



Cisco Broadband Access Center for Cable Administrator's Guide

Release 2.7

Corporate Headquarters

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
<http://www.cisco.com>
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

Text Part Number: OL-4409-01



THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

CCSP, CCVP, the Cisco Square Bridge logo, Follow Me Browsing, and StackWise are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn, and iQuick Study are service marks of Cisco Systems, Inc.; and Access Registrar, Aironet, ASIST, BPX, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unity, Empowering the Internet Generation, Enterprise/Solver, EtherChannel, EtherFast, EtherSwitch, Fast Step, FormShare, GigaDrive, GigaStack, HomeLink, Internet Quotient, IOS, IP/TV, iQ Expertise, the iQ logo, iQ Net Readiness Scorecard, LightStream, Linksys, MeetingPlace, MGX, the Networkers logo, Networking Academy, Network Registrar, *Packet*, PIX, Post-Routing, Pre-Routing, ProConnect, RateMUX, ScriptShare, SlideCast, SMARTnet, StrataView Plus, TeleRouter, The Fastest Way to Increase Your Internet Quotient, and TransPath are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or Website are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0502R)

Cisco Broadband Access Center for Cable Administrator's Guide
Copyright © 2002 - 2005 Cisco Systems, Inc.
All rights reserved.



Preface	xix
Audience	xix
How This Guide Is Organized	xx
Conventions	xxi
Related Documentation	xxi
Obtaining Documentation	xxii
Cisco.com	xxii
Product Documentation DVD	xxii
Ordering Documentation	xxiii
Documentation Feedback	xxiii
Cisco Product Security Overview	xxiii
Reporting Security Problems in Cisco Products	xxiv
Obtaining Technical Assistance	xxiv
Cisco Technical Support & Documentation Website	xxiv
Submitting a Service Request	xxv
Definitions of Service Request Severity	xxv
Obtaining Additional Publications and Information	xxvi

CHAPTER 1

Broadband Access Center for Cable Overview	1-1
Features and Benefits	1-1
Supported Technologies	1-1
DOCSIS High-Speed Data	1-2
PacketCable Voice Services	1-2
Non-Secure PacketCable Voice Services	1-2
Euro-PacketCable Voice Services	1-2
Non-Secure CableHome Provisioning	1-2

CHAPTER 2

Broadband Access Center for Cable System Architecture	2-1
Architecture	2-1
Registration Modes	2-2
Standard Mode	2-2
Promiscuous Mode	2-2

- Roaming Mode 2-2
- Mixed Mode 2-2
- Regional Distribution Unit 2-3
 - Generating Device Configurations 2-3
 - Service Level Selection 2-3
 - Regional Distribution Unit Failover 2-4
- Device Provisioning Engines 2-4
 - Types of Device Provisioning Engine 2-4
 - Hardware Device Provisioning Engines 2-5
 - Solaris Device Provisioning Engines 2-5
 - DPE License Keys 2-5
 - TACACS+ and DPE Authentication 2-6
 - TACACS+ Privilege Levels 2-6
 - TACACS+ Client Settings 2-6
 - DPE Server Assignments 2-7
 - TFTP Server 2-7
 - DOCSIS Shared Secret 2-7
 - Resetting the DOCSIS Shared Secret 2-8
- Provisioning Groups 2-8
- Cisco Network Registrar 2-9
 - Dynamic Host Configuration Protocol 2-9
 - Domain Name System 2-9
 - Lease Reservation 2-10
- Key Distribution Center 2-10
 - Default KDC Properties 2-10
 - Euro-PacketCable Support 2-12
 - KDC Certificates 2-12
 - KDC Licenses 2-12
 - Multiple Realm Support 2-13
- BACC MIBs 2-13
- BACC Agents 2-14
 - SNMP Agent 2-14
 - MIB Support 2-14
 - BACC Agent 2-15
 - Monitored Processes 2-15
 - BACC Agent Command Line 2-16
- Logging 2-16
 - Regional Distribution Unit Logs 2-17
 - Device Provisioning Engine Logs 2-17

Log Levels and Structures	2-17
Configuring Log Levels	2-18
Viewing the dpe.log File	2-19
Administrator's User Interface	2-19
Sample User Interface	2-19

CHAPTER 3**Configuration Workflows and Checklists 3-1**

Component Workflows	3-1
RDU Checklist	3-2
Hardware DPE Checklist	3-2
Solaris DPE Checklist	3-4
Network Registrar Checklist	3-6
Technology Workflows	3-7
DOCSIS Checklist	3-7
PacketCable Checklists	3-8
PacketCable	3-8
Non-Secure PacketCable	3-10
Euro-PacketCable	3-12
Non-Secure CableHome Provisioning Checklist	3-14

CHAPTER 4**DOCSIS Configuration 4-1**

DOCSIS Workflow	4-2
Using MIBs with Dynamic DOCSIS Templates	4-4
BACC Features for DOCSIS Configurations	4-5
Dynamic Configuration TLVs	4-5
DPE TFTP IP Validation	4-5
.DOCSIS 1.0, 1.1, 2.0 Support	4-6
DOCSIS Configuration File Based on DOCSIS Version	4-6
Troubleshooting DOCSIS Networks	4-6
Network Layer Configuration and Above	4-7
Establish DHCP State	4-7
Establishing the ToD State	4-8
Security Association State	4-8
Configuration File State	4-9
Registration State	4-9
Establish Privacy State	4-9
Operational State	4-10

- Troubleshooting Cable Modem States 4-10
 - Online State 4-10
 - Offline State 4-11
 - Ranging Process - init(r1),init(r2), and init(rc) state 4-16
 - DHCP - init(d) state 4-18
 - DHCP - init(i) state 4-20
 - TOD exchange- init(t) state 4-23
 - Option File Transfer Started-init(o) State 4-25
 - Online, Online(d), Online(pk), Online(pt) state 4-27
 - Reject(pk) and Reject(pt) state 4-30
 - Registration - reject (m) state 4-31
 - Registration - reject (c) state 4-32

CHAPTER 5

- PacketCable Voice Configuration 5-1**
 - PacketCable EMTA Secure Provisioning 5-1
 - BACC PacketCable Secure Provisioning Flow 5-1
 - PacketCable EMTA BASIC Provisioning 5-4
 - PacketCable TLV 38 and MIB Support 5-5
 - SNMP v2C Notifications 5-5
 - Euro-PacketCable 5-5
 - Euro-PacketCable MIBs 5-5
 - Configuring Euro-PacketCable MIBs 5-6
 - Troubleshooting eMTA Provisioning 5-6
 - Components 5-6
 - Embedded Media Terminal Adapter 5-7
 - DHCP Server 5-7
 - DNS Server 5-7
 - Key Distribution Center 5-8
 - PacketCable Provisioning Server 5-8
 - Call Management Server 5-8
 - Key Variables 5-8
 - Certificates 5-8
 - Scope Selection Tag(s) 5-9
 - MTA Configuration File 5-9
 - Troubleshooting Tools 5-9
 - Logs 5-10
 - Ethereal, SnifferPro, or Other Packet Capture Tools 5-10
 - Troubleshooting Scenarios 5-10

Certificate Trust Hierarchy	5-14
Certificate Validation	5-15
MTA Device Certificate Hierarchy	5-15
MTA Root Certificate	5-16
MTA Manufacturer Certificate	5-16
MTA Device Certificate	5-17
MTA Manufacturer Code Verification Certificates	5-18
CableLabs Service Provider Certificate Hierarchy	5-18
CableLabs Service Provider Root Certificate	5-19
Service Provider CA Certificate	5-19
Local System CA Certificates	5-20
Operational Ancillary Certificates	5-21
Certificate Revocation	5-24
Code Verification Certificate Hierarchy	5-24
Common CVC Requirements	5-24
CableLabs Code Verification Root CA Certificate	5-25
CableLabs Code Verification CA Certificate	5-25
Manufacturer Code Verification Certificate	5-26
Service Provider Code Verification Certificate	5-26
Certificate Revocation Lists for CVCs	5-27

CHAPTER 6**CableHome Configuration 6-1**

Non-Secure CableHome Provisioning Flow	6-1
Configuring CableHome	6-2
Configuring Network Registrar	6-2
Configuring the RDU	6-2
Configuring CableHome WAN-Man	6-2
Configuring CableHome WAN-Data	6-3
Configuring the Device Provisioning Engine	6-3

CHAPTER 7**Database Management 7-1**

Understanding Failure Resiliency	7-1
Database Files	7-1
Database Storage File	7-2
Database Transaction Log Files	7-2
Automatic Log Management	7-2
Miscellaneous Database Files	7-3
Disk Space Requirements	7-3
Handling Out of Disk Space Conditions	7-3

- Backup and Recovery 7-4
 - Database Backup 7-4
 - Database Recovery 7-5
 - Database Restore 7-5
- Changing Database Location 7-6
- RDU Database Migration 7-7

CHAPTER 8

Understanding the Administrator’s User Interface 8-1

- Accessing the BACC Administrators Graphical User Interface 8-1
 - Logging In 8-1
 - Logging Out 8-3
- Administrator’s User Interface 8-4
- Selecting Navigation Bar Items 8-4
- The Main Menu 8-5
 - Configuration 8-5
 - Devices 8-5
 - Servers 8-6
 - Nodes 8-6
 - Users 8-6
 - Administrator 8-6
 - Read/Write User 8-6
 - Read Only User 8-6
- Scrolling Backward and Forward 8-7

CHAPTER 9

Using the Broadband Access Center for Cable Administrator User Interface 9-1

- User Management 9-1
 - Adding a New User 9-2
 - Modifying Users 9-3
 - Deleting Users 9-3
- Device Management 9-3
 - Manage Devices Page 9-4
 - Device Management Controls 9-5
 - Searching for Devices 9-6
 - Search Types 9-7
 - Viewing Device Details 9-8
 - Managing Devices 9-12
 - Adding Devices 9-13
 - Modifying Devices 9-13

- Deleting Devices 9-14
- Regenerating Device Configurations 9-14
- Relating and Unrelating Devices 9-15
- Resetting Devices 9-15
- Unregistering a Device 9-15
- Node Management 9-16
 - Managing Node Types 9-16
 - Adding a Node Type 9-16
 - Modifying Node Types 9-17
 - Deleting Node Types 9-17
 - Managing Nodes 9-17
 - Adding a New Node 9-17
 - Modifying a Node 9-18
 - Deleting Nodes 9-18
 - Relating/Unrelating Node Types to Nodes 9-18
 - Viewing Node Details 9-18
- Viewing Servers 9-18
 - Listing Device Provisioning Engines 9-19
 - Listing Network Registrar Extension Points 9-21
 - Listing Provisioning Groups 9-23
 - Viewing Regional Distribution Unit Details 9-24

CHAPTER 10

Configuring Broadband Access Center for Cable 10-1

- Configuring the Class of Service 10-1
 - Adding a Class of Service 10-3
 - Modifying a Class of Service 10-4
 - Deleting a Class of Service 10-5
- Configuring Custom Properties 10-6
- Configuring Defaults 10-7
 - Selecting Configuration Options 10-7
 - ATA 186 Defaults 10-7
 - ATA 188 Defaults 10-9
 - CableHome WAN Defaults 10-9
 - CableHome WAN Data Defaults 10-10
 - CableHome WAN-Man Defaults 10-11
 - Computer Defaults 10-12
 - DOCSIS Defaults 10-13
 - Network Registrar Defaults 10-15
 - PacketCable Defaults 10-17

- RDU Defaults 10-19
- System Defaults 10-21
- Gateway (xGCP) Control Protocol Defaults 10-22
- Configuring DHCP Criteria 10-23
 - Adding DHCP Criteria 10-23
 - Modifying DHCP Criteria 10-24
 - Deleting DHCP Criteria 10-24
- Managing External Files 10-25
 - Adding External Files 10-26
 - Viewing External Files 10-26
 - Replacing External Files 10-28
 - Exporting External Files 10-28
 - Deleting External Files 10-29
- Managing License Keys 10-29
 - Adding and Modifying a License 10-30
- Managing Regional Distribution Unit Extensions 10-31
 - Writing a New Class 10-31
 - Installing RDU Custom Extension Points 10-31
 - Viewing RDU Extensions 10-32
- Publishing Provisioning Data 10-32
 - Publishing Datastore Changes 10-33
 - Modifying Publishing Plug-In Settings 10-33
- Configuring the SRV Record in the Network Registrar DNS Server 10-34
- Configuring SNMPV3 Cloning on the RDU and DPE for Secure Communication with PacketCable MTAs 10-34
 - Creating the Key Material and Generating the Key 10-34
- Automatic FQDN Generation 10-35
 - Automatically Generated FQDN Format 10-35
 - Properties for Automatically Generated FQDNs 10-36
 - FQDN Validation 10-36
 - Sample Automatic FQDN Generation 10-36

CHAPTER 11

Configuring and Using the Sample User Interface 11-1

- What is the Sample User Interface? 11-1
- Starting and Stopping the Sample User Interface 11-2
- Sample User Interface Configuration Options 11-2
 - Classes of Service 11-2
 - Promiscuous Mode 11-2

Selecting an Internet Service Provider	11-3
Using the Technician Login	11-3
Administrative Access Levels	11-3
Subscriber Provisioning Examples	11-3
Standard Customer Premise Equipment Registration	11-4
Provisioning a New Cable Modem and a New Computer	11-4
Provisioning a New Computer with an Existing Cable Modem	11-4
Altering an Existing Computer ISP	11-5
Promiscuous Customer Premises Equipment Registration	11-5
Provisioning a New Cable Modem and a New Computer	11-5
Provisioning an Existing Cable Modem and a New Computer	11-6
Administrator Provisioning Examples	11-6
Searching for Accounts	11-6
Searching by Account Number	11-6
Searching by IP Address	11-6
Searching by MAC Address	11-7
Account Maintenance	11-7
Registering a New Account	11-7
Managing Classes of Service	11-7
Managing Cable Modems	11-8
Managing Computers	11-8
Deleting an Account	11-9
Sample sampleui.properties File	11-9

CHAPTER 12**Broadband Access Center for Cable Support Tools and Advanced Concepts 12-1**

Developing Template Files	12-1
Template Grammar	12-2
Comments	12-3
Includes	12-3
Options	12-3
Instance Modifier	12-5
SNMP Varbind	12-5
DOCSIS MIBs	12-5
PacketCable MIBs	12-6
CableHome MIBs	12-6
Macro Variables	12-7
Encoding Types for Defined Options	12-8
BITS Value Syntax	12-10
OCTETSTRING Syntax	12-10

Adding SNMP TLVs Without a MIB	12-10
DOCSIS Option Support	12-12
PacketCable Option Support	12-23
Non-Secure CableHome Option Support	12-24
Using the Configuration File Utility	12-25
Running the Configuration File Utility	12-26
Using the Configuration File Utility	12-27
Converting a Binary File Into a Template File	12-28
Parsing a Local Template File	12-28
Parsing an External Template File	12-29
Parsing a Template File and Adding a User-Specified Shared Secret	12-31
Specifying Macro Variables at the Command Line	12-32
Specifying a Device for Macro Variables	12-33
Specifying Output to a Binary File	12-35
Viewing a Local Binary File	12-36
Viewing an External Binary File	12-37
Activating PacketCable BASIC Flow	12-38
The RDU Log Level Tool	12-38
Using the RDU Log Level Tool	12-39
Setting the RDU Log Level	12-40
Viewing the RDU's Current Log Level	12-40
Using the PKCert.sh Tool to Manage KDC Certificates	12-41
Running the PKCert Tool	12-41
Creating a KDC Certificate	12-42
Validating the KDC Certificates	12-43
Using the changeNRProperties.sh Tool	12-43
Using the Keygen Tool	12-45
Using the snmpAgentCfgUtil.sh Command	12-46
Adding a Host	12-47
Deleting a Host	12-47
Adding an SNMP Agent Community	12-48
Deleting an SNMP Agent Community	12-48
Starting the SNMP Agent	12-49
Stopping the SNMP Agent	12-49
Configuring an SNMP Agent Listening Port	12-49
Changing the SNMP Agent Location	12-50
Setting Up SNMP Contacts	12-50
Listing SNMP Agent Settings	12-50
Specifying SNMP Notification Types	12-51

Using the disk_monitor.sh Tool to Monitor Available Disk Space 12-51

Troubleshooting Devices by MAC Address 12-52

APPENDIX A
Alert and Error Messages A-1

- Alert Messages A-1
 - Message Format A-1
 - RDU Alerts A-2
 - Solaris DPE Alerts A-2
 - Agent Alerts A-4
 - Network Registrar Extension Point Alerts A-5
- RDU Error Messages with CCM A-6
 - [OBJECT_EXISTS] A-7
 - [RemoveReservation: NOT_FOUND] A-7
 - [AddReservation: INVALID_PARENT] A-7
 - [AddReservation: INVALID_SECOND_PARENT] A-7
 - [AddReservation: FORWARD_FAILED] A-8
 - Example Case #1 A-8
 - Example Case #2 A-8
 - [AddReservation: AX_ETIME] A-8
 - [AddReservation: INVALID_OBJECT] A-9
 - Selection-criteria exclusion tags will be ignored A-9
 - [AX_EIO] A-9
 - [AX_EPIPE] A-9

APPENDIX B
PacketCable DHCP Options to BACC Properties Mapping B-1

- Option 122 and BACC Property Comparison B-2
- Option 177 and BACC Property Comparison B-2

APPENDIX C
Broadband Access Center for Cable Application Programming Interface Use Cases C-1

- Use Cases C-1
 - Self-Provisioned Modem and Computer in Fixed Standard Mode C-2
 - Adding a New Computer in Fixed Standard Mode C-5
 - Disabling a Subscriber C-7
 - Pre-Provisioning Modems/Self-Provisioned Computers C-9
 - Modifying an Existing Modem C-10
 - Unregistering and Deleting a Subscriber's Devices C-11
 - Self-Provisioning First-Time Activation in Promiscuous Mode C-14
 - Bulk Provisioning 100 Modems in Promiscuous Mode C-16

Pre-Provisioning First-Time Activation in Promiscuous Mode **C-18**

Replacing an Existing Modem **C-19**

Adding a Second Computer in Promiscuous Mode **C-20**

Self-Provisioning First-Time Activation with NAT **C-20**

Adding a New Computer Behind a Modem with NAT **C-22**

Move Device to Another DHCP Scope **C-23**

Log Device Deletions Using Events **C-24**

Monitoring an RDU Connection Using Events **C-25**

Logging Batch Completions Using Events **C-26**

Getting Detailed Device Information **C-26**

Searching Using the Default Class of Service **C-27**

Retrieving Devices Matching a Vendor Prefix **C-29**

Pre-Provisioning PacketCable eMTA **C-31**

SNMP Cloning on PacketCable eMTA **C-33**

Incremental Provisioning of PacketCable eMTA **C-34**

Pre-Provisioning DOCSIS Modems with Dynamic Configuration Files **C-36**

Optimistic Locking **C-38**

Temporarily Throttling a Subscriber's Bandwidth **C-40**

Pre-Provisioning CableHome WAN-Man **C-41**

CableHome with Firewall Configuration **C-42**

Retrieving Device Capabilities for CableHome WAN-Man **C-44**

Self-Provisioning CableHome WAN-Man **C-46**

Lease Reservation Use Cases **C-48**

 API Calls Affected by the Lease Reservation Feature **C-49**

 Bringing a Device Online Using a Service Provider IP Address **C-49**

 Removing and Recreating a Reservation **C-50**

 Assigning a New Device with an Old Device's IP Address **C-51**

 Removing a Reservation and Assigning a New IP Address **C-52**

 Rebooting a Device with the Same IP Address **C-53**

 Removing a Device from BACC **C-54**

 A Submitted Batch Fails when BACC Uses CCM **C-55**

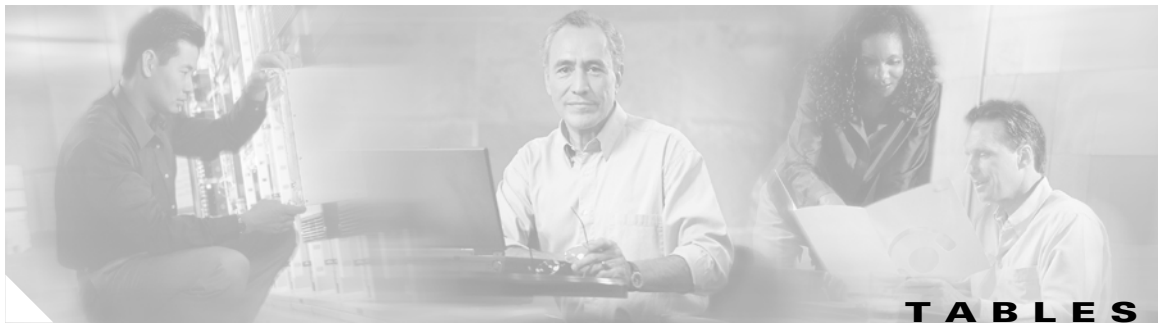
 A Submitted Batch Fails when BACC does not Use CCM **C-55**

GLOSSARY

INDEX



<i>Figure 4-1</i>	DOCSIS Provisioning Flow	4-2
<i>Figure 5-1</i>	Embedded-MTA Secure Power-on Provisioning Flow	5-2
<i>Figure 5-2</i>	PacketCable Certificate Hierarchy	5-14
<i>Figure 6-1</i>	Non-Secure CableHome Flows	6-1
<i>Figure 8-1</i>	Login Page	8-2
<i>Figure 8-2</i>	Main Menu Page	8-3
<i>Figure 8-3</i>	Administrator User Interface	8-4
<i>Figure 9-1</i>	Example Manage Users Page	9-2
<i>Figure 9-2</i>	Manage Devices Page	9-4
<i>Figure 9-3</i>	Device Details Page	9-9
<i>Figure 9-4</i>	Manage Nodes Page	9-16
<i>Figure 9-5</i>	View Device Provisioning Engine Details Page	9-19
<i>Figure 9-6</i>	View Cisco Network Registrar Details Page	9-22
<i>Figure 9-7</i>	View Provisioning Group Details Page	9-23
<i>Figure 9-8</i>	View Regional Distribution Unit Details Page	9-24
<i>Figure 10-1</i>	Manage Class of Service Page	10-2
<i>Figure 10-2</i>	Configure ATA 186 Defaults Page	10-8
<i>Figure 10-3</i>	Configure CableHome WAN-Data Defaults Page	10-10
<i>Figure 10-4</i>	Configure CableHome WAN-Man Defaults Page	10-11
<i>Figure 10-5</i>	Configure Computer Defaults Page	10-12
<i>Figure 10-6</i>	Configure DOCSIS Defaults Page	10-13
<i>Figure 10-7</i>	Configure Network Registrar Defaults Page	10-15
<i>Figure 10-8</i>	Configure PacketCable (Voice Technology) Defaults Page	10-17
<i>Figure 10-9</i>	Configure RDU Defaults Page	10-19
<i>Figure 10-10</i>	System Defaults Page	10-21
<i>Figure 10-11</i>	Configure XGCP Page	10-22
<i>Figure 10-12</i>	Manage External Files Page	10-25
<i>Figure 10-13</i>	Example Binary File Content	10-27
<i>Figure 10-14</i>	Example Jar File Content	10-27
<i>Figure 10-15</i>	Manage License Keys Page	10-30
<i>Figure 10-16</i>	Manage Publishing Page	10-32



<i>Table 1</i>	Document Chapters	xx
<i>Table 2-1</i>	TACACS+ Service Levels	2-6
<i>Table 2-2</i>	BACC Supported MIBs	2-14
<i>Table 2-3</i>	BACC Command Line Interface	2-16
<i>Table 3-1</i>	RDU Workflow Checklist	3-2
<i>Table 3-2</i>	Hardware DPE Configuration Checklist	3-2
<i>Table 3-3</i>	Solaris DPE Configuration Checklist	3-4
<i>Table 3-4</i>	Network Registrar Workflow Checklist	3-6
<i>Table 3-5</i>	DOCSIS Checklist	3-7
<i>Table 3-6</i>	PacketCable Checklist	3-8
<i>Table 3-7</i>	Non-Secure PacketCable Checklist	3-11
<i>Table 3-8</i>	Euro-PacketCable Checklist	3-12
<i>Table 3-9</i>	Non-secure CableHome Provisioning Checklist	3-14
<i>Table 4-1</i>	DOCSIS Workflow Description	4-3
<i>Table 4-2</i>	State Values	4-11
<i>Table 5-1</i>	Troubleshooting Scenarios	5-10
<i>Table 5-2</i>	MTA Root Certificate	5-16
<i>Table 5-3</i>	MTA Manufacturer Certificates	5-17
<i>Table 5-4</i>	MTA Device Certificates	5-18
<i>Table 5-5</i>	CableLabs Service Provider Root Certificates	5-19
<i>Table 5-6</i>	CableLabs Service Provider CA Certificates	5-20
<i>Table 5-7</i>	Local System CA Certificates	5-20
<i>Table 5-8</i>	Key Distribution Center Certificates	5-21
<i>Table 5-9</i>	DF Certificates	5-22
<i>Table 5-10</i>	PacketCable Server Certificates	5-23
<i>Table 5-11</i>	CableLabs Code Verification Root CA Certificates	5-25
<i>Table 5-12</i>	CableLabs Code Verification CA Certificates	5-25
<i>Table 5-13</i>	Manufacturer Code Verification Certificates	5-26
<i>Table 5-14</i>	Service Provider Code Verification Certificates	5-27
<i>Table 8-1</i>	Browser Platform Support	8-1
<i>Table 9-1</i>	Device Details Page	9-10

Table 9-2	Example Relate/Unrelate Process	9-15
Table 9-3	View Device Provisioning Engine Details Page	9-20
Table 9-4	View Network Registrar Extension Point Details Page	9-22
Table 9-5	View Provisioning Groups Details Page	9-24
Table 9-6	View RDU Details Page	9-25
Table 10-1	Configure Class of Service Page	10-2
Table 10-2	Add Class of Service Page	10-4
Table 10-3	Configure ATA 186 Defaults Page	10-8
Table 10-4	Configure WAN MAN /WAN Data Defaults Page	10-9
Table 10-5	Configure DOCSIS Defaults Page	10-14
Table 10-6	Configure Network Registrar Defaults Page	10-15
Table 10-7	Configure PacketCable (Voice Technology) Defaults Page	10-17
Table 10-8	Configure RDU Defaults Page	10-20
Table 10-9	Configure System Defaults Page	10-21
Table 10-10	Configure XGCP Defaults Page	10-23
Table 10-11	Manage External Files Page	10-26
Table 10-12	Modifying Publishing Plug-ins Page	10-33
Table 12-1	Template Grammar	12-2
Table 12-2	Defined Option Encoding Types	12-8
Table 12-3	DOCSIS Options and Version Support	12-12
Table 12-4	PacketCable MTA 1.0 Options	12-23
Table 12-5	Non-secure CableHome Options and Version Support	12-24
Table 12-6	Logging Levels	12-38
Table A-1	RDU Alerts	A-2
Table A-2	DPE Alerts	A-3
Table A-3	Agent Alerts	A-4
Table A-4	Network Registrar Extension Point Alerts	A-5
Table B-1	DHCP Option 122 to BACC Property Comparison	B-2
Table B-2	DHCP Option 177 to BACC Property Comparison	B-2



Preface

Welcome to the *Cisco Broadband Access Center for Cable Administrator's Guide*. This chapter provides an outline of the other chapters in this guide, and demonstrates the styles and conventions used in the guide.



Note

This document is to be used in conjunction with the documentation listed in [Related Documentation](#), page [xxi](#).

This section contains the following information:

- [Audience](#), page [xix](#)
- [How This Guide Is Organized](#), page [xx](#)
- [Conventions](#), page [xxi](#)
- [Related Documentation](#), page [xxi](#)
- [Obtaining Documentation](#), page [xxii](#)
- [Documentation Feedback](#), page [xxiii](#)
- [Cisco Product Security Overview](#), page [xxiii](#)
- [Obtaining Technical Assistance](#), page [xxiv](#)
- [Obtaining Additional Publications and Information](#), page [xxvi](#)

Audience

The *Cisco Broadband Access Center for Cable Administrator's Guide* is written for system administrators responsible for automating large scale provisioning for broadband access. The network administrator should be familiar with these topics:

- Basic networking concepts and terminology
- Network administration
- Cable networks

How This Guide Is Organized

This guide describes how to administer and maintain the Broadband Access Center for Cable (BACC):

Table 1 Document Chapters

Chapter 1, “Broadband Access Center for Cable Overview”	Describes BACC, its features and benefits, and gives you a detailed description of the BACC architecture.
Chapter 2, “Broadband Access Center for Cable System Architecture”	Describes the systems architecture implemented in this Broadband Access Center for Cable release.
Chapter 3, “Configuration Workflows and Checklists”	Provides checklists to follow when configuring BACC for use.
Chapter 4, “DOCSIS Configuration”	Identifies those functions to check or configure, and tools available to bring a BACC DOCSIS deployment into operation.
Chapter 5, “PacketCable Voice Configuration”	Describes all activities that must be performed to bring a PacketCable voice deployment into service.
Chapter 6, “CableHome Configuration”	Describes those activities that must be performed to ensure a satisfactory CableHome deployment, for the non-secure (DHCP) version of the CableHome technology.
Chapter 7, “Database Management”	Describes information on the management and maintenance of the RDU database. As with any database, it is essential that the administrator understand, and be familiar with, database backup and recovery procedures.
Chapter 8, “Understanding the Administrator’s User Interface”	Describes how to access BACC and explains various user interface components.
Chapter 9, “Using the Broadband Access Center for Cable Administrator User Interface”	Describes administration activities including searching for, and viewing device information.
Chapter 10, “Configuring Broadband Access Center for Cable”	Describes the configuration activities that are performed using the BACC administrator application.
Chapter 11, “Configuring and Using the Sample User Interface”	Describes the concepts, uses, and application of the sample workflow user interface.
Chapter 12, “Broadband Access Center for Cable Support Tools and Advanced Concepts”	Describes a series of tools intended to help, configure, maintain, speed and improve the installation, deployment, and use of BACC.
Appendix A, “Alert and Error Messages”	Lists and describes all BACC alert messages.
Appendix B, “PacketCable DHCP Options to BACC Properties Mapping”	Identifies the mapping of BACC properties to the PacketCable DHCP options used for PacketCable provisioning.
Appendix C, “Broadband Access Center for Cable Application Programming Interface Use Cases”	Presents a series of the most common provisioning API use cases, including pseudo-code segments that can be used to model typical service provider workflows.
Glossary	Defines terminology used in this guide and generally applicable to the technologies being discussed.

Conventions

This document uses the following conventions:

Item	Convention
Commands and keywords	boldface font
Variables for which you supply values	<i>italic</i> font
Displayed session and system information	screen font
Information you enter	boldface screen font
Variables you enter	<i>italic screen</i> font
Menu items and button names	boldface font
Selecting a menu item in paragraphs	Option > Network Preferences
Selecting a menu item in tables	Option > Network Preferences



Note

Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.



Caution

Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Related Documentation



Note

We sometimes update the printed and electronic documentation after original publication. Therefore, you should also review the documentation on Cisco.com for any updates.

Documentation for BACC 2.7 includes:

- *Release Notes for Cisco Broadband Access for Cable 2.7*
- *Cisco Broadband Access for Cable Installation Guide*
- *Cisco Broadband Access for Cable Administrator's Guide*
- *Cisco Broadband Access for Cable CLI Reference Guide*

To support the DPE-590:

- *Device Provisioning Engine 590 Recovery CD-ROM Release Notes*
- *Cisco Content Engine 500 Series Hardware Installation Guide*

To support the DPE-2115:

- *Device Provisioning Engine 2115 Recovery CD-ROM Release Notes*
- *Installation and Setup Guide for the Cisco 1102 VLAN Policy Server*

**Caution**

Refer to this guide for port and connector identification and to perform hardware installation only. Do not attempt to perform any of the configuration instructions found in that guide.

- *Cisco Network Registrar User's Guide*
- *Cisco Network Registrar CLI Reference*

Obtaining Documentation

Cisco documentation and additional literature are available on Cisco.com. Cisco also provides several ways to obtain technical assistance and other technical resources. These sections explain how to obtain technical information from Cisco Systems.

Cisco.com

You can access the most current Cisco documentation at this URL:

<http://www.cisco.com/techsupport>

You can access the Cisco website at this URL:

<http://www.cisco.com>

You can access international Cisco websites at this URL:

http://www.cisco.com/public/countries_languages.shtml

Product Documentation DVD

Cisco documentation and additional literature are available in the Product Documentation DVD package, which may have shipped with your product. The Product Documentation DVD is updated regularly and may be more current than printed documentation.

The Product Documentation DVD is a comprehensive library of technical product documentation on portable media. The DVD enables you to access multiple versions of hardware and software installation, configuration, and command guides for Cisco products and to view technical documentation in HTML. With the DVD, you have access to the same documentation that is found on the Cisco website without being connected to the Internet. Certain products also have .pdf versions of the documentation available.

