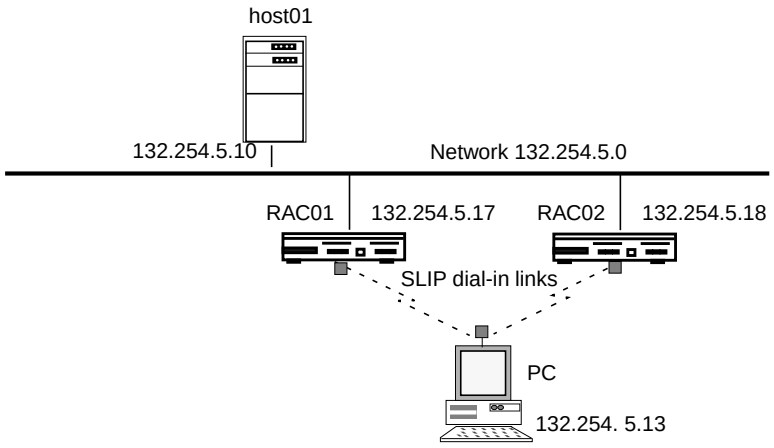

Figure 11-9. Configuration in Which Proxy ARP Can Fail

Overlapping Subnet/Network Addresses

Figure 11-10 shows a configuration that looks as if it should work but does not. In this configuration, *annex01* is attached to a backbone network and to another RAC that is, in turn, connected to a subnet. The problem with the configuration is the subnet mask for *RAC01* and *host01*, which designates the two rightmost bytes for host addresses. This mask creates the following kinds of difficulties:

- *RAC01* cannot determine where to send packets addressed to hosts whose addresses fall between 132.254.7.1 and 132.254.7.254. For example, where would it send a packet addressed to 132.254.7.20, which is the address of *host02*? As far as *RAC01* is concerned, host 7.20 could be on either of the two networks shown - *132.254.0.0* or *132.254.7.0*.

- Nodes on 132.254.0.0, such as *host01*, cannot reach nodes on subnet 132.254.7.0, such as *host02*. If *host01* wants to send a packet to *host02*, *host01* will try to use ARP to determine the Ethernet address to which it should deliver the packet, since *host02*'s address appears to be on the same directly attached network, *132.254.0.0*. No nodes will respond to the ARP request, since *132.254.7.20* is *not* on the local network.

To make the configuration shown in Figure 11-10 work, redefine the subnet mask for *annex01* and *host01* as 255.255.255.0. This indicates to the RAC and to *host01* that they are on subnet 132.254.5.0, and that, as a result, *host02* is on a different subnet and must be reached via a router (*annex02*).

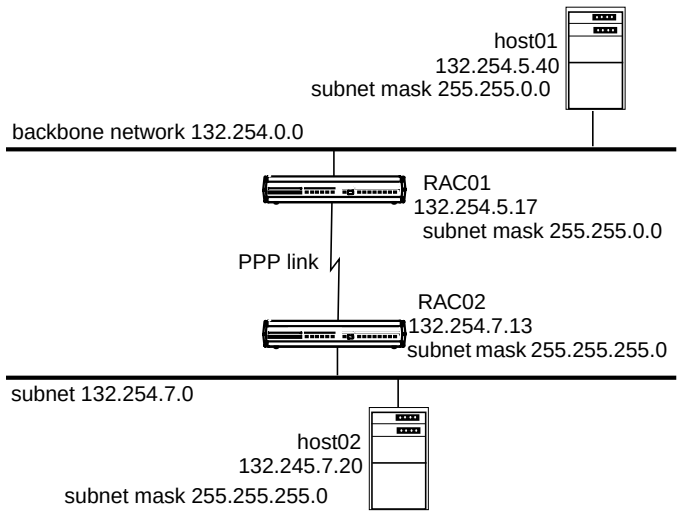

Figure 11-10. Overlapping Addresses

Non-contiguous Subnets

Figure 11-11 shows a network in which two RACs are configured to support, via modem pools, two dial-in remote PCs. The problem with this configuration is that both PC's are configured as if they were on subnet 132.254.7.0. Because the subnet masks on the RAC ports are set to 255.255.255.0, both RACs advertise that they can reach this subnet. Consequently, nodes on 132.254.5.0, such as *host01*, cannot determine which RAC is the appropriate next hop for either PC.

There are two solutions to this problem. The first is to configure both PCs with 132.254.7.*x* addresses to use the same RAC, reserving the other RAC for PCs and/or hosts on a different subnet. The second solution is to configure the ports on the RACs to use a subnet mask of 0.0.0.0 (the default), which is interpreted as 255.255.255.255, a host subnet mask. This causes the RACs to advertise host routes, rather than subnet routes. Provided that *host01* and other nodes on subnet 132.254.5.0 accept RIP host route advertisements, they will be able to determine which RAC to use as the next hop for a given PC.

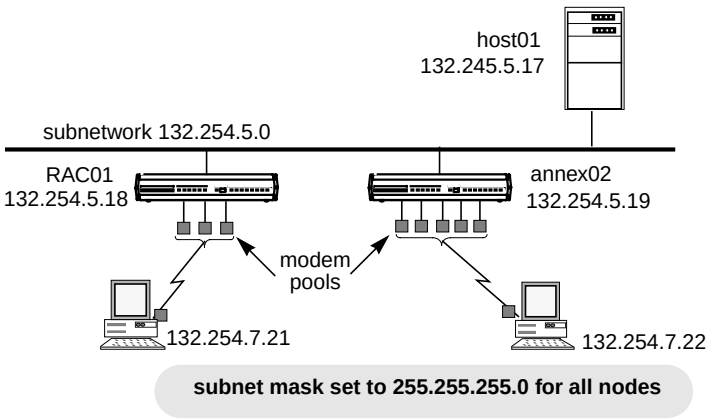

Figure 11-11. Non-contiguous Subnets

What To Do If the RAC Does Not Advertise Updates

If your RAC is not sending RIP updates as expected, check the following:

1. **Is the RAC parameter** routed **set to** Y**? See** ***Configuring Active RIP on page 11-47*****.**
2. **Did you reboot the RAC after setting** routed**?**
3. **Is the** rip_advertise **parameter set to** all **for all interfaces? To check this, issue the following** admin **commands:**

```
admin : show interface rip_advertise

interface en0:
     rip_advertise: all

interface asy1:
     rip_advertise: all

interface asy2:
     rip_advertise: all

interface asy3:
     rip_advertise: all

interface asy4:
     rip_advertise: all

     .
     .
     .
```

4. **If the display shows that** rip_advertise **is set properly above RIP as enabled, something else is wrong.**
5. **Is the RAC broadcast address set correctly? See** ***Setting the Broadcast Address on page 11-20*****.**

6. **Are at least two interfaces up and running?**
7. **If your network is divided into subnets, are the IP subnet addresses and subnet masks set correctly for the RAC and the SLIP and PPP ports? See *Subnetting Using Subnet Masks on page 11-15*.**
8. **If your network is divided into subnets, is the interface parameter** rip_sub_advertise **set to** Y **(the default)? See *Advertising Subnet Routes on page 11-49* and *rip_sub_advertise on page 11-59*.**
9. **Is** rip_horizon **set to** split**? If so, there may not be any routes to advertise on that interface. See *Using Split Horizon and Poison Reverse on page 11-50* and *rip_horizon on page 11-57*.**
10. **Are RIP packets being filtered out? For example, a filter that discards outgoing UDP packets also discards RIP packets, since RIP runs on UDP. To list all the defined filters, enter the CLI** superuser **command:**

    ```
    annex: su
    password:
    annex# filter list
    ```

    For more information, see *Using Filters for Security on page 6-146*.
11. **Are your hosts ignoring RIP version 2 updates? If so, set the interface parameter** rip_send_version **to** 1 **(see *Advertising RIP 1 and/or RIP 2 Updates on page 11-48*).**

What to Do if the RAC Does not Receive Updates

If the kernel routing table does not contain the expected learned routes, check the following:

1. **Are the routes really being advertised? Check to see if updates are being received by other routers on the network.**
2. **Did you reboot the RAC after setting** routed**?**
3. **Is** rip_accept **set to** all **(the default)? If not, are the correct network destination addresses being included or excluded via** rip_accept**?**
4. **If your network is divided into subnets, are the IP subnet addresses and subnet masks set correctly for the RAC and the SLIP and PPP ports? See *Subnetting Using Subnet Masks on page 11-15*.**
5. **Is the RAC parameter** routed **set to** Y**? See *Enabling Passive RIP Alone on page 11-26* and *routed on page 11-60*.**
6. **Is the RAC broadcast address set correctly? See *Setting the Broadcast Address on page 11-20*.**
7. **If subnet routes are not being learned, is** rip_sub_accept **set to** Y **(the default)? See *rip_sub_accept on page 11-59*.**
8. **Is** rip_recv_version **set correctly for the version(s) of RIP running on your network? See *Authenticating Incoming RIP 2 Updates and Requests on page 11-44* and *rip_auth on page 11-56*.**

Other Documentation

For outside sources of information on TCP/IP, routing, and RIP, see:

- Comer, Douglas E. (1991) *Internetworking with TCP/IP Volume I; Principles, Protocols, and Architecture* (3rd ed.) Englewood Cliffs, N.J., Prentice-Hall.
- Hedrick, C.L (1988) *Routing Information Protocol*. RFC 1058 (STD 34).
- Malkin, G. (1993), *RIP Version 2*. RFC 1388.

Chapter 12
Internetwork Packet Exchange Protocol

This chapter describes how to configure the RAC for access by remote nodes using the Internet Packet Exchange (IPX) protocol. IPX is the network-layer communications protocol that Novell networks use to deliver data packets. The RAC provides Novell dial-in connectivity and routing.

This chapter discusses:

- Novell Networks
- Standards-based IPX (IPX over PPP) Features
- Enabling IPX
- Default IPX Parameter Settings
- IPX Configuration Overview
- Configuring Standards-based IPX (IPXCP)
- Obtaining IPX Information

Novell Networks

Nodes on a Novell network are *servers* or *clients*. Servers provide shared access to files, printers, and specialized peripheral devices on the network. The RAC functions as a communications server, providing shared access to the network by non-Novell as well as Novell nodes. *Clients*, also called *workstations*, connect to the server(s) via a network interface (Ethernet, Token Ring, or Arcnet) to access files and services. The most common client and server hardware platforms are PCs.

The programs are stored on the server and retrieved for execution on the client. A client PC, unlike a terminal, can operate as a stand-alone computer since it has its own processor, storage, operating system, and application software.

IPXCP Features

The RAC implements standards-based IPX (IPX over PPP) via the IPX Control Protocol (IPXCP) described in RFC 1552. IPXCP allows a PC to dial into a RAC as an endpoint node on an IPX network. The same PC can also simultaneously run IP over the connection, allowing the user to use either IP or IPX services as the need arises. The same link can also be used for AppleTalk over PPP.

To dial into a RAC via IPXCP, a PC client can be running any operating system that supports IPXCP networking. This includes Windows 95, Windows NT, and DOS or Windows running FastLink II Version 2.*x* or higher.

In addition, IPX over PPP provides asynchronous routing, allowing an IPX network to run across a PPP LAN-to-LAN link. The RAC provides this routing by default. The RAC configured for IPXCP automatically sends and accepts RIP and SAP packets, provided that you set a network number for the link. However, a client can choose whether or not to receive RIPs and SAPs provided the client software allows this choice.

RAC IPXCP does not support NLSP routing.

Default IPX Parameter Settings

Table 12-1 and Table 12-2 list the default Serial Networking Protocol and IPX parameter settings, and the wan b parameter and default settings.

Table 12-1. Default Serial Networking Parameter Settings

Parameter	Default Setting
ipx-network	00000000
ipx-node	00-00-00-00-00-00
net_inactivity	off
allow_compression	N
address_origin	local
slip_ppp_security	N
net_inactivity_units	minutes
remote_user	pass

Table 12-2. Wan b Networking Parameter Settings

Parameter	Default Setting
ipx-network	00000000
ipx-node	00-00-00-00-00-00

IPX Configuration Overview

To configure the RAC for IPX sessions:

1. **Decide the method to use for handling IPX nodes.**
2. **Edit the configuration file to define SPBs for security files.**
3. **Review the default global port parameters, then reset the parameters you need for the IPX configuration.**

Step 1: Decide How to Handle IPX Addressing

The RAC handles IPX nodes using one of the following methods:

- Dial-up addressing
- Fixed addressing

You can configure the RAC for both methods, but dial-up addressing has priority. For information about how the RAC operates when both dial-up and fixed addressing are enabled, see *Determining Dial-up Addresses Using the acp_dialup File on page 6-145*.

Dial-up Addressing

Dial-up addressing is controlled through the global port parameter **address_origin**. When this parameter is set to **acp**, the RAC uses the host-resident **acp_dialup** file to handle addressing. The file resides in the RAC install directory. For information on making entries into the **acp_dialup** file, see *Creating the acp_dialup File on page 6-143*.

Any ACP dial-up address request that comes from the RAC includes the RAC address and an associated user name from PAP, CHAP, or ACP CLI security, which are used as the key in this file. Default match is by username (actually peername) and RAC IP addr, but port type, unit number, time of day, and other optional keys are also available. Once the keys are matched, the corresponding dial-up addresses are returned to the caller on the RAC. Dial-up addressing offers the ability to assign IPX nodes to individual users (e.g., when the **address_origin** parameter is set to **local** or the user is not found in acp_dialup).

Fixed Addressing

Fixed addressing for the RAC is controlled through the **ipx_network** and **ipx_node** parameters used with the **set wan b** command. This parameter and command associate IPX nodes with PRI B channels and WAN DSO channels.

When needed, the IPX nodes set using this command/parameter combination are automatically used for calls over the B channels. Unlike dial-up addressing, fixed addressing associates IPX nodes with B-channels, not with specific users.

For more information, see the *Remote Access Concentrator Software Reference Guide*.

Step 2: Edit the Configuration File

Session Parameter Blocks (SPBs) are structures within the configuration file. SPBs enable a RAC to handle one or more types of calls differently from the default call handler. If you need to create any SPBs to handle special IPX details, do it now. See *Configuring the WAN Interfaces, Global Ports, and Sessions on page 5-1*, for detailed information on SPBs.

To edit the configuration file:

1. **Open the configuration file.**

   The configuration file is located in the install directory on the UNIX load host (the default file is **/usr/spool/erpcd/bfs/config.annex**). Use any system editor (**vi**, **textedit**) to edit the file.

2. **Go to the** wan **or** pri **section in the file. This section contains examples. Do the following:**
   - Read the information that precedes each sample SPB provided in the configuration file. You should be familiar with this information before you enable or disable a default SPB.
   - Remove the comment character (#) from the beginning of each line of any sample SPB you want to enable.
   - Enter a comment character (#) at the beginning of each line of each SPB you want to disable.
   - Enter any new SPBs.
3. **Save the file.**
4. **Issue a** reset annex session **command from** na **or** admin**.**

Step 3: Review and Reset Global Port Parameters

The RAC ships with a set of default global port parameters already stored in non-volatile RAM.