The Product Documentation DVD is available as a single unit or as a subscription. Registered Cisco.com users (Cisco direct customers) can order a Product Documentation DVD (product number DOC-DOCDVD=) from the Ordering tool or Cisco Marketplace.

Cisco Ordering tool:

<http://www.cisco.com/en/US/partner/ordering/>

Cisco Marketplace:

<http://www.cisco.com/go/marketplace/>

Ordering Documentation

Beginning June 30, 2005, registered Cisco.com users may order Cisco documentation at the Product Documentation Store in the Cisco Marketplace at this URL:

<http://www.cisco.com/go/marketplace/>

Cisco will continue to support documentation orders using the Ordering tool:

- Registered Cisco.com users (Cisco direct customers) can order documentation from the Ordering tool:

<http://www.cisco.com/en/US/partner/ordering/>

- Instructions for ordering documentation using the Ordering tool are at this URL:

http://www.cisco.com/univercd/cc/td/doc/es_inpk/pdi.htm

- Nonregistered Cisco.com users can order documentation through a local account representative by calling Cisco Systems Corporate Headquarters (California, USA) at 408 526-7208 or, elsewhere in North America, by calling 1 800 553-NETS (6387).

Documentation Feedback

You can rate and provide feedback about Cisco technical documents by completing the online feedback form that appears with the technical documents on Cisco.com.

You can send comments about Cisco documentation to bug-doc@cisco.com.

You can submit comments by using the response card (if present) behind the front cover of your document or by writing to the following address:

Cisco Systems
Attn: Customer Document Ordering
170 West Tasman Drive
San Jose, CA 95134-9883

We appreciate your comments.

Cisco Product Security Overview

Cisco provides a free online Security Vulnerability Policy portal at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.html

From this site, you can perform these tasks:

- Report security vulnerabilities in Cisco products.
- Obtain assistance with security incidents that involve Cisco products.
- Register to receive security information from Cisco.

A current list of security advisories and notices for Cisco products is available at this URL:

<http://www.cisco.com/go/psirt>

If you prefer to see advisories and notices as they are updated in real time, you can access a Product Security Incident Response Team Really Simple Syndication (PSIRT RSS) feed from this URL:

http://www.cisco.com/en/US/products/products_psirt_rss_feed.html

Reporting Security Problems in Cisco Products

Cisco is committed to delivering secure products. We test our products internally before we release them, and we strive to correct all vulnerabilities quickly. If you think that you might have identified a vulnerability in a Cisco product, contact PSIRT:

- Emergencies—security-alert@cisco.com

An emergency is either a condition in which a system is under active attack or a condition for which a severe and urgent security vulnerability should be reported. All other conditions are considered nonemergencies.

- Nonemergencies—psirt@cisco.com

In an emergency, you can also reach PSIRT by telephone:

- 1 877 228-7302
- 1 408 525-6532



Tip

We encourage you to use Pretty Good Privacy (PGP) or a compatible product to encrypt any sensitive information that you send to Cisco. PSIRT can work from encrypted information that is compatible with PGP versions 2.x through 8.x.

Never use a revoked or an expired encryption key. The correct public key to use in your correspondence with PSIRT is the one linked in the Contact Summary section of the Security Vulnerability Policy page at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.htm

The link on this page has the current PGP key ID in use.

Obtaining Technical Assistance

Cisco Technical Support provides 24-hour-a-day award-winning technical assistance. The Cisco Technical Support & Documentation website on Cisco.com features extensive online support resources. In addition, if you have a valid Cisco service contract, Cisco Technical Assistance Center (TAC) engineers provide telephone support. If you do not have a valid Cisco service contract, contact your reseller.

Cisco Technical Support & Documentation Website

The Cisco Technical Support & Documentation website provides online documents and tools for troubleshooting and resolving technical issues with Cisco products and technologies. The website is available 24 hours a day, at this URL:

<http://www.cisco.com/techsupport>

Access to all tools on the Cisco Technical Support & Documentation website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register at this URL:

<http://tools.cisco.com/RPF/register/register.do>

**Note**

Use the Cisco Product Identification (CPI) tool to locate your product serial number before submitting a web or phone request for service. You can access the CPI tool from the Cisco Technical Support & Documentation website by clicking the **Tools & Resources** link under Documentation & Tools. Choose **Cisco Product Identification Tool** from the Alphabetical Index drop-down list, or click the **Cisco Product Identification Tool** link under Alerts & RMAs. The CPI tool offers three search options: by product ID or model name; by tree view; or for certain products, by copying and pasting **show** command output. Search results show an illustration of your product with the serial number label location highlighted. Locate the serial number label on your product and record the information before placing a service call.

Submitting a Service Request

Using the online TAC Service Request Tool is the fastest way to open S3 and S4 service requests. (S3 and S4 service requests are those in which your network is minimally impaired or for which you require product information.) After you describe your situation, the TAC Service Request Tool provides recommended solutions. If your issue is not resolved using the recommended resources, your service request is assigned to a Cisco engineer. The TAC Service Request Tool is located at this URL:

<http://www.cisco.com/techsupport/servicerequest>

For S1 or S2 service requests or if you do not have Internet access, contact the Cisco TAC by telephone. (S1 or S2 service requests are those in which your production network is down or severely degraded.) Cisco engineers are assigned immediately to S1 and S2 service requests to help keep your business operations running smoothly.

To open a service request by telephone, use one of the following numbers:

Asia-Pacific: +61 2 8446 7411 (Australia: 1 800 805 227)

EMEA: +32 2 704 55 55

USA: 1 800 553-2447

For a complete list of Cisco TAC contacts, go to this URL:

<http://www.cisco.com/techsupport/contacts>

Definitions of Service Request Severity

To ensure that all service requests are reported in a standard format, Cisco has established severity definitions.

Severity 1 (S1)—Your network is “down,” or there is a critical impact to your business operations. You and Cisco will commit all necessary resources around the clock to resolve the situation.

Severity 2 (S2)—Operation of an existing network is severely degraded, or significant aspects of your business operation are negatively affected by inadequate performance of Cisco products. You and Cisco will commit full-time resources during normal business hours to resolve the situation.

Severity 3 (S3)—Operational performance of your network is impaired, but most business operations remain functional. You and Cisco will commit resources during normal business hours to restore service to satisfactory levels.

Severity 4 (S4)—You require information or assistance with Cisco product capabilities, installation, or configuration. There is little or no effect on your business operations.

Obtaining Additional Publications and Information

Information about Cisco products, technologies, and network solutions is available from various online and printed sources.

- Cisco Marketplace provides a variety of Cisco books, reference guides, documentation, and logo merchandise. Visit Cisco Marketplace, the company store, at this URL:
<http://www.cisco.com/go/marketplace/>
- *Cisco Press* publishes a wide range of general networking, training and certification titles. Both new and experienced users will benefit from these publications. For current Cisco Press titles and other information, go to Cisco Press at this URL:
<http://www.ciscopress.com>
- *Packet* magazine is the Cisco Systems technical user magazine for maximizing Internet and networking investments. Each quarter, Packet delivers coverage of the latest industry trends, technology breakthroughs, and Cisco products and solutions, as well as network deployment and troubleshooting tips, configuration examples, customer case studies, certification and training information, and links to scores of in-depth online resources. You can access Packet magazine at this URL:
<http://www.cisco.com/packet>
- *iQ Magazine* is the quarterly publication from Cisco Systems designed to help growing companies learn how they can use technology to increase revenue, streamline their business, and expand services. The publication identifies the challenges facing these companies and the technologies to help solve them, using real-world case studies and business strategies to help readers make sound technology investment decisions. You can access iQ Magazine at this URL:
<http://www.cisco.com/go/iqmagazine>
or view the digital edition at this URL:
<http://ciscoiq.texterity.com/ciscoiq/sample/>
- *Internet Protocol Journal* is a quarterly journal published by Cisco Systems for engineering professionals involved in designing, developing, and operating public and private internets and intranets. You can access the Internet Protocol Journal at this URL:
<http://www.cisco.com/ipj>
- Networking products offered by Cisco Systems, as well as customer support services, can be obtained at this URL:
<http://www.cisco.com/en/US/products/index.html>
- Networking Professionals Connection is an interactive website for networking professionals to share questions, suggestions, and information about networking products and technologies with Cisco experts and other networking professionals. Join a discussion at this URL:
<http://www.cisco.com/discuss/networking>
- World-class networking training is available from Cisco. You can view current offerings at this URL:
<http://www.cisco.com/en/US/learning/index.html>



Broadband Access Center for Cable Overview

Broadband Access Center for Cable (BACC) automates the configuration and provisioning of network devices supported by a broadband service provider. It is a flexible product that can be scaled to suit virtually any size network. BACC is designed to handle rapid growth of service providers. It targets broadband service providers (including multiple service operators), Internet, and voice service providers who want to deploy IP data, voice, and video on hybrid fiber and coaxial cable networks. BACC also provides such critical features as redundancy and failover protection, and can be integrated into new or existing environments through the use of a provisioning application programming interface (API) that lets you control how BACC operates. You can use the provisioning API to enable BACC to register devices, device configurations, and configure the entire BACC provisioning system.

Features and Benefits

BACC lets multiple service operators (MSOs) meet the rapidly changing demands for data over cable services. Using BACC, you can realize these features and benefits of its architecture:

- Increased scalability
- Distributed architecture
- Redundancy
- Extensibility

Supported Technologies

This BACC release supports these technologies:

- DOCSIS high-speed data
- PacketCable voice services
- Non-secure PacketCable voice services
- Euro-PacketCable voice services
- Non-secure CableHome provisioning

DOCSIS High-Speed Data

The Data Over Cable Service Interface Specification defines functionality in cable modems involved in high-speed data distribution over cable television system networks. This allows MSOs to provide a variety of services through an “always-on” Internet connection. These services include broadband Internet connectivity, telephony, real-time interactive gaming, and video conferencing.

**Note**

Broadband Access Center for Cable supports DOCSIS 1.0, 1.1, and 2.0 devices.

PacketCable Voice Services

PacketCable voice technology enables the delivery of advanced, real-time multimedia services over a two-way cable network. PacketCable is built on top of the infrastructure supported by cable modems to enable a wide range of multimedia services such as IP telephony, multimedia conferencing, interactive gaming, and general multimedia applications.

The PacketCable voice technology enables additional services, for example, basic and extended telephony services, to be delivered more efficiently and cost-effectively, over the broadband cable access network.

**Note**

BACC currently supports versions 1.0 and 1.1 of the PacketCable specifications.

Non-Secure PacketCable Voice Services

Non-secure PacketCable voice services are the same as the standard PacketCable voice services except for the lack of security found in the non-secure variant.

Euro-PacketCable Voice Services

Euro-PacketCable services are the European equivalent to the North American PacketCable standard. The only significant difference between the two is that Euro-PacketCable uses different MIBs. See the [“Configuring Euro-PacketCable MIBs”](#) section on page 5-6 for additional information.

Non-Secure CableHome Provisioning

Non-secure CableHome 1.0 provisioning, hereafter referred to as home networking technology, is built on top of the existing DOCSIS standard and supports a ‘plug and play’ environment for home connectivity of residential broadband services. This form of home networking technology encompasses a DOCSIS home access device with support for CableHome functionality. This device is known as Portal Services and is considered to be the home’s entry point.



Broadband Access Center for Cable System Architecture

This chapter describes the system architecture implemented in this Broadband Access Center for Cable (BACC) release. The architecture described in this chapter is for information purpose only and is intended as background for those using the BACC product.

Architecture

This section describes the basic BACC architecture including:

- Regional distribution unit (RDU) that provides:
 - The authoritative data store of the BACC system.
 - Support for processing API requests.
 - A device detection/configuration/disruption engine, which invokes the various installed extensions to determine device types, generate configurations, and support disruption for specific device types.

See the [“Regional Distribution Unit” section on page 2-3](#) for additional information.

- Device provisioning engines (DPEs) that provide:
 - TFTP protocol server
 - Configuration cache
 - Redundancy
 - Time of day server
 - PacketCable provisioning services

See the [“Device Provisioning Engines” section on page 2-4](#) for additional information.

- Provisioning Groups

See the [“Provisioning Groups” section on page 2-8](#) for additional information.

- Cisco Network Registrar servers that provide:
 - Dynamic Host Configuration Protocol (DHCP)
 - Domain Name System (DNS)

See the [“Cisco Network Registrar” section on page 2-9](#) for additional information.

- A Kerberos server that authenticates PacketCable MTAs. See the [“Key Distribution Center” section on page 2-10](#) for additional information.
- An SNMP agent that provides:
 - SNMP version v2c support
 - SNMP Notification supportSee the [“SNMP Agent” section on page 2-14](#) for additional information.
- The BACC agent that provides:
 - Administrative monitoring of all critical BACC processes.
 - Automated process restart capability.
 - The ability to start and stop BACC component processes. See the [“BACC Agent” section on page 2-15](#) for additional information.

Registration Modes

Registration modes give the service provider a way to control the number of interactions with the subscriber. For any registered device, the service provider must be prepared to process any change to the device. So there is a significant difference between registering 100 cable modems with unregistered computers behind them and registering 100 cable modems each of which has a potentially large number of registered computers behind them. For this reason, the service provider must carefully choose among the standard, promiscuous, and roaming registration modes.

Standard Mode

When operating in the standard mode (sometimes called the fixed mode) a computer is given unprovisioned access when it is moved behind a different cable modem.

Promiscuous Mode

Only the device is registered; the DHCP server maintains lease information about a device operating behind another device. All devices behind a registered device receive network access.

Roaming Mode

In the roaming mode, a registered device can receive its class of service behind any other registered device. This permits, for example, the use of a laptop moving from location to location and obtaining service from multiple cable modems.

Mixed Mode

A mix of the promiscuous mode and the roaming mode can also be used as the two separate modes of operation can also coexist simultaneously.

Regional Distribution Unit

The RDU is the primary server in the BACC provisioning system. The RDU also provides:

- Management of device configuration generation.
- Authoritative datastore for the BACC system.
- Device disruption.
- A clearinghouse function, through which all application programming interface (API) requests must pass.
- Management of the BACC system.

The RDU supports the addition of new technologies and services through an extensible architecture. The RDU also supports:

- Data access and manipulation by means of the API.
- Distributing configurations to the DPEs for scalability.
- External clients, operations support systems (OSSs), and other provisioning related functionality by means of the API.

These sections describe these RDU concepts:

- [Generating Device Configurations, page 2-3](#)
- [Service Level Selection, page 2-3](#)
- [Regional Distribution Unit Failover, page 2-4](#)

**Note**

The RDU determines the names of DPEs and Network Registrar extensions by the interface they connect to the RDU on. That is, the name of the DPE or Network Registrar extensions is what the RDU machine thinks it is. This requires the RDU to be able to resolve the names (both forward and reverse).

Generating Device Configurations

When a device boots, it requests a configuration from BACC and this configuration determines the level of service for the device. Device configurations can include customer required provisioning information, such as: DHCP IP address selection, bandwidth, data rates, flow control, communication speeds, and level of service. A configuration can contain DHCP configuration and TFTP files for any device. When an unprovisioned device is installed, and the boot operation performed, a default configuration for the appropriate technology is obtained from BACC and sent to the device, by means of either DHCP or TFTP. The default configuration can be changed for each supported technology.

Service Level Selection

The service level selection extension point determines the criteria to be used by the RDU and notifies the RDU which criteria to use when generating a device configuration. The RDU stores this information for each device in the RDU database. Although a device may have been registered as having to receive one set of criteria, a second set may actually get selected. The configuration generation extensions look for the selected criteria and use them. Consequently, since the RDU automatic regeneration now knows that a second set of criteria are being used, the device configuration is regenerated if any changes occur to any of the criteria.

Service level selection extension points are entered on various technology defaults pages (“[Configuring Defaults](#)” section on page 10-7 for additional information). These are entered as a comma-separated list of extensions to instantiate creation of an object that can be used as the extension(s) for that specific extension point. By default, these properties are populated with zero or with one of the built-in extensions. These extensions should not be modified unless you are installing your own custom extensions.

Regional Distribution Unit Failover

Broadband Access Center for Cable currently supports one RDU per installation. For failover support, use your own hardware redundant system or obtain a similar system with hot-swap capability.

Device Provisioning Engines

The Cisco Device Provisioning Engine (DPE) communicates with the RDU to give devices their configurations. Each DPE caches information for up to one million devices and multiple DPEs can be used to ensure redundancy and scalability.

The DPE handles all configuration requests including providing configuration files for devices. It is integrated with the Cisco Network Registrar DHCP server to control the assignment of IP addresses for each device. Multiple DPEs can communicate with a single DHCP server.

The DPE manages these activities:

- Last step device configuration generation (DOCSIS timestamps for instance).
- Communication of the configuration files through an embedded TFTP server.
- Integration with Cisco Network Registrar.
- Time of day protocol server.
- Voice technology provisioning services.

Types of Device Provisioning Engine

Two types of DPE are supported by this BACC release; the traditional hardware device (either DPE-590 or DPE-2115) and the software only Solaris DPE. See the “[Hardware Device Provisioning Engines](#)” section on page 2-5 and the “[Solaris Device Provisioning Engines](#)” section on page 2-5 for additional information.

In addition to the different types of DPE, this section describes these major DPE components:

- [DPE License Keys, page 2-5](#)
- [TACACS+ and DPE Authentication, page 2-6](#)
- [DPE Server Assignments, page 2-7](#)
- [TFTP Server, page 2-7](#)
- [Provisioning Groups, page 2-8](#)

Hardware Device Provisioning Engines

BACC currently supports the DPE-590 and the DPE-2115 hardware device provisioning engines. Refer to these documents for additional information concerning the device itself, ports, connectors, and rear panel components:

- For the DPE-590, refer to the *Cisco Content Engine 500 Series Hardware Installation Guide*. This can be found at:

www.cisco.com/en/US/products/hw/contnetw/ps761/products_installation_guide_book09186a00800801e0.html

- For the DPE-2115, refer to the *Installation and Setup Guide for the Cisco 1102 VLAN Policy Server*. This can be found at:

www.cisco.com/en/US/products/sw/secursw/ps2136/products_installation_and_configuration_guide_book09186a00801f0d02.html

**Note**

Whenever an interface link between a DPE-2115 and a Catalyst switch is interrupted, a default 30-second delay occurs before data traffic flows.

Solaris Device Provisioning Engines

The Solaris DPE functions in the same way as the hardware DPE with the exception that this part of the BACC product is installed on a computer running either the Solaris 8 or 9 operating system. With few exceptions, the same command line interface CLI is used on both the hardware and Solaris DPEs. See the *Cisco Broadband Access Center for Cable Command Line Interface Reference* for specific information on the CLI commands that each DPE supports.

DPE License Keys

Licensing controls the number of DPEs (nodes) you can use. If you attempt to install more DPEs than you are licensed to have, those new DPEs will not be registered, and will be rejected. Existing valid DPEs remain online and register themselves again with the RDU at will.

**Note**

For licensing purposes, a registered DPE is considered to be one node.

The number of DPE licenses you register with the RDU includes hardware and Solaris DPEs, regardless of release number or type; including those used as part of a BACC lab installation. See the *Broadband Access Center for Cable Installation Guide* for additional information.

Whenever you change licenses, either by adding a license, extending an evaluation license, or through the expiration of an evaluation license, the changes take immediate effect.

When you delete a DPE from the RDU database, a license is freed.

Deleted DPEs are removed from all provisioning groups they serve and all Network Registrar extensions are notified that the DPE is no longer available. Consequently, when a previously deleted DPE is registered again, it is considered to be licensed again and will remain so until it is deleted from the RDU again or its license expires.

DPEs that are not licensed through the RDU do not appear in the administrator's user interface. You can determine the license state only by examining the DPE and RDU log files (dpe.log and rdu.log).

Any deleted or unlicensed hardware DPE, running either BPR 2.0.x or BACC 2.5, will constantly attempt to register with the RDU. This is normal behavior for devices running these versions of software and all rejected registrations are recorded in the RDU log and in the generation of SYSLOG alerts identifying this situation.

TACACS+ and DPE Authentication

Terminal Access Controller Access Control System plus (TACACS+) is a TCP-based protocol that supports centralized access control for large numbers of network devices and user authentication for the DPE CLI. Through the use of TACACS+ a DPE can support multiple users, with each username and login/enable password configured at the TACACS+ server. TACACS+ is used to implement the TACACS+ client/server protocol (ASCII login only).

TACACS+ Privilege Levels

The TACACS+ server uses the TACACS+ protocol to authenticate any user logging into a DPE. The TACACS+ client specifies a certain service level that is configured for the user. [Table 2-1](#) identifies the two service levels used to authorize DPE user access:

Table 2-1 TACACS+ Service Levels

Mode	Description
Login	User-level commands at router> prompt.
Enable	Enable-level commands at router# prompt.

TACACS+ Client Settings

TACACS+ uses a number of properties that are configured using the command line interface (CLI). See the *Cisco Broadband Access Center for Cable Command Line Reference* for information on these TACACS+ related CLI commands.

When TACACS+ is enabled, the administrator must specify either the IP addresses of all TACACS+ servers or their FQDNs with non-default values.

The administrator can also specify these settings, using their default values if applicable:

- The shared secret key for each TACACS+ server. This key is used for data encryption between the DPE and the TACACS+ server. If you choose to omit the shared secret for any specific TACACS+ server, TACACS+ message encryption is not used.
- The TACACS+ server timeout value. This is the maximum time that the TACACS+ client will wait for a TACACS+ server to reply to protocol requests.
- The TACACS+ server number of retries. This identifies the number of times that the TACACS+ client attempts a valid protocol exchange with a TACACS+ server.



Note

These commands are used in both the hardware and Solaris DPEs. In the hardware DPE, you can use only console mode.

DPE Server Assignments

BACC supports multiple DPEs. These communicate with devices, a DHCP failover configuration, and the RDU. During installation, you must make these DPE assignments:

- Assign a DPE to be responsible for one or more logical groups of devices or provisioning groups.
- The IP address and port number of the RDU.

The DPE's primary objective is to send configurations to customer devices whenever those devices are powered up or rebooted. To do this quickly, the DPE keeps a copy of the configuration for each device in a local cache database.

**Note**

A DPE should be assigned to only one provisioning group.

TFTP Server

The integrated TFTP server receives requests for files, including DOCSIS configuration files, both from device and non device entities. This server then transmits the file to the requesting entity.

The TFTP server is located in a home directory that is used for local file system access. The local files are stored in the BACC_DATA/dpe/tftp directory.

By default the TFTP server only looks in its cache for a TFTP read. However, if the tftp allow-read-access command has been run, the TFTP server first looks at the local file system before looking in the cache. If the file exists in the local file system, it is read from there. If not, the TFTP server looks in the cache. If the file is there, the server uses it. Otherwise, the server sends a request to the RDU for the file. When you can enable read access from the local file system, directory structure read requests are allowed only from the local file system.

BACC only allows writes to the local file system; never to the DPE cache. If you run the tftp allow-write-access DPE CLI command, BACC will allow a write to the TFTP home directory. By default, the creation of directories or the override files is not permitted. However, to change this, run either the tftp allow-create-dirs or the tftp allow-override commands.

DOCSIS Shared Secret

BACC lets you define multiple different DOCSIS shared secrets (DSS), for dynamic DOCSIS configuration files only, on devices belonging to different CMTSs. In this way, a compromised shared secret results in a limited number of compromised CMTSs and not every CMTS in the entire deployment.

Although the DSS can be set for each DPE, you should set it on a provisioning group basis and it must match what has been configured for the CMTSs in that provisioning group.

**Caution**

Configuring multiple DSS within one provisioning group could, under some conditions, result in degraded CMTS performance. This should have virtually no effect on BACC.

You can enter the shared secret as clear text or as IOS-encrypted format.

When entered in clear text, the DSS is encrypted to suit IOS version 12.2BC. You can also set the DSS from the RDU using the Administrator's user interface or the API. In this case, the DSS is entered, stored at the RDU, and passed to all DPEs in clear text. Consequently, before a DSS entered this way is stored on the DPE, it is encrypted.

If you set the DSS directly at the DPE, using the appropriate CLI command, this will take precedence over what is set from RDU.

Resetting the DOCSIS Shared Secret

If the security of the DSS becomes compromised, or if you simply want to change the shared secret for administrative purposes, you can run the `show running config` CLI command and then copy and paste the DOCSIS shared secret line from the displayed configuration back into the DPE configuration. In this way you can copy what you enter in a Cisco CMTS to the DPE CLI. See the *Cisco Broadband Access Center for Cable Command Line Reference* for additional information.



Note To change the shared secret as described above, the CMTS must be running a software version newer than V 12.2BC.

When the DSS must be changed, you must:

-
- Step 1** Identify the provisioning group with the DOCSIS shared secret to be reset.
 - Step 2** Examine the list of DPEs and CMTSs associated with the provisioning group.
 - Step 3** Change the primary DSS on the CMTS.
 - Step 4** Change the compromised DSS on the CMTS to the secondary DSS. This is required to allow cable modems to continue to register until all of the DOCSIS configuration files have been successfully changed to use the new DSS.
 - Step 5** Determine which DPEs were affected and change the DSS accordingly on each.
 - Step 6** Confirm that the DOCSIS configuration files are using the new DSS and then remove the CMTS compromised secondary shared secret from the CMTS configuration.
-

Provisioning Groups

A provisioning group is designed to be a regional grouping of servers usually consisting of two or more DPEs and a failover pair of DHCP servers that can handle the provisioning needs of up to one million devices. As the number of devices grows past one million, you can add additional provisioning groups to the deployment.



Note The servers for a provisioning group are not required to reside at a regional location, they can just as easily be deployed in the central NOC.

To support redundancy and load sharing each provisioning group can support any number of DPEs. As the requests come in from the DHCP servers, they are distributed between the DPEs in the provisioning group and an affinity is established between the devices and a specific DPE. This affinity is retained as long as the DPE state within the provisioning group remains stable. Because a specific device will normally be assigned to the same DPE, expensive Kerberos reticketing is avoided.

Cisco Network Registrar

Network Registrar provides Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) functionality. It has a complete administrative user interface that, when coupled with customized BACC configuration screens, can be used within a larger enterprise management system.

**Note**

For additional information on Network Registrar, refer to these documents: *Network Registrar User's Guide*, *Network Registrar CLI Reference*, and the *Network Registrar Installation Guide*.

Dynamic Host Configuration Protocol

The Dynamic Host Configuration Protocol (DHCP) server automates the process of configuring IP addresses on TCP/IP networks. This protocol performs many of the functions that a system administrator carries out when connecting a device to a network. DHCP automatically manages network policy decisions and eliminates the need for manual configuration. This adds flexibility, mobility, and control to networked device configurations.

DHCP failover allows pairs of DHCP servers to act in such a way that one can take over if the other stops functioning. The server pairs are known as the primary and backup server. Under normal circumstances the primary server performs all DHCP functions. If the primary server becomes unavailable, the backup server takes over. In this way, DHCP failover prevents loss of access to the DHCP service if the primary fails.

Domain Name System

The Domain Name System (DNS) server contains information on hosts throughout the network, including IP address hostnames and routing information, and DNS uses them primarily to translate between IP addresses and domain names. This conversion of names, such as `www.cisco.com`, to IP addresses simplifies accessing Internet-based applications.

The DNS directory service consists of:

- DNS data.
- DNS servers.
- Internet protocols for fetching data from the servers.
- Dynamic DNS for voice provisioning.

Lease Reservation

BACC lease reservation works in conjunction with Network Registrar's Central Configuration Management (CCM) to assign a device with a static IP address during provisioning.

When you provision a new device, BACC determines whether the IP address is specified and then determines which Network Registrar server identifies it is a valid IP address. After validation, the lease reservation function creates a reservation for the device using the Network Registrar CCM.

Lease reservation operates with all technologies supported by BACC and:

- Lets you add and remove IP address reservations using the BACC graphical user interface. See [Managing Devices, page 9-12](#) for additional information.
- Reports all errors resulting from attempts to reserve an IP address that is already in use or if a reservation is removed from the CCM server.



Note

For lease reservation to operate properly, you must have CNS Network Registrar version 6.1.2.3, with a licensed CCM server present, installed in your network.

You must configure the CCM address, port, username, and password before BACC can implement lease reservation. These parameters are set from the RDU Defaults page. Changes are dynamic and take effect immediately after being entered. (See the [“RDU Defaults” section on page 10-19](#) for information on these configuration parameters.)



Note

The lease reservation function is disabled by default and times out whenever the CCM cannot be reached for a specified amount of time.

Key Distribution Center

The key distribution center (KDC) is an authentication server used to authenticate MTAs and grant security tickets to them.



Note

The KDC is supported on multi-processor computers.

Default KDC Properties

The KDC has several default properties that get populated during BACC installation into a `<BACC_HOME>/kdc/solaris/kdc.ini` properties file. This file can be edited to change values as operational requirements dictate. After changes have been made, you must restart the KDC before any property file, key or certificate changes can take affect. The default properties are:

- interface address—This is the IP address of the local Ethernet interface that you want the KDC to monitor for incoming Kerberos messages. For example:

```
interface address = 10.10.10.1
```

- FQDN—Identifies the fully qualified domain name (FQDN) on which the KDC is installed. For example:

```
FQDN = kdc.cisco.com
```

**Note**

The interface address and FQDN are entered through the KDC Realm Name screen during installation. Refer to the *Broadband Access Center for Cable Installation Guide* for specific information.

- maximum log file size—The KDC generates a set of log files. This property specifies the maximum size, in kilobytes, that the log file can reach. Therefore, the KDC will create a new log file only when the current file reaches this maximum size. For example:

```
maximum log file size = 1000
```

- *n* saved log files—This property defines the number of old log files that the KDC saves. The default value is 7, and you can specify as many as required. For example:

```
n saved log files = 10
```

- log debug level—This property specifies the logging level for the log file.

```
log debug level = 5
```

- minimum (maximum) ps backoff—This property specifies the minimum (or maximum) time, in tenths of a second, that the KDC will wait for BACC to respond to the FQDN-REQUEST. For example:

```
minimum ps backoff = 150
```

Using the example values shown above, a sample INI file might contain data similar to that shown in [Example 2-1](#).

Example 2-1 Sample KDC INI Configuration File

```
interface address = 10.10.10.1
FQDN = kdc.cisco.com
maximum log file size = 1000
n saved log files = 10
log debug level = 5
minimum ps backoff = 150
maximum ps backoff = 300
```

You can set the times for both minimum and maximum ticket duration to effectively smooth out excessive numbers of ticket requests that could occur during deployment. This is beneficial given that most deployments occur during traditional working hours and excessive loading might, from time to time, adversely affect performance.

**Note**

Shortening the ticket duration forces the MTA to authenticate to the KDC much more frequently. While this results in much greater control over the authorization of telephony endpoints, it also causes much heavier message loads on the KDC and increased network traffic. For most circumstances, the default setting is appropriate and should not be changed.

- maximum ticket duration—This property defines the maximum duration for tickets generated by the KDC. The default unit is hours; however, by appending an **m** or **d**, the units can be changed to minutes or days, respectively.

The default value is 168, or seven days, and Cisco recommends that you not change this value because this is the duration required to conform to the PacketCable security specification. For example:

```
maximum ticket duration = 168
```

- minimum ticket duration—This property defines the minimum duration for tickets generated by the KDC. The default unit is hours; however, by appending an **m** or **d**, the units can be changed to minutes or days, respectively.

The default value is 144, or six days, and Cisco recommends that you not change this value. For example:

```
minimum ticket duration = 144
```

Euro-PacketCable Support

The Key Distribution Center (KDC) component supports Euro-PacketCable (tComLabs) certificate chains. [Example 2-2](#) shows a sample Euro-PacketCable enabled KDC configuration file.

Example 2-2 Example Euro-PacketCable Enabled KDC Configuration File

```
[general]
interface address = 10.10.10.1
FQDN = servername.cisco.com
maximum log file size = 10000
n saved log files = 100
log debug level = 5 minimum
ps backoff = 150 maximum
ps backoff = 300
euro-packetcable = true
```

KDC Certificates

The certificates used to authenticate the KDC are not shipped with BACC. You must obtain the required certificates from Cable Television Laboratories Inc. (CableLabs) and the content of these certificates must match those that are installed in the MTA. See the [“Using the PKCert.sh Tool to Manage KDC Certificates”](#) section on page 12-41 for additional information.



Caution

Without the certificates installed, the KDC will not function.

KDC Licenses

Obtain a KDC license from your Cisco representative and then install it in the correct directory.

To install a KDC license file:

-
- Step 1** Obtain your license file.
 - Step 2** Copy the license file to the <BACC_HOME>/kdc directory.



Caution

Be careful not to copy this as an ASCII file. The file contains binary data susceptible to unwanted modification during an ASCII transfer.

- Step 3** If the KDC license file is not called **bacckdc.license**, rename the file to **bacckdc.license**.

- Step 4** Run the `bprAgent restart kdc` command, from the `/etc/init.d` directory, to restart the KDC server and make the changes take effect.



Note KDC license files should not be copied between operating systems, as the transfer process may damage the file.

Multiple Realm Support

The BACC KDC supports the management of multiple realms.

To configure additional realms:

- Step 1** Locate the directory containing your KDC certificates.
- Step 2** Create a subdirectory there with a name matching the desired realm name. This subdirectory must be created using upper case characters only.
- Step 3** Place the KDC certificate and the private key for the realm in this directory
- Step 4** If the new realm is not chained to the same Service Provider as the KDC certificate, include all additional higher level certificates which differ from those in the certificates directory. However, since all realms must be rooted in the same certificate chain, only one locale (PacketCable, Euro-PacketCable) is supported per KDC installation.



Note Any given DPE is restricted to providing a service to a single realm.

BACC MIBs

Broadband Access Center for Cable supports several different MIBs. These include:

- CISCO-BACC-DPE-MIB—See [MIB Support, page 2-14](#).
- CISCO-APPLIANCE-MIB—See [MIB Support, page 2-14](#).
- CISCO-BACC-RDU-MIB—See [SNMP Agent, page 2-14](#).
- CISCO-BACC-SERVER-MIB—The CISCO-BACC-SERVER-MIB defines managed objects that are common to all BACC servers. This MIB supports the monitoring of multiple BACC servers when they are installed on the same device. The `ciscoBaccServerStateChanged` notification is generated every time a server state change takes place.

Table 2-2 summarizes BACC MIB support based on installation type. See the *Cisco Broadband Access Center for Cable Installation Guide* for descriptions of each installation type.

Table 2-2 BACC Supported MIBs

Installation type	MIBs Supported
Solaris DPE	CISCO-BACC-SERVER-MIB
	CISCO-BACC-DPE-MIB
Hardware DPE	RFC1213 - MIB II
	CISCO-APPLIANCE-MIB
	CISCO-BACC-SERVER-MIB
	CISCO-BACC-DPE-MIB
RDU	CISCO-BACC-SERVER-MIB
	CISCO-BACC-RDU-MIB

BACC Agents

This section describes BACC agents; what they are and why they are important. Subsequent descriptions provide all the details required to use and understand the agents. These agents are described:

- [SNMP Agent, page 2-14](#)
- [BACC Agent, page 2-15](#)

SNMP Agent

BACC provides basic SNMP v2 based monitoring of both the DPE and RDU servers. The BACC SNMP agents support both SNMP informs and traps. You can configure the SNMP agent on the DPE using `snmp-server` CLI commands and on RDU by using SNMP configuration CLI commands.

See the “[Using the snmpAgentCfgUtil.sh Command](#)” section on [page 12-46](#) for additional information on the SNMP configuration command line tool, and the *Cisco Broadband Access Center for Cable Command Line Reference* for additional information on the DPE CLI.

MIB Support

In addition to RFC 1213 (MIB-II), the SNMP agent supports the CISCO-CW-APPLIANCE-MIB. This MIB defines the managed objects for the software components installed on a hardware DPE. It monitors CPU, memory and disk utilization, and generates notifications whenever use exceeds certain thresholds. Notifications can be selectively enabled and disabled. Resource use is polled at regular intervals and notifications are generated when the average of two consecutive data points exceeds the threshold.

The SNMP agent also supports the CISCO-BACC-DPE-MIB. This MIB defines the managed objects for the software components installed on a Solaris DPE. The DPE manages local caching of device configurations and configuration files used by all supported devices. This MIB provides some basic DPE configuration and statistics information, including entries for TFTP and ToD servers.

The SNMP agent supports the CISCO-NMS-APPL-HEALTH-MIB which defines the Cisco NMS application health status notifications and related objects. These notifications are sent to the OSS/NMS to inform them about the NMS application status, including: started, stopped, failed, busy, or any abnormal exit of applications. The default MIB used is MIB-II.

The SNMP agent generates a cnaHealthNotif trap that announces that the RDU server has started, shutdown, failed, or there is a change in the exit status.

The SNMP agent supports the CISCO-BACC-RDU-MIB, which defines managed objects for the RDU. This MIB also contains managed objects defining statistics between the RDU and DPE and between the RDU and Network Registrar.

**Note**

These MIBs are located in the <BACC_HOME>/rdu/mibs directory.

BACC Agent

The BACC agent is an administrative agent that monitors the run time health of all BACC processes. This watchdog process ensures that if a process stops unexpectedly, it is automatically restarted.

The BACC agent can be used as a command line tool to start, stop, restart, and determine the status of any monitored processes.

Monitored Processes

If a monitored application fails, it is restarted automatically. If, for any reason, the restart process also fails, the BACC agent server will wait a prescribed amount of time before attempting to restart again.

**Note**

You do not have to use the BACC agent and the SNMP agent to monitor Network Registrar extensions.

The period between restart attempts increases exponentially until it reaches a length of 5 minutes. After that, the process restart is attempted at 5 minute intervals until successful. Five minutes after a successful restart, the period is automatically reset to 1 second again.

For example:

- Process A fails.
- The BACC agent server attempts to restart it and the first restart fails.
- The BACC agent server waits 2 seconds and attempts to restart the process and the second restart fails.
- The BACC agent server waits 4 seconds and attempts to restart the process and the third restart fails.
- The BACC agent server waits 16 seconds and attempts to restart the process and the fourth restart fails.

BACC Agent Command Line

The BACC agent automatically starts whenever the system boots up. Consequently, this agent also starts those BACC system components it is configured to control. The BACC agent can also be controlled through a simple command line interface. This is performed by running the `bprAgent` command from the `/etc/init.d` directory.

Table 2-3 describes the CLI commands available for use with the BACC agent. You can run the CLI from the `etc/init.d` directory.

Table 2-3 BACC Command Line Interface

Command	Description
<code>bprAgent start</code>	Starts the BACC agent, including all monitored processes.
<code>bprAgent stop</code>	Stops the BACC agent, including all monitored processes.
<code>bprAgent restart</code>	Restarts the BACC agent, including all monitored processes.
<code>bprAgent status</code>	Gets the status of the BACC agent, including all monitored processes.
<code>bprAgent start <process-name></code>	Starts one particular monitored process. The value <code><process-name></code> identifies that process.
<code>bprAgent stop <process-name></code>	Stops one particular monitored process. The value <code><process-name></code> identifies that process.
<code>bprAgent restart <process-name></code>	Restarts one particular monitored process. The value <code><process-name></code> identifies that process.
<code>bprAgent status <process-name></code>	Gets the status of one particular monitored process. The value <code><process-name></code> identifies that process.
Note	The <code><process-name></code> mentioned in Table 2-3 can be one of the <code>rdu</code> , <code>kdc</code> , <code>dpe</code> , <code>SnmpAgent</code> , and <code>jrnl</code> (which runs the administrators and sample user interface) processes. The CLI process is also monitored in both Lab and DPE installations.



Note

The RDU operates in a Solaris environment and may not shut down satisfactorily each time the Solaris **reboot** command is used. The preferred command to bring down the system is **shutdown**.

Logging

Logging of events is performed at both the DPE and RDU, and in some unique situations, DPE events are logged at the RDU. Log files are located in their own log directories and can be examined using any text processor. The files can be compressed to allow them to be easily emailed to the TAC or system integrators for troubleshooting and fault resolution.

Regional Distribution Unit Logs

The RDU has two logs that it maintains in the BACC_DATA/rdu/logs directory:

- rdu.log—Records all RDU events with the configured default level. See the “[Setting the RDU Log Level](#)” section on page 12-40 for instructions on setting the default log levels.
- audit.log—Records all high level changes to the BACC configuration or functionality including the user who made the change.

Device Provisioning Engine Logs

The DPE maintains a dpe.log file, in the BACC_DATA/dpe/logs directory, that also records all events having the configured default level. In situations where the DPE undergoes catastrophic failure, such as engaging in a series of system crashes, the catastrophic errors are also logged into the rdu.log file.

The SNMPService.logyyy.log log file is used by the DPE, when PacketCable is enabled on the DPE server, to provide detailed debugging information. You use the **show packetcable snmp log** DPE CLI command to view this file, which is also located in the BACC_DATA/dpe/logs directory. See the *Cisco Broadband Access Center for Cable Command Line Reference* for PacketCable command usage.

**Note**

PacketCable logging messages are sent to the dpe.log file and the detailed SNMP debugging is sent to the SNMPService.logyyy.log file.

Log Levels and Structures

The log file structure is described here, and illustrated in [Example 2-3](#), and includes:

- Domain Name—This is the name of the computer generating the log files.
- Date and Time—This is the date on which a message is logged. This information also identifies the applicable time zone.
- Facility—This identifies the system which, in this case is the BACC.
- SubFacility—This identifies the BACC subsystem or component.
- Severity Number—The logging system defines seven levels of severity (log levels) that are used to identify the urgency with which you might want to address log issues. The process of configuring log levels is described in the “[Configuring Log Levels](#)” section on page 2-18:
 - 0—Emergency: System unstable.
 - 1—Alert: Immediate action is needed.
 - 2—Critical: A critical condition exists.
 - 3—Error: Error conditions exist.
 - 4—Warning: Warning condition exists.
 - 5—Notification: A normal but significant condition exists.
 - 6—Information: Informational messages only.



Note Another level known as DEBUG is used exclusively by Cisco for debugging purposes. Do not use this level except at the direction of the Cisco TAC.

- Message ID—This is a unique identifier for the message text.
- Message—This is the actual log message.

**Note**

See the [“The RDU Log Level Tool” section on page 12-38](#) for additional logging information.

Configuring Log Levels

You can configure logging levels for both the RDU and the DPE to suit your specific requirements. For example, the logging level for the RDU could be set to Warning and the level for the DPE could be set to Alert.

Log messages are written based on certain events taking place. Whenever an event takes place, the appropriate log message and level are assigned and, if that level is less than or equal to the configured level the message is written to the log. The message is not written to the log if the level is higher than the configured value.

For example, assume that the log level is set to 4-Warning. All events generating messages with a log level of 4 or less are written into the log file. If the log level is set to 6-Information, the log file will receive all messages. Consequently, configuring a higher log level results in a larger log file size.

Example 2-3 Sample Log File

Domain Name	Data and Time	Facility	Sub-facility	Security Level	Msg ID	Message
BACC1:	2005 3 16 03:06:11 EST:	BPR-	RDU-		0236:	BPR Regional Distribution Unit starting up
BACC1:	2005 3 16 03:06:15 EST:	BPR-	RDU-	5	0566:	Initialized API defaults
BACC1:	2005 3 16 03:06:15 EST:	BPR-	RDU-	5	0567:	Initialized CNR defaults
BACC1:	2005 3 16 03:06:15 EST:	BPR-	RDU-	5	0568:	Initialized server defaults
BACC1:	2005 3 16 03:06:18 EST:	BPR-	RDU-	5	0570:	Initialized DOCSIS defaults
BACC1:	2005 3 16 03:06:18 EST:	BPR-	RDU-	5	0571:	Initialized computer defaults
BACC1:	2005 3 16 03:06:19 EST:	BPR-	RDU-	5	0573:	Initialized CableHome defaults
BACC1:	2005 3 16 03:06:19 EST:	BPR-	RDU-	5	0572:	Initialized PacketCable defaults
BACC1:	2005 3 16 03:06:19 EST:	BPR-	RDU-	5	0569:	Created default admin user
BACC1:	2005 3 16 03:06:19 EST:	BPR-	RDU-	5	0574:	Loaded 6 license keys
BACC1:	2005 3 16 03:06:20 EST:	BPR-	RDU-	5	0575:	Database initialization completed in 471 msec
BACC1:	2005 3 16 03:06:25 EST:	BPR-	RDU-	3	0015:	Unable to locate manifest file
BACC1:	2005 3 16 03:06:28 EST:	BPR-	RDU-	3	0280:	Command error

**Note**

The KDC is not considered in this log file.

Viewing the dpe.log File

You use the **show log** command, from the DPE CLI, to view the log file. See the *Cisco Broadband Access Center for Cable Command Line Reference* for additional information.

BACC generates log messages from the DHCP server extensions based on the extension trace level setting. You can change the extension trace level using the administrator's user interface.

To do this:

-
- Step 1** Select the DHCP Server, and expand the extension settings.
- Step 2** Reload the DHCP server for the change to take effect.
-

Administrator's User Interface

The BACC administrator's user interface is an HTML-based application from which you can configure and manage users and devices registered with the system. Refer to these chapters for specific instructions on the use of this user interface:

- [Chapter 8, "Understanding the Administrator's User Interface"](#) describes how to access the BACC administrative user interface and explains the various interface components.
- [Chapter 9, "Using the Broadband Access Center for Cable Administrator User Interface"](#) provides instructions for performing administrative activities involving the monitoring of various BACC components.
- [Chapter 10, "Configuring Broadband Access Center for Cable"](#) describes those activities that you perform to configure BACC.

Sample User Interface

The BACC product also comes with a Sample User Interface (SUI) application, which is found in [Chapter 11, "Configuring and Using the Sample User Interface."](#) This is used to demonstrate how BACC can be used to perform both self-provisioning and pre-provisioning, and other basic BACC functions in lab testing scenarios. In full BACC deployments, SUI functionality is expected to be provided by billing, OSS, and/or workflow applications.

**Caution**

The SUI is not intended for use in any live environment and is for demonstration purposes only.



Configuration Workflows and Checklists

This chapter is divided into two major sections that define the processes to follow when configuring BACC components to support various technologies. These sections are:

- [Component Workflows, page 3-1](#)
- [Technology Workflows, page 3-7](#)

Component Workflows

This section describes the workflows you must follow to configure each BACC component for the technologies supported by BACC. These configuration activities are performed before configuring BACC to support specific technologies. In some instances certain procedures may only be applicable to a lab or component installation. In these cases that appropriate indication is made.

The component workflows described in this section are arranged in a checklist format and include:

- [RDU Checklist](#)
- DPE Checklists including:
 - [Hardware DPE Checklist](#)
 - [Solaris DPE Checklist](#)
- [Network Registrar Checklist](#)



Note

Items marked with an asterisk (*) are mandatory tasks or procedures.

RDU Checklist

Table 3-1 identifies the workflow to follow when configuring the RDU.

Table 3-1 RDU Workflow Checklist

No.	Procedure	Refer to...	Installation Type
1.	Configure the system syslog service for use with BACC.	<i>Cisco Broadband Access Center for Cable Installation Guide.</i>	Both
2.	Access the BACC administrative user interface.	“Accessing the BACC Administrators Graphical User Interface” section on page 8-1.	Both
3.	Change the admin password.	“Accessing the BACC Administrators Graphical User Interface” section on page 8-1.	Both
4.	Add the appropriate license keys.	“Managing License Keys” section on page 10-29.	Both
5.	Configure the RDU database backup procedure.	“Backup and Recovery” section on page 7-4.	Component Only
6.	Configure the RDU SNMP agent.	“Using the snmpAgentCfgUtil.sh Command” section on page 12-46.	Component Only

Hardware DPE Checklist

You must perform the activities described in Table 3-2 after those described in Table 3-1, [RDU Workflow Checklist](#).



Note

Items marked with an asterisk (*) are mandatory tasks or procedures.

Table 3-2 Hardware DPE Configuration Checklist

No.	Procedure	Refer to ...	Installation Type
1.	Change the passwords.	“password” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
2.	Configure the system syslog service for use with BACC.	<i>Cisco Broadband Access Center for Cable Installation Guide.</i>	Both
3.	Configure your IP address.*	“interface ethernet 0..1 ip address” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only

Table 3-2 Hardware DPE Configuration Checklist (continued)

No.	Procedure	Refer to ...	Installation Type
4.	Configure the provisioning interface.*	“interface <0..1> provisioning enabled” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
5.	Configure the default hardware gateway.*	“ip default-gateway” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
6.	Configure the provisioning FQDN.	“Automatic FQDN Generation” section on page 10-35 for more information on enabling and configuring the auto generation of FQDNs.	Component Only
7.	Configure the BACC shared secret.*	“dpe shared-secret” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
8.	Configure the DPE to connect to the desired RDU.*	“dpe rdu-server (IP)” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
9.	Configure the network time protocol (NTP).	“ntp server (IP)” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
10.	Configure the primary provisioning group.*	“dpe provisioning-group primary” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
11.	Configure a hostname.*	“hostname” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
12.	Configure a domain name.*	“ip domain-name” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only

Table 3-2 Hardware DPE Configuration Checklist (continued)

No.	Procedure	Refer to ...	Installation Type
13.	Configure a minimum of one name server.*	“ip name-server” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
14.	Configure the required routes to the other BACC components as well as to the devices in the network.	“ip route” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
15.	Configure the DPE SNMP agent.	SNMP Agent Commands section in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
16.	Verify that you are connected to RDU.	“ Viewing Servers ” section on page 9-18 .	Component Only

Solaris DPE Checklist

You must perform the activities described in [Table 3-3](#) after those described in [Table 3-1, RDU Workflow Checklist](#).



Note

This checklist applies to component installation of the Solaris DPE. A lab installation prompts for the required parameters, and automatically configures the selected technologies. Lab installations also use a single SNMP agent to monitor both the DPE and the RDU. You can configure this using either the DPE CLI or the `snmpAgentCfgUtil.sh` tool. See the “[Using the snmpAgentCfgUtil.sh Command](#)” section on [page 12-46](#) for additional information.



Note

Items marked with an asterisk (*) are mandatory tasks or procedures.

Table 3-3 Solaris DPE Configuration Checklist

No.	Procedure	Refer to ...	Installation Type
1.	Configure the system syslog service for use with BACC.	<i>Cisco Broadband Access Center for Cable Installation Guide</i> .	Both
2.	Change the passwords.	“password” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Both

Table 3-3 Solaris DPE Configuration Checklist (continued)

No.	Procedure	Refer to ...	Installation Type
3.	Configure the provisioning interface.*	“interface ethernet [intf0 intf1]” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
4.	Configure the provisioning FQDN.	“Automatic FQDN Generation” section on page 10-35 for more information on enabling and configuring the auto generation of FQDNs.	Component Only
5.	Configure the BACC shared secret.*	“dpe shared-secret” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
6.	Configure the DPE to connect to the desired RDU.*	“dpe rdu-server (IP)” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
7.	Configure the network time protocol (NTP).	See the Solaris documentation for configuration information.	Component Only
8.	Configure the primary provisioning group.*	“dpe provisioning-group primary” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
9.	Configure the required routes to the other BACC components as well as to the devices in the network.	“ip route” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
10.	Configure the DPE SNMP agent.	SNMP Agent Commands section in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> .	Component Only
11.	Verify that you are connected to RDU.	“Viewing Servers” section on page 9-18.	Both

Network Registrar Checklist

You must perform the activities described in [Table 3-4](#) after those described in either [Table 3-2, Hardware DPE Configuration Checklist](#) or [Table 3-3, Solaris DPE Configuration Checklist](#).



Caution

The BACC DHCP option settings always replace any DHCP option values set within Network Registrar.



Note

Items marked with an asterisk (*) are mandatory tasks or procedures.

Table 3-4 Network Registrar Workflow Checklist

No.	Procedure	Refer to ...	Installation Type
1.	Validate the Network Registrar extensions.	<i>Broadband Access Center for Cable Installation Guide</i> for information on configuring valid extensions.	Both
2.	Configure the system syslog service for use with BACC.	<i>Broadband Access Center for Cable Installation Guide</i> for more information on configuring the system syslog service for BACC.	Both
3.	Configure client-classes/scope selection- tags that match those defined in the RDU.*	<i>Network Registrar User's Guide</i> for more information about configuring client-classes and scope-selection-tags.	Both
4.	Configure scopes.*	<i>Network Registrar User's Guide</i> for more information about configuring scopes.	Both
5.	Configure policies.*	<i>Network Registrar User's Guide</i> for more information about configuring policies.	Both
6.	Configure the backup procedure for the Network Registrar database.	<i>Network Registrar User's Guide</i> for more information about backing up the Network Registrar database.	Component Only
7.	Verify that you are connected to the correct RDU.	"Viewing Servers" section on page 9-18.	Both

Technology Workflows

This section describes the activities that you must perform when configuring BACC to support specific technologies. These configuration activities are performed subsequent to configuring BACC components and may, in limited circumstances, apply only to a lab or component installation. In these cases that appropriate indication is made.

The technology workflows described in this section are arranged in a checklist format and include:

- [DOCSIS Checklist, page 3-7](#)
- PacketCable Checklists including:
 - [PacketCable, page 3-8](#)
 - [Non-Secure PacketCable, page 3-10](#)
 - [Euro-PacketCable, page 3-12](#)
- [Non-Secure CableHome Provisioning Checklist, page 3-14](#)


Note

Items marked with an asterisk (*) are mandatory tasks or procedures.

DOCSIS Checklist

You must perform the activities described in the “[Component Workflows](#)” section on [page 3-1](#), in addition to those described in [Table 3-5](#) in order to successfully configure BACC for DOCSIS operation.

Perform the activities in this checklist in the order specified.

Table 3-5 DOCSIS Checklist

Step	Action	Refer to ...
Configure the RDU		
1.	Configure all provisioned DHCP criteria.	“ Configuring DHCP Criteria ” section on page 10-23 for more information.
2.	Configure provisioned classes of service. Add all classes of service that may be used by any provisioned DOCSIS modem.	“ Configuring the Class of Service ” section on page 10-1 .
3.	Configure the promiscuous mode of operation.	“ System Defaults ” section on page 10-21 .
Configure Network Registrar		
1.	Configure client-classes/scope-selection-tags to match those added for the provisioned DOCSIS modem DHCP criteria.	<i>Network Registrar User’s Guide</i> for more information about configuring client-classes and scope-selection-tags.

PacketCable Checklists

BACC supports three different variations of PacketCable. This section identifies the activities that must be performed for each, including:

- [PacketCable, page 3-8](#)
- [Non-Secure PacketCable, page 3-10](#)
- [Euro-PacketCable, page 3-12](#)



Note

The checklists in this section assume that an appropriate PacketCable configuration file and the correct MIBs have been loaded.

PacketCable

You must perform the PacketCable related activities described in [Table 3-6](#) after those described in the “[Component Workflows](#)” section on [page 3-1](#). The PacketCable checklist involves working with almost every BACC component.

Perform the activities in this checklist in the order specified.



Note

The default maximum clock skew between the KDC and the DPEs is 5 minutes.



Note

Items marked with an asterisk (*) are mandatory tasks or procedures.

Table 3-6 *PacketCable Checklist*

Step	Action	Refer to...
Configure the RDU		
1.	Enable the autogeneration of MTA FQDNs.	“ Automatic FQDN Generation ” section on page 10-35 for more information on enabling and configuring the auto generation of FQDNs.
2.	Configure all provisioned DHCP criteria.	“ Configuring DHCP Criteria ” section on page 10-23 for more information.
3.	Configure all provisioned classes of service.	“ Configuring the Class of Service ” section on page 10-1 for more information.
4.	Configure an SNMPv3 cloning key.*	the “ <i>packetcable snmp key-material</i> ” command described in the <i>Cisco Broadband Access Center for Cable Broadband Access Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> . Note This command must be run from the console mode.

Table 3-6 PacketCable Checklist (continued)

Step	Action	Refer to...
Configure the DPE		
1.	Configure a KDC service key.*	the “packetcable registration kdc-service-key” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
2.	Configure a privacy policy.*	the “packetcable registration policy-privacy” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
3.	Configure an SNMPv3 cloning key.*	the “packetcable snmp key-material” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> . Note This command must be run from the console mode.
4.	Enable PacketCable.*	the “packetcable enable” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
5.	Configure the optional MTA file encryption.	the “packetcable registration encryption” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
Configure the KDC		
1.	Obtain a KDC license from your Cisco representative and copy that file to the <BACC_HOME>/kdc directory.	“KDC Licenses” section on page 2-12.
2.	Configure a certificate chain.	“Using the PKCert.sh Tool to Manage KDC Certificates” section on page 12-41.
3.	Configure a service key pair for each DPE’s provisioning FQDN.	“Using the Keygen Tool” section on page 12-45.
4.	Configure service keys for the ticket-granting-ticket (TGT).	“Using the Keygen Tool” section on page 12-45.
5.	Configure service keys for the Call Management Server.	“Using the Keygen Tool” section on page 12-45.
6.	Configure network time protocol (NTP).	Solaris documentation for more information on configuring NTP for Solaris.

Table 3-6 PacketCable Checklist (continued)

Step	Action	Refer to...
Configure DHCP		
1.	Configure all necessary PacketCable voice technology properties.	“Using the <code>changeNRProperties.sh</code> Tool” section on page 12-43.
2.	Configure dynamic DNS for the MTA scopes.	<i>Network Registrar User’s Guide</i> for more information on configuring dynamic DNS.
3.	Configure client-classes/scope-selection-tags that match those defined in the RDU.*	<i>Network Registrar User’s Guide</i> for more information about configuring client-classes and scope-selection-tags.
Configure DNS		
1.	Configure dynamic DNS for each DHCP server.	<i>Network Registrar User’s Guide</i> for more information on configuring dynamic DNS.
2.	Configure a zone for the KDC realm.	<i>Network Registrar User’s Guide</i> for more information on configuring zones.
3.	Configure an SRV record for the KDC.	“Configuring the SRV Record in the Network Registrar DNS Server” section on page 10-34 and the <i>Network Registrar User’s Guide</i> for more information on configuring SRV records.
4.	Configure records for the KDC and DPE provisioning interface names.	<i>Network Registrar User’s Guide</i> for more information on configuring records.
Note	Cisco recommends that the DNS procedure be used to configure a reverse zone for the DNS server’s IP address. Some DNS clients, including nslookup, attempt to resolve the DNS server IP address to an FQDN. This may fail to retrieve any records from the DNS unless the reverse zone is present and properly configured.	

Non-Secure PacketCable

You must perform the PacketCable related activities described in [Table 3-7](#) after those described in the “Component Workflows” section on page 3-1. The non-secure PacketCable checklist involves working with almost every BACC component.

Perform the activities in this checklist in the order specified.



Note The default maximum clock skew between the KDC and the DPEs is 5 minutes.



Note Items marked with an asterisk (*) are mandatory tasks or procedures.

Table 3-7 Non-Secure PacketCable Checklist

Step	Action	Refer to...
Configure the DPE		
1.	Configure a KDC service key.*	the “packetcable registration kdc-service-key” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
2.	Configure a privacy policy.*	the “ipacketcable registration policy-privacy” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
3.	Configure an SNMPv3 cloning key.*	the “packetcable snmp key-material” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> . Note This command must be run from the console mode.
4.	Enable PacketCable.*	the “packetcable enable” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
Configure DHCP		
1.	Configure all necessary non-secure PacketCable voice technology properties.	Using the changeNRProperties.sh Tool, page 12-43.
2.	Configure dynamic DNS for the MTA scopes.	<i>Network Registrar User's Guide</i> for more information on configuring dynamic DNS.
3.	Configure client-classes/scope-selection-tags that match those defined in the RDU.*	<i>Network Registrar User's Guide</i> for more information about configuring client-classes and scope-selection-tags.
Configure DNS		
1.	Configure dynamic DNS for each DHCP server.	<i>Network Registrar User's Guide</i> for more information on configuring dynamic DNS.
Note	Cisco recommends that the DNS procedure be used to configure a reverse zone for the DNS server's IP address. Some DNS clients, including nslookup, attempt to resolve the DNS server IP address to an FQDN. This may fail to retrieve any records from the DNS unless the reverse zone is present and properly configured.	

Euro-PacketCable

You must perform the Euro-PacketCable related activities described in [Table 3-8](#) after those described in the “[Component Workflows](#)” section on [page 3-1](#). The Euro- PacketCable checklist involves working with almost every BACC component.

Perform the activities in this checklist in the order specified.



Note

Items marked with an asterisk (*) are mandatory tasks or procedures.

Table 3-8 Euro-PacketCable Checklist

Step	Action	Refer to...
Configure the RDU		
1.	Enable the autogeneration of MTA FQDNs.	“ Automatic FQDN Generation ” section on page 10-35 for more information on enabling and configuring the auto generation of FQDNs.
2.	Configure all provisioned DHCP criteria.	“ Configuring DHCP Criteria ” section on page 10-23 for more information.
3.	Configure all provisioned classes of service.	“ Configuring the Class of Service ” section on page 10-1 for more information.
4.	Configure an SNMPv3 cloning key.*	the “packetcable snmp key-material” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> . Note This command must be run from the console mode.
Configure the DPE		
1.	Configure a KDC service key.*	the “packetcable registration kdc-service-key” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
2.	Configure a privacy policy.*	the “ipacketcable registration policy-privacy” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
3.	Configure an SNMPv3 cloning key.*	the “packetcable snmp key-material” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> . Note This command must be run from the console mode.

Table 3-8 Euro-PacketCable Checklist (continued)

Step	Action	Refer to...
4.	Enable PacketCable.*	the “packetcable enable” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
5.	Configure the optional MTA file encryption.	the “packetcable registration encryption” command described in the <i>Cisco Broadband Access Center for Cable Command Line Interface Reference</i> and the <i>Cisco Broadband Access Center for Cable Installation Guide</i> .
Configure the KDC		
1.	Obtain a KDC license from your Cisco representative and copy that file to the <BACC_HOME>/kdc directory.	“KDC Licenses” section on page 2-12.
2.	Configure a certificate chain using the -e switch in the PKCert.sh tool.	“Using the PKCert.sh Tool to Manage KDC Certificates” section on page 12-41.
3.	Configure a service key pair for each DPE’s provisioning FQDN.	“Using the Keygen Tool” section on page 12-45.
4.	Configure service keys for the ticket-granting-ticket (TGT).	“Using the Keygen Tool” section on page 12-45.
5.	Configure service keys for the Call Management Server.	“Using the Keygen Tool” section on page 12-45.
6.	Configure network time protocol (NTP).	Solaris documentation for more information on configuring NTP for Solaris.
Configure DHCP		
1.	Configure all necessary PacketCable voice technology properties.	“Using the changeNRProperties.sh Tool” section on page 12-43.
2.	Configure dynamic DNS for the MTA scopes.	<i>Network Registrar User’s Guide</i> for more information on configuring dynamic DNS.
3.	Configure client-classes/scope-selection-tags that match those defined in the RDU.*	<i>Network Registrar User’s Guide</i> for more information about configuring client-classes and scope-selection-tags.
Configure DNS		
1.	Configure dynamic DNS for each DHCP server.	<i>Network Registrar User’s Guide</i> for more information on configuring dynamic DNS.
2.	Configure a zone for the KDC realm.	<i>Network Registrar User’s Guide</i> for more information on configuring zones.
3.	Configure an SRV record for the KDC.	“Configuring the SRV Record in the Network Registrar DNS Server” section on page 10-34 and the <i>Network Registrar User’s Guide</i> for more information on configuring SRV records.
4.	Configure records for the KDC and DPE provisioning interface names.	<i>Network Registrar User’s Guide</i> for more information on configuring records.

Table 3-8 Euro-PacketCable Checklist (continued)

Step	Action	Refer to...
Note	Cisco recommends that the DNS procedure be used to configure a reverse zone for the DNS server's IP address. Some DNS clients, including nslookup, attempt to resolve the DNS server IP address to an FQDN. This may fail to retrieve any records from the DNS unless the reverse zone is present and properly configured.	

Non-Secure CableHome Provisioning Checklist

You must perform the activities described in the “[Component Workflows](#)” section on page 3-1, in addition to those described in [Table 3-9](#) to successfully configure BACC for non-secure CableHome provisioning operation.

Perform the activities in this checklist in the order specified.

Table 3-9 Non-secure CableHome Provisioning Checklist

Step	Action	Refer to ...
Configure the RDU		
1.	Configure provisioned DHCP criteria. Add all the DHCP criteria that will be used by the non-secure CableHome devices you will provision.	“ Configuring DHCP Criteria ” section on page 10-23.
2.	Configure provisioned classes of service. Add all classes of service that may be used by any provisioned non-secure CableHome device.	“ Configuring the Class of Service ” section on page 10-1.
3.	Configure the promiscuous mode of operation.	“ System Defaults ” section on page 10-21.
Configure Network Registrar		
1.	Configure the client-classes/scope-selection-tags to match those added for the provisioned non-secure CableHome DHCP criteria.	<i>Network Registrar User's Guide</i> for more information about configuring client-classes and scope-selection tags.



DOCSIS Configuration

This chapter identifies those functions you must check or configure to bring a BACC DOCSIS deployment into operation. It also provides information required prior to configuration, and describes the available tools.