Review the defaults to determine which ones you need to change to satisfy your configuration requirements for PPP or security.

The remainder of this section provides the following information:

- A list of the default settings for the Serial Networking and PPP global port parameter groups.
- Instructions for changing a global port parameter setting.
- Instructions for using the **set wan b** command to associate IPX nodes with RAC PRI B channels.

To view the entire set of default global port parameters use **na** or **admin** to issue the **show port all** command.

Default PPP-Related Global Port Parameters

Table A-1 lists the default parameters related to the PPP protocol stored in the RAC nonvolatile memory when shipped. You can view these PPP-specific parameters through the **show port ppp** command issued from the **na** utility or the **admin** utility.

Table 12-3. Default PPP-related Global Port Parameter Settings

Parameter	Default Setting
allow_compression	N
address_origin	local
net_inactivity	off
net_inactivity_units	minutes
ppp_acm	0x0
ppp_mru	1500
ppp_ncp	all
ppp_password_remote	“<unset>”
ppp_sec_auto	N
ppp_security_protocol	none
ppp_username_remote	““
slip_ppp_security	N

How to Change a Global Port Parameter Setting

To change a global session parameter setting using **na**:

1. **At a terminal connected to a UNIX host, enter:**

   % na

   The following prompt displays on the screen:

   ```
   Annex network administrator Rx.x January 1, 1997
   COMMAND:
   ```

2. **Specify the RAC on which you intend to change global port parameter settings at the** COMMAND: **prompt. Specify the administrative password for host at the** password: **prompt.**

   You can specify the RAC by its IPX node or name. If you intend to change global port parameter settings on more than one RAC, separate their IPX nodes or names using a comma (,). The password is the administrative password for this host.

   For example:

   ```
   COMMAND:annex 132.245.6.40 or
           annex 132.245.6.40,132.245.6.45
           password:
   ```

3. **Specify a new setting for the global port parameter at the** COMMAND: **prompt.**

   For example, to change the default setting of the **address_origin** parameter (local) to enable dial-up addressing through the **acp_dialup** file, enter the following:

   ```
   COMMAND:set port address_origin acp
   ```

   

   The new parameter setting is stored automatically in nonvolatile RAM.

4. **To review your changes, issue the** show port all **command at the** COMMAND: **prompt.**

   This command displays all of the global port parameter settings. To locate the parameters you changed, press the return key, which allows you to scroll through the file.

   ```
   COMMAND: show port all
   ```

5. **Enter** quit **at the** COMMAND: **prompt to exit** na**.**

   ```
   COMMAND: quit
   ```

Assigning IPX Networks or Nodes to B Channels

The **set wan b** command (issued with the **ipx_network** or **ipx_node** parameter) associates IPX networks or nodes with PRI B channels.

Command Syntax

Use the following command syntax when creating B channel IPX network or node assignments:

`set wan b=`*`ch_range`* `{ipx_network` *`net_no`*`|ipx_node` *`node_no`*`} [`*`increment`*`]`

ch-range is a single B channel number, a list of B channels separated by commas, a range of B channels separated by a hyphen, or the keyword **all**.

net_no is the IPX network (in hexadecimal) to which you want to assign a single B channel or the first channel of a set of B channels.

node_no is the IPX node (in hexadecimal, with dashes separating the octets) that you want to assign to a single B channel or to the first channel of a set of B channels.

increment is the value by which you want to increment automatic IPX node or network assignment to the B channels specified by *ch_range*. For networks, you can specify increment as an integer. For nodes, you must specify increment in the same format you would specify a node - in hexadecimal, with dashes separating the octets.

Usage Rules

You can associate IPX networks or nodes with PRI B/WAN DSO channels in the following ways:

- For a single B channel (4).
- For the entire set of PRI B channels by using one of the following choices:
  - Specifying channels as a range of two numbers separated by a dash (1-23 for T1; 1-30 for E-1).
  - Using the keyword **all**.

When assigning IPX nodes to the entire set, you can specify an increment by which IPX nodes are assigned in sequence based on the increment value.

The following example specifies the entire set of B channels available with a T1-based RAC PRI module (23), an IPX node of 00-00-00-00-00-a0, and an increment of 2:

```
set wan b=1-23 ipx_node 00-00-00-00-00-a0 00-00-00-00-00-02
```

In this case, B channel #1 has an IPX node of `00-00-00-00-a0`. B channel #2 is assigned `00-00-00-00-a2`. B channel #3 is assigned `00-00-00-00-a4`, and B channel #23 is assigned `00-00-00-00-cc`.

When you do not specify any B channels, the command makes 23 or 30 IPX node or network B channel assignments based on the increment value.

When an increment is not specified, the command assumes a default increment of zero (0).

Configuration Samples

The following samples illustrate how to set global port parameters to enable PPP configurations.

Sample Configuration Using Dial-up Addresses

Figure 12-1 shows a configuration in which a remote PC (i.e., user *green*) is connected to a RAC through a PPP link. The PC appears to the network as directly attached device.

This configuration uses dial-up addressing through the **acp_dialup** file. The PC is connected through a BRI line with a V.120 terminal adapter and the PRI line to the RAC.

Based on the entries in the **acp_dialup** file, user *green* has access from all RACs and Remote Annexes since the **acp_dialup** file entry is a wildcard (*). User *green*'s **acp_dialup** file remote address is 00446688:00802d0077bc. The IPX address is 00802d0077bc and the IP address is 132.245.5.18. Following Figure 12-1 are the steps to complete that implement this configuration.

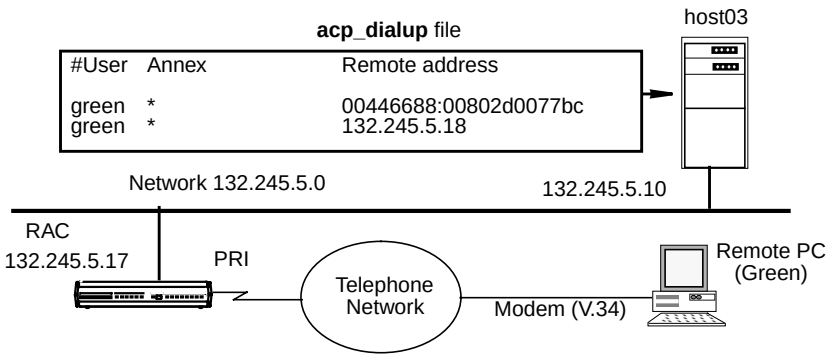

Figure 12-1. Connecting a Single Host Using PPP

To enable this configuration:

1. **Edit the** acp_dialup **file. Provide user *green* with access from all RACs and Remote Annexes by specifying a wildcard** (*) **and a remote address of 132.245.5.18.**

   See *Dynamic Allocation of Network Addresses on page 6-142* for instructions.

2. **Use the** set wan b **command with the** ipx_node **parameter to associate a set of IPX nodes with the PRI B channels.**

   

   Step 2 is optional since the RAC ignores the IPX node/B channel assignments created using the **set pri b** command when using dial-up addressing.

   However, if the host where the ACP server resides is unreachable by the RAC, or there is no entry in the **acp_dialup** file for a particular user, the RAC relies on the IPX nodes assigned to the B channel to provide a remote address for the link.

3. **Edit the configuration file to define an SPB.**

   You can use the default SPBs provided as part of the **config.annex** file or create them specifically for your requirements.

4. **Reset default global port parameters as required to the following settings:**

   - Enable CLI and/or connection security using the security parameters: **cli_security** and **connect_security**.
   - The **slip_ppp_security** parameter controls dial-in PPP access and use of ACP for PPP and protocol security such as **pap**, **chap**, and **chap-pap**. If **enable_security** and **slip_ppp_security** are enabled, access to the PPP command is restricted via ACP and call access is logged in the ACP log file.
   - Set the **ppp_security_protocol** parameter to **pap**, **chap**, **chap-pap**, or, for CLI users, **none** in some instances.
   - Set the **ppp_username_remote** and **ppp_password_remote** parameters to the values expected by the remote node, (the PC in Figure 12-1.
   - Set the **allow_compression** parameter to **Y** if you want the RAC to accept compressed packets.

- Set the **address_origin** parameter to **acp** so that the RAC requests the endpoint addresses, based on the user's login, from the **acp_dialup** file.
- If no remote network and node address is specified in **acp_dialup**, the RAC uses the values set for the **wan b ipx_node** and **ipx_network** parameters.
- You can leave **ppp_mru** parameter set to its default.

Sample Configuration Using Fixed Addressing

Figure 12-2 illustrates a configuration in which a single remote PC is connected to a RAC through an asynchronous PPP session. The session is running via a BRI line with a V.120 terminal adapter and the PRI line to the RAC.

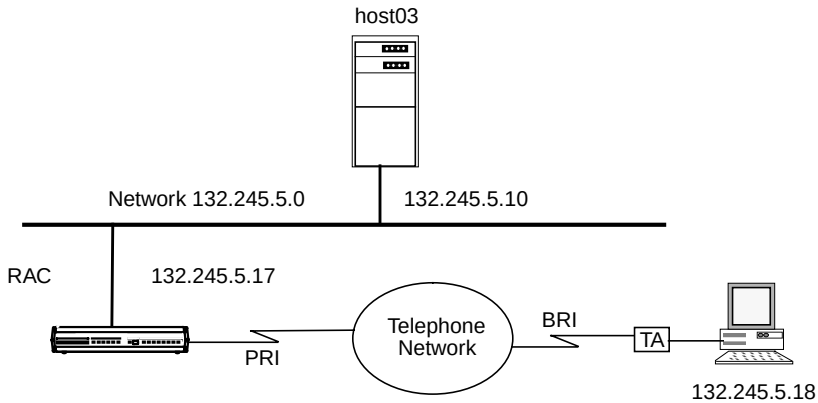

Figure 12-2. Connecting a Single Host Using PPP with Fixed Addresses

To enable this configuration, follow these steps:

1. **Use the** set pri b **command with the** ipx_node **parameter to associate a set of IPX nodes with the PRI B channels.**

   See *Assigning IPX Networks or Nodes to B Channels on page 12-9* for instructions to perform this step.

2. **Edit the configuration file to define an (SPB).You can use the default SPBs provided as part of the** config.annex **file or create specific SPBs for your requirements.**

3. **Reset default global port parameters as required to the following settings:**

   - Enable CLI and/or connection security using the security parameters **cli_security** and **connect_security**.
   - The **slip_ppp_security** parameter controls dial-in PPP access and use of ACP for PPP and protocol security such as **pap**, **chap**, and **chap-pap**. If **enable_security** and **slip_ppp_security** are enabled, access to the PPP command is restricted via ACP and RAC access is logged in the ACP log file.
   - Set the **ppp_security_protocol** parameter to **pap**, **chap-pap**, or, for CLI users, **none**.
   - Set the **address_origin** parameter to **local**.
   - Leave the **ppp_acm** and **ppp_mru** parameters set to their defaults.

Obtaining IPX Information

IPX information is available from several sources, including log messages the RAC creates automatically and output that various commands display.

System Logs

The RAC automatically logs **ppp** and **auto_detect** events to a 4.3BSD system log daemon (**syslogd**) on the RAC.

Negotiated PPP addresses are logged in the **acp_logfile**. For more information on syslogging, see *Using Event Logging on page 4-3* and *Encrypting Security Messages on page 6-59*.

IPXCP Interface Statistics

The **netstat -ip** command displays the IPXCP state and IPXCP options. The following is an example of IPXCP statistics.

annex: **netstat -ip 6**

```
                        LCP Status
State                Current: Open     Prior: Ack sent
Options              Local:            Remote:
MRU                  1500              1500
Auth type            None              None
LQM                  None              None
ACFC                 On                On
ACCM                 0x00000000        0x000a0000
Magic                0xbb1ee499        0x0047501b
PFC                  On                On

                      NCP (IPXCP)Status
State                Current: Open     Prior: Ack sent
Options              Local:            Remote:
Network No           12345678          12345678
Node No              00802d009c30      00802d009c30
Compression          None              None
Routing Port         RIP/SAP           RIP/SAP
Router Name          LM009c30          None
```

Negotiating the LCP Options on page 8-18 explains the fields displayed for *LCP Status*. Table 12-4 explains the fields displayed for *IPXCP Status*.

Table 12-4. Fields in (NCP) IPXCP Status Display

Field	Explanation
State	Shows the current and prior state of the IPXCP link. The states are:
Closed	The link has shut down via an administrative or peer request.
Request sent	The RAC has sent a configure request and is waiting for an answer.
ACK received	The RAC has received a configure ACK and is waiting for a configure request.
ACK sent	The RAC received and answered a configure request, and awaits a configure ACK.
Open	IPXCP negotiation has completed successfully.
Closing	The link is in the process of closing. The RAC has sent a terminate request and is waiting for a terminate ACK.
Options	Shows the current values of the negotiated options. The *Local:* column displays the value suggested by the RAC. The *Remote:* column displays the value suggested by the remote client. The options are:
Network No	The 8-digit hexadecimal IPX network number of the remote client.
Node No	The 12-digit hexadecimal IPX node number of the remote client.
Compression	The kind of IPXCP header compression used. This can be *Telebit* or *None.*
Routing Proto	The routing protocol used by the RAC and (optionally) the client. This can be *RIP/SAP* or *None*.
Router Name	The name by which the RAC is known as an IPX router.

IPX Interfaces, Memory Buffers, Routes, and Servers

The CLI command **netstat -x** provides options for displaying information about:

- IPX in general
- Using the **netstat -x** command itself
- IPX network interfaces
- The amount of memory available in the IPX buffer pools
- IPX routes (RIPs)
- IPX servers

The **netstat -x** syntax is:

netstat -x [**i** | **r** [*network_number*] | **s** [*server_name*] | **?** | **m**]
or
netstat -x [**i** | **r** [*network_number*] | **S** [*server_name*] | **?** | **m**]

IPX in General

Issuing the **netstat -x** command displays the number of NICs, RIPs, and Service Advertising Protocol (SAP) services on the RAC. NICs indicates the number of active IPX interfaces (including **en0**) on the RAC, and RIPs indicates the number of Novell networks the RAC can reach.

The **netstat -x** command display looks like this:

```
annex: netstat -x
There are 2 NICs, 3 RIPs, and 4 SAPs
```

Using the netstat -x Command

Issuing the **netstat -x ?** command displays information about the use of **netstat x**, as follows:

```
annex: netstat -x?
Usage: netstat -x
               -xi
               -xm
               -xr [network]
               -xs [server_name]
               -xS [server_name]
```

IPX Network Interfaces

Issuing the **netstat -xi** command displays information about the RACs currently in use for dial-in.