- [DOCSIS Workflow, page 4-2](#)
- [Using MIBs with Dynamic DOCSIS Templates, page 4-4](#)
- [BACC Features for DOCSIS Configurations, page 4-5](#)
- [Troubleshooting DOCSIS Networks, page 4-6](#)
- [Network Layer Configuration and Above, page 4-7](#)
- [Troubleshooting Cable Modem States, page 4-10](#)



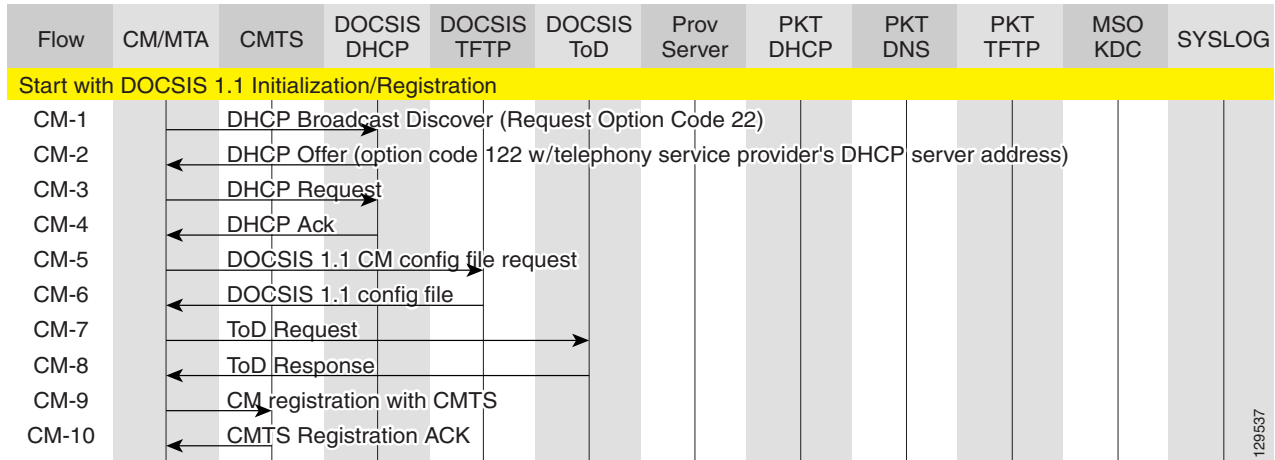
Note

See the “[DOCSIS Option Support](#)” section on [page 12-12](#) for information concerning the DOCSIS options supported by this BACC release.

DOCSIS Workflow

The provisioning (workflow) steps contained in the DOCSIS Provisioning Specification are shown in [Figure 4-1](#).

Figure 4-1 DOCSIS Provisioning Flow



[Table 4-1](#) describes the potential problems that can exist at various DOCSIS provisioning steps illustrated in [Figure 4-1](#).

Table 4-1 DOCSIS Workflow Description

Step	DOCSIS Workflow step	Potential Problems
CM-1	DHCP Discover	<ul style="list-style-type: none"> • The init(d) state • No addresses available • Verify BACC shared-secret is correct • Incorrectly configured Class of Service • DOCSIS template parsing errors (invalid option, include file - not found, and so on) <p>CNR DHCP</p> <ul style="list-style-type: none"> • Incorrect CNR DHCP configuration • Verify DHCP server is in the provisioning group <p>BACC CNR Extension</p> <ul style="list-style-type: none"> • BACC CNR extension cannot contact DPEs • BACC CNR extension fails to find any DPEs in provisioning group • Verify extensions are connected to the RDU • BACC CNR extension gets DPE cache miss, sends request to RDU <p>RDU</p> <ul style="list-style-type: none"> • No appropriate scopes defined (or not match BACC RDU configuration) • Verify RDU IP address is correct • Verify RDU port is correct (default 49187) • Verify the RDU can be pinged from the DPE • Configuration generation is failing at the RDU • RDU licenses exceeded, not configured • Device detection is failing at the RDU <p>DPE</p> <ul style="list-style-type: none"> • Verify DPEs assigned to the provisioning group • Verify DPEs can be pinged from the DHCP server • Verify DPE interfaces enabled for provisioning
CM-2	DHCP Offer	Routing issues between DHCP and CMTS
CM-3	DHCP Request	init(i) state DHCP server did not provide all the parameters required
CM-4	DHCP Ack	

Table 4-1 DOCSIS Workflow Description (continued)

Step	DOCSIS Workflow step	Potential Problems
CM-5	TFTP Request	<ul style="list-style-type: none"> • init(o) state • routing issues between CMTS and DPE • No route from TFTP server (DPE) to modem • DPE cache miss (static file, and RDU down or doesn't have file) • File not found at TFTP server (DPE) • DPE cache miss (dynamic file) • DPE IP validation failure (for example, the IP address of the device is not what was expected, the Dynamic Shared Secret is enabled on CMTS, or a hacker is spoofing as a DOCSIS modem.)
CM-6	TFTP Response	Routing issues between DPE and CMTS
CM-7	ToD Request	init(t) state - No route from time server (DPE) to Modem
CM-8	ToD Response	
CM-9	CM register with CMTS	<ul style="list-style-type: none"> • reject(m) - * CMTS shared secret mismatch with BACC or DPE DSS shared secret • reject(c) - * delivered incorrect DOCSIS configuration file (1.1 file to 1.0 CM)
CM-10	CMTS registration Ack	Acceptable states are: <ul style="list-style-type: none"> • online • online(d) • online(pk) • online(pt)

Using MIBs with Dynamic DOCSIS Templates

For a full list of MIBs shipped with BACC, see [SNMP Varbind](#), page 12-5.

Two versions of the DOCSIS MIB are loaded into the RDU:

- DOCS-CABLE-DEVICE-MIB-OBSOLETE (experimental branch)
- DOCS-CABLE-DEVICE-MIB (mib2 branch)

For information on how to use them, see [DOCSIS MIBs](#), page 12-5.

You can add MIBs using an API call or by modifying rdu.properties. For more details, see [Configuring Euro-PacketCable MIBs](#), page 5-6.

You can add a SNMP TLV to a template when no MIB is available. See [Adding SNMP TLVs Without a MIB](#), page 12-10 for more information.

BACC Features for DOCSIS Configurations

This section describes BACC value-added features as they relate to the DOCSIS technology.

Dynamic Configuration TLVs

These TLVs are added by the DPE whenever a TFTP request for dynamic DOCSIS configuration is received:

- TLV 19: TFTP Server Timestamp (optional)—Displayed in the Configure DOCSIS Defaults page as the TFTP Time Stamp Option. See [Table 10-5 on page 10-14](#) for more information. This TLV requires NTP synchronization on CMTS and DPE.
- TLV 20: TFTP Server Provisioned Modem Address (optional)—Displayed in the Configure DOCSIS Defaults page as the TFTP Modem Address Option. See [Table 10-5 on page 10-14](#) for more information.
- TLV 6: CM MIC Configuration Setting (required)
- TLV 7: CMTS MIC Configuration Setting (required)—Displayed in the Configure DOCSIS Defaults page as the CMTS Shared Secret. See [Table 10-5 on page 10-14](#) for more information.

**Note**

When configuring CMTS MIC, note the following CMTS IOS release dependencies:

- DOCSIS 2.0 CMTS MIC requires CMTS IOS 12.3BC when including TLV 39 or TLV 40.
- Certain CMTS IOS commands are assumed to be configured by BACC:
 - ip dhcp relay information option
 - no ip dhcp relay information check
 - cable helper-address <x.x.x.x>
 - cable dhcp-giaddr primary

DPE TFTP IP Validation

For dynamic configuration files, the DPE TFTP server verifies that the IP address of the TFTP client matches the expected DOCSIS cable modem IP address. If it does not match, the request is dropped. This feature is incompatible with the Cisco CMTS DMIC feature.

Use the `no tftp verify-ip` command to disable the verification of requestor IP addresses on dynamic configuration TFTP requests:

```
dpe# no tftp verify-ip
%OK
```

DOCSIS 1.0, 1.1, 2.0 Support

BACC 2.7 supports DOCSIS 1.0, 1.1, and 2.0. See [Chapter 12, “Broadband Access Center for Cable Support Tools and Advanced Concepts”](#) for additional information describing the TLVs and options supported in each DOCSIS version by BACC.

Dynamic DOCSIS Version Selection

BACC can detect a cable modem’s DOCSIS version from an incoming DHCP request. It can also detect the CMTS DOCSIS version from any customer-supplied source that provides a mapping of GIADDR to DOCSIS version numbers.

You can specify the DOCSIS version in the Configuration File utility. See [Using the Configuration File Utility, page 12-25](#) for more information.

DOCSIS Configuration File Based on DOCSIS Version

The default DOCSIS code generation extension uses the version of DOCSIS for provisioning the modem, to determine the filename of the DOCSIS configuration file to send to the modem.

DOCSIS Version Dependent Class of Service Properties

The following new class of service properties are supported by the Administrator’s user interface and by the BACC API:

```
/cos/docsis/file/1.0  
/cos/docsis/file/1.1  
/cos/docsis/file/2.0
```

These properties can optionally be added to a DOCSIS class of service to associate a DOCSIS configuration filename with a particular DOCSIS version. Each of these properties (if set) causes the RDU to establish a database relationship between the class of service and the file named by the property value, as is done for the existing DOCSIS configuration filename property.

Troubleshooting DOCSIS Networks

DOCSIS provides the bandwidth and latency guarantees needed to provide toll-quality voice, data services, and multimedia applications across a shared HFC network. It is designed to be backward compatible, enabling DOCSIS 1.0, 1.1, and 2.0 modems to operate in the same spectrum on the same network.



Note

The bulk of this section discusses the DOCSIS technology with respect to BACC and the Cisco uBR7246 CMTS. The examples are for the Cisco uBR7246 CMTS; however the content is relative to the DOCSIS standard, which other CMTS vendors must support.

Network Layer Configuration and Above

After ranging is successful, the cable modem needs additional configuration from the operational support servers. The collection of services they provide and configure are called operational support services (OSS).



Note

At this point, the modem is functional; it could act like a bridge and be a fairly quick transmission device, but there would be no administrator configuration, diagnosis, class of services, security, privacy, or remote software upgrade capability, *and therefore* it would not be DOCSIS compliant.

Establish DHCP State

```
MAC State --->>> 'establish_dhcp_state'
cmWriteFlashFile("CM_MACCONFIG", 0x8313db6, 0x12e) by TID 0x82eb0f8 (tConfigNV)
usrEraseSysFlash(1, 0x1e0400, 0x12e) by TID 0x82eb0f8 (tConfigNV)
.....cmWriteFlashFile("CM_DHCPLEASE", 0x810ebe4, 0x24) by TID 0x82d825c (tLease-0)
usrEraseSysFlash(1, 0x1e0200, 0x24) by TID 0x82d825c (tLease-0)
```

The first task in configuring the OSS is acquiring network configuration via DHCP. A broadcast DHCP DISCOVER message is issued by the modem. Normally, routers will not forward broadcast messages, but they can be configured to do so for a collection of UDP protocols, one of which is DHCP. If a DHCP server responds to the DISCOVER with an OFFER, the modem may choose to send a REQUEST for the offered configuration.

The DHCP server can respond with an ACK (acknowledged) or NAK (not acknowledged). A NAK may be the result of an incompatible IP address and gateway address, which might occur if a modem hopped from one downstream channel to another that resides on a different subnet. When the modem seeks renewal of the lease, the IP address and the gateway address of the DHCP REQUEST message will be different network numbers and the DHCP server will refuse the REQUEST with a NAK. These situations are rare and the modem will simply release the lease and start over with a DHCP DISCOVER message.

Frequently, errors at `establish_dhcp_state` manifest as timeouts rather than NAKs. To troubleshoot effectively:

- Start at the lower layers and work up.
- Start locally and work towards remote possibilities.
- Look at the modem, before diagnosing the DHCP server via DHCP logs.

The log mask `0xffff00bf` used above in ranging can also help you to troubleshoot almost all the problems a modem might encounter in bring-up stage and is a good place to start when troubleshooting a bring-up error. The order of DHCP messages should be DISCOVER, OFFER, REQUEST, ACK. If the modem is transmitting a DISCOVER with no OFFER response from the DHCP server, turn on UDP debugging on the uBR, using the following command:

```
uBR# debug ip udp
```



Note

If connected to the router via a telnet session or any session other than a console session, the **terminal monitor** command (abbreviated **term mon**) is necessary to view debug messages. To turn off **term mon**, use the **term no mon** command or simply logout.

**Caution**

Running debug commands on a uBR7246 with more than a few modems may cause the uBR7246 to halt the system in order to keep up with the debugging. In this case, all the modems may lose sync and debugging will be useless.

If there are no packets in the debug messages, check the configuration of the “ip helper-address” statement on the cable interface to which this modem is attached. If this is configured correctly and a packet trace of the DHCP server subnet also reveals no DHCP packets from the modem, then look at the output errors of the modem’s cable interface, or the input errors of the cable interface of the uBR. It might be a good idea to boost the transmitter power a bit more with more attenuation.

If packets are being transmitted onto the DHCP server subnet, double-check the modem debug messages to see if there are parameter request or assignment errors. At this stage of troubleshooting you should investigate the routing between the modem and the DHCP server. It is also advisable to double-check the DHCP server configuration.

Establishing the ToD State

```
MAC State --->>> 'establish_tod_state'
```

After a modem has acquired its network parameters, it must request the time of day from a Time Of Day (TOD) server. TOD uses a UTC timestamp (seconds from January 1, 1970) combined with the time offset option value from DHCP to calculate the current time. The time is used for syslog and event log timestamps.

Time of day errors almost always point to a DHCP misconfiguration. Possible misconfigurations that can result in TOD errors are IP address, gateway address misconfigurations, or the wrong TOD server address. Again, start troubleshooting locally by examining the DHCP parameters the modem has stored, using the following command:

```
-> cmShowDhcpParameters
```

or:

```
-> cmAddLogValues ("SEV_ALL FAC_DHCP")
```

The latter will give debug output of DHCP actions and tasks.

Also check the modem routing table with the vxWorks **routeShow** command.

Security Association State

```
MAC State --->>> 'security_association_state'
```

This is a placeholder for a state yet to be defined. It is envisioned that a security association with a security server will provide IPsec-like security for the modems. Its design and implementation are still being discussed. In the meantime, DOCSIS 1.0 requires modems to support a future definition of this state including the DHCP option for a security server. It is unlikely to find a modem having a problem with this state until it is defined in DOCSIS and implemented on the modem.

Configuration File State

The main configuration and administration interface to the CM is the configuration file downloaded from the provisioning server. This configuration file contains downstream channel and upstream channel identification and characteristics, as well as class of service settings, baseline privacy settings, general operational settings, network management information, software upgrade fields, filters, and vendor specific settings.

Common reasons for failure at this state are missing file, wrong file permissions, TFTP server is unreachable, file is wrong format, file has missing required options, misconfigured required options, or incorrect options, for example, unknown or invalid TLVs.

A debug command for configuration file transactions and parsing is:

```
-> cmAddLogValues ("SEV_ALL FAC_CONFIGFILE")
```

Registration State

```
MAC State --->>> 'registration_state'
```

After configuration, the modem sends a registration request (REG-REQ) with a required subset of the configuration settings, as well as the CM and CMTS message integrity checks (MIC). The CM MIC is a hashed calculation over the configuration file settings which provides a method for the modem to be sure the configuration file was not tampered with in transit. The CMTS MIC is similar but also includes a setting for a shared-secret authentication string. This shared secret is known by the CMTS and the provisioning server, and ensures that only modems configured by authenticated provisioning servers will be allowed to register with the CMTS.

Problems with registration state almost always point to a configuration file error. Make sure the modem and the CMTS both support the settings in the configuration file. Make sure the CMTS allows the creation of class of service profiles or use a profile created by the CMTS. Check the authentication strings in the CMTS cable interface configuration and the configuration file.

Establish Privacy State

```
MAC State --->>> 'establish_privacy_state'
```

The modem must negotiate baseline privacy with the CMTS through the Baseline Privacy Key Management protocol, if all of the following are true:

- The modem software supports baseline privacy.
- The class of service is a privacy enabled profile.
- Baseline privacy settings are present in the configuration file.

If errors occur in this state, the likely causes are configuration file misconfigurations. Be certain that both the CM and the CMTS support baseline privacy and are enabled in the interface configuration for the CMTS and the configuration file for the CM.

You might also encounter errors due to encryption export restrictions, some vendor modems may require the following command on the uBR7246 in the interface configuration:

```
uBR(config-if)# cable privacy 40-bit-des
```

Operational State

```
MAC State --->>> 'operational_state' cmWriteFlashFile("CM_BOOT", 0x8109180, 0x13e) by TID
0x83049b8 (tMACCtrl)
usrEraseSysFlash(1, 0x1e0900, 0x13e) by TID 0x83049b8 (tMACCtrl)
```

If registration and baseline privacy negotiation (if required) succeed, the modem is operational and is ready to pass traffic.

A modem may become operational but not remain in operational state. This could be caused by sync loss or DHCP lease renewal failure, for example.

Troubleshooting Cable Modem States

This section discusses these cable modem states:

- [Online State, page 4-10](#)
- [Offline State, page 4-11](#)
- [Ranging Process - init\(r1\),init\(r2\), and init\(rc\) state, page 4-16](#)
- [DHCP - init\(d\) state, page 4-18](#)
- [DHCP - init\(i\) state, page 4-20](#)
- [TOD exchange- init\(t\) state, page 4-23](#)
- [Option File Transfer Started-init\(o\) State, page 4-25](#)
- [Online, Online\(d\), Online\(pk\), Online\(pt\) state, page 4-27](#)
- [Reject\(pk\) and Reject\(pt\) state, page 4-30](#)
- [Registration - reject \(m\) state, page 4-31](#)
- [Registration - reject \(c\) state, page 4-32](#)

Online State

This section provides information on understanding the *online state* of the cable modem, and troubleshooting problems indicated by this state, including RF problems.



Note

Many but not all of the problems addressed in this section are RF-related. You should review this section to generally understand troubleshooting the problems between the Cisco uBR7246VXR and the CMs in its domain.

The first and most useful command to use at the Cisco uBR7246VXR is **show cable modem**:

```
#show cable modem
```

Interface	Prim Sid	Online State	Timing Offset	Rec Power	QoS	CPE	IP address	MAC address
Cable2/0/U0	5	online(pt)	2290	-0.25	5	0	10.1.1.25	0050.7366.2223
Cable2/0/U0	6	offline	2287	-0.25	2	0	10.1.1.26	0050.7366.2221

The Online State field identifies the status. Table 4-2 displays the possible values for the state.

Table 4-2 State Values

State Value	Description
offline	Cable modem considered offline
init (r1)	Cable modem sent initial ranging
init (r2)	Cable modem is ranging
init (rc)	Cable modem ranging complete
init (d)	DHCP request received
init (i)	DHCP reply received; IP address assigned
init (t)	TOD exchange started
init (o)	Option file transfer started
online	Cable modem registered, enabled for data
online(d)	Cable modem registered, but network access for the cable modem is disabled
online(pk)	Cable modem registered, BPI enabled and KEK assigned
online(pt)	Cable modem registered, BPI enabled and TEK assigned
reject (pk)	KEK modem key assignment rejected
reject (pt)	TEK modem key assignment rejected
reject (m)	Cable modem did attempt to register; registration was refused due to bad MIC (Message Integrity Check)
reject (c)	Cable modem did attempt to register; registration was refused due to bad COS (Class of Service)

The following sections discuss each *State* value, the possible causes, and the steps that can be taken to arrive at the correct state (online).

Offline State

In some cases the cable modem may cycle through other states then back to offline. The following list gives the most common reasons for a cable modem not to achieve QAM lock:

- Weak carrier signal (too much noise).
- Incorrect downstream center frequency.
- Incorrect frequency specified in the DOCSIS file.
- Absence of downstream digital QAM modulated signal.
- Incorrect frequency specified in **cable modem change-frequency** on the Cisco uBR7246VXR.

The following is a portion of the output from **show controllers cable-modem 0** entered at the modem:

```
#sh cont c 0
```

```
BCM Cable interface 0:
CM unit 0, idb 0x8086C88C, ds 0x8086E460, regaddr = 0x2700000, reset_mask 0x80
station address 0030.96f9.65d9 default station address 0030.96f9.65d9
PLD VERSION: 1
```

```
Concatenation: ON Max bytes Q0: 2000 Q1: 2000 Q2: 2000 Q3: 2000

MAC State is ds_channel_scanning_state, Prev States = 3
MAC mcfilter 01E02F00 data mcfilter 00000000

MAC extended header ON
DS: BCM 3300 Receiver: Chip id = BCM3300
US: BCM 3300 Transmitter: Chip id = 3300

Tuner: status=0x00
Rx: tuner_freq 529776400, symbol_rate 5361000, local_freq 11520000
    snr_estimate 166(TenthdB), ber_estimate 0, lock_threshold 26000
    QAM not in lock, FEC not in lock, qam_mode QAM_64 (Annex B)
Tx: tx_freq 27984000, symbol_rate 8 (1280000 sym/sec)
    power_level: 6.0 dBmV (commanded)
        7 (gain in US AMP units)
        63 (BCM3300 attenuation in .4 dB units)
...[Rest of the displayed output is omitted]
```

The Signal to Noise ratio (SNR) estimate is 16.6 dB in the previous output example. Ideally this should be at least 30dB for the CM to operate properly for 64 QAM. See the RF specifications for DOCSIS Downstream and Upstream signals. In some cases you may have a good SNR (for example, 34dB) but still have noise present, such as impulse noise. This can be detected only by a spectrum analyzer operating in the zero span mode. One indication of impulse noise is errors that cannot be corrected. These are seen in the output of **show interfaces cable 2/0 upstream 0** as shown in the following output:

```
#show interfaces cable 2/0 upstream 0

Cable2/0: Upstream 0 is up
Received 46942 broadcasts, 0 multicasts, 205903 unicasts
0 discards, 12874 errors, 0 unknown protocol
```

**Note**

If a value is greater than 1 in 10,000, most likely impulse noise is present.

```
252845 packets input, 1 uncorrectable
12871 noise, 0 microreflections Total Modems On This Upstream Channel : 3 (3 active)
Default MAC scheduler
Queue[Rng Polls] 0/64, fifo queueing, 0 drops
Queue[Cont Mslots] 0/104, fifo queueing, 0 drops
Queue[CIR Grants] 0/64, fair queueing, 0 drops
Queue[BE Grants] 0/64, fair queueing, 0 drops
Queue[Grant Shpr] 0/64, calendar queueing, 0 drops
Reserved slot table currently has 0 CBR entries
Req IEs 77057520, Req/Data IEs 0
Init Mtn IEs 1194343, Stn Mtn IEs 117174
Long Grant IEs 46953, Short Grant IEs 70448
Avg upstream channel utilization : 1%
Avg percent contention slots : 96%
Avg percent initial ranging slots : 4%
Avg percent minislots lost on late MAPs : 0%
Total channel bw reserved 0 bps
CIR admission control not enforced
Current minislot count : 7192093 Flag: 0
Scheduled minislot count : 7192182 Flag: 0
```

The optimal input power level at the CM is 0dBmV; the receiver has a range of -15dBmV to +15dBmV. This can be measured by the spectrum analyzer. If the power is too low you may need to configure the upconverter as described in the *Cisco uBR 7246 Hardware Installation Guide*. If the signal is too strong, you may need to add more attenuation at the high frequency port connection. If a particular frequency has too much noise present, you may need to select another frequency in the spectrum.

To confirm that the CM has not been able to achieve QAM lock, turn on **debug cable-modem mac log verbose**. You should see output similar to the following:

```

2d18h: 239198.516          CMAC_LOG_LINK_DOWN
2d18h: 239198.516    CMAC_LOG_LINK_UP
2d18h: 239198.516    CMAC_LOG_STATE_CHANGE          ds_channel_scanning_state
2d18h: 239198.520    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 99/805790200/99770
2d18h: 239198.520    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 98/601780000/79970
2d18h: 239198.520    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 97/403770100/59570
2d18h: 239198.524    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 96/73753600/115750
2d18h: 239198.524    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 95/217760800/39770
2d18h: 239198.528    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 94/121756000/16970
2d18h: 239198.528    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 93/175758700/21170
2d18h: 239198.528    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 92/79753900/857540
2d18h: 239198.532    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 91/55752700/677530
2d18h: 239198.532    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 90/177000000/21300
2d18h: 239198.532    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 89/219000000/22500
2d18h: 239198.536    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 88/141000000/17100
2d18h: 239198.536    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 87/135012500/13500
2d18h: 239198.540    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 86/123012500/12900
2d18h: 239198.540    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 85/405000000/44700
2d18h: 239198.540    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 84/339012500/39900
2d18h: 239198.544    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 83/333025000/33300
2d18h: 239198.544    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 82/231012500/32700
2d18h: 239198.544    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 81/111025000/11700
2d18h: 239198.548    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 80/93000000/105000
2d18h: 239198.548    CMAC_LOG_WILL_SEARCH_DS_FREQUENCY_BAND 79/453000000/85500

NOTE—unable to lock on:

2d18h: 239198.552    CMAC_LOG_WILL_SEARCH_SAVED_DS_FREQUENCY 453000000
2d18h: 239199.672    CMAC_LOG_DS_NO_QAM_FEC_LOCK          453000000
2d18h: 239200.788    CMAC_LOG_DS_NO_QAM_FEC_LOCK          453000000
2d18h: 239201.904    CMAC_LOG_DS_NO_QAM_FEC_LOCK          453000000
2d18h: 239203.020    CMAC_LOG_DS_NO_QAM_FEC_LOCK          459000000

```

Another reason the CM might not achieve QAM lock is if the incorrect downstream center frequency was configured on the upconverter. For example, on the NTSC frequency map for standard 6-MHz channel bands in North America, channel 100-100 uses 648.0-654.0 with a center frequency of 651 MHz. However, if you are using an upconverter that is designed to work with the video carrier frequency (for example, GI C6U which is 1.75MHz below the center frequency), you must set a frequency of 649.25 MHz for Channel 100-100.

Another common mistake is to specify an incorrect frequency value in the *Downstream Frequency* field under the Radio Frequency Info in the Cisco Broadband Configurator (available for download on cisco.com). Usually there is no need to specify a frequency value under this option. However, if there is a need (for example, when certain modems need to lock on a different frequency), you must select the

proper frequency values, as explained previously. The following debug output illustrates this: a CM locks on initially at 453MHz and then at 535.25MHz (which was specified in the DOCSIS configuration file), thus causing the cable modem to reset and cycle through this process indefinitely:

```

4d00h: 345773.916 CMAC_LOG_WILL_SEARCH_SAVED_DS_FREQUENCY 453000000
4d00h: 345774.956 CMAC_LOG_UCD_MSG_RCVD
4d00h: 345775.788 CMAC_LOG_DS_64QAM_LOCK_ACQUIRED 453000000
4d00h: 345775.792 CMAC_LOG_DS_CHANNEL_SCAN_COMPLETED
4d00h: 345775.794 CMAC_LOG_STATE_CHANGE wait_ucd_state
4d00h: 345776.946 CMAC_LOG_UCD_MSG_RCVD 1
4d00h: 345778.960 CMAC_LOG_UCD_MSG_RCVD 1
4d00h: 345778.962 CMAC_LOG_ALL_UCDS_FOUND
4d00h: 345778.966 CMAC_LOG_STATE_CHANGE wait_map_state
4d00h: 345778.968 CMAC_LOG_FOUND_US_CHANNEL 1
4d00h: 345780.996 CMAC_LOG_UCD_MSG_RCVD 1
4d00h: 345781.000 CMAC_LOG_UCD_NEW_US_FREQUENCY 27984000
4d00h: 345781.004 CMAC_LOG_SLOT_SIZE_CHANGED 8
4d00h: 345781.084 CMAC_LOG_UCD_UPDATED
4d00h: 345781.210 CMAC_LOG_MAP_MSG_RCVD
4d00h: 345781.212 CMAC_LOG_INITIAL_RANGING_MINISLOTS 40
4d00h: 345781.216 CMAC_LOG_STATE_CHANGE ranging_1_state
4d00h: 345781.220 CMAC_LOG_RANGING_OFFSET_SET_TO 9610
4d00h: 345781.222 CMAC_LOG_POWER_LEVEL_IS 22.0 dBmV (comma)
4d00h: 345781.226 CMAC_LOG_STARTING_RANGING
4d00h: 345781.228 CMAC_LOG_RANGING_BACKOFF_SET 0
4d00h: 345781.232 CMAC_LOG_RNG_REQ_QUEUED 0
4d00h: 345781.272 CMAC_LOG_RNG_REQ_TRANSMITTED
4d00h: 345781.280 CMAC_LOG_RNG_RSP_MSG_RCVD
4d00h: 345781.282 CMAC_LOG_RNG_RSP_SID_ASSIGNED 3
4d00h: 345781.284 CMAC_LOG_ADJUST_RANGING_OFFSET 2288
4d00h: 345781.288 CMAC_LOG_RANGING_OFFSET_SET_TO 11898
4d00h: 345781.292 CMAC_LOG_ADJUST_TX_POWER 7
4d00h: 345781.294 CMAC_LOG_POWER_LEVEL_IS 24.0 dBmV (comma)
4d00h: 345781.298 CMAC_LOG_STATE_CHANGE ranging_2_state
4d00h: 345781.302 CMAC_LOG_RNG_REQ_QUEUED 3
4d00h: 345782.298 CMAC_LOG_RNG_REQ_TRANSMITTED
4d00h: 345782.300 CMAC_LOG_RNG_RSP_MSG_RCVD
4d00h: 345782.304 CMAC_LOG_RANGING_SUCCESS
4d00h: 345782.316 CMAC_LOG_STATE_CHANGE dhcp_state
4d00h: 345782.450 CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS 10.1.1.25
4d00h: 345782.452 CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS 10.10.20.1
4d00h: 345782.456 CMAC_LOG_DHCP_TOD_SERVER_ADDRESS 10.10.20.2
4d00h: 345782.460 CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
4d00h: 345782.464 CMAC_LOG_DHCP_TZ_OFFSET 0

```

```

4d00h: 345782.466 CMAC_LOG_DHCP_CONFIG_FILE_NAME frequency.cm
4d00h: 345782.470 CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
4d00h: 345782.474 CMAC_LOG_DHCP_COMPLETE
4d00h: 345782.598 CMAC_LOG_STATE_CHANGE establish_tod_state
4d00h: 345782.606 CMAC_LOG_TOD_REQUEST_SENT
4d00h: 345782.620 CMAC_LOG_TOD_REPLY_RECEIVED 3178880491
4d00h: 345782.628 CMAC_LOG_TOD_COMPLETE
4d00h: 345782.630 CMAC_LOG_STATE_CHANGE security_associate_state
4d00h: 345782.634 CMAC_LOG_SECURITY_BYPASSED
4d00h: 345782.636 CMAC_LOG_STATE_CHANGE configuration_file
4d00h: 345782.640 CMAC_LOG_LOADING_CONFIG_FILE frequency.cm
4d00h: %LINEPROTO-5-UPDOWN: Line protocol on Interface cable-modem0,changed state to up
4d00h: 345783.678 CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE

```

NOTE—frequency override:

```

4d00h: 345783.682 CMAC_LOG_DS_FREQ_OVERRIDE 535250000
4d00h: 345783.686 CMAC_LOG_STATE_CHANGE reset_hardware_state
4d00h: 345784.048 CMAC_LOG_STATE_CHANGE wait_for_link_up_state
4d00h: 345784.052 CMAC_LOG_DRIVER_INIT_IDB_RESET 0x082A5226
4d00h: 345784.054 CMAC_LOG_LINK_DOWN
4d00h: 345784.056 CMAC_LOG_LINK_UP
4d00h: 345784.062 CMAC_LOG_STATE_CHANGE ds_channel_scanning_state
4d00h: 345785.198 CMAC_LOG_DS_NO_QAM_FEC_LOCK 535250000
4d00h: 345785.212 CMAC_LOG_DS_TUNER_KEEPALIVE
4d00h: 345787.018 CMAC_LOG_UCD_MSG_RCVD 1
4d00h: 345787.022 CMAC_LOG_DS_64QAM_LOCK_ACQUIRED 453000000

```

Incorrect frequency specified in **cable modem change-frequency** on the Cisco uBR7246VXR can also cause the CM to switch frequencies, and if the frequency configured on the router is not chosen carefully, you will see results similar to that above. The **cable modem change-frequency** command on the Cisco uBR7246VXR is optional and is typically left out by default.

After a downstream channel has been acquired and latency has been calculated, the next task is to locate a suitable upstream channel. The cable modem listens for an upstream channel descriptor (UCD) which contains the physical properties of the upstream channel, such as frequency, modulation, channel width, and other parameters defined in the burst descriptors.

A cable modem that cannot find a usable upstream channel descriptor may be on a downstream channel for which no upstream service is provided. This is likely to be a faulty configuration of the router. You can check this using the **show controller cable x** command. Another possible reason a cable modem may not find a usable UCD is that its hardware or MAC may not support the parameters in the burst descriptors. This is likely to be either a faulty configuration of the Cisco uBR7246VXR or a CM that is not DOCSIS-compliant.

After a usable UCD is found, the cable modem begins to listen to MAP (Bandwidth Allocation Map) messages which contain the upstream bandwidth allocation map of time. A section of time is mapped out into mini-slots and assigned to individual modems. There are also regions in the MAP for broadcast, contention-based initial maintenance (or broadcast) ranging. It is these regions of the MAP that the cable modem must send its initial ranging requests until the Cisco uBR7246VXR responds with a ranging response (RNG-RSP).

If a CM cannot find an initial maintenance region before a T2 timer expires, check whether the Cisco uBR7246VXR has a faulty configuration. You should also check the **insertion-interval** for the cable interface on the router.

Ranging Process - init(r1),init(r2), and init(rc) state

At this stage, the CM begins a ranging process to calculate the necessary transmit power level to reach the Cisco uBR7246VXR at its desired input power level. A reasonably good transmit power is roughly 40 - 50 dBmV (based on a uBR7246 input power of 0 dBmV.) Other hardware may vary. Like the downstream channel, the carrier in the upstream channel should be sufficiently strong for the Cisco uBR7246VXR receiver to discern the symbols yet not too high to prevent increased bit error-rates.

The CM sends a ranging request (RNG-REQ) message to the Cisco uBR7246VXR and waits for a ranging response (RNG-RSP) message or a T3 timer expiry. If a T3 timeout occurs, the retry count increments. If the retry count is less than the maximum number of retries, the cable modem transmits another RNG-REQ at a higher power level. This ranging process occurs in the initial maintenance or broadcast regions of the MAP because the Cisco uBR7246VXR has not assigned the cable modem a service identifier (SID) for unicast transmissions in the MAP. Thus, broadcast ranging is contention based and subject to collisions. To compensate for this, the cable modems have a ranging backoff algorithm to calculate a random backoff time between RNG-REQ transmissions. This can be configured using **cable upstream range-backoff** command. When the transmit power reaches a sufficient level for the Cisco uBR7246VXR, it responds to the RNG-REQ with a RNG-RSP containing a temporary SID. This SID is used to identify unicast transmission regions in the MAP for unicast ranging.

The following output shows a CM in init(r1) state indicating the CM cannot get past the initial ranging stage:

```
r#show cable modem
```

Interface	Prim	Online	Timing	Rec	QoS	CPE	IP address	MAC address
Sid		State	Offset	Power				
Cable2/0/U0	6	init(r1)	2813	12.00	2	0	10.1.1.22	0050.7366.1e01

The debug below shows how the CM fails to complete the ranging process and resets after a T3 timer expiry. Note the **CMAC_LOG_ADJUST_TX_POWER** messages coming from the Cisco uBR7246VXR asking the CM to adjust its power:

```
1w3d:      871160.618  CMAC_LOG_STATE_CHANGE                ranging_1_state
1w3d:      871160.618  CMAC_LOG_RANGING_OFFSET_SET_TO        9610
1w3d:      871160.622  CMAC_LOG_POWER_LEVEL_IS               19.0  dBmV (comman)
1w3d:      871160.622  CMAC_LOG_STARTING_RANGING
1w3d:      871160.622  CMAC_LOG_RANGING_BACKOFF_SET          0
1w3d:      871160.622  CMAC_LOG_RNG_REQ_QUEUED                0
1w3d:      871160.678  CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d:      871160.682  CMAC_LOG_RNG_RSP_MSG_RCVD
1w3d:      871160.682  CMAC_LOG_RNG_RSP_SID_ASSIGNED          6
1w3d:      871160.682  CMAC_LOG_ADJUST_RANGING_OFFSET         2813
1w3d:      871160.682  CMAC_LOG_RANGING_OFFSET_SET_TO        12423
1w3d:      871160.686  CMAC_LOG_ADJUST_TX_POWER                48
```

```

1w3d:      871160.686  CMAC_LOG_STATE_CHANGE          ranging_2_state
1w3d:      871160.686  CMAC_LOG_RNG_REQ_QUEUED        6
1w3d:      871161.690  CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d:      871161.690  CMAC_LOG_RNG_RSP_MSG_RCVD
1w3d:      871161.694  CMAC_LOG_ADJUST_TX_POWER      -36
1w3d:      871161.694  CMAC_LOG_RANGING_CONTINUE
1w3d:      871162.698  CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d:      871162.898  CMAC_LOG_T3_TIMER
1w3d:      871163.734  CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d:      871163.934  CMAC_LOG_T3_TIMER
1w3d:      871164.766  CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d:      871164.966  CMAC_LOG_T3_TIMER
131.CABLEMODEM.CISCO: 1w3d: %UBR900-3-RESET_T3_RETRIES_EXHAUSTED: R03.0 Ranging
1w3d:      871164.966  CMAC_LOG_RESET_T3_RETRIES_EXHAUSTED
1w3d:      871164.966  CMAC_LOG_STATE_CHANGE          reset_interface_state
1w3d:      871164.966  CMAC_LOG_STATE_CHANGE          reset_hardware_state
Note: init(r1) is ranging_1_state and init(r2) is ranging_2_state

```

You can get an indication of the Transmit power on the CM by displaying the following command:

```

#sh cont cable-modem 0
BCM Cable interface 0:
CM unit 0, idb 0x2010AC, ds 0x86213E0, regaddr = 0x800000, reset_mask 0x80
station address 0050.7366.2223 default station address 0050.7366.2223
PLD VERSION: 32

MAC State is wait_for_link_up_state, Prev States = 2
MAC mcfilter 00000000 data mcfilter 00000000

MAC extended header ON
DS: BCM 3116 Receiver: Chip id = 2
US: BCM 3037 Transmitter: Chip id = 30AC

Tuner: status=0x00
Rx: tuner_freq 0, symbol_rate 5055932, local_freq 11520000
    snr_estimate 30640, ber_estimate 0, lock_threshold 26000
    QAM not in lock, FEC not in lock, qam_mode QAM_64
Tx: tx_freq 27984000, power_level 0x20 (8.0 dBmV), symbol_rate 8 (1280000 sym/s)

```

If a cable modem cannot proceed out of ranging state, the likely cause is an insufficient transmit power level. Transmit power can be adjusted by adjusting attenuation at the low frequency port. Increased attenuation will result in increased transmit power levels. 20 - 30 dBmV of attenuation is a good place to start.

After initial ranging `init(r1)` the cable modem proceeds to `init(r2)`. This is where the cable modem must configure the transmit timing offset and power level to ensure that transmissions from the modem are received at the correct time and are at an acceptable input power level at the Cisco uBR7246VXR receiver. This is performed through a conversation of unicast RNG-REQ and RNG-RSP messages. The RNG-RSP messages contain the power and timing offset corrections that the cable modem must make.

The cable modem continues to transmit RNG-REQ and to perform adjustments per RNG-RSP until the RNG-RSP message indicates ranging success or ranging complete by reaching the `init(rc)` state. If a cable modem cannot proceed out of `init(r2)` the transmit power needs to be refined. Below is an output

display of a CM in `init(r2)` state. Note the asterisk (*) symbol next to the Rec Power column indicating that the noise power adjustment method is active for this modem. An exclamation mark (!) means the cable modem has reached its maximum transmit power.

```
#show cable modem
```

Interface Sid	Prim	Online State	Timing Offset	Rec Power	QoS	CPE	IP address	MAC address
Cable2/0/U0	5	init(r2)	2289	*4.00	2	0	10.1.1.25	0050.7366.2223

DHCP - init(d) state

The next stage after successful ranging is acquiring network configuration via DHCP.

Below is a an output display of `show cable modem` showing a cable modem in `init(d)`, which indicates that the DHCP request was received from the CM:

```
#show cable modem
```

Interface Sid	Prim	Online State	Timing Offset	Rec Power	QoS	CPE	IP address	MAC address
Cable2/0/U0	7	init(d)	2811	0.25	2	0	10.1.1.20	0030.96f9.65d9

Note that the CM can cycle through `init(r1)` to `init(d)` indefinitely. Following are some possible causes:

- IP connectivity issue from the Cisco uBR7246VXR to the DHCP server.
- DHCP server down.
- Wrong default gateway configured at the DHCP server.
- Low transmit power at the CM.



Note

Although you can ping the DHCP server from the Cisco uBR7246VXR, the problem could be that the DHCP has the incorrect gateway set since it may be able to respond to the primary IP address of the Cisco uBR7246VXR cable interface, but not the secondary IP address, which is used as the source IP address during the DHCP discovery phase.

Frequently, errors at DHCP state manifest themselves as timeouts rather than NAKs. The order of DHCP messages should be DISCOVER, OFFER, REQUEST, ACK. If the cable modem is transmitting a DISCOVER with no OFFER response from the DHCP server, turn on UDP debugging on the Cisco uBR7246VXR using the following command:

```
# debug ip udp
```

```
4d01h:  UDP:   rcvd src=0.0.0.0(68), dst=255.255.255.255(67), length=584
4d01h:  BOOTP:  opcode 1 from host 0.0.0.0 on Cable2/0, 0 secs, 0 hops
4d01h:  UDP:   forwarded broadcast 67 from 10.1.1.10 to 10.20.20.9 on Ethernet0
4d01h:  UDP:   rcvd src=0.0.0.0(68), dst=255.255.255.255(67), length=584
4d01h:  BOOTP:  opcode 1 from host 0.0.0.0 on Cable2/0, 0 secs, 0 hops
4d01h:  UDP:   forwarded broadcast 67 from 10.1.1.10 to 10.20.20.9 on Ethernet0
4d01h:  UDP:   rcvd src=172.17.110.136(67), dst=10.1.1.10(67), length=314
4d01h:  BOOTP:  opcode 2 from host 10.20.20.9 on Ethernet1/0, 0 secs, 0 hops
4d01h:  BOOTP:  Broadcasting response 10.20.20.9 -> 10.1.1.20 (Cable2/0)
```



```

4d01h: UDP: forwarded broadcast 68 from 10.20.20.9 to 255.255.255.255 on Ca0
4d01h: UDP: rcvd src=0.0.0.0(68), dst=255.255.255.255(67), length=584
4d01h: BOOTP: opcode 1 from host 0.0.0.0 on Cable2/0, 0 secs, 0 hops
4d01h: UDP: forwarded broadcast 67 from 10.1.1.10 to 110.20.20.9 on Ethernet0
4d01h: UDP: rcvd src=10.20.20.9(67), dst=10.1.1.10(67), length=314
4d01h: BOOTP: opcode 2 from host 10.20.20.9 on Ethernet1/0, 0 secs, 0 hops
4d01h: BOOTP: Broadcasting response 10.20.20.9 -> 10.1.1.20 (Cable2/0)
4d01h: UDP: forwarded broadcast 68 from 10.20.20.9 to 255.255.255.255 on 1
All possible debugging has been turned off

```

**Caution**

Running debug commands on a uBR7246 (Universal Broadband Router) with more than a handful of modems may cause the Cisco uBR7246VXR to halt the system in order to keep up with the debugging. In this case, all the cable modems may lose sync and debugging will be useless.

If no packets are seen through debug messages, check the configuration of the **cable helper-address** statement on the cable interface to which this modem is attached. If this is configured correctly and a packet trace of the DHCP server subnet also reveals no DHCP packets from the cable modem, then look at the output errors of the cable modem's cable interface or the input errors of the cable interface of the Cisco uBR7246VXR. It might be a good idea to boost the transmitter power of the CM a bit more with more attenuation.

If packets are seen to be transmitted onto the DHCP server subnet, check the cable modem debug messages to see if there are parameter request or assignment errors. At this stage of troubleshooting you should investigate the routing between the CM and the DHCP server. Also, double-check the DHCP server configuration and the DHCP logs.

Below is a sample debug taken at the CM by running the **debug cable-modem mac log verbose** command:

```

1w3d: 865015.920 CMAC_LOG_RANGING_SUCCESS
1w3d: 865015.920 CMAC_LOG_STATE_CHANGE dhcp_state
1w3d: 865053.580 CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d: 865053.584 CMAC_LOG_RNG_RSP_MSG_RCVD
1w3d: 865055.924 CMAC_LOG_WATCHDOG_TIMER
131.CABLEMODEM.CISCO: 1w3d: %UBR900-3-RESET_DHCP_WATCHDOG_EXPIRED:
Cable Interface Reset due to DHCP watchdog timer expiration
1w3d: 865055.924 CMAC_LOG_RESET_DHCP_WATCHDOG_EXPIRED
1w3d: 865055.924 CMAC_LOG_STATE_CHANGE reset_interface_state
1w3d: 865055.924 CMAC_LOG_DHCP_PROCESS_KILLED
1w3d: 865055.924 CMAC_LOG_STATE_CHANGE reset_hardware_state

```

As you can see, the above the DHCP process failed and the cable modem was reset.

DHCP - init(i) state

After a reply to the DHCP request has been received and an IP address assigned to the cable modem the next **show cable modem** command gives its state as init(i):

#show cable modem

Interface	Prim	Online	Timing	Rec	QoS	CPE	IP address	MAC address
Sid		State	Offset	Power				
Cable2/0/U0	7	init(i)	2815	-0.25	2	0	10.1.1.20	0030.96f9.65d9

In the output above, the CM never gets beyond state init(i). Repetitive **show cable modem** displays will usually show the cable modem cycling between init(r1), init(r2), init(rc), init(d) and init(i) indefinitely.

Following is a list of the more common reasons for a cable modem not getting further than init(i):

- Incorrect or invalid DOCSIS file specified in the DHCP server.
- TFTP server issues, for example: incorrect IP address, TFTP server unreachable.
- Problems getting TOD or Timing Offset.
- Incorrect Router setting in the DHCP configuration.

Since the Cable Modem has reached as far as init(i) we know that it has got as far as obtaining an IP address. This is shown in the output display of the **debug cable-modem mac log verbose** command from the cable modem below:

```
3d20h: 334402.548 CMAC_LOG_RANGING_SUCCESS
3d20h: 334402.548 CMAC_LOG_STATE_CHANGE dhcp_state
NOTE-IP address Assigned to CM:
3d20h: 334415.492 CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS 10.1.1.20
3d20h: 334415.492 CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS 10.20.20.9
3d20h: 334415.492 CMAC_LOG_DHCP_TOD_SERVER_ADDRESS 10.20.20.9
3d20h: 334415.492 CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
3d20h: 334415.492 CMAC_LOG_DHCP_TZ_OFFSET 0
NOTE-DOCSIS file CM is trying to load:
3d20h: 334415.496 CMAC_LOG_DHCP_CONFIG_FILE_NAME nofile
3d20h: 334415.496 CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
3d20h: 334415.496 CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRESS
3d20h: 334415.496 CMAC_LOG_DHCP_COMPLETE
3d20h: 334415.508 CMAC_LOG_STATE_CHANGE establish_tod_state
3d20h: 334415.512 CMAC_LOG_TOD_REQUEST_SENT 10.20.20.9
3d20h: 334415.524 CMAC_LOG_TOD_REPLY_RECEIVED 3178343318
3d20h: 334415.524 CMAC_LOG_TOD_COMPLETE
3d20h: 334415.528 CMAC_LOG_STATE_CHANGE security_association_state
3d20h: 334415.528 CMAC_LOG_SECURITY_BYPASSED
3d20h: 334415.528 CMAC_LOG_STATE_CHANGE configuration_file
NOTE-DOCSIS file name:
3d20h: 334415.528 CMAC_LOG_LOADING_CONFIG_FILE nofile
```

```

133.CABLEMODEM.CISCO: 3d20h: %LINEPROTO-5-UPDOWN: Line protocol on Interface cap
3d20h: 334416.544   CMAC_LOG_CONFIG_FILE_TFTP_FAILED           -1
3d20h: 334416.548   CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE
3d20h: 334416.548   CMAC_LOG_RESET_CONFIG_FILE_READ_FAILED

```

Similarly, TFTP server issues give similar errors resulting in the cable modem resetting and cycling through the same process indefinitely:

```

3d21h: 336136.520   CMAC_LOG_STATE_CHANGE                               dhcp_state
3d21h: 336149.404   CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS                  10.1.1.20
NOTE—TFTP Server address
3d21h: 336149.404   CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS                  10.1.1.10
3d21h: 336149.404   CMAC_LOG_DHCP_TOD_SERVER_ADDRESS                  10.1.1.20
3d21h: 336149.404   CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
3d21h: 336149.404   CMAC_LOG_DHCP_TZ_OFFSET                            0
3d21h: 336149.408   CMAC_LOG_DHCP_CONFIG_FILE_NAME                    platinum.cm
3d21h: 336149.408   CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
3d21h: 336149.408   CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRES
3d21h: 336149.408   CMAC_LOG_DHCP_COMPLETE
3d21h: 336149.420   CMAC_LOG_STATE_CHANGE                               establish_tod_state
3d21h: 336149.424   CMAC_LOG_TOD_REQUEST_SENT                          10.1.1.20
3d21h: 336149.436   CMAC_LOG_TOD_REPLY_RECEIVED                        3178345052
3d21h: 336149.436   CMAC_LOG_TOD_COMPLETE
3d21h: 336149.440   CMAC_LOG_STATE_CHANGE                               security_association_state
3d21h: 336149.440   CMAC_LOG_SECURITY_BYPASSED
3d21h: 336149.440   CMAC_LOG_STATE_CHANGE                               configuration_file
3d21h: 336149.440   CMAC_LOG_LOADING_CONFIG_FILE                       platinum.cm
133.CABLEMODEM.CISCO: 3d21h: %LINEPROTO-5-UPDOWN: Line protocol on Interface cap
3d21h: 336163.252   CMAC_LOG_RNG_REQ_TRANSMITTED
3d21h: 336163.252   CMAC_LOG_RNG_RSP_MSG_RCVD
NOTE—TFTP process failing:
3d21h: 336165.448   CMAC_LOG_CONFIG_FILE_TFTP_FAILED                   -1
3d21h: 336165.448   CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE
3d21h: 336165.452   CMAC_LOG_RESET_CONFIG_FILE_READ_FAILED
3d21h: 336165.452   CMAC_LOG_STATE_CHANGE                               reset_interface_state

```

Problems getting TOD (Time of Day) or Timing Offset can also result in the cable modem not achieving online status:

```

3d21h: 338322.500   CMAC_LOG_STATE_CHANGE                               dhcp_state
3d21h: 338334.260   CMAC_LOG_RNG_REQ_TRANSMITTED
3d21h: 338334.260   CMAC_LOG_RNG_RSP_MSG_RCVD
3d21h: 338335.424   CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS                  10.1.1.20

```

```

3d21h: 338335.424 CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS 10.10.10.1
3d21h: 338335.424 CMAC_LOG_DHCP_ERROR_ACQUIRING_TOD_ADDRESS
3d21h: 338335.424 CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
3d21h: 338335.424 CMAC_LOG_DHCP_ERROR_ACQUIRING_TZ_OFFSET
3d21h: 338335.424 CMAC_LOG_DHCP_CONFIG_FILE_NAME platinum.cm
3d21h: 338335.428 CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
3d21h: 338335.428 CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRESS
3d21h: 338335.428 CMAC_LOG_DHCP_COMPLETE
3d21h: 338335.428 CMAC_LOG_RESET_DHCP_FAILED
3d21h: 338335.432 CMAC_LOG_STATE_CHANGE reset_interface_state
3d21h: 338335.432 CMAC_LOG_STATE_CHANGE reset_hardware_state
3d21h: 338336.016 CMAC_LOG_STATE_CHANGE wait_for_link_up_state

```

**Note**

Prior to IOS version 12.1(1) TOD needed to be specified in the DHCP server in order for the cable modem to go online. However, after 12.1(1) TOD is not required but the cable modem still needs to get the timing offset, as shown below.

```

344374.528 CMAC_LOG_STATE_CHANGE dhcp_state
344377.292 CMAC_LOG_RNG_REQ_TRANSMITTED
344377.292 CMAC_LOG_RNG_RSP_MSG_RCVD
344387.412 CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS 10.1.1.20
344387.412 CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS 10.10.10.1
NOTE- TOD server IP address obtained:
344387.412 CMAC_LOG_DHCP_TOD_SERVER_ADDRESS 10.10.10.1
344387.412 CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
NOTE- Timing offset not specified in DHCP server:
344387.412 CMAC_LOG_DHCP_ERROR_ACQUIRING_TZ_OFFSET
344387.412 CMAC_LOG_DHCP_CONFIG_FILE_NAME platinum.cm
344387.412 CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
344387.412 CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRESS
344387.412 CMAC_LOG_DHCP_COMPLETE
344387.412 CMAC_LOG_RESET_DHCP_FAILED
NOTE-Modem resetting:
344387.412 CMAC_LOG_STATE_CHANGE reset_interface_state

```

In the debug below there is no time-server specified, but there is a timing offset configured in the DHCP server, and therefore the cable modem goes online:

```

3d23h: 345297.516 CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS 10.1.1.20
3d23h: 345297.516 CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS 172.17.110.136
3d23h: 345297.516 CMAC_LOG_DHCP_ERROR_ACQUIRING_TOD_ADDRESS
3d23h: 345297.516 CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS

```

```

3d23h: 345297.516 CMAC_LOG_DHCP_TZ_OFFSET 0
3d23h: 345297.516 CMAC_LOG_DHCP_CONFIG_FILE_NAME platinum.cm
3d23h: 345297.520 CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
3d23h: 345297.520 CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRESS
3d23h: 345297.520 CMAC_LOG_DHCP_COMPLETE
3d23h: 345297.532 CMAC_LOG_STATE_CHANGE establish_tod_state
3d23h: 345297.532 CMAC_LOG_TOD_NOT_REQUESTED_NO_TIME_ADDR
3d23h: 345297.532 CMAC_LOG_STATE_CHANGE security_association_state
3d23h: 345297.536 CMAC_LOG_SECURITY_BYPASSED
3d23h: 345297.536 CMAC_LOG_STATE_CHANGE configuration_file
3d23h: 345297.536 CMAC_LOG_LOADING_CONFIG_FILE platinum.cm
3d23h: 345297.568 CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE
3d23h: 345297.568 CMAC_LOG_STATE_CHANGE registration_state
3d23h: 345297.592 CMAC_LOG_REG_RSP_MSG_RCVD
3d23h: 345297.592 CMAC_LOG_COS_ASSIGNED_SID 1/7
3d23h: 345297.596 CMAC_LOG_RNG_REQ_QUEUED 7
3d23h: 345297.596 CMAC_LOG_REGISTRATION_OK
3d23h: 345297.596 CMAC_LOG_STATE_CHANGE establish_privacy_state
3d23h: 345297.596 CMAC_LOG_PRIVACY_NOT_CONFIGURED
3d23h: 345297.596 CMAC_LOG_STATE_CHANGE maintenance_state
133.CABLEMODEM.CISCO: 3d23h: %LINEPROTO-5-UPDOWN: Line protocol on Interface changed state
to up

```

Not including a Router option setting in the DHCP server or specifying an invalid IP address in the Router option field also results in a cable modem not getting beyond init(i) state, as shown in the output from **debug cable-modem mac log verbose** below:

```

1d16h: 146585.940 CMAC_LOG_CONFIG_FILE_TFTP_FAILED -1
1d16h: 146585.940 CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE
1d16h: 146585.944 CMAC_LOG_RESET_CONFIG_FILE_READ_FAILED
1d16h: 146585.944 CMAC_LOG_STATE_CHANGE reset_interface_state
1d16h: 146585.944 CMAC_LOG_STATE_CHANGE reset_hardware_state

```

TOD exchange- init(t) state

After a cable modem has acquired its network parameters, it must request the time of day from a Time Of Day (TOD) server. TOD uses a UTC timestamp (seconds from January 1, 1970) combined with the time offset option value from DHCP, to calculate the current time. The time is used for syslog and event log timestamps.

Below we have modems with SID 1 and 2 in init(t). Note that with IOS later than 12.1(1) the cable modem still comes online even though the TOD exchange failed, as you can see in the following output from the **show cable modem** command:

```
#show cable modem
```

Interface	Prim Sid	Online State	Timing Offset	Rec Power	QoS	CPE	IP address	MAC address
Cable2/0/U0	1	init(t)	2808	0.00	2	0	10.1.1.20	0030.96f9.65d9
Cable2/0/U0	2	init(t)	2809	0.25	2	0	10.1.1.21	0030.96f9.6605
Cable2/0/U0	3	init(i)	2810	-0.25	2	0	10.1.1.22	0050.7366.1e01


```

2d01h: 177933.712 CMAC_LOG_STATE_CHANGE dhcp_state
2d01h: 177933.716 CMAC_LOG_RNG_REQ_TRANSMITTED
2d01h: 177933.716 CMAC_LOG_RNG_RSP_MSG_RCVD
2d01h: 177946.596 CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS 10.1.1.20
2d01h: 177946.596 CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS 10.20.20.1
2d01h: 177946.596 CMAC_LOG_DHCP_TOD_SERVER_ADDRESS 10.30.30.1
2d01h: 177946.596 CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
2d01h: 177946.596 CMAC_LOG_DHCP_TZ_OFFSET 0
2d01h: 177946.600 CMAC_LOG_DHCP_CONFIG_FILE_NAME platinum.cm
2d01h: 177946.600 CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
2d01h: 177946.600 CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRESS
2d01h: 177946.600 CMAC_LOG_DHCP_COMPLETE
2d01h: 177946.612 CMAC_LOG_STATE_CHANGE establish_tod_state
2d01h: 177946.716 CMAC_LOG_RNG_REQ_TRANSMITTED
2d01h: 177946.716 CMAC_LOG_RNG_RSP_MSG_RCVD
133.CABLEMODEM.CISCO: 2d01h: %LINEPROTO-5-UPDOWN: Line protocol on Interface cap
2d01h: 177947.716 CMAC_LOG_RNG_REQ_TRANSMITTED
2d01h: 177947.716 CMAC_LOG_RNG_RSP_MSG_RCVD
2d01h: 177948.616 CMAC_LOG_TOD_REQUEST_SENT 10.30.30.1
2d01h: 177948.716 CMAC_LOG_RNG_REQ_TRANSMITTED
2d01h: 177954.616 CMAC_LOG_TOD_REQUEST_SENT 10.30.30.1
2d01h: 177954.716 CMAC_LOG_RNG_REQ_TRANSMITTED
2d01h: 177954.716 CMAC_LOG_RNG_RSP_MSG_RCVD
2d01h: 177960.616 CMAC_LOG_TOD_REQUEST_SENT 10.30.30.1
2d01h: 177960.712 CMAC_LOG_RNG_REQ_TRANSMITTED
2d01h: 177960.716 CMAC_LOG_RNG_RSP_MSG_RCVD
2d01h: 177961.716 CMAC_LOG_RNG_REQ_TRANSMITTED
131.CABLEMODEM.CISCO: 2d01h: %UBR900-3-TOD_FAILED_TIMER_EXPIRED:TOD failed, but Cable Interface proceeding to operational state
2d01h: 177986.616 CMAC_LOG_TOD_WATCHDOG_EXPIRED
2d01h: 177986.616 CMAC_LOG_STATE_CHANGE security_association_state
2d01h: 177986.616 CMAC_LOG_SECURITY_BYPASSED
2d01h: 177986.616 CMAC_LOG_STATE_CHANGE configuration_file
2d01h: 177986.620 CMAC_LOG_LOADING_CONFIG_FILE platinum.cm
2d01h: 177986.644 CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE
2d01h: 177986.644 CMAC_LOG_STATE_CHANGE registration_state

```

```

2d01h: 177986.644 CMAC_LOG_REG_REQ_MSG_QUEUED
2d01h: 177986.648 CMAC_LOG_REG_REQ_TRANSMITTED
2d01h: 177986.652 CMAC_LOG_REG_RSP_MSG_RCVD
2d01h: 177986.652 CMAC_LOG_COS_ASSIGNED_SID 1/1
2d01h: 177986.656 CMAC_LOG_RNG_REQ_QUEUED 1
NOTE- Modem online:
2d01h: 177986.656 CMAC_LOG_REGISTRATION_OK
2d01h: 177986.656 CMAC_LOG_STATE_CHANGE establish_privacy_statee
2d01h: 177986.656 CMAC_LOG_PRIVACY_NOT_CONFIGURED
2d01h: 177986.656 CMAC_LOG_STATE_CHANGE maintenance_state
2d01h: 177988.716 CMAC_LOG_RNG_REQ_TRANSMITTED

```

Time of day errors almost always point to a DHCP misconfiguration. Possible misconfigurations that can result in TOD errors include the following:

- Gateway address misconfigurations.
- Incorrect TOD server address.

Make sure you can ping the time-server to rule out IP connectivity issues and to verify the time-server is available.

Option File Transfer Started-init(o) State

The main configuration and administration interface to the cable modem is the configuration file downloaded from the provisioning server. This configuration file contains downstream channel and upstream channel identification and characteristics as well as class of service settings, baseline privacy settings, general operational settings, network management information, software upgrade fields, filters, and vendor specific settings.

A cable modem stuck in init(o) state usually indicates that the cable modem started or is ready to start downloading the configuration file, but was unsuccessful. This can be due to the following reasons:

- Incorrect, corrupt, or missing DOCSIS configuration file.
- Unable to reach the TFTP server, because it is unavailable or there is no IP connectivity.
- Invalid or missing configuration parameters in DOCSIS file.
- Wrong file permissions on the TFTP server.

Note that you may not always see init(o), instead you might see init(i) and then cycling through from init(r1) to init(i). A more accurate state can be derived by displaying the output of **show controller cable-modem 0 mac state**.

Following is an abbreviated display of that output:

```

#show controller cable-modem 0 mac state

MAC State: configuration_file_state
Ranging SID: 4
Registered: FALSE
Privacy Established: FALSE

```

When you see `init(o)` in the `show cable modem` output, running the `debug cable-modem mac log verbose` will not tell you if it is a corrupt configuration file or a TFTP server failure that is the cause. The debug points to both of them.

Examples of invalid configuration parameters in the Cisco Broadband Configurator (available for download on cisco.com) are invalid or missing Vendor ID or Vendor Specific Information. The result is similar to the following output display:

```
w3d:      880748.992   CMAC_LOG_STATE_CHANGE                dhcp_state
1w3d:     880751.652   CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d:     880751.656   CMAC_LOG_RNG_RSP_MSG_RCVD
1w3d:     880761.876   CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS    10.1.1.20
1w3d:     880761.876   CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS    10.10.10.1
1w3d:     880761.876   CMAC_LOG_DHCP_TOD_SERVER_ADDRESS     10.10.10.1
1w3d:     880761.876   CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
1w3d:     880761.876   CMAC_LOG_DHCP_TZ_OFFSET              0

NOTE—Corrupt configuration file:
1w3d:     880761.880   CMAC_LOG_DHCP_CONFIG_FILE_NAME       data.cm
1w3d:     880761.880   CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
1w3d:     880761.880   CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRESS
1w3d:     880761.880   CMAC_LOG_DHCP_COMPLETE
1w3d:     880761.892   CMAC_LOG_STATE_CHANGE                establish_tod_state
1w3d:     880761.896   CMAC_LOG_TOD_REQUEST_SENT            10.10.10.1
1w3d:     880761.904   CMAC_LOG_TOD_REPLY_RECEIVED          3180091733
1w3d:     880761.908   CMAC_LOG_TOD_COMPLETE
1w3d:     880761.908   CMAC_LOG_STATE_CHANGE                security_association_state
1w3d:     880761.908   CMAC_LOG_SECURITY_BYPASSED
1w3d:     880761.912   CMAC_LOG_STATE_CHANGE                configuration_file_state
1w3d:     880761.912   CMAC_LOG_LOADING_CONFIG_FILE         data.cm
1w3d:     880762.652   CMAC_LOG_RNG_REQ_TRANSMITTED
1w3d:     880762.652   CMAC_LOG_RNG_RSP_MSG_RCVD

133.CABLEMODEM.CISCO: 00:13:07: %LINEPROTO-5-UPDOWN: Line protocol on Interface cable-modem0,
changed state to up
00:13:08: 788.004     CMAC_LOG_CONFIG_FILE_CISCO_BAD_TYPE  155
00:13:08: 788.004     CMAC_LOG_CONFIG_FILE_CISCO_BAD_TYPE  115
00:13:08: 788.004     CMAC_LOG_CONFIG_FILE_CISCO_BAD_TYPE  116
00:13:08: 788.004     CMAC_LOG_CONFIG_FILE_CISCO_BAD_ATTR_MAX LENG 128
00:13:08: 788.008     CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE
00:13:08: 788.008     CMAC_LOG_RESET_CONFIG_FILE_READ_FAILED
```


Online, Online(d), Online(pk), Online(pt) state

These states indicate that the cable modem has achieved online status and is able to transmit and receive data:

- Online
- Online(pk)
- Online(pt)

Online(d), however indicates that the cable modem has come online but has been denied network access.