Name	Network	Tics	C0	NB	S0	Ipkts	Ierrs	0pkts	0errs	Collis
en0	00001234	2	n	y	n	21592	0	201380	0	0
asy18	00003456	4	n	y	n	72	0	98	0	0

The field headings in the above sample display indicate the following:

- *Name* is the interface name of the corresponding IPX port over which IPX dial-in or routing is currently occurring.
- *Network* is the number of the network to which interface *Name* connects.
- *Tics* indicate the amount of time associated with the cost of using interface *Name*. A tic is approximately 55 milliseconds.
- The *CO* field is not used.
- *NB* indicates whether or not this interface propagates NetBIOS information.
- *SO* indicates whether or not this interface propagates Server information only.
- *Ipkts* is the number of IPX packets received on this interface.
- *Ierrs* is the number of incoming IPX packets that contained errors.
- *Opkts* is the number of IPX packets transmitted over this interface.

- *Oerrs* is the number of outbound IPX packets that contained errors.
- *Collis* is the number of times a packet transmission was terminated due to a collision.

IPX Buffer Pools

Issuing the **netstat -xm** command displays the amount of memory available in the large and small IPX buffer pools. The RAC creates these buffer pools when it boots, allotting the appropriate amount of memory for the configuration. If you change the configuration, reboot the RAC so it can allot the proper amount of buffer memory.

```
annex: netstat -xm
Large IPX Buffer Pool: Free = 0125 Total = 0125 Min =  0109
Small IPX Buffer Pool: Free = 0125 Total = 0125 Min =  0117
```

IPX Routes

Issuing the **netstat -xr** command displays the routes defined in the RAC's IPX routing table. In the following example, **netstat -xr** displays five routes.

```
annex: netstat -xr
Network     Gateway          Tics     Hops     Interface
2d90ab99    0000a2816349     3        2        en0
00000042    0000a2816349     24       5        en0
00000043    0000a2816349     3        2        en0
00000044    0020af07dec4     3        2        en0
00001234    ffffffffffff     0        0        en0
```

The field headings in the above display indicate the following:

- *Network* is the number of a destination Netware network.

- *Gateway* is the number of the next hop on the path to *Network*. A gateway of ffffffffffff indicates a directly attached network.
- *Tics* indicate the amount of time required to reach *Network* when *Gateway* is the next hop. A tic is approximately 55 milliseconds.
- *Hops* are the number of routers that must be crossed to reach *Network.*
- *Interface* is the network interface using the route.

Issuing the **netstat -xr** command followed by a network number displays the RAC route for that network. The following example shows how to display the route for network 42 (you can omit the leading zeros when specifying the network number):

```
annex: netstat -xr 42

Network    Gateway          Tics     Hops     Interface

00000042   0000a2816349     24       5        en0
```

IPX Servers

Issuing the **netstat -xs** command displays server names, types, and addresses.

```
annex: netstat -xs

OSCAR     File Server   [2e80703c] 000000000001 [0451]
CTEST     Annex NAS     [00000055] 00802d01d252 [e480]
VENUS     File Server   [00006501] 000000000001 [0451]
SMTPQ     Advert Print  [00000043] 000000000001 [8060]
SNOWY     Annex NAS     [00000063] 00802d01ea57 [e480]
```

From left to right, the fields in the previous display are as follows:

- The first field is the server name, for example, *SUPT_TJB_INT.* If the name is longer than 34 characters, **netstat -xs** displays only the first 34 characters.

- The second field is the server type, which can be:
  - *File Server*
  - *Job*
  - *Print*
  - *Archive*
  - *Job Queue*
  - *NAS SNA Gate(way)*
  - *TimeSync VAP*
  - *Dynamic SAP*
  - Annex NCS
  - *Annex NAS*
  - *Advert(ised) Print*
  - *Btrieve (5.0) VAP*
  - *SQL VAP*
  - *TES-NetW(are) VMS*
  - *NetW(are)* Access
  - Named Pipes
  - *NetW(are)* UNIX
  - Netware 386
  - *NETW(are) manage(ment) (type 0x6601)*
  - *NETW(are) manage(ment) (type 0x6a02)*
  - *Unknown <type>*

In the list above, text in parentheses is provided for clarity; **netstat -xs** does not display it.

- The third field is the server's hexadecimal address, displayed in the format [*network*] *address* [*socket*].

Issuing the **netstat -xS** command displays an additional line of information for each server. The additional line contains the RAC route for the server.

```
annex: netstat -xS
HOBBESAnnex NAS       [00000012] 00802d009930 [e480]
  Gateway = [00000009] 0000a2816349  Hops = 2   IF = en0
ARAMIS NetWare 386    [0beef123] 000000000001 [8104]
  Gateway = [00000009] 0000a2816349  Hops = 3   IF = en0
ARAMIS File           [0beef123] 000000000001 [0451]
  Gateway = [00000009] 0000a2816349  Hops = 2   IF = en0
ROSA NetWare 386      [1a2a3b4c] 000000000001 [8104]
  Gateway = [00000009] 0000a2816349  Hops = 3   IF = en0
ROSA File             [1a2a3b4c] 000000000001 [0451]
  Gateway = [00000009] 0000a2816349  Hops = 2   IF = en0
```

When issued with a *server_name* argument after the **-s** or **-S** option, **netstat -sx** or **netstat -Sx** displays information for that specified server only.

Server names are typically in upper case.

IPX Frame Type and Network Number

The CLI **stats** command displays various RAC statistics, including the RAC Netware network number.

The following is part of a **stats** display; IPX information is on the last line shown:

```
annex: stats

S/W: Remote Access Rx.x          Fri Jan 10 00:33:16 EST 1997
H/W: 5393/Turbo, MLB Rev 128.0   ROM Rev: 1013, PRI VERSION A
Ports: eth 1pri 32mod 32syn/ta   8MB RAM 64KB EE 67.7KB SLC1 2MB FLSH
Boot from: 132.245.33.71         Date: Mon Jan 13 16:23:24 1997 EST
Image: oper/oper.63.enet/B13.3.6 Uptime: 1 day 21 hours 14 min.
Inet addr: 132.255.12.14         Subnet mask: 255.255.255.0
Ethernet addr: 00-60-2d-04-ec-bb Broadcast addr: 132.245.11.255
Primary NS: 132.255.32.7         Domain: xylogics.com
QUICC Ver: 130
IPX Frame Type: EthernetII       IPX Network Number: 11
```

If a zero is displayed for the IPX network number, either the **ipx_frame_type** parameter was not configured properly or there is no IPX file server on the network. IPX cannot be used.

IPX State

The CLI **stats -o** command shows whether or not IPX is enabled (see *IPX Configuration Overview on page 12-3*).

IPX Connections

For all IPX ports, the CLI **who** command displays specific information about an IPX connection, including what protocol the connection is using, the user name associated with the connection, where the connection is located, when the connection was created, how long the connection has been idle, and the address from which the connection was made.

The following is an example of a **who** command display:

```
annex: who

Port   What  User   Location     When    Idle Address
v1     CLI   ---    ---          10:00am      132.245.9.4
2      PPP   ---    ---          11:00am :20  [local]
```

Statistics for All Interfaces and for 802.2

Use the CLI command **netstat -i** to display statistics for current RAC interfaces and for the 802.2 data-link layer. For example:

```
annex: netstat -i

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

                  *** Hardware Interface Statistics ***

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

Chapter 13
AppleTalk

The RAC implementation of AppleTalk provides dial-in connectivity in a multi-protocol network. Using the RAC as a dial-in AppleTalk Remote Access (ARA) server, a remote ARA user can dial into the RAC and become a directly connected ARA network user. The RAC is transparent to the ARA user; it behaves like an AppleTalk end node.

AppleTalk Remote Access Protocol (ARAP)

ARAP allows Apple PowerBook and Macintosh computers to communicate with one another or with an AppleTalk network over standard telephone lines. A remote ARA user can dial into an AppleTalk network and take advantage of all the services available on the network, including:

- File transfer
- Electronic mail
- Database access
- Printing
- Mounting remote disks

AppleTalk on the RAC supports ARAP V1 and V2.

Configuring the RAC for AppleTalk

After booting, the RAC automatically determines the appropriate network information, e.g., its AppleTalk node ID, etc. The AppleTalk-specific RAC parameters **a_router**, **zone**, and **node_id** are hints for the RAC to use at start-up (*AppleTalk-Specific Configuration Parameters on page 13-3* describes these parameters).

The RAC behaves like an AppleTalk phase II end node. At start-up, it listens for an AppleTalk router in the start-up network range and begins the process of finding its address. The RAC selects as its A_Router the first router it detects broadcasting an RTMP Route Data Request. It acquires an address within the A_Router's net-range and then uses a ZIP GetNetInfo and ZIP GetZoneList to find the network's zones; otherwise, the RAC obtains an available address within the start-up range.

The RAC also installs a net-range route and an AppleTalk default route from the A_Router. If another router broadcasts an RTMP message, and its Ethernet address matches the address defined in the RAC parameter **a_router**, the RAC discards the current router information and tracks to this new router. If the RAC does not hear from the current A_Router for 50 seconds it selects a new A_Router. This 50-second hold-down prevents the RAC from bouncing between routers.

When ARA clients connect to a RAC port, the **node_id** parameter acts as a hint to acquire an address for the client. The RAC then installs a proxy *aarp* entry and the client's zone multicast address.

AppleTalk-Specific Configuration Parameters

You can use either **na** or **admin** to configure RAC parameters.

The AppleTalk-specific configuration parameters are divided into two groups:

- AppleTalk-specific RAC parameters.
- AppleTalk-specific global port parameters.

AppleTalk-Specific RAC Parameters

These parameters provide some AppleTalk protocol control, limits, and identification. Table 13-1 lists these parameters; the following subsections describe them in detail.

Table 13-1. AppleTalk-specific RAC Parameters

Parameter	Default	Description
a_router	00-00-00-00-00-00	The Ethernet address of the network's A_Router.
default_zone_list	""	This zone list is sent to ARAP clients as the local back-up to ACP.
node_id	0.0	The address the RAC tries to acquire at start-up.
zone	""	The AppleTalk zone for use at start-up.

Since AppleTalk uses dynamic addressing, AppleTalk addresses are acquired at boot time. The **a_router**, **zone**, and **node_id** parameters are hints for the RAC to use at start-up. If another AppleTalk node is using an address defined as a hint, the RAC chooses a different address. The **stats** command displays the run-time values for these parameters.

a_router

The Ethernet address of the network's A_Router. The RAC uses this value as a hint at start-up. 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, e.g., 00-7F-12-33-44-55. The default is **00-00-00-00-00-00**.

default_zone_list

This zone list is sent to ARA clients as the local back-up to ACP. The parameter is a 100-character string with spaces separating the zones, e.g., **marketing engineering sales**. When this parameter is not set, the RAC provides the network zone list. The default is a **null string** ("").

You must use the backslash (\) character to escape embedded spaces.

node_id

This is the address the RAC tries to acquire at start-up. 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** to **65534**; valid *node* values are **0** to **254**. The default is **0.0**.

zone

The **zone** parameter provides the AppleTalk zone for use at start-up. It is a 32-byte string variable.This is the zone in which ARA clients are located unless overridden by security. The default is a **null string** ("").

AppleTalk-specific Global Port Parameters

The **set <parameter>** command modifies the AppleTalk-specific port parameters (where <parameter> is one of the parameters described in this section). The **show port appletalk** command displays them. Table 13-2 lists these parameters; the following subsections describe them in detail.

Table 13-2. AppleTalk-specific Global Port Parameters

Parameter	Default	Purpose
arap_v42bis	Y	Enables/disables V.42bis compression during an ARA session.
at_guest	N	Allows ARA guest login service.
at_nodeid	0.0	The node ID given to an ARA client during connection establishment.
at_security	N	Enables/disables ACP service for this port.

arap_v42bis

The **arap_v42bis** parameter enables/disables V.42bis compression during an ARA session. A **Y** enables the parameter; an **N** disables it. The default is **Y**.

If you disable this parameter, you may want to change the Communications Control Language (CCL) script for the remote modem to improve performance. Sample CCL scripts are included in the software distribution.

at_guest

The **at_guest** parameter allows guests to log into an ARA service. When this parameter is enabled and a client requests guest access, the RAC asks ACP for user name (guest) privileges. A **Y** enables guest privileges; an **N** disables guest privileges. The default is **N**.

at_nodeid

The **at_nodeid** 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** to **65534**. The valid *node* values are **0** to **254**. The default is **0.0**.

at_security

The **at_security** parameter enables/disables ACP service for this port. When both **at_security** and **enable_security** are set, the RAC uses ACP to get per-user security information about the client (authentication, logging, and zone access) from the **acp_userinfo** file (see *Creating the acp_userinfo File on page 6-74*). If **at_security** is not set, the RAC uses only local security (**port_password** and **username** for authentication, and the **default_zone_list**). A **Y** enables this parameter; an **N** disables it. The default is **N**.

CLI AppleTalk Commands

The Command Line Interface (CLI) is the command interface for the RAC. At the CLI, you enter commands that connect to hosts, manage jobs (or sessions), display and modify port parameters, and display information for the RAC and the network.

The CLI provides two groups of commands: user and superuser. You administer the RAC using the superuser commands. Table 13-3 lists the CLI commands for use with AppleTalk; the following subsections describe them. (*The Remote Access Concentrator Software Reference describes all of the CLI commands.*)

Table 13-3. CLI AppleTalk Commands

Command	Description
arap	Converts a CLI line into an ARA connection. When the port is reset, it reverts to its original mode. After entering the command, the RAC prompts: *Annex switching line to ARAP*. The **arap** command does not apply to VCLI connections.
arp	This superuser command displays ARP cache on the RAC.
netstat	Displays information about AppleTalk interfaces.
ping	This superuser command generates AppleTalk Echo Packets (AEP).
stats	Displays AppleTalk information.
who	Displays a line's type as ARA.

Command Syntax

You can shorten any CLI command or host name to the minimum number of letters that make the name unique. This is referred to as *minimum uniqueness*. If you do not want the RAC to interpret a host name using minimum uniqueness, enclose the name in double quotes (""). For example, entering hosts "new" prevents ambiguities between hosts newark and new. You can enter commands and host names in all lower case, all upper case, or a combination of both. The RAC performs any necessary case conversion.

arap

The **arap** command converts a CLI line into an ARA connection. Resetting the port returns the CLI to its original mode. The syntax is:

arap

The command display looks like this:

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

arp

The **arp** command displays and, optionally, modifies the IP-to-hardware address translation table that the Address Resolution Protocol (ARP) uses. Since the RAC builds the ARP table dynamically, you rarely need to modify it. Table 13-4 lists the arguments for this command.

Although the **arp** command shows AppleTalk information, you cannot manipulate it. Since **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.

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. For example:

```
annex01# arp -a

xenna (192.9.200.95) at 08-00-4C-00-2a-c0 tt1=20
2356.189 at 08-00-4e-34-22-39 tt1=16
```

Table 13-4. Arguments for the 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 the host, specified using either *host* or an Internet address, at the hardware address specified using *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.

AppleTalk over ARA

AppleTalk over ARA allows Apple PowerBook and Macintosh computers to communicate with one another or with an AppleTalk network over standard telephone lines. An ARA user can dial into a remote AppleTalk network and use all the available services as if that user is physically connected to the network through EtherTalk.

AppleTalk Configuration Overview

To configure the RAC for PPP sessions:

1. **Edit the configuration file to define SPBs for AppleTalk.**
2. **Review the default global port parameters, then reset the parameters you need for the PPP configuration.**

Step 1: Edit the RAC Configuration File

Session Parameter Blocks (SPBs) are structures within the RAC configuration file. SPBs enable a RAC to handle calls properly.

Before editing the file, determine whether you intend to use one or all of the default SPBs provided in the PRI section of the configuration file, disable one or all of the default SPBs, or write your own set of SPBs. See *Configuring the WAN Interfaces, Global Ports, and Sessions on page 5-1*, for detailed information on SPBs.

The following instructions describe how to enable and disable the default SPBs that exist within the configuration file.

To edit the configuration file:

1. **Open the configuration file.**

   The configuration file is located in the install directory on the UNIX load host (default filename is **/usr/spool/erpcd/bfs/config.annex**). Use any system editor (e.g., **vi**, **textedit**) to edit the file.

2. **Go to the** pri **section in the file. The section begins with the percent symbol (%) and the name** pri**. Do the following:**

   - Read the information that precedes each default SPB provided in the configuration file. You should be familiar with this information before you enable or disable a default SPB.
   - Remove the comment character (#) from the beginning of each line of each SPB you want to enable.
   - Enter a comment character (#) at the beginning of each line of each SPB you want to disable.
   - Following the *called_number* field in an SPB that has one, replace the string with the telephone number callers will use from remote nodes that will use this SPB.

     For example, to modify a default SPB to handle AppleTalk calls:

```
#begin_session appletalk
#called_no Replace_this_string_with_your_appletalk_telephone_number
#call_action modem
#set mode arap
#end_session
```

Remove the comment character (#) from the beginning of each line.

Replace this string with the telephone number callers using AppleTalk will use.

3. **Save the file.**

4. **Issue a** reset annex sessions **command from** na **or** admin**.**

Step 3: Review and Reset Global Port Parameters

The RAC ships with a set of default global port parameters already stored in non-volatile RAM. Review the defaults to determine which ones you need to change to satisfy your configuration requirements for PPP, security, etc.

The remainder of this section provides instructions for changing a global port parameter setting.

To view the entire set of default global port parameters use **na** or **admin** to issue the **show port all** command.

How to Change a Global Port Parameter Setting

To change a global port parameter setting using **na**:

1. **At a terminal connected to a UNIX host, enter:**

   `% na`

   The following prompt displays on the screen:

   ```
   Annex network administrator Rx.x January 1, 1997
   COMMAND:
   ```

2. **Specify the RAC on which you intend to change global port parameter settings at the** COMMAND: **prompt. Specify the administrative password for host at the** password: **prompt.**

   You can specify the RAC by its IP addresses or name. If you intend to change global port parameter settings on more than one RAC, separate their IP addresses or names using a comma (,). If prompted for a password, the password is the administrative password for the host on which **na** is running.

   For example:

   ```
   COMMAND:annex 132.245.6.40 or
           annex 132.245.6.40,132.245.6.45
           password:
   ```

3. **Specify a new setting for the global port parameter at the** COMMAND: **prompt.**

   For example, to change the default setting of the **at_security** parameter (N) to enable security enter the following:

   ```
   COMMAND:set port at_security Y
   ```

   The new parameter setting is stored automatically in nonvolatile RAM.

4. **To review your changes, issue the** show port all **command at the** COMMAND: **prompt.**

   This command displays all of the global port parameter settings. To locate the parameters you changed, press the return key which allows you to scroll down through the file.

   ```
   COMMAND: show port all
   ```

5. **Enter** quit **at the** COMMAND: **prompt to exit** na**.**

   ```
   COMMAND: quit
   ```

Sample AppleTalk Configuration

Figure 13-1 illustrates the following sample settings. In this configuration, a Macintosh connected to a RAC through an ARA link appears to the network as an attached node.

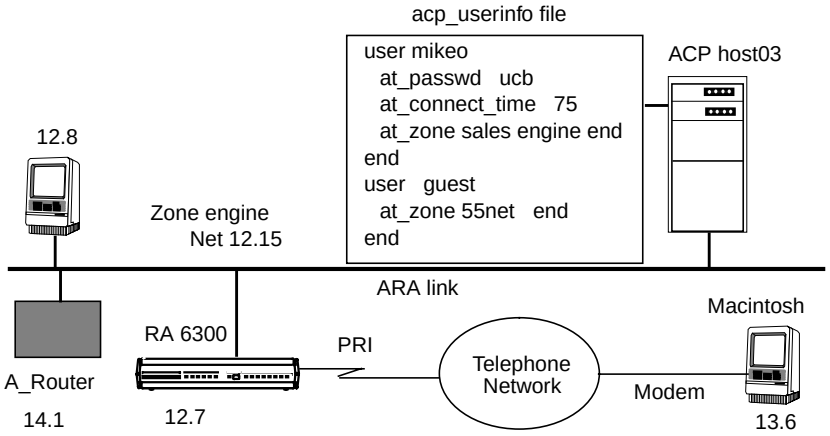

Figure 13-1. Connecting a Macintosh Using ARA

To enable this configuration:

1. **Edit the configuration file to define an SPB.**

   You can use the default SPBs provided as part of the **config.annex** file or create them specifically for your requirements. For more details, see *Step 1: Edit the RAC Configuration File on page 13-10*.

2. **Reset default global port parameters as required to the following settings:**

   - Enable ARA security by setting **at_security** to **Y**. ACP and port access is logged in the ACP log file.
   - Use the supplied defaults for the **data_bits** (8), **stop_bits** (1), and **parity** (none) parameters.

   ARA is an 8-bit protocol. If **data_bits** is set to **7**, and **parity** is not set to **none**, the RAC forces the **data_bits** setting to **8** and the **parity** setting to **none**. Otherwise, the RAC generates an error message for the port.

- Set the **arap_v42bis** parameter to **Y**.
- Set the **at_guest** parameter to **Y** to enable guest access.
- Set the **at_nodeid** parameter to define the AppleTalk node address for the remote Apple PowerBook or Macintosh.

If you intend to use AppleTalk over PPP, see *Point-to-Point Protocol on page 8-1*.

ARA Security

The RAC provides comprehensive security features that assist you in securing your RACs and the network from unauthorized access. Using these features, you can select between host-based security (where at least one host on the network is functioning as a security server) and local password protection (where the passwords are stored on the RAC). Optionally, you can use local password protection as a back-up to host-based security.

If you are using host-based security, you must define the user name and password in the **acp_userinfo** file (see *Creating the acp_userinfo File on page 6-74*).

The RAC provides protection through the use of an administrative password that controls access to the superuser CLI commands. This password also protects access to a RAC through **na**. The security system provides audit trails that monitor users and their activities. The RAC also provides the source code for the Access Control Protocol (ACP) security system, and the flexibility to integrate RAC security with existing security for a network-wide system.

The following subsections briefly describe RAC security as it relates to ARA. For a detailed description of ACP, host-based security, and the **acp_userinfo** file, see *Configuring Security on page 6-1*.

Security Features

The RAC implementation of ARA provides three areas of security:

- ARA security
- Zone security
- Logging

ARA Security

The basic ARA security features are:

- **username and password authentication**

  The RAC authenticates the client using Apple's DES encryption algorithm. To define a user name and password for a registered (as opposed to guest) user, see *Creating the acp_userinfo File on page 6-74*.

- **guest access**

  The RAC allows anonymous access to the network. Restrictions can be applied to *guests* by setting up an ACP *guest* profile with limitations. For more details, see *at_zone on page 6-88*.

- **connection timer**

  The connection timer is stored in the **acp_userinfo** file. For more details, see *Creating the acp_userinfo File on page 6-74*.

Zone Security

Every user can have a zone list assigned via remote ACP. If a list is not available via ACP, the RAC provides all the zones it has learned from the network. If local security is used, use the per RAC parameter **default_zone_list.** For more details, see *at_zone on page 6-88*.

Logging

The RAC logs activity and errors from the ARA session. The log is accessed via remote ACP and **syslog** (see *Using Event Logging on page 4-3* for more details).

Network-Visible Entity (NVE) Filtering

NVE filtering controls a remote access Apple user's view of network resources: when using *Chooser* to select resources, only the resource set defined for the user by the administrator will be visible. The administrator can specify the NVE filter on a per-user basis. This feature complements the existing zone list, by offering a higher level of control.

The administrator uses the **nve_filter** entry in the **acp_userinfo** file to specify a list of filters on a per-user basis. See *at_nve_filter on page 6-90* for detailed information on creating **nve_filter** entries.

This method of limiting NBP traffic is not secure, and can be circumvented by a person willing to write some code to probe the network without using NBP. Also, this feature has no local RAC security equivalent.

AppleTalk over PPP

AppleTalk over PPP allows Apple PowerBook and Macintosh computers to connect as an endpoint node to an AppleTalk network. The same Macintosh can also run IP over the connection simultaneously, allowing the user to use either IP or AppleTalk services as the need arises.

When the RAC opens a PPP connection, it negotiates for link-level options, and then runs an optional security phase to authenticate the user. Finally, the two ends negotiate for network control protocol (NCP) options. The link is then opened and becomes a generic interface for the RAC. An AppleTalk point-to-point link is configured, enabled, and disabled using AppleTalk Control Protocol (ATCP). See *Point-to-Point Protocol on page 8-1* for information about setting PPP parameters.

The RAC implementation of ATCP currently supports dial-in only.

Appendix A
Digital Modem Configuration Parameters

This appendix describes parameters used to configure the RAC's digital modems. These parameters are set using "config_bytes" entries in the **%digital_modem** section of the **config.annex** file.

Custom Modem Configuration

For normal operation, digital modems do not require custom configuration as described here. These configuration instructions are provided for users who have a compelling need to alter the digital modems' behavior at a basic level.

Parameters 26 through 63 are provided for use by qualified modem engineers only. Changing these parameters from their default settings could cause unexpected negative results, and is discouraged.

Using the %digital_modem Section

The **%digital_modem** section of the **config.annex** file can be used to establish user-defined modem types and modify the digital modem configuration parameters described in this document.

An example of the **%digital_modem** section follows below:

```
%wan

begin_session credit_card
called_no 103
call_action modem
set type_of_modem credit_card
end_session
```

(continued on next page)

```
%digital_modem

type_of_modem credit_card
config_bytes 16 =15,15,7,7
end
```

In this example, when the RAC receives a call on a number that ends with 103, the call is treated as a modem call, with a user-defined modem type named *credit_card*. Before the modem answers the call, its digital modem configuration parameters 16, 17, 18, and 19 are set to the values 15, 15, 7, and 7, respectively. This sets the options Maximum Tx Line Speed and Maximum Rx Line Speed to Speed 31200, and the options Minimum Tx Line Speed and Minimum Rx Line Speed to Speed 4800.

Setting Parameter Values

You can modify a digital modem configuration parameter by specifying the parameter number and its corresponding value in the **%digital_modem** section of the config.annex file. You can modify a series of parameters by specifying the number of the first parameter in the series, then specifying the corresponding values for all of the parameters in that series.

Each parameter's value is a decimal , octal, or hexadecimal number that is the equivalent of the parameter's binary bit settings; the **config_bytes** keyword does not use values expressed in binary. Decimal is the default base for setting parameter values. If you set values in octal base, precede them with "0"; if you set values in hexadecimal base, precede them with "0x". Each parameter value in a series may be set separately, so it is not necessary to use the same base for a single series of configuration parameters.

For example, the parameter map for *Parameter 3* on page A-12 shows that the default setting for the parameter is 11101 expressed in binary; this value is 29 expressed in decimal, 035 in octal, and 0x1D in hexadecimal. Changing the MNP Frame Size option to Block Size 64

causes the bit setting to be 00101, or 5 in decimal, 05 in octal, and 0x5 in hexadecimal.

The corresponding **config_bytes** entry in the **%digital_modem** section looks like this (with the parameter's value expressed in decimal base):

```
config_bytes 3 =5
```

Standard Digital Modem Configurations

This section describes five standard digital modem configurations. Each description lists certain parameters and options which must be set a particular way for the modem to operate correctly.

U.S. Operation, V.42bis Primary/MNP5 Secondary

This is the digital modems' default configuration. No **config_bytes** entries are needed to enable this configuration.

Parameter 0:
L: LAPM - Enabled (*Default)*
M: MNP - Disabled *(Default)*
B: Buffer - Enabled (*Default)*

Parameter 1:
D: V42 Detection - Enabled (*Default)*

Parameter 2:
T: Tx V.42bis - Enabled (*Default)*
R: Rx V.42bis - Enabled (*Default)*
M: Max String Length - String Length 16 *(Default. Setting the option to other values may cause unexpected results.*)
D: Dictionary Size - Dictionary Size 2048 (*Default)*

Parameter 3:
C: MNP Class - Class 5 (*Default)*
B: MNP Frame Size - Frame Size 256 (*Default)*

Parameter 21:
Modulation - V34+ (bit rates up to 33.6Kbps) (*Default)*

***Parameter 30*:**

A/μ: Compander Type 1 - μ (mu) law (*Default)*

***Parameter 31*:**

m1: v34 Full Duplex - Enabled (*Default)*

***Parameter 33*:**

T**rC: Trellis Coding** - 4D 16 States (*Default)*

***Parameter 34*:**

V8: V8 Handshake - Enabled (*Default)*

E: V34 Extension - v.34 Extended *(Default)*

U.S. Operation, Disable V.42bis Compression (V.42 only)

Leave all parameter options set to their default values except:

***Parameter 2*:**

T: Tx V.42bis - Disabled (*Not Default)*

R: Rx V.42bis - Disabled (*Not Default)*

required **config_bytes** setting:

config_bytes 2 =4

U.S. Operation, MNP5 Primary: V.42/V.42bis Disabled

Leave all parameter options set to their default values except:

***Parameter 0*:**

L: LAPM - Disabled *(Not Default)*

Although other parameters affect V.42 operation, only parameter 0 needs to be changed to disable V.42 operation completely.

required **config_bytes** setting:

config_bytes 0 =4

U.S. Operation, Disable V.34 Extended Speeds (33.6 Kbps)

Leave all parameter options set to their default values except:

***Parameter 21*:**
Modulation - V34 (bit rates up to 28.8Kbps) (*Not Default)*
required **config_bytes** setting:
config_bytes 21 =1

International Operation

Leave all parameter options set to their default values except:

***Parameter 30*:**
A/μ: Companding type - A law (*Not Default*)
required **config_bytes** setting:
config_bytes 30 =1

Parameter Maps and Option Settings

The default settings for each parameter are shown in binary in the map for the parameter and are displayed in bold in the description for each option. All settings are listed as binary values.

Unused and reserved bits are shaded in the parameter maps.

Parameter 0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
Descr.	**Unused**								**Reserved**				**T**	**B**	**M**	**L**

Default **config_bytes** setting:

Decimal base: 5

Octal base: 05

Hexadecimal base: 0x5

L: LAPM - When this option is enabled, the modem attempts to establish a Link Access Protocol for Modems (LAPM) connection. This is the link protocol used by V.42 for error correction. Disabling this option prevents V.42 or V.42bis operation. In this case, an MNP (Microcom Networking Protocol) connection or Buffer mode connection is made if the corresponding option is enabled.

0: Disable

1: Enable

M: MNP - (MNP is not available in the first release.) When this option is enabled, the modem attempts to establish a Microcom Networking Protocol connection. As with LAPM, disabling MNP prevents an MNP connection from being made. The modem then attempts to make a V.42 connection (if enabled) or a Buffer mode connection as a last resort.

0: Disable

1: Enable

B: Buffer - This option enables or disables data buffering.

0: Disable - Data is not buffered and speed conversion is not allowed.

1: Enable - Buffers about one screenfull of data between the DTE and the modem.

Modem connections will not be successful if all three options (L, M, and B) are disabled.

Parameter 1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**
Descr.	**Unused**								**Reserved**				**SR**	**B**		**D**

Default **config_bytes** Setting:

Decimal base: 13

Octal base: 015

Hexadecimal base: 0xD

D: V42 Detection - (Changing this option from its default value is not recommended.) This option causes the modem to skip the V.42 detection phase. It may be required for compatibility with some older non-standard modems that expect V.42 detection to be performed during the V.8 sequence.

0: Disable

1: Enable

B: V42 Break - A break is an attention key provided on some keyboards for interrupting a transmission from a host. Breaks were widely used by terminals communicating with mainframe computers, but are rarely used in modern applications such as Internet browsing. A break character consists of several character periods of constant Space bits. Table A-1 and Table A-2 below describe how the modem handles a break character:

0: Disable - Will not forward a break

1: Destructive/Expedited

10: Nondestructive/Expedited

11: Nondestructive/Nonexpedited

Table A-1. Transmitting DCE Break Handling with Respect to Data

Break Handling Option	Going to Remote DCE	Going to local DTE	Coming from remote DCE	Coming from local DTE
Destructive/ Expedited *	Complete data transmission in progress,then transmit break. Discard data not yet transmitted.	Discard data not yet delivered.	Discard data until acknowledgement received.	Discard data until acknowledge ment received.
Non-destructive/ Expedited	Complete data transmission in progress, then transmit break. Hold data until receive acknowledgment.	Continue delivering data.	Continue receiving data.	Continue receiving data.
Non-destructive/ Non-expedited	Wait for acknowledgment of data previously transmitted, and then transmit break. Hold data until acknowledgment received.	Continue delivering data.	Continue receiving data..	Continue receiving data.
All state variables pertaining to control function and error control function operation, except those pertaining to break transfer, are reset to their initial values. DCE: Data Communications Equipment (MODEM) DTE: Data Terminal Equipment (PC)				

Table A-2. Receiving DCE Break Handling with Respect to Data

Break Handling Option	Going to Remote DCE	Going to local DTE
Destructive/ Expedited *	Discard data not yet transmitted.	Discard data not yet delivered. Deliver break signal.
Non-Destructive/Expedited	No effect.	Deliver break signal immediately. Resume normal data delivery.
Non-Destructive/Non-Expedited	No effect.	Deliver break signal in sequence with respect to data.
All state variables pertaining to control function and error control function, except those pertaining to break transfer, are reset to their initial value. For all break options, acknowledgment should be returned as soon as possible. DCE Data Communications Equipment (MODEM) DTE Data Terminal Equipment (PC)		

SR: Selective Retransmission - This option supports the ability to retransmit a single frame that was received in error without having to retransmit the entire series of frames that contained the distorted packet. For example, assume the receiving modem identified frame 5 as bad and is not able request a retransmission until frame 8 has been transmitted. With selective retransmission turned off, the transmitting modem must retransmit frames 5 through 8. With selective transmission on, frame 5 could be retransmitted alone and the receiving modem would place it back into the proper sequence. This results in a more efficient, slightly faster exchange of data.

0: Disable

1: Enable

Parameter 2

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**
Descr.	**Unused**								**Reserved**		**D**		**M**		**R**	**T**

Default **config_bytes** Setting:

Decimal base: 7

Octal base: 07

Hexadecimal base: 0x7

T: Tx V.42bis - V.42bis is the extension to V.42 that provides data compression. This parameter allows the modem to selectively disable compression in the transmitting direction.

0: Disable

1: Enable

R: Rx V.42bis - V.42bis is the extension to V.42 that provides data compression. This parameter allows the modem to selectively disable compression in the receiving direction.

0: Disable

1: Enable

M: Max String Length - This parameter defines the maximum number of consecutive characters that are stored in the V.42bis dictionary. It optimizes the dictionary, and therefore compression, for different data types.

0: String Length 6

1: String Length 16

10: String Length 32

11: String Length 64

D: Dictionary Size - The dictionary stores recurrent data patterns which are used in V.42bis data compression. As a rule, the larger the dictionary, the better the modem is able to compress data. Both modems attempting to make a connection must use the same dictionary size and will use the largest dictionary available to both modems. The first digital modem release supports a dictionary of 512 bytes only.
0: Dictionary Size 512 bytes
1: Dictionary Size 1024 bytes
10: Dictionary Size 2048 bytes

Parameter 3

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**1**
Descr.	**Unused**								**Reserved**			**B**		**C**		

Default **config_bytes** Setting:
Decimal base: 29
Octal base: 035
Hexadecimal base: 0x1D

C: MNP Class - This option limits MNP operation to the specified MNP class or lower. Classes 1 through 4 support error correction only, while class 5 also supports data compression. Some early MNP-only modems improperly negotiate MNP class and may require the host modem to disable higher classes of operation.
0: Unused
1: Class 1
10: Class 2
11: Class 3
100: Class 4
101: Class 5

B: MNP Frame Size - This option specifies the maximum MNP frame size. It also allows operation with very early implementations of MNP. Note that V.42 (LAPM) frame size is fixed at 128 bytes.
0: Block Size 64
1: Block Size 128
10: Block Size 192
11: Block Size 256

Parameter 4

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 5

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 6

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 7

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 8

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**1**
Descr.	**Unused**								**Wait CD Time**							

Default **config_bytes** Setting:

Decimal base: 45

Octal base: 055

Hexadecimal base: 0x2D

Wait CD Time: The modem waits for a carrier for the specified number of seconds; the default value is 45 seconds. If a carrier is not received within this time, the modem hangs up and reports:

Message: No carrier

Disconnect reason: CD Timer

Parameter 9

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**
Descr.	**Unused**								**Retrain Timeout**							

Default **config_bytes** Setting:

Decimal base: 20

Octal base: 024

Hexadecimal base: 0x14

Retrain Timeout: The modem enters a retrain and waits for completion and reconnection for the time selected (the default is 20 seconds). If a carrier is not received within this time, the modem hangs up and reports:

Message: No carrier

Disconnect reason: Retrain timeout

Parameter 10

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 11

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**
Descr.	**Unused**								**Lost CD Time**							

Default **config_bytes** Setting:

Decimal base: 20

Octal base: 024

Hexadecimal base: 0x14

Lost CD Time: The modem waits the specified number of milliseconds (the default value is 20) after loss of carrier to report a failure and drop the connection.

Message: No carrier

Disconnect reason: Carrier lost

Parameter 12

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 13

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 14

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 15

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 16

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**
Descr.	**Unused**								**Maximum Tx Line Speed**							

Default **config_bytes** Setting:

Decimal base: 17

Octal base: 021

Hexadecimal base: 0x11

Maximum Tx Line Speed: The maximum transmission bit rate the modem supports.

0: Reserved

1: Speed 0 - 300

10: Speed 600

11: Speed 1200

100: Speed 2400

101: Speed 4800

110: Speed 7200

111: Speed 9600

1000: Speed 12000

1001: Speed 14400

1010: Speed 16800

1011: Speed 19200

1100: Speed 21600

1101: Speed 24000

1110: Speed 26400

1111: Speed 28800

10000: Speed 31200

10001: Speed 33600

Parameter 17

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**
Descr.	**Unused**								**Maximum RX Line speed**							

Default **config_bytes** Setting:

Decimal base: 17

Octal base: 021

Hexadecimal base: 0x11

Maximum Rx Line speed: The maximum reception bit rate the modem supports.

0: Reserved

1: Speed 0 - 300

10: Speed 600

11: Speed 1200

100: Speed 2400

101: Speed 4800

110: Speed 7200

111: Speed 9600

1000: Speed 12000

1001: Speed 14400

1010: Speed 16800

1011: Speed 19200

1100: Speed 21600

1101: Speed 24000

1110: Speed 26400

1111: Speed 28800

10000: Speed 31200

10001: Speed 33600

Parameter 18

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
Descr.	**Unused**								**Minimum TX Line speed**							

Default **config_bytes** Setting:

Decimal base: 1

Octal base: 01

Hexadecimal base: 0x1

Minimum Tx Line Speed: The minimum transmission bit rate the modem supports.

0: Reserved

1: Speed 0 - 300

10: Speed 600

11: Speed 1200

100: Speed 2400

101: Speed 4800

110: Speed 7200

111: Speed 9600

1000: Speed 12000

1001: Speed 14400

1010: Speed 16800

1011: Speed 19200

1100: Speed 21600

1101: Speed 24000

1110: Speed 26400

1111: Speed 28800

10000: Speed 31200

10001: Speed 33600

Parameter 19

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
Descr.	**Unused**								**Minimum RX Line speed**							

Default **config_bytes** Setting:

Decimal base: 1

Octal base: 01

Hexadecimal base: 0x1

Minimum Rx Line Speed: The minimum reception bit rate the modem supports.

0: Reserved

1: Speed 0 - 300

10: Speed 600

11: Speed 1200

100: Speed 2400

101: Speed 4800

110: Speed 7200

111: Speed 9600

1000: Speed 12000

1001: Speed 14400

1010: Speed 16800

1011: Speed 19200

1100: Speed 21600

1101: Speed 24000

1110: Speed 26400

1111: Speed 28800

10000: Speed 31200

10001: Speed 33600

Parameter 20

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Descr.	**Unused**								**Max RX baud**				**Max TX baud**			

Default **config_bytes** Setting:

Decimal base: 85

Octal base: 0125

Hexadecimal base: 0x55

Max Tx baud: The maximum transmission baud rate the modem supports. This parameter is provided for cases where certain non-standard modems do not support all baud rates.

0: 2400

1: 2743

10: 2800

11: 3000

100: 3200

101: 3429

Max Rx baud: The maximum reception baud rate the modem supports. This parameter is provided for cases where certain non-standard modems do not support all baud rates.

0: 2400

1: 2743

10: 2800

11: 3000

100: 3200

101: 3429

Parameter 21

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**					**Modulation**		

Default **config_bytes** Setting:

Decimal base: 0

Octal base: 00

Hexadecimal base: 0x0

Modulation: Sets the maximum modulation the modem supports.

0: V34+ (bit rates up to 33.6Kbps)

1: V34 (bit rates up to 28.8Kbps)