#show cable modem

Interface	Prim Sid	Online State	Timing Offset	Rec Power	QoS	CPE	IP address	MAC address
Cable2/0/U0	4	online	2810	-0.75	6	0	10.1.1.20	0030.96f9.65d9
Cable2/0/U0	5	online(pt)	2290	0.25	5	0	10.1.1.25	0050.7366.2223
Cable2/0/U0	6	online(d)	2815	0.00	6	0	10.1.1.27	0001.9659.4461

Online(d) is typically caused by disabling the Network Access option under Radio Frequency info in the Cisco Broadband Configurator (available for download on cisco.com). The default for Network Access is **enabled**. This can be seen in the display of **show cable modem** above and in the following output from **debug cable-modem mac log verbose**:

```
04:11:34: 15094.700    CMAC_LOG_STATE_CHANGE                dhcp_state
04:11:46: 15106.392    CMAC_LOG_RNG_REQ_TRANSMITTED
04:11:46: 15106.396    CMAC_LOG_RNG_RSP_MSG_RCVD
04:11:47: 15107.620    CMAC_LOG_DHCP_ASSIGNED_IP_ADDRESS    10.1.1.20
04:11:47: 15107.620    CMAC_LOG_DHCP_TFTP_SERVER_ADDRESS    172.17.110.136
04:11:47: 15107.620    CMAC_LOG_DHCP_TOD_SERVER_ADDRESS     172.17.110.136
04:11:47: 15107.620    CMAC_LOG_DHCP_SET_GATEWAY_ADDRESS
04:11:47: 15107.620    CMAC_LOG_DHCP_TZ_OFFSET              0
NOTE—Network Access disabled:
04:11:47: 15107.624    CMAC_LOG_DHCP_CONFIG_FILE_NAME       noaccess.cm
04:11:47: 15107.624    CMAC_LOG_DHCP_ERROR_ACQUIRING_SEC_SVR_ADDR
04:11:47: 15107.624    CMAC_LOG_DHCP_ERROR_ACQUIRING_LOG_ADDRES
04:11:47: 15107.624    CMAC_LOG_DHCP_COMPLETE
04:11:47: 15107.636    CMAC_LOG_STATE_CHANGE                establish_tod_state
04:11:47: 15107.640    CMAC_LOG_TOD_REQUEST_SENT            172.17.110.136
04:11:47: 15107.648    CMAC_LOG_TOD_REPLY_RECEIVED          3179226080
04:11:47: 15107.652    CMAC_LOG_TOD_COMPLETE
04:11:47: 15107.652    CMAC_LOG_STATE_CHANGE                security_association_state
04:11:47: 15107.652    CMAC_LOG_SECURITY_BYPASSED
04:11:47: 15107.652    CMAC_LOG_STATE_CHANGE                configuration_file_state
04:11:47: 15107.652    CMAC_LOG_LOADING_CONFIG_FILE         noaccess.cm
133.CABLEMODEM.CISCO: 04:11:48: %LINEPROTO-5-UPDOWN: Line protocol on Interface
cable-modem0, changed state to up
```


online(pk), followed shortly by online(pt). The following debug output display was taken on the cable modem side using the **debug cable-modem mac log verbose** command, showing only the registration part:

```

5d03h: 445197.804 CMAC_LOG_STATE_CHANGE registration_state
5d03h: 445197.804 CMAC_LOG_REG_REQ_MSG_QUEUED
5d03h: 445197.812 CMAC_LOG_REG_REQ_TRANSMITTED
5d03h: 445197.816 CMAC_LOG_REG_RSP_MSG_RCVD
5d03h: 445197.816 CMAC_LOG_COS_ASSIGNED_SID 1/4
5d03h: 445197.816 CMAC_LOG_RNG_REQ_QUEUED 4
5d03h: 445197.816 CMAC_LOG_REGISTRATION_OK
5d03h: 445197.816 CMAC_LOG_STATE_CHANGE establish_privacy_state
5d03h: 445197.820 CMAC_LOG_PRIVACY_FSM_STATE_CHANGE
machine: KEK, event/state: EVENT_1_PROVISIONED/STATE_A_START, new state:
STATE_B_AUTH_WAIT
5d03h: 445197.828 CMAC_LOG_BPKM_REQ_TRANSMITTED
5d03h: 445197.848 CMAC_LOG_BPKM_RSP_MSG_RCVD
5d03h: 445197.848 CMAC_LOG_PRIVACY_FSM_STATE_CHANGE
machine: KEK, event/state: EVENT_3_AUTH_REPLY/STATE_B_AUTH_WAIT, new state:
STATE_C_AUTHORIZED
5d03h: 445198.524 CMAC_LOG_PRIVACY_FSM_STATE_CHANGE
machine: TEK, event/state: EVENT_2_AUTHORIZED/STATE_A_START, new state: STATE_B_OP_WAIT
5d03h: 445198.536 CMAC_LOG_RNG_REQ_TRANSMITTED
5d03h: 445198.536 CMAC_LOG_RNG_RSP_MSG_RCVD
5d03h: 445198.536 CMAC_LOG_BPKM_REQ_TRANSMITTED
5d03h: 445198.536 CMAC_LOG_BPKM_RSP_MSG_RCVD
5d03h: 445198.540 CMAC_LOG_PRIVACY_FSM_STATE_CHANGE
machine: TEK, event/state: EVENT_8_KEY_REPLY/STATE_B_OP_WAIT, new state:
STATE_D_OPERATIONAL
5d03h: 445198.548 CMAC_LOG_PRIVACY_INSTALLED_KEY_FOR_SID 4
5d03h: 445198.548 CMAC_LOG_PRIVACY_ESTABLISHED
5d03h: 445198.552 CMAC_LOG_STATE_CHANGE maintenance_state
5d03h: 445201.484 CMAC_LOG_RNG_REQ_TRANSMITTED
5d03h: 445201.484 CMAC_LOG_RNG_RSP_MSG_RCVD

```

If there is a problem with BPI in general you will see reject(pk), meaning it could not get through the key authentication stage. This state is covered in the reject(pk) and reject (pt) section.

For correct BPI operation ensure that the Cisco uBR7246VXR and the cable modem are both running a BPI enabled image. This is indicated by the symbol **K1** in the image name.

Also, ensure that the field **Baseline Privacy Enable** is set to 1 under the Class of Service option in the Cisco Broadband Configurator (available for download on cisco.com). If the Cisco uBR7246VXR is running a BPI enabled image and the cable modem is not, and BPI is enabled in the Cisco Broadband Configurator, then you will observe the cable modem cycling between online and offline.

Reject(pk) and Reject(pt) state

The following output display from **show cable modem** on the Cisco uBR7246VXR shows a reject(pk) state:

#show cable modem

Interface	Prim	Online	Timing	Rec	QoS	CPE	IP address	MAC address
Sid		State	Offset	Power				
Cable2/0/U0	2	reject(pk)	2812	0.00	6	0	10.1.1.20	0030.96f9.65d9

```
01:58:51: %UBR7200-5-UNAUTHSIDTIMEOUT: CMTS deleted BPI unauthorized Cable Modem
0030.96f9.65d9
```

Often, when there is a problem with the BPI configuration you will see a reject(pk) state. This state is typically caused by the following:

- Corrupt public key by the CM in the auth request, refer to sample debug cable privacy for proper sequence of events.
- Presence of cable privacy authenticate-modem configuration command on the CMTS router but no Radius server present.
- Improperly configured Radius server.

Reject(pt) is typically caused by an invalid traffic encryption key (TEK). Following is the output display using **debug cable privacy** on the Cisco uBR7246VXR:

```
02:32:08: CMTS Received AUTH REQ.
02:32:08: Created a new CM key for 0030.96f9.65d9.
02:32:08: CMTS generated AUTH_KEY.
02:32:08: Input : 70D158F106B0B75
02:32:08: Public Key:
02:32:08: 0x0000: 30 68 02 61 00 DA BA 93 3C E5 41 7C 20 2C D1 87
02:32:08: 0x0010: 3B 93 56 E1 35 7A FC 5E B7 E1 72 BA E6 A7 71 91
02:32:08: 0x0020: F4 68 CB 86 A8 18 FB A9 B4 DD 5F 21 B3 6A BE CE
02:32:08: 0x0030: 6A BE E1 32 A8 67 9A 34 E2 33 4A A4 0F 8C DB BD
02:32:08: 0x0040: D0 BB DE 54 39 05 B0 E0 F7 19 29 20 8C F9 3A 69
02:32:08: 0x0050: E4 51 C6 89 FB 8A 8E C6 01 22 02 34 C5 1F 87 F6
02:32:08: 0x0060: A3 1C 7E 67 9B 02 03 01 00 01
02:32:08: RSA public Key subject:
02:32:08: 0x0000: 30 7C 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01 05
02:32:08: 0x0010: 00 03 6B 00 30 68 02 61 00 DA BA 93 3C E5 41 7C
02:32:08: 0x0020: 20 2C D1 87 3B 93 56 E1 35 7A FC 5E B7 E1 72 BA
02:32:08: 0x0030: E6 A7 71 91 F4 68 CB 86 A8 18 FB A9 B4 DD 5F 21
02:32:08: 0x0040: B3 6A BE CE 6A BE E1 32 A8 67 9A 34 E2 33 4A A4
02:32:08: 0x0050: 0F 8C DB BD D0 BB DE 54 39 05 B0 E0 F7 19 29 20
02:32:08: 0x0060: 8C F9 3A 69 E4 51 C6 89 FB 8A 8E C6 01 22 02 34
02:32:08: 0x0070: C5 1F 87 F6 A3 1C 7E 67 9B 02 03 01 00 01
02:32:08: RSA encryption result = 0
02:32:08: RSA encrypted output:
02:32:08: 0x0000: B6 CA 09 93 BF 2C 05 66 9D C5 AF 67 0F 64 2E 31
02:32:08: 0x0010: 67 E4 2A EA 82 3E F7 63 8F 01 73 10 14 4A 24 ED
02:32:08: 0x0020: 65 8F 59 D8 23 BC F3 A8 48 7D 1A 08 09 BF A3 A8
02:32:08: 0x0030: D6 D2 5B C4 A7 36 C4 A9 28 F0 6C 5D A1 3B 92 A2
02:32:08: 0x0040: BC 99 CC 1F C9 74 F9 FA 76 83 ED D5 26 B4 92 EE
02:32:08: 0x0050: DD EA 50 81 C6 29 43 4F 73 DA 56 C2 29 AF 05 53
02:32:08: CMTS sent AUTH response.
02:32:08: CMTS Received TEK REQ.
02:32:08: Created a new key for SID 2.
02:32:08: CMTS sent KEY response.
```

Below is a sample debug output on the cable modem when there is an authorization failure:

```
6d02h: 527617.480  CMAC_LOG_CONFIG_FILE_PROCESS_COMPLETE
6d02h: 527617.480  CMAC_LOG_STATE_CHANGE                registration_state
6d02h: 527617.484  CMAC_LOG_REG_REQ_MSG_QUEUED
6d02h: 527617.488  CMAC_LOG_REG_REQ_TRANSMITTED
6d02h: 527617.492  CMAC_LOG_REG_RSP_MSG_RCVD
6d02h: 527617.492  CMAC_LOG_COS_ASSIGNED_SID            1/2
6d02h: 527617.492  CMAC_LOG_RNG_REQ_QUEUED              2
6d02h: 527617.492  CMAC_LOG_REGISTRATION_OK
6d02h: 527617.496  CMAC_LOG_STATE_CHANGE                establish_privacy_state
6d02h: 527617.496  CMAC_LOG_PRIVACY_FSM_STATE_CHANGE
machine: KEK, event/state: EVENT_1_PROVISIONED/STATE_A_START, new state:
STATE_B_AUTH_WAIT
6d02h: 527617.504  CMAC_LOG BPKM_REQ_TRANSMITTED
6d02h: 527617.504  CMAC_LOG BPKM_RSP_MSG_RCVD
6d02h: 527617.508  CMAC_LOG_PRIVACY_FSM_STATE_CHANGE
machine: KEK, event/state: EVENT_2_AUTH_REJECT/STATE_B_AUTH_WAIT, new state:
STATE_E_AUTH_REJ_WAIT
129.CABLEMODEM.CISCO: 6d02h: %CMBPKM-1-AUTHREJECT: Authorization request rejected by
CMTS: Unauthorized CM
6d02h: 527618.588  CMAC_LOG_RNG_REQ_TRANSMITTED
6d02h: 527618.592  CMAC_LOG_RNG_RSP_MSG_RCVD
```

Similarly, a **debug cable privacy** on the Cisco uBR7246VXR provides the following errors:

```
02:47:00: CMTS Received AUTH REQ.
02:47:00: Sending KEK REJECT.
02:47:05: %UBR7200-5-UNAUTHSIDTIMEOUT: CMTS deleted BPI unauthorized Cable Modem
0030.96f9.65d9
```



Note

The cable modem keeps cycling from reject(pk) to init(r1) indefinitely.

Another error you might encounter is that due to encryption export restrictions, some vendor modems may require the following command on the Cisco uBR7246VXR in the interface configuration:

```
(config-if)#cable privacy 40-bit-des
```

Registration - reject (m) state

After configuration, the cable modem sends a registration request (REG-REQ) with a required subset of the configuration settings, and the cable modem and uBR7246 message integrity checks (MIC). The cable modem MIC is a hashed calculation over the configuration file settings, which provides a method for the cable modem to be sure the configuration file was not tampered with in transit. The Cisco uBR7246VXR MIC is similar except it also includes a setting for a cable shared-secret authentication string. This shared secret is known by the Cisco uBR7246VXR and the provisioning server, and ensures that only cable modems authenticated by provisioning servers will be allowed to register with the router.

A cable modem in reject(m) state has a bad Message Integrity Check (MIC), typically caused by:

- Mismatch between cable shared-secret under the cable interface and CMTS Authentication value under Miscellaneous option in the Cisco Broadband Configurator (downloadable from cisco.com).
- Corrupt configuration file (DOCSIS file).

To correct the problem, verify that you have a valid configuration file, and that the value under **CMTS Authentication** is identical to the value configured in **cable shared-secret <line>** under the cable interface. Also verify that the Cisco uBR7246VXR allows the creation of class of service profiles, or use a profile created by the Cisco uBR7246VXR.

Registration - reject (c) state

A cable modem that fails registration due to bad class of service (COS) has a state of reject(c). Typically this is caused by:

- uBR7246 is unable or unwilling to grant a particular requested COS.
- Misconfigured parameter(s) in Class of Service option in Cisco Broadband Configurator (available for download on cisco.com), for example, two classes of service with the same ID.

Following is the output from **debug cable-modem mac log verbose** taken on the cable modem side showing failure due to bad COS:

```

w3d:      885643.820   CMAC_LOG_STATE_CHANGE           registration_state
1w3d:      885643.820   CMAC_LOG_REG_REQ_MSG_QUEUED
1w3d:      885643.824   CMAC_LOG_REG_REQ_TRANSMITTED
1w3d:      885643.828   CMAC_LOG_REG_RSP_MSG_RCVD
1w3d:      885643.828   CMAC_LOG_SERVICE_NOT_AVAILABLE   0x01, 0x01, 0x01
1w3d:      885643.828   CMAC_LOG_RESET_SERVICE_NOT_AVAILABLE
1w3d:      885643.828   CMAC_LOG_STATE_CHANGE           reset_interface_state
1w3d:      885643.832   CMAC_LOG_STATE_CHANGE           reset_hardware_state
1w3d:      885644.416   CMAC_LOG_STATE_CHANGE           wait_for_link_up_state
1w3d:      885644.420   CMAC_LOG_DRIVER_INIT_IDB_RESET   0x8039E23C
1w3d:      885644.420   CMAC_LOG_LINK_DOWN
1w3d:      885644.420   CMAC_LOG_LINK_UP
1w3d:      885644.420   CMAC_LOG_STATE_CHANGE           ds_channel_scanning_state
133.CABLEMODEM.CISCO:
1w3d:      %LINEPROTO-5-UPDOWN: Line protocol on Interface cable-modem0, changed state to
down
1w3d:      885645.528   CMAC_LOG_UCD_MSG_RCVD           1
1w3d:      885646.828   CMAC_LOG_DS_64QAM_LOCK_ACQUIRED  453000000

```

Similarly, **debug cable registration** on the Cisco uBR7246VXR gives the following output:

```

sniper#debug cable registration
CMTS registration debugging is on
sniper#
1d04h: %UBR7200-5-CLASSFAIL: Registration failed for Cable Modem 0001.9659.4461 on
interface Cable2/0/U0:
Bad/Missing Class of Service Config in REG-REQ

```

Note how the cable modem eventually resets and starts all over again.



PacketCable Voice Configuration

This chapter describes all activities you must perform to bring a PacketCable voice deployment into service. This chapter contains information on these variants of PacketCable:

- [PacketCable EMTA Secure Provisioning, page 5-1](#)
- [PacketCable EMTA BASIC Provisioning, page 5-4](#)
- [Euro-PacketCable, page 5-5](#)

This chapter also includes information that will help to solve issues that might arise and hinder PacketCable voice technology deployment:

- [Troubleshooting eMTA Provisioning, page 5-6](#)
- [Troubleshooting Tools, page 5-9](#)
- [Troubleshooting Scenarios, page 5-10](#)
- [Certificate Trust Hierarchy, page 5-14](#)

This chapter assumes that you are familiar with the contents of the PacketCable MTA Device Provisioning Specification, PKT-SPPROV1.5-I01-050128. See www.packetcable.com for details.

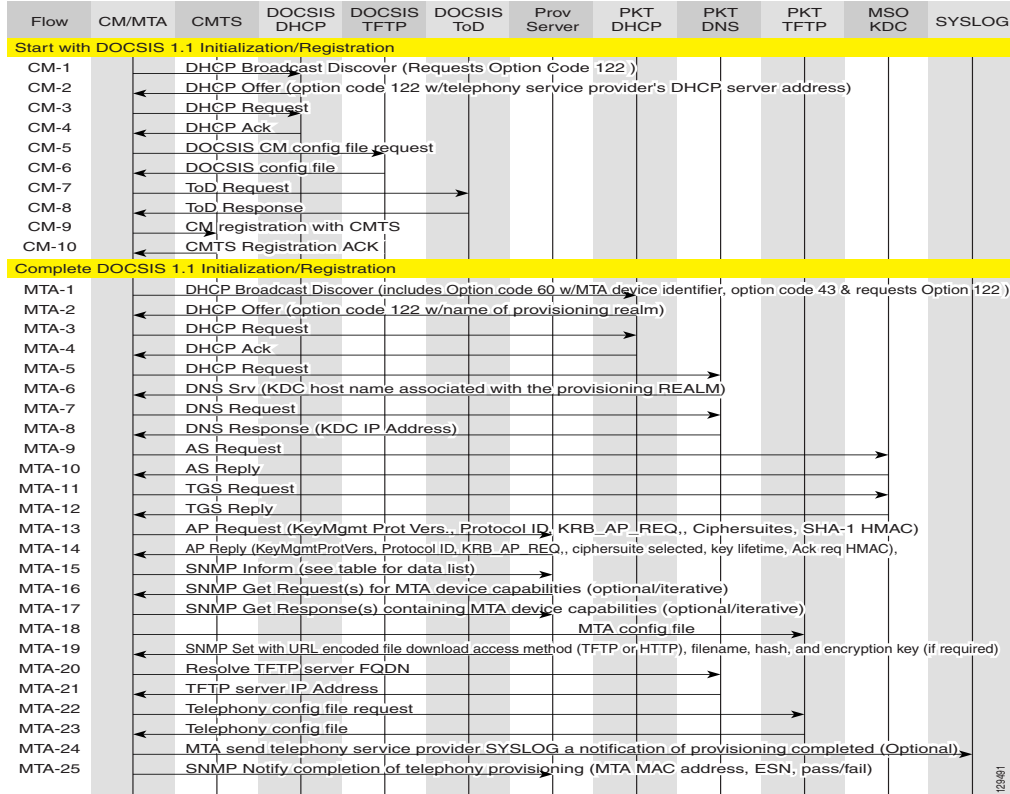
PacketCable EMTA Secure Provisioning

This section deals exclusively with the secure PacketCable voice provisioning. PacketCable Secure is designed to minimize the possibility of theft of telephony service, malicious disruption of service, and so on. PacketCable Secure depends on the Kerberos infrastructure to mutually authenticate the MTA and provisioning system. SNMPv3 is also used to secure the conversation between the MTA and the provisioning system.

BACC PacketCable Secure Provisioning Flow

All PacketCable provisioning flows are defined as a sequence of steps. When diagnosing an EMTA provisioning problem, this flow description helps identify which step in the PacketCable provisioning flow is failing. [Figure 5-1](#) illustrates the secure provisioning flow for PacketCable embedded MTAs.

Figure 5-1 Embedded-MTA Secure Power-on Provisioning Flow

**Note**

It is strongly recommended that you use a protocol analyzer (protocol sniffer) with the ability to capture data packets, to understand exactly which step is failing.

In addition, the KDC log file content is critical to understanding the root cause of any KDC failure.

1. CM 1 through 10—This is the usual DOCSIS CM boot flow with DHCP option 122 added to provide the MTA with a list of PacketCable DHCP servers from which the MTA is allowed to accept DHCP OFFERS.
2. MTA 1 through 4—Using DHCP, the MTA announces itself as a PacketCable MTA and provides information re: which capabilities and provisioning flows it supports (SECURE, BASIC, and so on.). The MTA also obtains addressing information and DHCP option 122. DHCP option 122 will contain PacketCable ProvServ address and security realm name.

This information is needed to allow the MTA to contact the KDC and ProvServer. Some key troubleshooting hints are:

- Check the DHCP relay agent on the CMTS for the correct configuration; ensure that your CMTS points to the correct DHCP server.
- Verify that you have the correct routing between the MTA, CMTS, DHCP server, and the DPE.
- Verify that secondary subnets are configured correctly on the CMTS.
- Check the Network Registrar DHCP configuration—Verify that the scopes are configured, IP addresses are available, and that all secondary subnets are configured.

- Check the BACC configuration—Check the `cnr_ep.properties` file and ensure that the required PacketCable CNR extension properties are configured. See [Appendix B, “PacketCable DHCP Options to BACC Properties Mapping”](#) for additional information on this properties file.
If a packet trace reveals that the MTA is cycling between flow steps MTA 1 and 2, there is probably a problem with the configuration of DHCP option 122 (realm name or provisioning server FQDN sub-options), DHCP option 12 (host name), or DHCP option 15 (domain name). All are required DHCP content for the MTA.
3. MTA 5 through 8—MTA uses the security realm name (delivered within DHCP option 122) to perform a DNS SRV lookup on the KDC service and then resolve the KDC IP address. Some key troubleshooting hints are:
 - Use a packet sniffer to watch for misdirected or malformed DNS packets sent to the Network Registrar DNS.
 - Set the Network Registrar DNS log level to detailed packet tracing and verify what arrives there.
 - Check the DNS configuration—The DNS server specified in `cnr_ep.properties` must contain the realm zone, the SRV record, and the DNS ‘A’ record for the KDC.
 4. MTA-9—The AS_REQ request message is used by the KDC to authenticate the MTA. Some key troubleshooting hints are:
 - Check the KDC log file to determine if the AS_REQ arrives and to observe any errors or warnings.
 - Check that the KDC is configured with the correct MTA_Root certificate. The Manufacturer and Device certificates sent by the MTA within the AS_REQ message *must* chain with the MTA_Root certificate installed at the KDC.
 5. FQDN_REQ/FQDN_REP (not shown in flow)—The KDC extracts the MTA MAC address from the MTA certificate and sends it to the ProvServ for validation. If the ProvServ has the FQDN for that MAC address, it is returned to the KDC. The KDC then compares the FQDN received from the MTA to the FQDN received in the FQDN_REP reply message.
Some key troubleshooting hints are:
 - Use a packet sniffer to watch for misdirected or malformed DNS packets. The MTA passes the ProvServ FQDN (which the MTA received in DHCP option 122) within the AS-REP message to the KDC. The KDC then uses this FQDN to resolve the IP address of the ProvServ
 - Check the file names and content of the KDC key file; the KDC service key in the DPE must match the service key at the KDC. The names of the service key files at KDC are critical.
 6. MTA-10 (AS_REP)—The KDC grants a provisioning service ticket to the MTA and also sends the Service Provider, Local System Provider (optional), and KDC certificate to the MTA. The MTA then verifies that the certificates sent by the KDC chain to the Service Provider Root certificate stored in the MTA. If these certificates do not chain, the MTA will loop back to step MTA 1 of the provisioning flow. See the [“Using the PKCert.sh Tool to Manage KDC Certificates”](#) section on [page 12-41](#) for additional information on the `KDC.cer` file. Some key troubleshooting hints are:
 - Verify that the KDC log files show that the AS-REP message was sent to the device. If a packet trace reveals the MTA is cycling between steps MTA 1 and MTA 10, there is a problem with the service provider certificate chain.
 7. MTA-13 (AP_REQ)—The MTA presents the ticket (received at MTA 10) to the ProvServ specified by DHCP option 122.
 8. MTA-14 (AP_REP)—The ProvServ uses the KDC shared secret to decrypt AP_REQ, validate the ProvServ ticket presented by the MTA, and sends AP_REP with SNMPv3 keys. Subsequent SNMPv3 will now be authenticated and (optionally) encrypted.

9. MTA-15 (SnmpV3 Inform)—The MTA signals the ProvServ that it is ready to receive provisioning information.
10. MTA-19 (SNMPv3 SET)—The ProvServ performs an SNMPv3 SET to the MTA containing the URL for the MTA configuration file, encryption key for the file, and the file's hash value.
11. MTA 22 through 23—The MTA proceeds to download the VoIP configuration file from the specified TFTP server. Note that BACC integrates the TFTP server into the DPE component.
12. MTA-25 (SnmpV3 Inform)—The MTA signals the ProvServ whether the new configuration is acceptable.

PacketCable EMTA BASIC Provisioning

BACC also supports PacketCable BASIC, which offers a simpler, DOCSIS-like, non-secure provisioning flow.

The following describes the BASIC.1 flow:

1. MTA-1—Executes as for the Secure flow.
2. MTA-2—If the prov system is configured to provision the MTA in BASIC.1 mode, the prov system returns a DHCP OFFER containing option 122 sub-option 6, which contains the special reserved realm name “BASIC.1”. This reserved realm name commands the MTA to use the BASIC.1 prov flow. This OFFER also contains the prov system IP address in option 122.3, and the file and siaddr fields are populated with the MTA's configuration file location.
3. MTA-3,4—The remainder of the MTA DHCP exchange is executed (REQUEST and ACK exchanged).
4. MTA skips directly to step MTA-22. Using the file and siaddr information, the MTA TFTP's its configuration file from the prov system.



Note No authentication of MTA/ProvServ or encryption occurs.

The BASIC.2 flow is identical to BASIC.1, with the following exceptions:

- “BASIC.2” is populated into the MTA's DHCP option 122 sub-option 6.
- The MTA issues a provisioning status SNMPv2c INFORM at the very end of the flow, MTA-25 (DHCP option 122 sub-option 3 specifies the INFORM target).

The BASIC PacketCable flow is similar to the DOCSIS flow with the following differences:

- There is no ToD exchange between MTA and the provisioning system.
- The MTA configuration file **MUST** contain an integrity hash. Specifically, the SHA1 hash of the entire content of the config file must be populated into a pkcMtadevConfigFileHash SNMP varbind and placed within a TLV 11 just before the end of file TLV.
- BASIC.2 flow issues a provisioning status SNMPv2c INFORM after the MTA has received and processed its configuration file. This INFORM notifies BACC whether the MTA's provisioning completed successfully or if there was a provisioning problem. If there is a problem, an error can be generated and an event sent from the DPE to the RDU, then on to a BACC client. This INFORM is very useful for debugging configuration file issues.

(See [Chapter 4, “DOCSIS Configuration”](#) for additional information about the DOCSIS flow.)

**Note**

Before using the PacketCable BASIC provisioning flow, ensure that you are using a PacketCable BASIC-capable eMTA. The eMTA must report that it is BASIC capable with its DHCP DISCOVER option 60, TLV 5.18 (supported flows).

PacketCable TLV 38 and MIB Support

BACC supports the complete set of PacketCable 1.5 MIBs.

BACC supports TLV 38 in PacketCable configuration templates. This TLV lets you configure multiple SNMP notification targets. Configuration of this TLV means that all notifications are also issued to the targets configured through TLV 38.

SNMP v2C Notifications

BACC supports both SNMP v2C TRAP and INFORM notifications from the PacketCable MTA.

Euro-PacketCable

Euro-PacketCable services are essentially the European equivalent of North American PacketCable services with the following differences:

- Euro-PacketCable uses different MIBs.
- Euro-PacketCable uses a different set of device certificates (MTA_Root.cer) and service provider certificates (Service Provider Root).

For Euro-PacketCable certificates, the kdc.ini must have the property “euro-packetcable = true”. See [Euro-PacketCable Support, page 2-12](#) for more information.

When using Euro-PacketCable, ensure that the value of the PacketCable property /pktcbl/prov/locale is set to EURO. (The default is NA for North America.) You can specify the locale in the Configuration File utility. See [Using the Configuration File Utility, page 12-25](#) for more information.

Euro-PacketCable MIBs

Euro-PacketCable MIBs are essentially snapshots of draft-IETF MIBs. MTA configuration files consist essentially of SNMPVarBinds that reference the MIBs. There are substantial differences between the PacketCable and Euro MIBs; therefore, the PacketCable and Euro-PacketCable configuration files are incompatible. During installation sample files for PacketCable (cw29_config.tmpl) and Euro-PacketCable (ecw15_mta_config.tmpl) are copied to the <BACC_HOME>/rdu/samples directory.

The following Euro-PacketCable MIBS are shipped with BACC:

- DOCS-IETF-BPI2-MIB
- INTEGRATED-SERVICES-MIB
- DIFFSERV-DSCP-TC
- DIFFSERV-MIB

- TCOMLABS-MIB
- PKTC-TCOMLABS-MTA-MIB
- PKTC-TCOMLABS-SIG-MIB

Configuring Euro-PacketCable MIBs

To configure BACC to use Euro-PacketCable MIBs, you must change the BACC RDU property that specifies the MIBs to be loaded. By default, this property contains the PacketCable MIBs

You can do this in one of the following ways:

- Modify `rdu.properties` and restart RDU.
- Use the Administrator's User Interface. Navigate to Configuration > Defaults > System Defaults and replace the MIB list with the list shown below. You do not need to restart RDU.
- Use Prov API `changeSystemDefaults()` call. You do not need to restart RDU.

The property name is `/snmp/mibs/mibList` (properties file), or `SNMPPropertyKeys.MIB_LIST` (the Prov API constant name). The property value is a comma-separated value (CSV) consisting of the required MIB names, as shown:

```
/snmp/mibs/mibList=SNMPv2-SMI,SNMPv2-TC,INET-ADDRESS-MIB,CISCO-SMI,CISCO-TC,SNMPv2-MIB,RFC
1213-MIB,IANAifType-MIB,IF-MIB,DOCS-IF-MIB,DOCS-IF-EXT-MIB,DOCS-BPI-MIB,CISCO-CABLE-SPECTR
UM-MIB,CISCO-DOCS-EXT-MIB,SNMP-FRAMEWORK-MIB,DOCS-CABLE-DEVICE-MIB,DOCS-CABLE-DEVICE-MIB-O
BSOLETE,DOCS-QOS-MIB,CISCO-CABLE-MODEM-MIB,DOCS-IETF-BPI2-MIB,INTEGRATED-SERVICES-MIB,DIFF
SERV-DSCP-TC,DIFFSERV-MIB,TCOMLABS-MIB,PKTC-TCOMLABS-MTA-MIB,PKTC-TCOMLABS-SIG-MIB
```

Troubleshooting eMTA Provisioning

Provisioning PacketCable Embedded Media Terminal Adapters (eMTAs) is a relatively complex process; however, with the right tools and ‘tricks of the trade,’ getting eMTAs operational can be straightforward.

This chapter assumes that Network Registrar and BACC are both in use; however, much of the information also applies for other deployments. Basic knowledge of Network Registrar (scopes, policies, basic DNS zone setup, and record entry) and BACC (class of service, DHCP criteria, external files, and BACC directory structure) is assumed.

The PacketCable eMTA provisioning process consists of 25 steps for the Secure flow; the BASIC flow has far fewer steps. To troubleshoot eMTAs, knowledge of these 25 steps from the PacketCable provisioning specification is absolutely essential.

This section contains the following topics:

- [Components, page 5-6](#)
- [Key Variables, page 5-8](#)

Components

Before troubleshooting eMTAs, you should be familiar with the following system components as described in the subsections that follow.

- [Embedded Media Terminal Adapter](#)
- [DHCP Server](#)
- [DNS Server](#)
- [Key Distribution Center](#)
- [PacketCable Provisioning Server](#)
- [Call Management Server](#)

Embedded Media Terminal Adapter

The eMTA is a cable modem and a Media Terminal Adapter (MTA) in one box, with a common software image. The CM and MTA each have their own MAC address and each performs DHCP to get its own IP address. The eMTA contains, at minimum, three certificates. One certificate is a unique MTA certificate. A second certificate identifies the MTA's manufacturer. Both the device and manufacture certificates are sent by the MTA to authenticate itself to the key distribution center (KDC). The third certificate is a telephony root certificate used to verify the certificates sent by the KDC to the MTA. The KDC's certificates will be chained from the telephony root, therefore the telephony root must reside on the MTA to validate the authenticity of the KDC certificates. The MTA portion receives its own configuration file, which it uses to identify its controlling call agent, among other things.

DHCP Server

The DOCSIS specifications mandate that cable modems negotiate their IP address using the Dynamic Host Configuration Protocol (DHCP). The MTA, like most CPEs on a DOCSIS network, must use DHCP to obtain its IP address and other crucial information (DNS servers, PacketCable option 122 for Kerberos realm name, provisioning server FQDN).