10: V32/V32bis (bit rates up to 19.2Kbps)

11: V22bis (bit rates up to 2400bps)

Parameter 22

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**RN**	**R**	**QF**	**QR**	**Reserved**			

Default **config_bytes** Setting:

Decimal base: 176

Octal base: 0260

Hexadecimal base: 0xB0

QR: Retrain when quality is bad. When this option is enabled, the modem pauses to retrain or re-establish the modem connection when the quality of the connection falls below the Signal Quality threshold value specified in Parameter 33. This is done in an attempt to respond to changes in the telephone connection and restore a high quality connection.
0: Disable
1: Enable

QF: Fallback when quality is bad. When this option is enabled, the modem will fall back to a lower speed of operation that is less demanding of the phone line quality during the initial handshake . This occurs when the received signal quality is below the Signal Quality threshold specified in Parameter 33.
0: Disable
1: Enable

RN: Renegotiates the rate upward when quality improves during a call. When enabled, the modem pauses and selects a higher speed during a call if line conditions improve from their initial values.
0: Disable
1: Enable

Parameter 23

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 24

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 25

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 26

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
Descr.	**Unused**								**MC**	**R**	**AB**	**PR**	**Pre-emphasis Filter**			

Default **config_bytes** Setting:

Decimal base: 31

Octal base: 037

Hexadecimal base: 0x1F

Pre-emphasis Filter - Pre-emphasis is an equalization method in which the transmission signal spectrum compensates for amplitude distortion. During channel probing, the modem selects one of ten possible filters. When this option is set to **Negotiable**, the modem determines which pre-emphasis filter to use based on channel probing.
0 - 1110: Reserved
1111: Negotiable

PR: Channel Probing - This option specifies whether or not the modem probes the phone line/channel to determine its characteristics and adjusts itself accordingly when establishing a V.34 connection.
0: Disable
1: Enable

AB: Asymmetric Baud Rates - This is the ability of the modem to operate at different baud rates for transmission and reception. It allows the modem to take advantage of differing line conditions in either direction.
0: Disable
1: Enable

MC: Mandatory Carrier Frequency - This option, effective only if Channel Probing is disabled, allows you to manually specify which carrier frequency (high or low) to use in V.34 modulation. Six baud rates are specified in V.34 modulation: 2400, 2743, 2800, 3000, 3200, and 3429. Out of these, the first five can operate at high or low carrier frequencies. During channel probing, one of these two carrier frequencies is selected in order to optimize the channel bandwidth usage.
0: Low
1: High

Parameter 27

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**NE**		**SHAP**		**PREC**		**P4_TR**	

Default **config_bytes** Setting:

Decimal base: 96

Octal base: 0140

Hexadecimal base: 0x60

P4_TR: Phase 4 Training - The Phase 4 Training constellation option allows you to specify a 4 point or 16 point signal constellation during the final phase (phase 4) of training. The default setting (**Negotiable**) allows the modem to make this determination.

0: Negotiable

1: Training 4 points

10: Training 16 points

11: Reserved

PREC: Precoding - Precoding is a non-linear equalization method that uses information provided by the remote modem to reduce equalizer noise enhancement caused by amplitude distortion.

0: Negotiable

1: Mandatory

10: Disable

11: Reserved

SHAP: Shaping in Receiver - Constellation shaping is a method for improving performance in the presence of Gaussian noise by shaping the signal constellation (the amount of spacing between individual amplitude and phase change points).

0: Negotiable
1: Mandatory
10: Disable
11: Reserved

NE: Nonlinear Encoder - Nonlinear encoding is a method for improving distortion immunity near the perimeter of a signal constellation by introducing non-uniform two-dimensional signal point spacing.

0: Negotiable
1: Mandatory
10: Disable
11: Reserved

Parameter 28

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**
Descr.	**Unused**								**Reserved**				**O3**	**O2**	**O1**	**R**

Default **config_bytes** Setting:

Decimal base: 14
Octal base: 016
Hexadecimal base: 0xE

O1: Optional Baud Rate 2743 - This supports certain country-specific spectral requirements as well as some unusual modems.

0: Disable
1: Enable

O2: Optional Baud Rate 2800 - This suppors certain country-specific spectral requirements as well as some unusual modems.
0: Disable
1: Enable

O3: Optional Baud Rate 3429 - This supports certain country-specific spectral requirements as well as some unusual modems.
0: Disable
1: Enable

Parameter 29

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
Descr.	**Unused**								**Reserved**			**R5**	**R4**	**R3**	**R2**	**R1**

Default **config_bytes** Setting:
Decimal base: 31
Octal base: 037
Hexadecimal base: 0x1F

Options R1 through R5 refer to combinations of baud or symbol rate and carrier frequency pairs. You can disable pairs selectively in order to troubleshoot certain international modem problems.

R1: 3000/1800 - Enables/disables 3000 baud, low-carrier frequency operation.
0: Disable
1: Enable

R2: 3000/2000 - Enables/disables 3000 baud, high-carrier frequency operation.
0: Disable
1: Enable

R3: 3200/1829 - Enables/disables 3200 baud, low-carrier frequency operation.
0: Disable
1: Enable

R4: 3200/1920 - Enables/disables 3200 baud, high-carrier frequency operation.
0: Disable
1: Enable

R5: 3429/1959 - Enables/disables 3429 baud rate operation.
0: Disable
1: Enable

Parameter 30

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**
Descr.	**Unused**								**Reserved**						**A/μ**	**D**

Default **config_bytes** Setting:
Decimal base: 3
Octal base: 03
Hexadecimal base: 0x3

D: Usb1 Early Detector - Detection of unscrambled binary ones supports V.34 modems under certain South American line conditions.
0: Disable
1: Enable

A/μ: Compander Type - Companding (compress/expand) is a method used to limit the dynamic range (strongest to weakest signal) encoded into PCM (pulse code modulation) signals used by the PSTN. The U.S. and Japan utilize μlaw (mu law) companding while Europe and most of the rest of the world require A law.

0: A law

1: μ law

Parameter 31

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**0**
Descr.	**Unused**								**Reserved**			**m4**	**m3**	**m2**	**m1**	**m0**

Default **config_bytes** Setting:

Decimal base: 26

Octal base: 032

Hexadecimal base: 0x1A

m1: v34 Full Duplex - Enables/disables V.34 operation.

0: Disable - The modem will falls back to V.32bis operation.

1: Enable

m3: v32/v32bis - Enables/disables V.32 operation.

0: Disable - The modem still operates at V.34 speeds and at V.22bis.

1: Enable

m4: v22/v22bis - Enables/disables V.22bis operation.

0: Disable - The modem still supports V.32 and V.34 operation.

1: Enable

Parameter 32

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 33

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**1**
Descr.	**Unused**								**Reserved**		**Q**	**CD**		**T**	**TrC**	

Default **config_bytes** Setting:

Decimal base: 21

Octal base: 025

Hexadecimal base: 0x15

TrC: Trellis Coding - This allows you to define the type of trellis coding (a type of forward error correction) used by the modem receiver during V.34 connections.

0: Negotiable

1: 4D 16States

10: Reserved

11: Reserved

T: v32 9600 Trellis Coding - Enables/disables trellis coding at 9600 bps.

0: Disable

1: Enable

CD: CD Threshold - This sets the level of the received signal necessary to enable the Carrier Detect signal from the modem.
0: -26dBm
1: -33dBm
10: -43dBm
11: Reserved

Q: Signal Quality Threshold - This sets the bit error rate threshold for the error indicator. For example, when this option is set to 0, the modem indicates an error when it detects one bit in error for every $1x10^4$ bits received.
0: 10^4
1: 10^6

Parameter 34

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**1**	**1**
Descr.	**Unused**								**R**	**E**	**V8**	**R**			**Eq**	

Default **config_bytes** Setting:
Decimal base: 99
Octal base: 0143
Hexadecimal base: 0x63

Eq: Equalizer Type - This is for V.32 operation only, and is provided to support certain international line conditions.
0: No compromise
1: Equalizer 1
10: Equalizer 2
11: Equalizer 3

V8: V8 Handshake - This feature must be enabled for V.34 operation. It may be disabled to support certain non-compliant V.32bis modems.
0: Disable
1: Enable

E: V34 Extension - Often mistakenly referred to as V.34bis (this ITU specification has not been released), this option disables extended (33.6Kbps) operation.
0: v.34 Bis
1: v.34 Extended

Parameter 35

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**CT**	**R**	**S1**	**RS**	**Reserved**		**EP**	

Default **config_bytes** Setting:
Decimal base: 0
Octal base: 00
Hexadecimal base: 0x0

EP: Echo Protection Tone - When this option is enabled, the modem sends an unmodulated carrier signal prior to the training and data signals to disable the telephone network echo suppressor.
0: Disable
1: Short
10: Long
11: Reserved

RS: Rx Space Disconnect - When this option is enabled, reception of a continuous space of specified duration from the remote modem will cause the modem to disconnect. The space required to cause a disconnect is either 1 second (or greater) or 2 seconds (or greater), depending on the setting of the TX/RX Space option.

0: Long

1: Short

S1: V22bis S1 Sequence - This option affects the duration of the S1 signal during a V22bis handshake. It is used to support certain older V.22bis modems.

0: 100 ms

1: 150 ms

RL: V25 Calling Tone - V25 calling tone. This tone is used in some international modems to identify a data call from a voice call.

0: Disable

1: Enable

Parameter 36

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 37

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 38

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							**NE**

Default **config_bytes** Setting:

Decimal base: 0

Octal base: 00

Hexadecimal base: 0x0

NE: Near End Echo Canceller - Echoes on the modem's phone line are due to transitions from digital or four-wire connections to two-wire analog service as found in domestic telephone connections. In most situations where the modem has a direct digital connection to the PSTN (T1, E1, or PRI ISDN), the signal is only converted to two-wire analog once, just before being routed to the subscriber's phone jack (client modem). In this case, only one echo is generated from the far end. This far end echo is expected and is compensated for by the modem's normal far end echo canceller.

In a very few situations, the telco converts the digital connection to two-wire analog and then back to digital before it is routed to the distant telco where the expected two-wire conversion is then performed. This additional conversion results in an unexpected near-end echo which may be compensated for by enabling this parameter. This should be done only when normal operation is not possible and th etelco indicates that such a conversion is taking place. Enabling this feature may result in an inability to achieve connect speeds in excess of 31.2 Kbps. In addition, as in any situation where a downlink digital to analog conversion (PCM hop) is made, 56K (V.pcm) connections are not possible.

0 - Disable

1 - Enable

Parameter 39

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 40

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 41

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 42

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 43

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 44

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 45

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
Descr.	**Unused**								**CPM Threshold**				**CPM Filter**			

Default **config_bytes** Setting:

Decimal base: 5

Octal base: 05

Hexadecimal base: 0x5

CPM Filter: Call Progress Filter Type - This supports the differing call progress tone (busy, ring-back, etc.) frequencies throughout the world.
0: General purpose
1: Belgium
10: Holland
11: Not used
100: Austria
101: USA
110: Reserved

CPM Threshold: Call Progress Threshold, dBm - This sets the minimum level of call progress tones that will be detected by the modem.
0: -50
1: -48
10: -46
11: -44
100: -42
101: -40
110: -38
111: -36
1000: -34
1001: -32
1010: -30
1011: -28
1100: -26
1101: -24
1110: -22
1111: -20

Parameter 46

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**1**
Descr.	**Unused**								**GF**	**GT**	**AP**	**AF**	**AT Threshold**			

Default **config_bytes** Setting:

Decimal base: 35

Octal base: 043

Hexadecimal base: 0x23

AT Threshold: Answer Tone Threshold, dBm - Used in originate mode only, this sets the minimum level that the modem requires to detect the answering modem's answer tone.

0: -50

1: -48

10: -46

11: -44

100: -42

101: -40

110: -38

111: -36

1000: -34

1001: -32

1010: -30

1011: -28

1100: -26

1101: -24

1110: -22

1111: -20

AF: Answer Tone Frequency, Hz - Sets the frequency of the modem's answer tone. This changes in some international modems.

0: 2100

1 - 2225

AP: Answer Tone Phase Change - This enables phase changes in the answering modems answer tones. These phase changes are used to disable echo cancellers found in the telephone network. This can be heard as the "ding ding ding" sound heard during the early portion of the modem's handshake. It almost never should be disabled except in some unusual international lines.

0 - Disable

1 - Enable

GT: Guard tone - Some national regulatory agencies require guard tones on systems that use international telephone circuits (for V.22 and V.22bis modes only).

0: Disable

1: Enable

GF: Guard tone frequency, Hz

0: 550

1: 1800

Parameter 47

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 48

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 49

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 50

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 51

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 52

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 53

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 54

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 55

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 56

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 57

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 58

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 59

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 60

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 61

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 62

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Parameter 63

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Descr.	**Unused**								**Reserved**							

Symbols

/etc/group file 6-69
/etc/services file 4-5

Numerics

4.2BSD
    hosts, accessing 4-1
4.3BSD
    syslogging 4-3, 12-15
        setting up host for 4-2

A

a_router parameter 13-2, 13-4
accesscode 6-77 to 6-79
    entries example 6-76, 6-79
ACE/Server software
    installing 4-23
ACP dialup file 8-4, 10-3
ACP files
    editing 6-49, 6-50
    use in mixed Remote Annex environments 6-49
ACP security 6-50, 6-50 to 6-57
    See also SafeWord, SecurID, security
    changing file names 6-102
    creating acp_dialup file 6-142
    creating acp_userinfo file 6-74 to 6-93
    encryption 6-59
    integrating SecurID into 6-115 to 6-118, 6-118
    linking to NIS 6-100
    server configuration 6-57 to 6-58
acp_dialup file
    creating 6-142, 6-143
    fields 6-144
acp_key parameter 6-60
acp_keys file
    creating 6-59
    syntax rules 6-59
acp_logfile 8-12, 8-13, 12-12, 12-14
    locking 6-106
acp_passwd file 6-110
    creating 5-40
    for use with Kerberos 6-112
acp_policy.h file 6-106
acp_regime file 6-70 to 6-72
acp_restrict file
    arguments 6-95
    using for connection security 6-94
acp_userinfo file 3-44, 6-154, 13-15 to 13-17
    creating 6-74 to 6-93
        using accesscode option 6-77 to 6-79
        using at_connect_time option 6-89
        using at_nve_filter option 6-90 to 6-91
        using at_passwd option 6-91
        using at_zone option 6-88
        using chap_secret option 6-93
        using clicmd option 6-80
        using climask option 6-81
        using deny option 6-83
        using filter option 6-84
    using for dial-out routes 3-45
    using include files in 6-99
ACP-related files
    guidelines for creating 6-48
active RIP. *See* RIP
Additional Number Information (ANI) 7-2
addr command 4-2, 4-12
address origin parameter
    configuring for dial–in PPP 8-14
address_origin parameter 6-18, 6-142
addressing
    broadcast 4-2
    IP 11-11 to 11-18
    RAC Internet 3-10 to 4-3
admin command 1-11, 11-43, 11-48
    using to configure RAC parameters 3-7 to 3-10, 13-3

administrative password
configuring 6-4
using with su command 2-3
AFD
See automated firmware download
allow_compression parameter
configuring for dial-in PPP 8-14
configuring for PPP link 8-12, 8-16, 10-10, 10-12, 12-12
ANI (Additional Number Information) 7-2
annex...end blocks 3-22, 8-18
route cache and 10-14
using to configure active RIP 11-47
AppleTalk 13-3, 13-17
configuring RAC for 13-2
over ARA 13-9
over PPP 13-17
AppleTalk Configuration Overview 13-9 to 13-13
editing the RAC configuration file 13-10
reviewing and resetting global port parameters 13-12
AppleTalk Configuration Sample 13-13 to 13-15
AppleTalk Control Protocol 13-17
See also ACP security
AppleTalk Remote Access Protocol. *See* ARAP
AppleTalk Remote Access. *See* ARA
AppleTalk security 6-133 to 6-134
ARA security and 6-133
logging and 6-134
NVE filtering and 6-134
zone security and 6-134
ARA 13-1
security 6-133, 13-15 to 13-16
ARAP 13-1
arap command 13-8
arap_v42bis parameter 13-5
arp command 13-8
at_connect_time 6-89
at_guest parameter 13-6
at_nodeid parameter 13-6
at_nve_filter 6-90 to 6-91
at_passwd 6-91
at_security parameter 13-6
at_zone 6-88
ATCP 13-17
attn_string parameter 5-39
audit trail 6-154
auth_protocol parameter 6-18
authoritative_agent parameter 4-3
authorized_groups parameter 4-27
auto_busyout_enable parameter 7-4
automated firmware download (AFD) 3-52

B

Bay Networks Press xxv
bfs 4-8
installing software using 4-5
blacklisting 6-11
boot command 4-7, 11-43, 11-48
boot-d command 4-7, 4-8
booting
configuring for RAC 4-7 to 4-13
self- 4-11
from Flash ROM without Local Ethernet Interface 4-12
using tftp 4-12
boot-l command 4-11
BOOTP requests 8-24
over SLIP 10-15
broadcast address
configuring for RAC 4-2
broadcast_addr parameter 4-2
broadcasting
disabling for files during boot 4-11
disabling for security server 6-58
enabling for time server 4-14
for name server 4-19
buffer pools, IPX 12-20
Bundling Scenarios 9-6

C

call delivery 5-1
calls
  Multilink PPP 1-8
  synchronous PPP 1-7
  V.120 1-6
  voice over PRI 1-5
CAS
  auto-busyout of DS0 channels 7-4
CAS interface 1-3
ch_passwd command 6-102
ch_passwd utility
  command syntax 6-111
  description of 6-111
  supported arguments 6-112
Challenge-Handshake protocol. *See* CHAP
channels
  allocation by protocol 1-3
CHAP 6-137
  ACP logging for 6-139
  challenge
    receiving 6-138
    re-issuing 6-139
    sending 6-138
chap_auth_name parameter 6-138
chap_secret 6-93, 6-137
chat script 3-45 to 3-52
  default global timeout values and 3-50
  default timeout values and 3-50
  examples 3-50 to 3-52
  field definitions 3-47
  reserved keywords 3-49
  string formatting extensions and 3-48
CLI
  See also CLI commands, CLI port parameters
  command syntax definition 2-2
  errors (squelch) 2-2
  masking commands 6-107
  prompt, setting for environment customization 3-10 to 3-11
  protecting 6-5
  virtual (VCLI) connections
    implementing local password protection 6-2
    setting limit on 3-12
CLI commands
  admin superuser command 1-11, 3-7 to 3-10
  arap user command 13-8
  arp superuser command 13-8
  edit superuser command 3-12
  filter superuser command 6-148
  for use with AppleTalk 13-6 to 13-9
  list of user and superuser commands 2-3
  masking 6-107
  preventing access to 6-107
  providing routing information 11-72
CLI sessions
  configuring 5-35
cli_inactivity parameter 3-34
cli_prompt parameter 3-10
cli_security parameter 6-52, 6-55, 6-56
  configuring for dial-in PPP 8-13, 12-14
  configuring for dial-in SLIP 10-10, 10-11
  configuring for dialup PPP 8-12, 12-12
clicmd 6-80
climask 6-81
Closing Member Links 9-6
codes, formatting
  list of, for Annex prompts 3-11
config.annex file 11-47
  editing for SLIP 10-4 to 10-5
  editing pri section for PPP 8-5 to 8-6
config_file parameter 3-12, 3-14
Configuration file
  editing for AppleTalk 13-10
configuration file
  creating dialout entries in 3-39 to 3-52
  creating macro entries in 3-27 to 3-37
  creating rotary entries in 3-39
  creating service entries in 3-37

digital_modem section 7-6, A-1
include statements 3-14
managing macro entries in 3-36
setting up 3-12, 3-15
configuration parameters
Annex
pref_dhcp1_addr 6-142
pref_dhcp2_addr 6-142
asynchronous port
address_origin 6-142
global port
AppleTalk-specific 13-5 to 13-6
arap_v42bis 13-5
at_guest 13-6
at_nodeid 13-6
at_security 13-6
set port 13-5
show port 13-5
RAC
a_router 13-4
default_zone_list 13-4
node_id 13-4
rip_auth 11-56
routed 11-60
zone 13-5
RIP-specific interface
option_key 11-53
rip_accept 11-53
rip_advertise 11-55
rip_auth 11-56
rip_default_route 11-57
rip_horizon 11-57
rip_next_hop 11-58
rip_recv_version 11-58
rip_routers 11-58
rip_send_version 11-59
rip_sub_accept 11-59
rip_sub_advertise 11-59
configuration, typical
PRI 1-5
configuring
IPX, standards-based 12-15
dial-in 12-15
LAT services 3-14
ports
multiple 5-24
RAC 3-12
for use with SecurID 6-114
RAC parameters 3-2
using CLI admin command 3-7 to 3-10
using na 3-3, 3-3 to 3-6
security for RAC FTP daemon 6-152
using RADIUS attributes 6-25
connect command 4-28
connect_security parameter 6-54
configuring for dial-in PPP 8-13, 12-14
configuring for dialup PPP 8-12, 12-12
connection security 6-54
setting up 6-94 to 6-97
connections
IPX protocol 12-24
control_lines parameter
setting for dynamic dialing 5-54
CSLIP
introduction to 10-1
customer support
programs xxv
Technical Solutions Centers xxvi
customizing RAC environment 3-10

D

data_bits parameter
configuring for PPP link 8-16
data–b
slot support for LAT 4-31
daylight_savings parameter 4-14
default route, RIP 11-51
default_zone_list parameter 6-134, 13-4, 13-16
deny
option for acp_userinfo 6-83

DHCP. See *Dynamic Host Configuration Protocol*
Dialed Number Information Service (DNIS) 7-2
dialing, dynamic 5-50
  enabling 5-51
  network inactivity and 5-51
  routes 5-59
  sample configurations 5-55
dial-out
  passwords 3-44
  routes 3-45
dialout entries
  creating in configuration file 3-39 to 3-52
  list of field definitions 3-42
dialup_addresses parameter 8-16
  configuring for dial–in PPP 12-14
  configuring for dial–up PPP 8-12, 12-13
digital modems
  administration of 7-11
  assignment of 7-2
  communication sessions 7-2
  custom configuration 7-6, A-1
  dynamic assignment of 7-2
  making unavailable 7-3
  statistics 7-15
  support 7-1
digital_modem section, config.annex 7-6, A-1
disabled_modules parameter 3-45
disabling
  CLI commands 6-107
displaying
  data for asynchronous PPP ports 8-25
  RAC routing table 11-62
  RIP statistics 11-61
  route cache 11-66
DNIS (Dialed Number Information Service) 7-2
DNS. *See* Domain Name System server
Domain Name System server 4-17
  See also name servers
  example of PTR entry 4-17
  using for RAC configuration 4-16
Dropped 9-22

DS0 channels
automatic busy-out 7-4
dual WAN interfaces 1-2
dump host
file naming 4-8
services 4-8
setting for RAC configuration 4-9
dumpboot command 4-8
dumping
configuring for RAC 4-7 to 4-13
using tftp 4-12
Dynamic Host Configuration Protocol (DHCP) 6-142 to 6-143

E

edit command 3-12, 4-13
enable_radius_acct parameter 6-19
enable_security parameter 3-44, 5-40, 6-1 to 6-4, 6-19, 6-139, 6-152
Enigma. *See* SafeWord
erpcd 4-9, 4-12
and file loading 1-12
as proxy RADIUS client 6-18
recompiling 6-110
using to set up file server 4-5
Ethernet 3-13
event logging
priority levels 4-5
using for RAC configuration 4-3 to 4-5
using host-based security 6-153
exclude filters 6-147

F

file
configuration
creating dialout entries in 3-39 to 3-52
creating gateway entries in 3-19 to 3-21
creating macro entries in 3-27 to 3-37
creating rotary entries in 3-39
creating service entries in 3-37
loading host table from 3-24
managing macro entries in 3-36
sample 3-15 to 3-18
sections 3-15
setting up 3-15 to 3-52
configuration, parsing 3-15
servers
installing on multiple hosts 4-6
setting for RAC configuration 4-7
setting up using bfs 4-5
setting up using tftp 4-6
system, local
for configuring RAC 3-1
files
loading 1-12
filter
option for acp_userinfo 6-84
filter lists 6-150
filter numbers 6-150
filtering
See also NVE filtering
accessing filter subcommands and 6-148
configuring RAC for 6-148
enabling 6-148
filter lists and 6-150
filter numbers and 6-150
filters
include and exclude 6-147
using for security 6-146
Flash ROM
booting RAC from 4-12
Frames discarded 9-22
Frames received 9-22
Frames sent 9-22
FTP daemon, RAC
configuring security for 6-152
using 4-13

G

gateway
defining for preferred load host 4-7
entries
creating in configuration file 3-19 to 3-21
for SLIP link 10-13
routing services and 3-25
supported keywords for 3-20 to 3-22
extensions 3-22 to 3-24
host table loading 3-24
LAT–to–Telnet 4-30
RAC TCP/IP 4-30
Telnet-to-LAT 4-28 to 4-30
Global Port Parameters 13-12
reviewing and resetting for AppleTalk 13-12
global port parameters
changing values of 5-23
defaults for SLIP 10-5, 10-6
displaying 5-19
PPP defaults 8-6 to 8-7
resetting for PPP 8-6, 8-7 to 8-8
reviewing and resetting for SLIP 10-5 to 10-7
group profile criterion 6-64
group_value parameter 4-26, 4-31
groups
creating for security 6-69

H

help command 3-28, 3-36
hop 11-3
host table
built using RWHO messages 4-20
loading from configuration file 3-24
management 4-21
host_table_size parameter 4-21
host-based security
configuring for port server 5-39
for CLI 6-52
for RAC 4-23
for VCLI connections 5-40, 6-53
hosts
accessing 4.2BSD 4-1
configuring 4-1
limiting access to 6-94
multiple server 4-6
Hunt groups
special considerations 7-5

I

ICMP
redirect messages 11-9
router discovery 11-8
IEN-116 name server
See also name servers
using for RAC configuration 4-18
image file 4-10
location of 4-7
image_name parameter 4-7
include filters 6-147
include statement 3-15
inet_addr parameter 4-2
input_flow_control parameter
setting for dynamic dialing 5-54
installing
file server software 6-57
time server software 4-15
Internet
addressing 3-10 to 4-3
Internet Protocol. *See* IP
Internetwork Packet Exchange protocol. *See* IPX protocol
IP
See also RIP
addressing 11-11 to 11-18
Basic Security Option (IPSO)
configuring for RAC security 6-126
encapsulation type
setting 3-13

forwarding versus routing 11-4
routing 11-1 to 11-80
troubleshooting 11-72 to 11-79
IP addresses
assigning B channels for SLIP 10-7
choosing a method for SLIP 10-2
dial-up IP addressing for SLIP 10-3
fixed IP addressing for SLIP 10-3
IP addressing 8-2 to 8-5
choosing an assignment method 8-3
dial-up addressing 8-4
ipencap_type parameter 3-13
IPSO 6-126
IPX protocol 12-1 to 12-25
buffer pools 12-20
configuring standards-based 12-15
information, obtaining 12-15 to 12-25
and statistics for interfaces/ 802.2 12-25
for frame type and network number 12-23
for interfaces, memory buffers, routes (RIPs), and servers 12-18 to 12-23
for IPX connections 12-24
for IPX state 12-24
for memory buffer pools 12-20
using IPXCP interface statistics 12-16
using netstat -x command 12-18
using system logs (syslogging) 12-15
network interfaces 12-19
routes 12-20
servers 12-21
standards-based 12-2
features 12-2
ipx_frame_type parameter 12-24
ISDN connections
modem calls 1-5
synchnronous PPP calls 1-7
V.120 calls 1-6

K

Kerberos user authentication
configuring 6-113
enabling 6-113
introduction to 6-112

L

LAT protocol
services 3-14, 4-23
accessing 4-25
from RAC port 4-27
from VCLI 4-27
advertised 4-24
data–b slot support 4-31
group codes 4-25
LAT–to–Telnet gateway 4-30
learned 4-25
miscellaneous LAT parameters 4-32
restricting access to 4-26
reverse LAT 4-27
reverse LAT vcli 4-27
telnet–to–LAT gateway 4-28 to 4-30
lat_key parameter 3-14
latb_enable parameter 4-31
link control protocol (LCP) 8-18
load host
setting for RAC configuration 4-7
setting up 4-5
load server
setting for RAC configuration 4-10
load_broadcast parameter 3-12, 3-27, 4-11, 4-12
load_dump_gateway parameter 4-7, 4-9
load_dump_sequence parameter 4-10, 4-11
load-dump sequence
setting for RAC configuration 4-10
loading files 1-12
Local Area Transport. *See* LAT protocol
local password protection

configuring for port server 5-39
for RAC 4-23
for virtual CLI (VCLI) connections 6-2
overview 6-2 to 6-6
local_address parameter 8-16
configuring for dial–in PPP 8-14
configuring for dial–up PPP 10-10, 10-12
logging 6-134
security 13-16
ls command 4-11, 4-13

M

macro
alias definitions 3-28
entries
creating in configuration file 3-27 to 3-37
examples of aliases and menus 3-30 to 3-34
managing in configuration file 3-36
supported keywords for 3-29
keyin 3-29
menu definition 3-28
port_set 3-28
resetting 3-37
masking CLI commands 6-107
max_chap_chall_int parameter 6-139
max_vcli parameter 3-12, 5-39
Maximum Transmission Unit (MTU) 11-9
Member Links and Bundle Links 9-1
message-of-the-day
file set up 3-12
resetting 3-12
metric parameter 8-16
min_unique_hostnames parameter 4-15, 4-21
minimum uniqueness
configuring for RAC 4-21
definition of 4-21
Missing mbuf 9-22
MMP
configuring 9-15
connections 9-14
groups 9-14
mmp_enabled parameter 9-12
mode parameter 6-55
setting 5-33
setting for dynamic dialing 5-54
modem -a command 5-54
modem command 7-11
modems
See digital modems

motd_file parameter
configuring 3-13
MP Configuration 9-7
MP Operational Characteristics 9-9
MP Overview 9-1
MP Parameters 9-9
MP Statistics 9-21
mp_endpoint_address 9-9
mp_endpoint_address parameter 9-14
mp_endpoint_class parameter 9-14
mp_max_links 9-11
Multilink Point-to-Point Protocol (MP) 9-1
Multi-system Multilink PPP (MMP) 9-11

N

na