Note

The cable modem portion, in addition to its normally required DHCP options, also requests, and must receive, Option 122 suboption 1, which it passes to the MTA portion as the IP address of the correct DHCP server from which to accept offers.

When using BACC with PacketCable support, be aware that a correctly-configured BACC will automatically populate the ToD server, DNS servers, TFTP server, as well as the Option 122 fields; these do not need to be explicitly set in the Network Registrar policy.

DNS Server

The Domain Name System (DNS) server is fundamental in PacketCable provisioning. The PacketCable provisioning server, which is the device provisioning engine (DPE) in a BACC architecture, must have an address (A) record in the appropriate zone, because its fully qualified domain name (FQDN) is provided to the MTA in Option 122 by the DHCP server. The KDC realm must have a zone of the same name as the realm name containing a server (SRV) record that contains the FQDN of the Kerberos server.

The Kerberos server identified in the SRV record must itself have an A record in the appropriate zone. The call management server (CMS) identified in the MTA configuration file must also have an A record in the appropriate zone. Lastly, the MTAs themselves must have A records in the appropriate zone, since the CMS reaches the MTA by resolving its FQDN. Dynamic DNS (DDNS) is the preferred method of creating A records for the MTA. See the Cisco Network Registrar documentation for information on configuring and troubleshooting DDNS.

Key Distribution Center

The KDC is responsible for authenticating MTAs. As such, it must check the MTA's certificate, and provide its own certificates so that the MTA can authenticate the KDC. It also communicates with the DPE (the provisioning server) to validate that the MTA is provisioned on the network.

PacketCable Provisioning Server

The PacketCable provisioning server is responsible for communicating the location of the MTA configuration file to the MTA, and/or provisioning MTA parameters via SNMP. SNMPv3 is used for all communication between the MTA and the provisioning server. The keys used to initiate SNMPv3 communication are obtained by the MTA during its authentication phase with the KDC. Provisioning server functionality is provided by the DPE in a BACC architecture.

Call Management Server

The call management server (CMS) is essentially a soft switch, or call-agent, with additional PacketCable functionality to control, among other things, quality of service on a cable network. The MTA sends a network call signaling (NCS) restart in progress (RSIP) message to the CMS upon successful PacketCable provisioning.

Key Variables

This section describes the key variables required to provision an eMTA correctly.

- [Certificates, page 5-8](#)
- [Scope Selection Tag\(s\), page 5-9](#)
- [MTA Configuration File, page 5-9](#)

Certificates

The MTA_Root.cer file contains the MTA root certificate (a certificate that is rooted in official PacketCable MTA root). The MTA_Root.cer will not usually cause significant problems.

You must know in advance what telephony root certificate is required for the MTAs you want to provision. Deployments in production networks use telephony certificates rooted in the PacketCable real root. There is also a PacketCable test root used in lab and testing environments. The KDC certificates used by the KDC to authenticate itself to the MTA must be rooted in the same telephony root that is stored on the MTA (PacketCable real or test root). Most MTA vendors support test images that have Telnet and/or HTTP login capabilities so that you can determine which telephony root is enabled, and change the root used (in most cases, you can only select between the PacketCable real or test root).

The most common scenario has the KDC loaded with certificates (from the \$BACC_HOME/kdc/solaris/packetcable/certificates dir) as follows:

- CableLabs_Service_Provider_Root.cer
- Service_Provider.cer
- Local_System.cer
- KDC.cer
- MTA_Root.cer

The first four certificates comprise the telephony certificate chain. The MTA_Root.cer file contains the MTA root used by the KDC to validate the certificates sent by the MTA.

**Note**

Refer to the [“Using the PKCert.sh Tool to Manage KDC Certificates”](#) section on page 12-41 for information on installation and management of KDC certificates.

To determine if you are using PacketCable test root, open the CableLabs_Service_Provider_Root.cer file in Windows, and validate that the Subject OrgName entry is **O = CableLabs**, and/or check that the Subject Alternative name reads **CN=CABLELABS GENERATED TEST ROOT FOR EQUIPMENT TEST PURPOSES ONLY**.

The KDC certificate (KDC.cer) contains the realm name to use. The realm name that BACC (and the corresponding DNS zone) is configured to use must match this realm name. Additionally, the MTA configuration file realm org name must match the organization name as seen in the telephony root.

The KDC certificate has a corresponding private key that must be installed in the \$BACC_HOME/kdc/solaris directory. Usually it is named KDC_private_key.pkcs8 or KDC_private_key_proprietary. When changing certificates, you must also change the private key.

Scope Selection Tag(s)

In most scenarios, BACC is involved in processing all DHCP requests from scopes with scope selection tags that match selection criteria specified in the DHCP criteria page of the BACC administrative user interface. Client Class can also be used to tie scopes to BACC processing; ensure you make this association before you attempt to provision devices.

MTA Configuration File

The MTA configuration file contains the location of the CMS. Additionally, it must contain an entry for Realm Name. This value must match that of the certificate chain in use.

Certain table entries within the MTA configuration file are indexed by the realm name delivered to the MTA in Option 122. This realm name entry in the MTA configuration file must match that delivered in Option 122. For example, if **DEF.COM** was the realm name delivered in Option 122, MTA configuration file entries in the pktcMtaDevRealm table would be indexed with a suffix made up of the ASCII-coded character values (in dot delimited decimal format when using the Cisco Broadband Configurator) of the realm name, for example 68.69.70.46.67.79.77. There are many free ASCII conversion pages available on the web to make this conversion easier.

Troubleshooting Tools

The 25 eMTA secure provisioning steps contained in PacketCable MTA Device Provisioning Specification, are shown in [Figure 5-1](#).

This section contains the following topics:

- [Logs, page 5-10](#)
- [Ethereal, SnifferPro, or Other Packet Capture Tools, page 5-10](#)

Logs

These log files are used to maintain the following information:

- The Network Registrar has two logs (name_dhcp_1_log and name_dns_1_log), which contain the most recent logging entries from Network Registrar. Look in these files for DHCP- or DNS-related problems.
- The \$BACC_HOME/kdc/logs/kdc.log file shows all KDC interactions with MTAs, and KDC interactions with the DPE.
- The \$BACC_DATA/dpe/logs/dpe.log file shows the major steps related to SNMPv3 interaction with the MTA. You can also use the **show log** CLI command if you are working with the hardware DPE.



Note

Turning on the tracing of snmp, registration server and registration server detail messages, using the command line interface, helps to troubleshoot potential PacketCable problems. See the *Cisco Broadband Access Center for Cable Command Line Interface Reference* for information on the appropriate troubleshooting commands.

Ethereal, SnifferPro, or Other Packet Capture Tools

A packet capture tool is indispensable when troubleshooting the eMTAs. The Ethereal version, as packaged by CableLabs, includes numerous packet decoders specific to PacketCable. These include the Kerberos AS and AP packets.

- If you suspect that a specific failure is DHCP-related, capture packets while filtering on packets sourced from, or destined to, the CMTS cable interface IP address and the DHCP server IP address.
- If you suspect that a specific failure is related to any of the 25 steps occurring after DHCP, filter all packets to and from the eMTA IP address. This provides a very concise, easy-to-follow trace of provisioning steps 5 through 25, as shown in [Figure 5-1](#).

Troubleshooting Scenarios

The scenarios listed in [Table 5-1](#) are possible failures involving embedded MTAs.

Table 5-1 Troubleshooting Scenarios

If this problem occurs...	Which indicates this potential cause ...	And to correct it, you should...
KDC does not start.	The KDC certificate does not correspond to the private key.	Ensure that you have matching certificates and private key.
	The KDC license expired or is missing.	Restore KDC license to \$BACC_HOME/kdc directory.

Table 5-1 Troubleshooting Scenarios (continued)

If this problem occurs...	Which indicates this potential cause ...	And to correct it, you should...
The MTA device does not appear in the BACC Devices page.	An incorrect cable helper address may have been configured.	Fix the helper address.
	The scope selection tags do not match the DHCPCriteria selected in the BACC user interface.	Verify that the MTA scope selection tags match those in the PacketCable DHCP criteria created, in BACC, for the relevant MTA(s).
	The Network Registrar Extension point is not properly installed.	Re-install the Network Registrar Extension Point. See the <i>Cisco Broadband Access Center for Cable Installation Guide</i> for the appropriate instruction.
	The cable modem portion did not receive Option 122.	Verify that the tags on the scope of the cable modem portion match the DOCSIS DHCP criteria configure for BACC.
MTA does not accept the DHCP offer and continually cycles through the DHCP flow.	There are invalid DHCP options configured.	Check that scope policy includes DNS server option, and/or check <code>cnr_ep.properties</code> file includes entry for primary and secondary DNS servers.
	The DHCP offer may have come from a DHCP server different from the one indicated in the cable modem portion's Option 122 suboption 1.	Check the <code>cnr_ep.properties</code> file to ensure that the primary and secondary DHCP servers are set correctly.
Both the KDC.log file and the ethereal trace indicate that the MTA never contacts the KDC.	An incorrect DNS server is specified in the <code>cnr_ep.properties</code> file and/or the MTA scope policy.	Check /correct <code>cnr_ep.properties</code> DNS servers.
	A zone is missing or has been incorrectly set up for the Kerberos realm.	Make sure a zone with same name as realm is created and contains an 'SRV' record of format ' <code>_kerberos._udp 0 0 88 <KDC FQDN></code> '.
	There is a missing or incorrect KDC 'A' record entry.	Ensure that an 'A' record exists for the FQDN contained in the Kerberos zone's 'SRV' record.
	The DPE FQDN cannot be resolved.	Ensure that <code>dpe.properties</code> <code>provFQDNs</code> entry has the correct FQDN and IP of the DPE.

Table 5-1 Troubleshooting Scenarios (continued)

If this problem occurs...	Which indicates this potential cause ...	And to correct it, you should...
KDC reports failure during the Kerberos AS-Request.	The MTA certificate does not match the MTA root used by KDC.	Verify that the MTA_Root.cer is correct by comparing the MTA_Root.cer against that used on a working system. If it is correct, the MTA itself could have a certificate problem. This situation is extremely rare and if this is the case, contact the MTA manufacturer.
	FQDN lookup by KDC to Prov Server failed. The device may not yet be provisioned in BACC.	Verify that the device appears. It should be given both a class of service and a DHCP criteria.
	A clock skew error. See the “ PacketCable Checklists ” section on page 3-8 for additional information.	Ensure that all BACC network elements are clock-synced via NTP. See the <i>Broadband Access Center for Cable Command Line Interface Reference</i> for additional information.
	A mismatch may exist between the KDC and the DPE.	Check that these three entries exist in the \$BACC_HOME/kdc/solaris/keys directory: <ul style="list-style-type: none"> • mtafqdnmap,dpe.abc.com@DEF.COM • mtaprovsrvr,dpe.abc.com@DEF.COM • krbtgt,DEF.COM@DEF.COM Your system will have the DPE FQDN, and Realm name different from this example. Contents of these entries must match the entry in either the dpe.properties ‘KDCServiceKey’ entry, or the keys generated using the keygen utility.
The KDC reports success at the AS-Request/Reply (steps 9 and 10 in Figure 5-1) but MTA never moves past step 9, and continually reprovisions up to that step.	There is a certificate mismatch between the telephony root loaded or enabled on the MTA, and that loaded on the KDC.	Check certificates on MTA and KDC.
	Although highly unlikely, it is possible that there is a corrupted telephony certificate chain.	Ensure that the correct certificate is loaded/enabled on MTA. If no devices can be provisioned correctly, try a different certificate on the KDC.
	Note If other devices are provisioned correctly, this is not the cause of the problem.	
Failure at AP Request/Reply (step 14 in Figure 5-1)	A clock skew error. See the “ PacketCable Checklists ” section on page 3-8 for additional information.	Ensure that all BACC network elements are clock-synced via NTP. See the <i>Broadband Access Center for Cable Command Line Interface Reference</i> for additional information.
	Cannot resolve Prov Server FQDN.	Make sure that the provisioning server (DPE) has a correct DNS entry. Ensure that dpe.properties provFQDNs entry has the correct FQDN and IP of the provisioning server (DPE)
	There is no route from the MTA to the DPE.	Correct the routing problem.

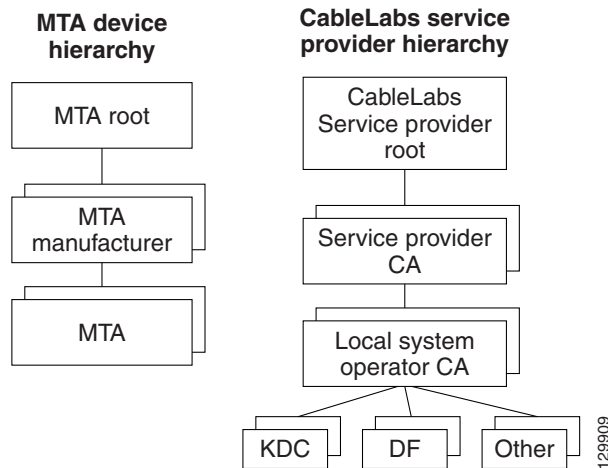
Table 5-1 Troubleshooting Scenarios (continued)

If this problem occurs...	Which indicates this potential cause ...	And to correct it, you should...
The MTA never issues a TFTP request for a configuration file.	There is no route to the TFTP server running on the DPE.	Correct the routing problem.
The MTA never receives the TFTP configuration file.	The configuration file is not cached at the DPE.	Wait until the next provisioning attempt, at which time the file should be cached. If this fails, reset the MTA.
	A conflicting TFTP server option is included in the network registrar MTA scope policy.	Since BACC inserts the DPE address for the TFTP server, you can safely remove this option from the policy.
The MTA receives a configuration file but the DPE fails to receive the SNMP Inform (step 25 in Figure 5-1) as seen in the dpe.log file.	One of: <ul style="list-style-type: none"> An internal conflict in the configuration file. A conflict with Realm origin of the telephony certificate chain. A conflict with the Realm Name provided in Option 122. 	Ensure that the MTA configuration file is consistent.
The MTA reports success (step 25 in Figure 5-1) although an RSIP is not sent.	The MTA cannot resolve the IP address of the CMS FQDN given in the MTA configuration file.	Verify that a DNS entry exists for the CMS.
	The MTA cannot reach the IP address(es) of the CMS. This is an indication that no route is configured.	Resolve all routing problems.
The MTA reports success (step 25 in Figure 5-1), although it proceeds to contact the KDC again for CMS service.	The MTA configuration file points to an incorrect cable modem.	Correct the configuration file, or reconfigure the Cisco BTS 10200 to use the FQDN listed in the configuration file.
	The MTA configuration file has its pktcMtaDevCmsIPsecCtrl value missing, or it is set to 1. This means it will perform secure NCS call signaling, or it will use an ASCII suffix that does not match that of the CMS FQDN.	Correct the configuration file. If you intend to perform secure signaling, take the necessary steps to configure the KDC and the BTS for support.
The MTA reports success (step 25 in Figure 5-1), RSIPs, but gets no response or gets an error in response from the soft switch.	The MTA is unprovisioned or has been incorrectly provisioned on the Cisco BTS 10200.	Provision MTA on the Cisco BTS 10200.
	An eMTA DNS entry does not exist.	Place an entry in the correct DNS zone for the eMTA. Dynamic DNS is the preferred method. See the Cisco Network Registrar documentation for information on enabling DDNS.

Certificate Trust Hierarchy

There are two certificate hierarchies affiliated with BACC PacketCable, the MTA Device Certificate Hierarchy and the CableLabs Service Provider Certificate Hierarchy, as shown in [Figure 5-2](#).

Figure 5-2 PacketCable Certificate Hierarchy



Prior to working with the BACC PacketCable implementation, you should thoroughly familiarize yourself with these technology documents:

- *RFC 2459 Internet X.509 Public Key Infrastructure Certificate and CRL Profile*
- *DOCSIS Baseline Privacy Plus Interface Specification, SP-BPI+-I11-040407, April 7, 2004*



Note

While Euro-PacketCable uses the security specifications from PacketCable [PKT-SP-SEC-I08-030415], some changes are needed in relation to the digital certificates that are used in a Euro-PacketCable environment. To keep Euro-PacketCable and PacketCable as much alike as possible, Euro-PacketCable uses all PacketCable security technology, including new revision of the security specifications [PKTSP-SEC-I08-030415].

The elements of the Euro-PacketCable certificates that are different from the PacketCable certificates are indicated in the tables below.

For Euro-PacketCable the Euro-PacketCable certificates are the only valid certificates, any requirements that are stated in [PKT-SP-SEC-I08-030415] for PacketCable which refer to PacketCable Certificates are changed to the corresponding requirements for the Euro-PacketCable certificates.

Euro-PacketCable compliant embedded MTAs must have the Euro-DOCSIS root CVC CA's public key stored in the CM's non-volatile memory instead of the DOCSIS CVC CA's public key. Euro-PacketCable compliant standalone MTAs must have the tComLabs CVC Root Certificate and the tComLabs CVC CA certificate stored in non-volatile memory. The CVC of manufacturers are verified by checking the certificate chain.

Certificate Validation

PacketCable certificate validation in general involves validation of an entire chain of certificates. For example, when the Provisioning Server validates an MTA Device certificate, the following chain of certificates is validated:

MTA Root Certificate + MTA Manufacturer Certificate + MTA Device Certificate

The signature on the MTA Manufacturer Certificate is verified with the MTA Root Certificate and the signature on the MTA Device Certificate is verified with the MTA Manufacturer Certificate. The MTA Root Certificate is self-signed and is known in advance to the Provisioning Server. The public key present in the MTA Root Certificate is used to validate the signature on this same certificate.

Usually the first certificate in the chain is not explicitly included in the certificate chain that is sent over the wire. In the cases where the first certificate is explicitly included it must already be known to the verifying party ahead of time and must *not* contain any changes to the certificate with the possible exception of the certificate serial number, validity period, and the value of the signature. If changes other than these exist in the CableLabs Service Provider Root certificate that was passed over the wire in comparison to the known CableLabs Service Provider Root certificate, the device making the comparison must fail the certificate verification.

The exact rules for certificate chain validation must fully comply with RFC 2459, where they are referred to as Certificate Path Validation. In general, X.509 certificates support a liberal set of rules for determining if the issuer name of a certificate matches the subject name of another. The rules are such that two name fields may be declared to match even though a binary comparison of the two name fields does not indicate a match. RFC 2459 recommends that certificate authorities restrict the encoding of name fields so that an implementation can declare a match or mismatch using simple binary comparison.

PacketCable security follows this recommendation. Accordingly, the DER-encoded `tbsCertificate.issuer` field of a PacketCable certificate must be an exact match to the DER-encoded `tbsCertificate.subject` field of its issuer certificate. An implementation may compare an issuer name to a subject name by performing a binary comparison of the DER-encoded `tbsCertificate.issuer` and `tbsCertificate.subject` fields.

The sections below specify the required certificate chain, which must be used to verify each certificate that appears at the leaf node (at the bottom) in the PacketCable certificate trust hierarchy illustrated in [Figure 5-2](#).

Validity period nesting is not checked and intentionally not enforced. Thus, the validity period of a certificate need not fall within the validity period of the certificate that issued it.

MTA Device Certificate Hierarchy

The device certificate hierarchy exactly mirrors that of the DOCSIS1.1/BPI+ hierarchy. It is rooted at a CableLabs issued PacketCable MTA Root certificate, which is used as the issuing certificate of a set of manufacturer's certificates. The manufacturer's certificates are used to sign the individual device certificates.

The information contained in the following tables contains the PacketCable specific values for the required fields according to RFC 2459. These PacketCable specific values must be followed according to [Table 5-2](#), except that Validity Periods should be as given in the respective tables. If a required field is not specifically listed for PacketCable then the guidelines in RFC 2459 must be followed.

MTA Root Certificate

This certificate must be verified as part of a certificate chain containing the MTA Root Certificate, MTA Manufacturer Certificate, and the MTA Device Certificate.

Table 5-2 MTA Root Certificate

MTA Root Certificate											
Subject Name Form	<table border="1"> <thead> <tr> <th>PacketCable</th> <th>Euro-PacketCable</th> </tr> </thead> <tbody> <tr> <td>C=US</td> <td>C=BE</td> </tr> <tr> <td>O=CableLabs</td> <td>O=tComLabs</td> </tr> <tr> <td>OU=PacketCable</td> <td>OU=Euro-PacketCable</td> </tr> <tr> <td>CN=PacketCable Root Device Certificate Authority</td> <td>CN=Euro-PacketCable Root Device Certificate Authority</td> </tr> </tbody> </table>	PacketCable	Euro-PacketCable	C=US	C=BE	O=CableLabs	O=tComLabs	OU=PacketCable	OU=Euro-PacketCable	CN=PacketCable Root Device Certificate Authority	CN=Euro-PacketCable Root Device Certificate Authority
PacketCable	Euro-PacketCable										
C=US	C=BE										
O=CableLabs	O=tComLabs										
OU=PacketCable	OU=Euro-PacketCable										
CN=PacketCable Root Device Certificate Authority	CN=Euro-PacketCable Root Device Certificate Authority										
Intended Usage	This certificate is used to sign MTA Manufacturer Certificates and is used by the KDC. This certificate is not used by the MTAs and thus does not appear in the MTA MIB.										
Signed By	Self-signed										
Validity Period	20+ years. It is intended that the validity period is long enough that this certificate is never reissued.										
Modulus Length	2048										
Extensions	keyUsage[c,m](keyCertSign, cRLSign) subjectKeyIdentifier[n,m] basicConstraints[c,m](cA=true, pathLenConstraint=1)										

MTA Manufacturer Certificate

This certificate must be verified as part of a certificate chain containing the MTA Root Certificate, MTA Manufacturer Certificate, and the MTA Device Certificate. The state/province, city and manufacturer's facility are optional attributes. A manufacturer may have more than one manufacturer's certificate, and there may exist one or more certificates per manufacturer. All certificates for the same manufacturer may be provided to each MTA either at manufacture time or during a field update. The MTA must select an appropriate certificate for its use by matching the issuer name in the MTA Device Certificate with the subject name in the MTA Manufacturer Certificate. If present, the authorityKeyIdentifier of the device certificate must match the subjectKeyIdentifier of the manufacturer certificate as described in RFC 2459. The <CompanyName> field that is present in O and CN may be different in the two instances.

Table 5-3 MTA Manufacturer Certificates

MTA Manufacturer Certificate																	
Subject Name Form	<table border="0"> <tr> <td style="vertical-align: top;">PacketCable</td> <td style="vertical-align: top;">Euro-PacketCable</td> </tr> <tr> <td>C=US</td> <td>C = <Country of Manufacturer></td> </tr> <tr> <td>O=CableLabs</td> <td>O = <Company Name></td> </tr> <tr> <td>OU=PacketCable</td> <td>[stateOrProvinceName = <state/province>]</td> </tr> <tr> <td>CN=PacketCable Root Device Certificate Authority</td> <td>[localityName = <City>]</td> </tr> <tr> <td></td> <td>OU = Euro-PacketCable</td> </tr> <tr> <td></td> <td>[organizationalUnitName = <Manufacturing Location>]</td> </tr> <tr> <td></td> <td>CN = <Company Name> Euro-PacketCable CA</td> </tr> </table>	PacketCable	Euro-PacketCable	C=US	C = <Country of Manufacturer>	O=CableLabs	O = <Company Name>	OU=PacketCable	[stateOrProvinceName = <state/province>]	CN=PacketCable Root Device Certificate Authority	[localityName = <City>]		OU = Euro-PacketCable		[organizationalUnitName = <Manufacturing Location>]		CN = <Company Name> Euro-PacketCable CA
PacketCable	Euro-PacketCable																
C=US	C = <Country of Manufacturer>																
O=CableLabs	O = <Company Name>																
OU=PacketCable	[stateOrProvinceName = <state/province>]																
CN=PacketCable Root Device Certificate Authority	[localityName = <City>]																
	OU = Euro-PacketCable																
	[organizationalUnitName = <Manufacturing Location>]																
	CN = <Company Name> Euro-PacketCable CA																
Intended Usage	This certificate is issued to each MTA manufacturer and can be provided to each MTA as part of the secure code download as specified by the PacketCable Security Specification (either at manufacture time, or during a field update). This certificate appears as a read-only parameter in the MTA MIB. This certificate along with the MTA Device Certificate is used to authenticate the MTA device identity (MAC address) during authentication by the KDC.																
Signed By	MTA Root Certificate CA																
Validity Period	20 years																
Modulus Length	2048																
Extensions	keyUsage[c,m](keyCertSign, cRLSign), subjectKeyIdentifier[n,m], authorityKeyIdentifier[n,m](keyIdentifier=<subjectKeyIdentifier value from CA certificate>), basicConstraints[c,m](cA=true, pathLenConstraint=0)																

MTA Device Certificate

This certificate must be verified as part of a certificate chain containing the MTA Root Certificate, MTA Manufacturer Certificate and the MTA Device Certificate. The state/province, city and manufacturer's facility are optional attributes. The MAC address must be expressed as six pairs of hexadecimal digits separated by colons, for example, "00:60:21:A5:0A:23". The alpha hexadecimal characters (A-F) must be expressed as uppercase letters. The MTA device certificate should not be replaced or renewed.

Table 5-4 MTA Device Certificates

MTA Device Certificate			
Subject Name Form	<table border="0"> <tr> <td style="vertical-align: top;"> PacketCable C=<country> O=<Company Name> [ST=<state/province>] [L=<city>], OU=PacketCable [OU=<Product Name>] [OU=<Manufacturer's Facility>] CN=<MAC Address> </td> <td style="vertical-align: top;"> Euro-PacketCable C = <Country of Manufacturer> O = <Company Name> [ST = <state/province>] [L = <City>] OU = Euro-PacketCable [OU= <Product Name>] [OU = <Manufacturing Location>] CN = <MAC Address> </td> </tr> </table>	PacketCable C=<country> O=<Company Name> [ST=<state/province>] [L=<city>], OU=PacketCable [OU=<Product Name>] [OU=<Manufacturer's Facility>] CN=<MAC Address>	Euro-PacketCable C = <Country of Manufacturer> O = <Company Name> [ST = <state/province>] [L = <City>] OU = Euro-PacketCable [OU= <Product Name>] [OU = <Manufacturing Location>] CN = <MAC Address>
PacketCable C=<country> O=<Company Name> [ST=<state/province>] [L=<city>], OU=PacketCable [OU=<Product Name>] [OU=<Manufacturer's Facility>] CN=<MAC Address>	Euro-PacketCable C = <Country of Manufacturer> O = <Company Name> [ST = <state/province>] [L = <City>] OU = Euro-PacketCable [OU= <Product Name>] [OU = <Manufacturing Location>] CN = <MAC Address>		
Intended Usage	This certificate is issued by the MTA manufacturer and installed in the factory. The provisioning server cannot update this certificate. This certificate appears as a read-only parameter in the MTA MIB. This certificate is used to authenticate the MTA device identity (MAC address) during provisioning.		
Signed By	MTA Manufacturer Certificate CA		
Validity Period	At least 20 years		
Modulus Length	1024, 1536 or 2048		
Extensions	keyUsage[c,o](digitalSignature, keyEncipherment) authorityKeyIdentifier[n,m](keyIdentifier=<subjectKeyIdentifier value from CA certificate>)		

MTA Manufacturer Code Verification Certificates

Code Verification Certificate (CVC) specification for embedded MTAs must be identical to the DOCSIS 1.1 CVC, specified in DOCSIS specification SP-BPI+-I11-040407.

CableLabs Service Provider Certificate Hierarchy

The Service Provider Certificate Hierarchy is rooted at a CableLabs issued CableLabs Service Provider Root certificate. That certificate is used as the issuing certificate of a set of service provider's certificates. The service provider's certificates are used to sign an optional local system certificate. If the local system certificate exists then that is used to sign the ancillary equipment certificates, otherwise the ancillary certificates are signed by the Service Provider's CA.

The information contained in [Table 5-5](#) contains the specific values for the required fields according to RFC 2459. These specific values must be followed. If a required field is not specifically listed then the guidelines in RFC 2459 must be followed exactly.

CableLabs Service Provider Root Certificate

Before any Kerberos key management can be performed, an MTA and a KDC need to perform mutual authentication using the PKINIT extension to the Kerberos protocol. An MTA authenticates a KDC after it receives a PKINIT Reply message containing a KDC certificate chain. In authenticating the KDC, the MTA verifies the KDC certificate chain, including KDC's Service Provider Certificate signed by the CableLabs Service Provider Root CA.

Table 5-5 CableLabs Service Provider Root Certificates

CableLabs Service Provider Root Certificate		
Subject Name Form	PacketCable C=US O=CableLabs CN=CableLabs Service Provider Root CA	Euro-PacketCable C=BE O=tComLabs CN=tComLabs Service Provider Root CA
Intended Usage	This certificate is used to sign Service Provider CA certificates. This certificate is installed into each MTA at the time of manufacture or with a secure code download as specified by the PacketCable Security Specification and cannot be updated by the Provisioning Server. Neither this root certificate nor the corresponding public key appears in the MTA MIB.	
Signed By	Self-signed	
Validity Period	20+ years. It is intended that the validity period is long enough that this certificate is never reissued.	
Modulus Length	2048	
Extensions	keyUsage[c,m](keyCertSign, cRLSign) subjectKeyIdentifier[n,m] basicConstraints[c,m](cA=true)	

Service Provider CA Certificate

This is the certificate held by the service provider, signed by the CableLabs Service Provider Root CA. It is verified as part of a certificate chain that includes the CableLabs Service Provider Root Certificate, Telephony Service Provider Certificate, optional Local System Certificate and an end-entity server certificate. The authenticating entities normally already possess the CableLabs Service Provider Root Certificate and it is not transmitted with the rest of the certificate chain.

The fact that a Service Provider CA Certificate is always explicitly included in the certificate chain allows a Service Provider the flexibility to change its certificate without requiring reconfiguration of each entity that validates this certificate chain (for example, MTA validating a PKINIT Reply). Each time the Service Provider CA Certificate changes, its signature must be verified with the CableLabs Service Provider Root Certificate. However, a new certificate for the same Service Provider must preserve the same value of the OrganizationName attribute in the SubjectName. The <Company> field that is present in O and CN may be different in the two instances.

Table 5-6 CableLabs Service Provider CA Certificates

CableLabs Service Provider Root Certificate		
Subject Name Form	PacketCable C=<Country> O=<Company> CN=<Company> CableLabs Service Provider CA	Euro-PacketCable C=<Country> O=<Company> CN=<Company> tComLabs Service Provider CA
Intended Usage	This certificate is used to sign Service Provider CA certificates. This certificate is installed into each MTA at the time of manufacture or with a secure code download as specified by the PacketCable Security Specification and cannot be updated by the Provisioning Server. Neither this root certificate nor the corresponding public key appears in the MTA MIB.	
Signed By	Self-signed	
Validity Period	20+ years. It is intended that the validity period is long enough that this certificate is never reissued.	
Modulus Length	2048	
Extensions	keyUsage[c,m](keyCertSign, cRLSign), subjectKeyIdentifier[n,m] basicConstraints[c,m](cA=true)	

Local System CA Certificates

A Service Provider CA may delegate the issuance of certificates to a regional Certification Authority called Local System CA (with the corresponding Local System Certificate). Network servers are allowed to move freely between regional Certification Authorities of the same Service Provider. Therefore, the MTA MIB does not contain any information regarding a Local System Certificate (which might restrict an MTA to KDCs within a particular region).

Table 5-7 Local System CA Certificates

Local System CA Certificate		
Subject Name Form	PacketCable C=<Country> O=<Company> OU=<Local System Name> CN=<Company> CableLabs Local System CA	Euro-PacketCable C = <Country> O = <Company> OU = <Local System Name> CN = <Company> tComLabs Local System CA
Intended Usage	A Service Provider CA may delegate the issuance of certificates to a regional Certification Authority called Local System CA (with the corresponding Local System Certificate). Network servers are allowed to move freely between regional Certification Authorities of the same Service Provider. Therefore, the MTA MIB does not contain any information regarding a Local System Certificate (which might restrict an MTA to KDCs within a particular region).	
Signed By	Service Provider CA Certificate	
Validity Period	20 years.	

Table 5-7 Local System CA Certificates (continued)

Local System CA Certificate	
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](keyCertSign, cRLSign), subjectKeyIdentifier[n,m], authorityKeyIdentifier[n,m](keyIdentifier=<subjectKeyIdentifier value from CA certificate>), basicConstraints[c,m](cA=true, pathLenConstraint=0)

Operational Ancillary Certificates

All these are signed by the either the Local System CA or by the Service Provider CA. Other ancillary certificates may be added to this standard at a later time.

Key Distribution Center Certificate

This certificate must be verified as part of a certificate chain containing the CableLabs Service Provider Root Certificate, Service Provider CA Certificate and the Ancillary Device Certificates. The PKINIT specification requires the KDC certificate to include the subjectAltName v.3 certificate extension, the value of which must be the Kerberos principal name of the KDC.