command notation 2-11 to 2-13
commands 2-1
introduction to 2-1
dumpboot command 4-7
protecting from unauthorized access 6-6
using for AppleTalk-specific configuration parameters 13-3
using for RAC configuration 3-3 to 3-6
name servers
See also Domain Name System server, IEN-116 name server
broadcasting for 4-19
configuring 4-22
setting configuration parameters 4-19
using for RAC configuration 4-15 to 4-21
name_server_1 parameter 4-19
name_server_2 parameter 4-19
nameserver_broadcast parameter 4-19
NCP
and AppleTalk over PPP 13-17
protocol stack and 8-18
Net down 9-22
netstat -C command 3-26, 11-6, 11-66
netstat command
RIP interface statistics 11-61
netstat -g command 11-61
netstat -i command 11-11
using to obtain information/statistics for all interfaces and for 802.2 12-25
netstat -ip command 12-16
CLI, using 8-24
netstat -r command 3-25, 5-54, 5-59, 11-7, 11-62
netstat -ri command 11-62
netstat -x command 12-18
using to obtain information for IPX protocol interfaces, memory buffers, routes (RIPs), and servers 12-18 to 12-23
netstat -xi command 12-19
netstat -xm command 12-20
netstat -xr command 12-20
using to display RAC route for network 12-21
netstat -xS command
using to display additional line of information for each server 12-23
netstat -xs command
using to display server names, types, and addresses 12-21
network
interfaces, for IPX 12-19
number, IPX 12-23
Network Control Protocol. *See* NCP
Network File System (NFS) protocol 6-146
network_turnaround parameter
configuring 6-58
Network-Visible Entity (NVE) Filtering. *See* NVE filtering
NIS password file
linking verification to ACP 6-100
No memory 9-22
node_id parameter 13-2, 13-4
Novell networks 12-1
See also IPX protocol
NVE filtering 6-90 to 6-91, 6-134, 13-17

nve_filter entries 13-17

O

operational code
location of 4-7
option_key parameter 11-53
output_flow_control parameter
setting for dynamic dialing 5-54

P

PAP 6-135, 8-16
parity parameter
configuring for PPP link 8-16
parsing
configuration file 3-14
passive RIP. *See* RIP
Password Authentication Protocol. *See* PAP
password files
creating for users 6-72 to 6-74
password histories 6-7
password parameter 6-2, 6-3, 6-54
Path MTU Discovery 11-9
Point–to–point Protocol. *See* PPP
poison reverse 11-50
port handling 5-18
port parameters
for AppleTalk over ARA 13-13
port security 6-6
port server security 6-56
port servers 5-38 to 5-41
configuring security for 5-39, 6-56
port_password parameter 3-44, 6-2, 6-52, 6-57
configuring for dial-in PPP 8-13, 12-14
configuring for dialup PPP 8-12, 12-12
configuring for port server 5-40
port_server_security parameter 5-40, 6-56
configuring for RAC 5-40
ports
changing global defaults for 5-23
differences on other servers 5-37
displaying global parameters for 5-19
numbering of 5-18
types of 5-18
using global parameters for configuring 5-19
ports profile criterion 6-66
PPP 8-1 to 8-24
AppleTalk over 13-17
authentication type 8-22
connecting single host using 8-11, 12-11
connecting to single host using with fixed addresses 8-13, 12-13
connecting two subnets 8-14
link
connecting two subnets 8-15
routing across 8-17
multilink synchronous 1-8
negotiating compression type 8-24
negotiating data compression 8-21
negotiating IP Address 8-23
negotiating LCP for asynchronous PPP 8-18
Network Control Protocol (NCP) and 8-21
overview of operation with the RAC 8-1
PAP and 6-135
protocol stack 8-18 to 8-24
security 6-135
security parameters, using 6-140
synchronous 1-7
ppp command 6-56
PPP Configuration Overview 8-2 to 8-10
assigning IP addresses to B channels 8-9 to 8-10
editing the config.annex file 8-5 to 8-6
IP addressing 8-2
reviewing and resetting global port parameters 8-6 to 8-10
PPP Configuration Samples 8-10 to 8-17
configuration using dial-up addressing 8-10 to 8-12

configuration using fixed IP addressing 8-12 to 8-14
connecting two subnets 8-14 to 8-17
PPP Statistics 9-22
ppp_acm parameter
configuring for dial-in PPP 8-14, 12-14
ppp_mru parameter 8-12, 12-13
configuring for dial-in PPP 8-14, 12-14
ppp_ncp
mp_mrru
mp_endpoint_class 9-9
ppp_ncp parameter 8-21
ppp_password_remote parameter 3-44, 6-2, 6-136, 6-138, 8-12, 8-16, 12-12
ppp_security_protocol parameter 6-139, 8-16
configuring for dial–in PPP 8-14, 12-14
configuring for dial–up PPP 8-12, 12-12
ppp_username_remote parameter 6-136, 8-12, 8-16, 12-12
pref_dhcp1_addr parameter 6-142
pref_dhcp2_addr parameter 6-142
pref_dump_addr parameter 4-9
pref_load_addr parameter 4-7
pref_name1_addr parameter 4-19
pref_name2_addr parameter 4-19
pref_secure1_host parameter 6-19, 6-58
pref_secure2_host parameter 6-19, 6-58
preferred dump host
setting for RAC configuration 4-9
preferred load host
setting for RAC configuration 4-7
PRI
typical configurations 1-5
PRI interface 1-2
profile criteria
group criterion 6-64
overview of 6-61
ports criterion 6-66
protocol criterion 6-67
syntax 6-64
time criterion 6-65
username criterion 6-64
prompt parameter 3-10

prompts
Annex
list of formatting codes for 3-11
CLI
setting for environment customization 3-10 to 3-11
protocol profile criterion 6-67
protocol stack 8-18 to 8-24
protocol support 1-4
Proxy ARP 11-19
using with SLIP 10-14
PSNDN 9-9
publications, ordering xxv

R

RAC
allocation of channels by protocol 1-3
CAS interface 1-3
dual WAN interfaces 1-2
overview 1-1
PRI interface 1-2
protocol support 1-4
RADIUS
configuring RAC functions 6-25
native client 6-18
Overview 6-17
parameters 6-18
supported attributes 6-20
RADIUS Dictionary File 6-23
radius_acct_level parameter 6-19
radius_acct_port parameter 6-19
radius_auth_port parameter 6-19
radius_port_encoding parameter 6-19
radius_retries parameter 6-19
radius_secret parameter 6-19
radius_timeout parameter 6-19
read command
using for port configuration 5-24
redirect messages 11-9
remote_address parameter 10-3
reset annex dialout command 3-45, 5-54
reset annex macros command 3-27, 3-37
reset annex motd command 3-12
reset annex security command 6-60
RIP 11-1 to 11-80
active RIP 3-13
advertising routes 11-48
advertising subnet routes 11-49
advertising the default route 11-51
advertising to subset of routers 11-52
configuring 11-47 to 11-52
defining routes 11-47
when to choose 11-4
authenticating incoming RIP 2 packets 11-44
authenticating outgoing packets 11-51
basic passive 8-17
broadcast address, setting 11-20
configuration parameters 11-22 to 11-23, 11-53 to 11-60
default configuration 11-48
default route 11-51
defining routes 11-27 to 11-42
disabling 3-26
displaying routing information 11-60 to 11-72
enabling 5-51
entering routes in config.annex 11-28
entering routes using the route command 11-39
hop 11-3
ICMP redirect messages 11-9
interface routes 11-10
IP addressing 11-11 to 11-18
address classes 11-12
obtaining IP network addresses 11-13
setting the RAC IP address 11-14
netstat -r option 11-62
non-operational interfaces 11-11
passive RIP

configuring 11-27 to 11-45
defining routes 11-27 to 11-42
disabling 11-26
using subnet masks 11-15 to 11-18
when to choose 11-4
Path MTU discovery 11-9
ping –t option 11-67
poison reverse 11-50
Proxy ARP 11-18
RIP 1 and 2 packets, accepting 11-42
rip_advertise parameter 11-55
rip_auth parameter 11-56
rip_default_route parameter 11-57
rip_horizon parameter 11-57
rip_next_hop parameter 11-58
rip_recv_version parameter 11-58
rip_routers parameter 11-58
rip_send_version parameter 11-59
rip_sub_accept parameter 11-59
rip_sub_advertise parameter 11-59
routed parameter 11-60
routing interfaces 11-10
routing table, displaying 11-62
supernetting 11-17
troubleshooting 11-72 to 11-79
using for environment customization 3-13
using subnet masks 11-15 to 11-18
versions 11-5, 11-48
rip_accept parameter 11-53
rip_advertise parameter 11-55
rip_auth parameter 11-43, 11-56
rip_default_route parameter 11-52, 11-57
rip_horizon parameter 11-57
rip_next_hop parameter 11-58
rip_recv_version parameter 11-43, 11-58
rip_routers parameter 11-52, 11-58
rip_send_version parameter 11-48, 11-59
rip_sub_accept parameter 11-59
rip_sub_advertise parameter 11-47, 11-49, 11-59
rlogin
connection requests 5-38
using to access 4.2BSD hosts 4-1
rlogin command 5-38
ROM monitor prompt 4-12
rotaries 5-41 to 5-49
configuring 5-41 to 5-48
assigning a phone number 5-47
assigning auxiliary Internet addresses 5-43
binary 5-46
defining multiple rotaries with one entry 5-42
port selection 5-47
protocol 5-46
raw 5-46
rlogin 5-46
telnet 5-46
using DNS server to define multiple rotaries 5-44
visibility 5-45
definition of 5-41
entries 5-41
creating in configuration file 3-39
route
cache 3-25, 8-18, 11-5
displaying 11-66
route command 3-25, 11-66
routed parameter 3-13, 3-25, 11-60
disabling RIP 3-26
router discovery 11-8
routes
default to gateways 3-26
defining gateways 3-22
definition of 11-3
dial-out 3-45
IPX 12-20
routing
See also RIP
across PPP link (basic passive RIP) 8-17

services
gateway entries and 3-25
table 11-7
displaying RAC 11-62
RWHO protocol 4-19
rwhod parameter 4-15, 4-19

S

SafeWord 6-119 to 6-126
backup security 6-126
installing 6-120
integrating into ACP 6-122
passwords 6-124
dynamic 6-125
fixed 6-124
security
backup 6-126
using 6-119 to 6-126
Sample 13-13
SecurID 6-114
backup security 6-118
card
assigning to user 6-128
PIN mode 6-132
user interface 6-129
clients 6-129
configuring RAC for 6-114
installing ACE/Server software and 4-23
integrating into ACP 6-115 to 6-118
security
See also ACP security, SafeWord, SecurID
ACP 6-50 to 6-57
changing ACP file names 6-102
creating acp_dialup file 6-142
creating acp_userinfo file 6-74 to 6-93
acp_dbm database 6-14
AppleTalk 6-133 to 6-134
application, modifying 6-99 to 6-110
ARA 6-133, 13-15 to 13-16
blacklisting
configuring 6-12
error handling 6-16
overview of 6-11
connection, setting up 6-54, 6-94 to 6-97
customizing policy 6-99, 6-110
disabling user name and password validation 6-100
event logging 6-153
for dial-in PPP 8-13, 12-14
for dialup PPP 8-12, 12-12
for port server 5-39
for ports 6-6
for VCLI connections 5-40
host-based 6-50
hosts, specifying 6-58
IP Basic Security Option (IPSO) 6-126
IPX over PPP 6-135
local password protection 6-2 to 6-6
masking CLI commands 6-107
messages, encrypting 6-59 to 6-60
modifying source code 6-109
password histories
configuring 6-8
error handling 6-16
overview of 6-7
with password aging 6-7
port server 6-56
PPP 6-135 to 6-141
profile criteria
overview of 6-61
syntax 6-64
profiles, defining 6-61 to 6-98
RAC 4-23
configuring for FTP daemon 6-152
re-compiling erpcd 6-110
SafeWord 6-119 to 6-126
SecurID 6-50, 6-114, 6-128
server
configuring for RAC 6-57 to 6-58
disabling broadcasting for 6-58

setting up ACP encryption key 6-59
SLIP and PPP 6-55
using filters for 6-146
using Kerberos authentication for 6-113
security profiles
configuring /etc/group file 6-69
configuring acp_regime file 6-70
configuring acp_restrict file 6-94
creating acp_group file 6-69
creating acp_userinfo file 6-74
creating user password files 6-72
defining 6-61 to 6-98
files used for, overview 6-68
security profiles, groups, creating for 6-69
security_broadcast parameter
configuring 6-58
self-booting 4-11
See also booting
sequence command 4-12
Serial Line Internet Protocol. *See* SLIP
server_capability parameter 4-10
arguments 4-11
server_name parameter 4-28
servers
See also name servers, Domain Name System server, IEN-116 name server
BIND 4-17
configuring 4-1
Domain Name System (DNS) 4-17
file
installing on multiple hosts 4-6
setting up using bfs 4-5
setting up using tftp 4-6
IPX 12-21
name, configuring 4-22
port, security for 6-56
service entries
creating in configuration file 3-37
services command 4-27, 4-28
Session Parameter Blocks
configuring 5-25
editing for AppleTalk 13-10
field definitions for 5-27
fields in 5-26
samples of 5-32
sections in 5-25

sessions
CLI
configuring 5-35
WAN
configuring 5-25
definition of 5-24
set annex command
for configuring RAC parameters 3-2
set port command 13-5
Short 9-22
show annex command
for configuring RAC parameters 3-2
show port command 13-5
SLIP 10-1 to 10-15
BOOTP requests 10-15
compressed (CSLIP) 10-1
configuring for host 10-9
extending single host to network using Proxy-ARP 10-14
link
routing across 10-12 to 10-14
route cache 10-14
routing between two networks 10-13
using gateways file 10-13
using Proxy-ARP with 10-14
SLIP and PPP security 6-55
slip command 6-56
using in chat scripts 3-45
SLIP Configuration Overview 10-2 to 10-8
editing the config.annex file 10-4
handling IP addressing 10-2
major steps 10-2
resetting global port parameters 10-5
SLIP Configuration Samples 10-9 to 10-12
connecting a single device 10-9 to 10-10
connecting two subnets 10-10
SLIP Overview for the RAC 10-1
slip_ppp_security parameter 6-138
configuring for a PPP link 8-12, 12-12
configuring for dial-in PPP 8-13, 12-14
SPBs. See Session Parameter Blocks
speed parameter
setting for dynamic dialing 5-54
squelch 2-2
statistics
for interfaces and 802.2 data-link layer 12-25
stats command 13-4
using to obtain information
for IPX protocol frame type 12-23
for IPX protocol network number 12-23
for IPX protocol state 12-24
stop_bits parameter
configuring for PPP link 8-16
stty attn command 5-39
su command 2-3
subnet
mask
configuring for RAC 4-2
routes
entering 11-38
subnet...end blocks 3-23, 8-18, 11-38
route cache and 10-14
using to configure active RIP 11-47
subnet_mask parameter 4-2, 11-16
Supported MP Features 9-3
syslog daemon 6-2, 13-16
event logging and 4-4
file 6-146
syslog_mask parameter 3-23, 4-4
syslogging 12-15
4.3BSD, setting up host for 4-2
configuring for RAC 4-3 to 4-5

T

TCP port numbers
defining
for telnet and rlogin 5-38
with port server 5-38
Technical Solutions Centers xxvi
telnet 4-28

connection requests 5-38
telnet command 4-20, 5-38
Telnet-to-LAT gateway 4-28 to 4-30
tftp 4-8, 6-146
booting/dumping and 4-7
file loading and 1-12
using for booting and dumping 4-12
tftp_dump_name parameter 4-9
The acp_userinfo File 9-11
The MP Process 9-4
time profile criterion 6-65
time server
enabling broadcasting for 4-14
installing 4-15
using for RAC configuration 4-13
time_broadcast parameter 4-14
timezone_minuteswest parameter 4-14
traceroute (ping -t) option 11-67 to 11-72
Trivial File Transfer Protocol. *See* tftp
troubleshooting
for Internet Protocol (IP) routing 11-72 to 11-79
tuple
in nve_filter entries 6-90 to 6-91
type parameter
setting for dynamic dialing 5-54
type_of_modem parameter
setting for dynamic dialing 5-54

U

Unknown codes 9-22
Unsupported MP Features 9-4
user validation
disabling 6-100
for port servers 5-39
on VCLI connections 5-40
username profile criterion 6-64
Using RADIUS Security 6-152
Using the ACE/Server 6-127
utilities
ch_passwd 6-111

V

VCLI connections
configuring for host-based security 5-40, 6-53
configuring for local password protection 5-40, 6-2
for port servers 5-38
setting limit on 3-12
vcli_groups parameter 4-27
vcli_password parameter 6-2 to 6-4, 6-54
configuring security for 6-2
security for VCLI connections 5-41
vcli_security parameter 6-53
security for VCLI connections 5-40
virtual CLI. *See* VCLI connections
VMS
environments
privileged VMS commands 2-7

W

WAN
sessions 5-24
wan commands
wan b 10-3, 10-7, 10-8
WAN interfaces
configuring 5-1
who command 6-152
using to obtain information for IPX protocol connections 12-24
write command
using for port configuration 5-24

Z

zone parameter 13-2, 13-5
zone security 6-134, 13-16