Table 5-8 Key Distribution Center Certificates

Key Distribution Center Certificate		
Subject Name Form	PacketCable C=<Country> O=<Company>, OU=<Local System Name> OU= CableLabs Key Distribution Center CN=<DNS Name>	Euro-PacketCable C = <Country> O = <Company> [OU = <Local System Name>] OU = tComLabs Key Distribution Center CN = <DNS Name>
Intended Usage	To authenticate the identity of the KDC server to the MTA during PKINIT exchanges. This certificate is passed to the MTA inside the PKINIT replies and is therefore not included in the MTA MIB and cannot be updated or queried by the Provisioning Server.	
Signed By	Service Provider CA Certificate or Local System Certificate	
Validity Period	20 years	
Modulus Length	1024, 1536 or 2048	
Extensions	keyUsage[c,o](digitalSignature)authorityKeyIdentifier[n,m](keyIdentifier=<subjectKeyIdentifier value from CA certificate>)subjectAltName[n,m]	

Delivery Function (DF)

This certificate must be verified as part of a certificate chain containing the CableLabs Service Provider Root Certificate, Service Provider CA Certificate and the Ancillary Device Certificates. This certificate is used to sign phase 1 IKE intra-domain exchanges between DFs (which are used in Electronic Surveillance). Although Local System Name is optional, it is required when the Local System CA signs this certificate. The IP address must be specified in standard dotted-quad notation, for example, 245.120.75.22.

Table 5-9 DF Certificates

DF Certificate		
Subject Name Form	PacketCable C=<Country> O=<Company> [OU=<Local System Name>] OU=PacketCable Electronic Surveillance CN=<IP address>	Euro-PacketCable C = <Country> O = <Company> [OU = <Local System Name>] OU = Euro-PacketCable Electronic Surveillance CNe = <IP address>
Intended Usage	To authenticate IKE key management, used to establish IPsec Security Associations between pairs of DFs. These Security Associations are used when a subject that is being legally wiretapped forwards the call and event messages containing call info have to be forwarded to a new wiretap server (DF).	
Signed By	Service Provider CA Certificate or Local System CA Certificate	
Validity Period	20 years	
Modulus Length	2048	
Extensions	keyUsage[c,o](digitalSignature) authorityKeyIdentifier[n,m](keyIdentifier=<subjectKeyIdentifier value from CA certificate>) subjectAltName[n,m] (dNSName=<DNSName>)	

PacketCable Server Certificates

These certificates must be verified as part of a certificate chain containing the CableLabs Service Provider Root Certificate, Service Provider Certificate, Local System Operator Certificate (if used) and the Ancillary Device Certificates. These certificates are used to identify various servers in the PacketCable system. For example, they may be used to sign phase 1 IKE exchanges or to authenticate a PKINIT exchange. Although the Local System Name is optional, it is REQUIRED when the Local System CA signs this certificate. 2IP address values must be specified in standard dotted decimal notation, for example, 245.120.75.22. DNS Name values must be specified as a fully qualified domain name (FQDN), for example, device.packetcable.com.

Table 5-10 PacketCable Server Certificates

PacketCable Server Certificates		
Subject Name Form	PacketCable	Euro-PacketCable
	<p>C=<Country></p> <p>O=<Company></p> <p>OU=PacketCable</p> <p>OU=[<Local System Name>]</p> <p>OU=<Sub-System Name></p> <p>CN=<Server Identifier[:<Element ID>]></p> <p>The value of <Server Identifier> must be the server's FQDN or its IP address, optionally followed by a colon (:) and an Element ID with no white space either before or after the colon.</p> <p><Element ID> is the identifier that appears in billing event messages and it must be included in the certificate of every server that is capable of generating event messages. This includes a CMS, CMTS and MGC. [8] defines the Element ID as an 5-octet right-justified space-padded ASCII-encoded numerical string. When converting the Element ID for use in a certificate, spaces must be converted to ASCII zeroes (0x48).</p> <p>For example, a CMTS that has the Element ID "311" and an IP address 123.210.234.12 will have a common name "123.210.234.12: 00311".</p> <p>The value of <Sub-System Name> must be one of the following:</p> <ul style="list-style-type: none"> • For Border Proxy: bp • For Cable Modem Termination System: cmts • For Call Management Server: cms • For Media Gateway: mg•For Media Gateway Controller: mgc • For Media Player: mp • For Media Player Controller: mpc • For Provisioning Server: ps • For Record Keeping Server: rks • For Signaling Gateway: sg 	<p>C = <Country></p> <p>O = <Company></p> <p>OU = Euro-PacketCable</p> <p>[OU = <Local System Name>]</p> <p>OU = <Sub-system Name></p> <p>CN = <Server Identifier[:<Element ID>]></p> <p>Please refer to [PKT-SP-SEC-IO8-030415] for additional specifications on the commonName.</p>
Intended Usage	These certificates are used to identify various servers in the PacketCable system. For example, they may be used to sign phase 1 IKE exchanges or to authenticate a device in a PKINIT exchange.	
Signed By	Telephony Service Provider Certificate or Local System Certificate	
Validity Period	Set by MSO policy	

Table 5-10 PacketCable Server Certificates (continued)

PacketCable Server Certificates	
Modulus Length	2048
Extensions	keyUsage[c,o](digitalSignaturekeyEncipherment) authorityKeyIdentifier[n,m](keyIdentifier=<subjectKeyIdentifier value from CA cert>) subjectAltName[n,m](dNSName=<DNSName> iPAddress=<IP AddressName>) The keyUsage tag is optional. When it is used it must be marked as critical. Unless otherwise described below, the subjectAltName extension must include the corresponding name value as specified in the CN field of the subject.

The CN attribute value for CMS certificates must be the Element ID. The subjectAltName extension must include either the IP Address or the FDQN of the CMS. The CN attribute value for CMTS certificates must be the Element ID. The subjectAltName extension must include either the IP Address or the FDQN of the CMTS.

The CN attribute value for MGC certificates must be the Element ID. The subjectAltName extension must include either the IP Address or the FDQN of the MGC.

Certificate Revocation

Out of scope for PacketCable at this time.

Code Verification Certificate Hierarchy

The CableLabs Code Verification Certificate (CVC) PKI is generic in nature and applicable to all CableLabs projects needing CVCs. This means the basic infrastructure can be re-used for every CableLabs project. There may be differences in the end-entity certificates required for each project, but in the cases where end-entity certificates overlap, one end-entity certificate could be used to support the overlap.

The CableLabs CVC hierarchy does not apply to E-MTAs. Refer to section 11 for more information.

Common CVC Requirements

The following requirements apply to all Code Verification Certificates:

- Certificates must be DER encoded.
- Certificates must be version 3.
- Certificates must include the extensions that are specified in the following tables and must NOT include any additional extensions.
- The public exponent must be F4 (65537 decimal).

CableLabs Code Verification Root CA Certificate

This certificate must be validated as part of the certificate chain containing the CableLabs Code Verification Root CA Certificate, the CableLabs Code Verification CA, and the Code Verification Certificates. See [“Certificate Validation” section on page 5-15](#) for additional information on how to validate certificates.

Table 5-11 CableLabs Code Verification Root CA Certificates

CableLabs Code Verification Root CA Certificate		
Subject Name Form	PacketCable	Euro-PacketCable
	C=US O=CableLabs CN=CableLabs CVC Root CA	C = BE O = tComLabs CN = tComLabs CVC Root CA
Intended Usage	This certificate is used to sign Code Verification CA Certificates. This certificate must be included in the S-MTAs non-volatile memory at manufacture time.	
Signed By	Self-signed	
Validity Period	20+ years	
Modulus Length	2048	
Extensions	KeyUsage [c,m] (keyCertSign, cRL Sign) subjectkeyidentifier [n,m] basicConstraints [c,m](cA=true)	

CableLabs Code Verification CA Certificate

The CableLabs Code Verification CA Certificate must be validated as part of a certificate chain containing the CableLabs Code Verification Root CA Certificate, CableLabs Code Verification CA Certificate and the Code Verification Certificate. See [“Certificate Validation” section on page 5-15](#) for additional information on how to validate certificates. There may be more than one CableLabs Code Verification CA. A S-MTA must support one CableLabs CVC CA at a time.

Table 5-12 CableLabs Code Verification CA Certificates

CableLabs Code Verification CA Certificate		
Subject Name Form	PacketCable	Euro-PacketCable
	C=US O=CableLabs CN=CableLabs CVC CA	C = BE O = tComLabs CN = tComLabs CVC CA
Intended Usage	This certificate is issued to CableLabs by the CableLabs Code Verification Root CA. This certificate issues Code Verification Certificates. This certificate must be included in the S-MTAs non-volatile memory at manufacture time.	
Signed By	CableLabs Code Verification Root CA	
Validity Period	Set by CableLabs policy	

Table 5-12 CableLabs Code Verification CA Certificates (continued)

CableLabs Code Verification CA Certificate	
Modulus Length	2048
Extensions	KeyUsage [c,m] (keyCertSign, cRL Sign) subjectKeyIdentifier [n,m] authorityKeyIdentifier [n,m] basicConstraints [c,m](cA=true, pathLenConstraint=0)

Manufacturer Code Verification Certificate

The CableLabs Code Verification CA issues this certificate to each authorized Manufacturer. It is used in the policy set by the cable operator for secure software download.

Table 5-13 Manufacturer Code Verification Certificates

Manufacturer Code Verification Certificate		
Subject Name Form	PacketCable C=<country> O=<Company Name> [ST=<state/province>] [L=<city>] CN=<Company Name> Mfg CVC	Euro-PacketCable C = <Country> O = <Company Name> [ST = <state/province>] [L = <City>] CN = <Company Name> Mfg CVC
Intended Usage	The CableLabs Code Verification CA issues this certificate to each authorized Manufacturer. It is used in the policy set by the cable operator for secure software download.	
Signed By	CableLabs Code Verification CA	tComLabs Code Verification CA Certificate
Validity Period	Set by CableLabs policy	
Modulus Length	1024, 1536, 2048	
Extensions	extendedKeyUsage [c,m] (id-kp-codeSigning) authorityKeyIdentifier [n,m]	

The Company Name in the Organization may be different than the Company Name in the Common Name.

Service Provider Code Verification Certificate

The Service Provider Code Verification Certificate must be validated as part of a certificate chain containing the CableLabs Code Verification Root CA Certificate, the CableLabs Code Verification CA Certificate, and the Service Provider Code Verification Certificate. Refer to the [“Certificate Validation” section on page 5-15](#) for additional information on how to validate certificates.

Table 5-14 Service Provider Code Verification Certificates

Service Provider Code Verification Certificate		
Subject Name Form	C=<country> O=<Company Name> [ST=<state/province>] [L=<city>] CN=<Company Name> Service Provider CVC	C = <Country> O = <Company Name> [ST = <state/province>] [L = <City>] CN = <Company Name> Service Provider CVC
Intended Usage	The CableLabs Code Verification CA issues this certificate to each authorized Service Provider. It is used in the policy set by the cable operator for secure software download.	
Signed By	CableLabs Code Verification CA	tComLabs Code Verification CA Certificate
Validity Period	Set by CableLabs policy	
Modulus Length	1024, 1536, 2048	
Extensions	extendedKeyUsage [c,m] (id-kp-codeSigning) authorityKeyIdentifier [n,m]	

The Company Name in the Organization may be different than the Company Name in the Common Name.

Certificate Revocation Lists for CVCs

The S-MTA is not required to support Certificate Revocation Lists (CRLs) for CVCs.



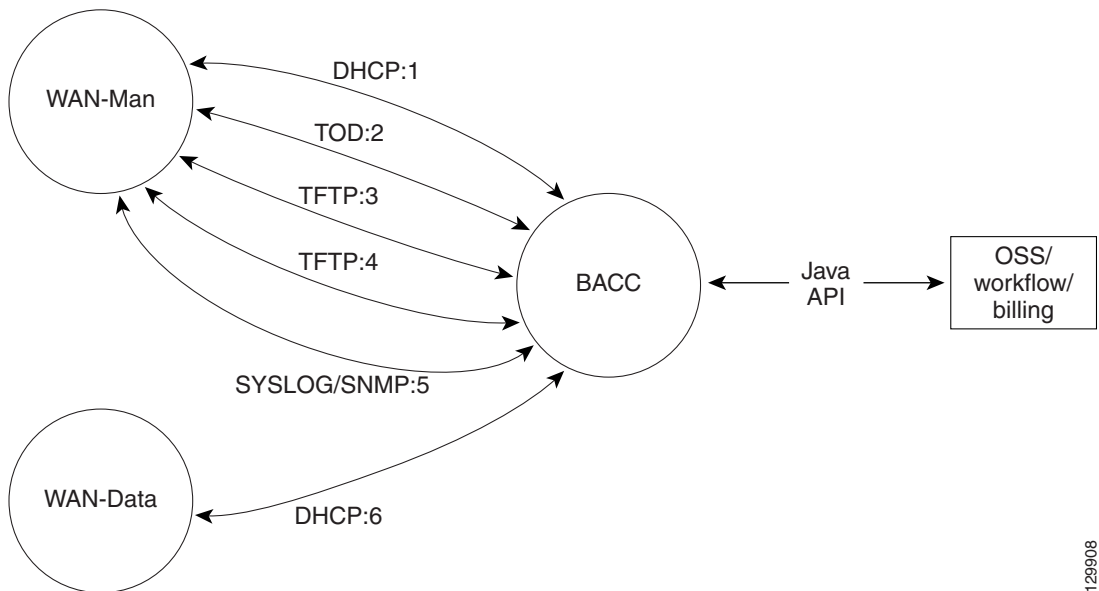
CableHome Configuration

This chapter describes the activities that must be performed to ensure a satisfactory CableHome deployment. There are two versions of the CableHome technology; secure (SNMP) and non-secure (DHCP). This chapter deals exclusively with the non-secure version.

Non-Secure CableHome Provisioning Flow

It is extremely useful to identify which step in the non-secure CableHome provisioning flow is failing before attempting to diagnose other details. [Figure 6-1](#) illustrates a summary of the key provisioning flows and the following steps describe those flows.

Figure 6-1 Non-Secure CableHome Flows



1. DHCP:1—The WAN-Man obtains its IP lease during this step. The IP lease also specifies that the portal service is to be provisioned in the DHCP (non-secure) mode, the TFTP server and the configuration file.
2. TOD:2—Portal services achieve time synchronization with a time of day server in the MSO’s network.

129908

3. TFTP:3—Portal services obtain a configuration file in the manner similar to that described in step 1.
4. TFTP:4—Provided that the configuration file acquired in step 3 contains firewall information, portal services may also acquire a firewall configuration file.
5. SYSLOG/SNMP:5—Once successfully configured. Portal services send either a SYSLOG message or a SNMP PDU to inform BACC that it has been successfully configured.
6. DHCP:6—Provided that the portal services configuration file acquired in step 3 contains a MIB setting that sets the number of WAN-Data IP address to one or more, the portal services WAN-Data obtains the required number of IP addresses.

Configuring CableHome

This section describes how to configure Network Registrar, the CMTS.

Configuring Network Registrar

-
- Step 1** Create selection tags for provisioned and unprovisioned WAN-Man and also for provisioned WAN-Data.
 - Step 2** Configure unprovisioned and provisioned client classes and scopes for cable modems, as specified in *Network Registrar User's Guide*.
 - Step 3** Configure unprovisioned and provisioned client classes and scopes for WAN-Man.
 - Step 4** Configure provisioned client classes and scopes for WAN-Data.
 - Step 5** Add routes to all the subnets.
-

Configuring the RDU

To configure CableHome support into the RDU perform these configurations:

- [Configuring CableHome WAN-Man, page 6-2](#)
- [Configuring CableHome WAN-Data, page 6-3](#)

Configuring CableHome WAN-Man

-
- Step 1** Create a DHCPCriteria for the provisioned WAN-Man. To do this, set the client class to a client-class name that is configured in Network Registrar CableHome WAN-Man.
 - Step 2** Create a class of service for the provisioned WAN-Man.
 - Set the `/cos/chWanMan/file` to a CableHome configuration file appropriate for the class of service.
 - Set the `/chWanMan/firewall/file` to the desired Firewall configuration file.
-

Configuring CableHome WAN-Data

Configure these WAN-Data parameters whenever you want portal services to obtain the WAN-Data IP addresses:

-
- Step 1** Create DHCPCriteria for WAN-Data.
 - Step 2** Create ClassOfService for WAN-Data.
-

Configuring the Device Provisioning Engine

To configure the DPE to support the CableHome technology:

-
- Step 1** Open the CableHome device provisioning WAN-Man config file and verify that DHCP Option 60 is set to either CableHome1.0 or CableHome1.1. Some manufacturers use a proprietary MIB object to instruct a device to behave as either a pure cable modem, a non-CableHome router, or as a CableHome router. The device appears as a Computer whenever the device DHCP packet does not contain CableHome1.0 or CableHome1.1 in the DHCP Option 60.
 - Step 2** If you want the portal services to obtain IP addresses for WAN-Data:
 - Ensure that the WAN-Man configuration file contains TLV 28 that sets cabhCdpWanDataIpAddrCount to a value that is greater than 0.
 - In the cable modem configuration file, set the max number of CPEs to include the number of WAN-Data IP addresses.
 - Step 3** To enable self-provisioning when the CableHome device boots:
 - In the unprov-wan-man.cfg portal services configuration file, set the portal services in the passthrough mode
 - In the cable modem configuration file, set the maximum number of CPEs to at least 2 to allow provisioning of the WAN-Man and a computer. The computer can directly access sign-up web pages to be self-provisioned.
 - Step 4** Close and save the CableHome configuration file.
-



Database Management

This chapter contains information on RDU database management and maintenance. The RDU database is the Broadband Access Center for Cable (BACC) central database. As with any database, it is essential that you, as the administrator, understand and be familiar with database backup and recovery procedures.

Understanding Failure Resiliency

The RDU database uses a technique known as *write-ahead logging* to protect against database corruption that could result from unforeseen problems, such as an application crash, system failure, or power outage.

Write-ahead logging involves writing a description of any database change to a log file prior to writing the change into the database files. This allows the repair of any incomplete database writes that can result from system failures.

The RDU server performs an automatic recovery each time it is started. During this recovery process, the log files are used to synchronize the data with the database files. Database changes that were written into the log, but not into the database, are written into the database during this automatic recovery.

In this way, write-ahead logging virtually guarantees that the database does not become corrupted when the RDU server crashes because the database is automatically repaired when the RDU server is restarted.

Write-ahead logging requires these conditions to work properly:

- The file system and physical storage must be set up so that they guarantee that the data is flushed to physical storage when requested. For example, a storage system with RAM-only write cache, which loses data during system failure, is not appropriate. However, a disk array with battery-backup write cache that guarantees that the data gets persisted, even in the event of a system failure, is acceptable.
- The file system must be set up with 8192 byte block size to match the RDU database block size. This is usually the default setting on Solaris unless explicitly adjusted.

Database Files

The RDU database stores data in binary files using the file system you have mounted on the partition containing the files. It is essential to choose and configure a file system so that it is not susceptible to long recovery times after system failures.

Database files are vital to the operation of the RDU. Therefore, take extra precaution measures to safeguard them against accidental removal or other manual manipulation. Follow standard system administration practices to safeguard these important files. For example, these files should always have

permissions that allow only root user access. Additionally, it is a good practice to never log into your production system as a root user, but rather log in as a less privileged user and use the sudo command to execute tasks requiring root privileges.

**Note**

Use the backup, restore, and recovery tools to perform corresponding tasks on the database. Previous Broadband Provisioning Registrar (BPR) releases allowed you to perform some of these tasks directly without using specific tools. Do not attempt to perform these tasks without using the tools. See the [“Backup and Recovery” section on page 7-4](#) for additional information on these tools.

Database Storage File

The RDU server stores its database in a file called bpr.db, which is found in the database directory. This directory is located in the <BACC_DATA>/rdu/db directory and is configured by specifying the BACC_DATA parameter during a component installation. See the [“Changing Database Location” section on page 7-6](#) for additional information on moving the database.

**Note**

The database file is normally accessed in a random fashion. You should therefore, select a disk with the fastest seek time and rotational access latency to obtain the best database performance.

Database Transaction Log Files

The RDU server stores database transaction logs in 10 MB files that are stored in the database log directory. You configure this directory during installation by specifying the BACC_DBLOG parameter. The log directory is located in the <BACC_DBLOG>/rdu/dblog directory. See the [“Changing Database Location” section on page 7-6](#) for additional information on moving the transaction logs to a new directory.

Database log files are named in numeric sequence, starting at log.00000001, then log.00000002, and so on.

**Note**

The disk on which transaction logs are stored is usually accessed in a sequential manner, with data being appended to the log files. Select a disk that can efficiently handle this access pattern to achieve the best database performance.

Automatic Log Management

Database transaction logs files are used to store transaction data until that data is completely written into the database. After that, the transaction log data becomes redundant and, because the RDU server database manages transaction log files automatically, the files are then automatically removed from the system. Under normal circumstances there should be only a few log files in the database transaction log directory. Over time, you will notice that older transaction logs disappear and newer ones are created.

**Note**

Database transaction logs are an integral part of the database. Manual deletion of transaction log files may result in database corruption.

Miscellaneous Database Files

The database directory contains additional files that are essential to database operation. These files, in addition to the bpr.db file, are found in the <BACC_DATA>/rdu/db directory and are copied as part of the database backup:

- DB_VERSION—Identifies the physical and logical version of the database and is used internally by the RDU.
- history.log—Used to log activity about essential database management tasks, such as automatic log file deletion, backup, recovery, and restore operations. In addition to providing useful historical information for the administrator, this log file is essential to RDU database operation.

Disk Space Requirements

The size of a fully populated database depends on a number of factors. The main factors are the number of device objects managed by the RDU and the number of custom properties stored on each object. The approximate estimates for disk space required on each partition are:

- BACC_DATA, approximately 3 to 5 KB per device object
- BACC_DBLOG, at least 500 MB

**Caution**

These numbers are provided as a guideline only and do not eliminate the need for normal system monitoring.

You can use the `disk_monitor.sh` command to monitor available disk space and alert the administrator. See the [“Using the disk_monitor.sh Tool to Monitor Available Disk Space”](#) section on page 12-51 for additional information.

Handling Out of Disk Space Conditions

When the RDU server runs out of disk space, it generates an alert through the syslog facility and the RDU log. The RDU server then tries to restart automatically. When attempting to restart, the RDU server may again encounter the out of disk space error and attempt to restart again.

The RDU server continues trying to restart until free disk space becomes available. Once you free up some disk space on the disk that is running into a limit, the next time the RDU restarts it will succeed.

If your database grows in size beyond the capacity of the current disk partition, then you should move the database to a new disk/partition. For information on how to do this, see the [“Changing Database Location”](#) section on page 7-6.

**Note**

It is a good practice, to monitor disk space utilization to prevent failure. See the [“Using the disk_monitor.sh Tool to Monitor Available Disk Space”](#) section on page 12-51 for additional information.

Backup and Recovery

The RDU server supports a highly efficient backup process that can be performed without stopping the server or suspending any of its activities. Database backup and recovery involves these stages:

- Backup—Takes a snapshot of the RDU database from a live server.
- Recovery—Prepares the database snapshot for re-use.
- Restore—Copies the recovered database snapshot to the RDU server.

Automated tools are provided for each of these steps. You can use these tools in either interactive mode or silent mode, but you must have root privileges to use the tools.

Database Backup

Backup is the process of copying the database files into a backup directory. The files can then be compressed and placed on tape or other archive.

RDU database backup is highly efficient because it involves just copying files without interrupting server activity. However, because it involves accessing the RDU database disk, backup may adversely affect RDU performance. The opposite is also true. RDU activity happening during backup will adversely affect backup performance. Therefore, you should perform backups during off-peak hours.

Other than concurrent system activity, backup performance also depends on the underlying disk and file system performance. Essentially, backup will perform as fast as database files can be copied from source to target.

Use the **backupDb.sh** command, in the <BACC_DATA>/rdu/bin directory, to perform database backups:

- To use this command, you must provide the target directory where the backup files will be placed. This directory should be on a disk or partition that has available disk space equivalent to 120% of the current database file size.
- As illustrated in the following example, this command automatically creates a time stamped subdirectory, under the directory you specify and places the backups there.

Examples

Here is an example of using the backupDb.sh command:

```
backupDb.sh /var/backup
```

Where:

- */var/backup*—identifies the database backup directory.

In this example, all backup database files are stored in a directory called */var/backup/rdu-backup-09252002-130345*. The last subdirectory (*rdu-backup-09252002-130345*) is automatically created with a current timestamp.



Note

The timestamped subdirectory format used is *rdu-backup-MMddyyyy-HHmms*. In this example, the subdirectory would be *rdu-backup-04272003-175430* meaning that the directory contains a backup that was started at 5:54:30 pm on April 27, 2003.

The backupDb.sh command also reports progress to the screen and logs its activity into *history.log*.

**Note**

When using the `backupDb.sh` command, you can use a `-help` option to obtain usage information. You can also use another optional flag called `-nosubdir` to disable, if necessary, the automatic creation of the subdirectory.

Database Recovery

Database recovery is the process of restoring the database to a consistent state. Since backup is performed on a live RDU, the database can be changing while it is being copied. The database log files however, ensure that the database can be recovered to a consistent state.

Recovery is performed on a snapshot of a database. In other words, this task does not involve touching the database on the live RDU server. The recovery task can be performed either immediately following a backup or prior to restoring the database to the RDU server.

**Note**

Cisco recommends that you perform database recovery immediately after each backup. This way, the backed up database can be more quickly restored in case of emergency.

The duration of database recovery depends on the number of database log files that were copied as part of the backup, which in turn depends on the level of RDU activity at the time of the backup. The more concurrent activity RDU experiences during the backup, the more transaction log files have to be copied as part of the backup and the longer is the recovery. Generally, recovering a database takes from 10 to 60 seconds per transaction log file.

Use the `recoverDb.sh` command, located in the `<BACC_DATA>/rdu/bin` directory, to perform recovery of the snapshot of a database. When you use this command, you must provide the location of the backup. This is also the directory in which the recovery will be performed.

Examples

Here is an example of using the `recoverDb.sh` command:

```
# recoverDb.sh /var/backup/rdu-backup-04272003-130345
```

In this example, the snapshot located in the `/var/backup/rdu-backup-04272003-130345` directory will be recovered to a consistent state. The progress of the recovery operation will appear on screen and the activity will be recorded in the `history.log` file in the snapshot directory.

**Note**

When using the `recoverDb.sh` command, you can use a `-help` option to obtain usage information on the command.

Database Restore

Restoring the database is the process of copying the previously recovered database snapshot to the database location used by the RDU server. Only a database that has been previously recovered can be restored.

Since restoring the database means replacing the current RDU database, it is very important that you first properly remove and archive the old database.

**Note**

Never delete the database you are replacing. You might need a copy of an old database to simplify future system diagnostics.

Use the **restoreDb.sh** command located in the <BACC_DATA>/rdu/bin directory, to replace the current RDU database with another database. When using this command, you must specify an input directory. This directory must contain the recovered backup snapshot of the database to be restored to the RDU server.

**Note**

Before running the restoreDb.sh utility, you must stop the RDU server.

Examples

Here is an example of running restoreDb.sh utility:

```
# restoreDb.sh /var/backup/rdu-backup-04272003-130345
```

In this example, the database found in the /var/backup/rdu-backup-04272003-130345 directory is restored to the RDU server.

**Note**

When using the restoreDb.sh command, you can use a **-help** option to obtain usage information on the command.

You must restart the RDU after the restore operation is completed. The RDU log file will contain successful start up messages.

This command displays progress on screen and logs its activity into the history.log file.

Changing Database Location

You can move the database from one partition or disk to another on the same system. You might sometimes want to do this for administrative reasons. This process requires stopping the RDU server and the BPR agent.

The process of changing the database location involves changing one or two system parameters and copying the appropriate files to the new location. You can adjust one or both of the following parameters:

- **BACC_DATA**—This parameter is initially set during installation and points to a directory that is used to store the database, and many other important files, such as logs, configuration files, and so on.
- **BACC_DBLOG**—This parameter is initially set during installation and points to the directory that stores database transaction log files.

The values for the above parameters are recorded in a file called <BACC_DATA>/bpr_definitions.sh. Any change to this file requires that you restart all BACC components running on the system.

To change the location of the database and transaction logs:

-
- Step 1** Run the **/etc/init.d/bprAgent stop** command to stop the BACC agent.
 - Step 2** Make a backup copy of <BACC_DATA>/bpr_definitions.sh file.

- Step 3** Edit the file and change BACC_DATA and/or BACC_DBLOG parameters to new directories.
 - Step 4** Save the file.
 - Step 5** Copy or move the directory structure and contents of the original BACC_DATA and/or BACC_DBLOG directories to new location(s). If you make a copy, make sure that all file and directory permissions are preserved.
 - Step 6** Run the `/etc/init.d/bprAgent start` command to start the BACC agent and all BACC components.
 - Step 7** Monitor the appropriate log files to ensure that all components have successfully started.
-

RDU Database Migration

All information concerning the migration of the RDU database is provided in the Cisco Broadband Access Center for Cable Installation Guide.



Understanding the Administrator's User Interface

This chapter describes how to access the Broadband Access Center for Cable (BACC) administrative user interface and explains important aspects about it.

[Chapter 9, "Using the Broadband Access Center for Cable Administrator User Interface"](#) explains how to use the BACC administrator user interface to perform administrative activities. In addition, the Cisco Broadband Access Center for Cable Command Line Interface Reference contains descriptions of the CLI commands used to access, monitor, and control the device provisioning engine (DPE) devices.

Accessing the BACC Administrators Graphical User Interface

You can access the BACC user interface from any computer that can access the URL corresponding to the BACC application.

Logging In

You can log into the BACC user interface as an administrative user, a Read/Write user, or a Read Only user. Although each user type has different capabilities, as described in the ["Users" section on page 8-6](#), you access the user interface in the same way.

Complete this procedure to access the BACC administrator user interface:

- Step 1** Launch your web browser. [Table 8-1](#) lists the browsers supported in this BACC release.

Table 8-1 Browser Platform Support

Platform	Supported Browsers
Windows 2000 (Service pack 2)	Internet Explorer 6.0 and above Netscape 4.7 and above
Red Hat Linux (7.1)	Netscape 4.7 and above
Solaris (2.9)	Netscape 4.7 and above

- Step 2** Enter the administrator's location using this syntax:

```
http://<machine_name>:8100/adminui/
```

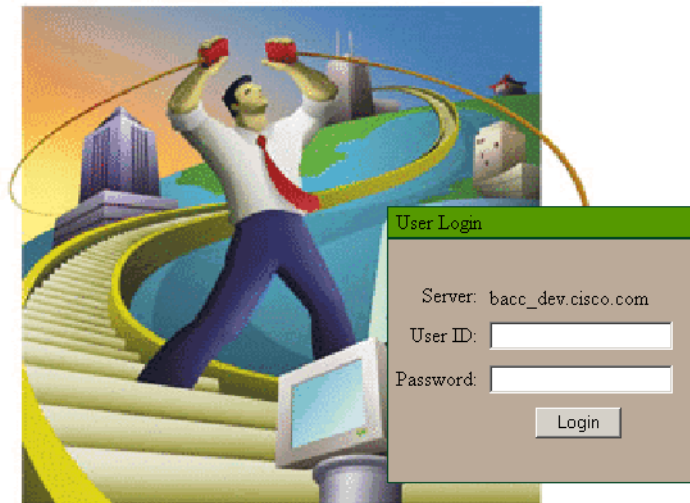
Where:

- `<machine_name>`—identifies the computer on which the regional distribution unit (RDU) is running.
- `8100`—identifies the computer port on which the server-side of the administrator application is running. This port number is fixed at 8100.

Step 3 The main login page, shown in [Figure 8-1](#), appears.

Figure 8-1 Login Page

Broadband Access Center for Cable



This product contains cryptographic features and is subject to United States and local country laws governing import, export, transfer and use. Delivery of Cisco cryptographic products does not imply third-party authority to import, export, distribute or use encryption. Importers, exporters, distributors and users are responsible for compliance with U.S. and local country laws. By using this product you agree to comply with applicable laws and regulations. If you are unable to comply with U.S. and local laws, return this product immediately. A summary of U.S. laws governing Cisco cryptographic products may be found at: <http://www.cisco.com/wll/export/crypto/tool/stqrg.html>. If you require further assistance please contact us by sending email to export@cisco.com.

129898

Step 4 Enter the default username (**admin**) and password (**changeme**).

Note that the FQDN of the RDU server you are logged into is displayed in the User Login area.

Step 5 If logging in for the first time:

- A prompt appears asking you change the default password. Enter a new password and click **OK**.
- A prompt appears with a link to the Add License Keys page. This page lets you enter the technology licenses that you are authorized to use.



Note Refer to [Chapter 10, “Configuring Broadband Access Center for Cable,”](#) for additional information on entering license keys.

Step 6 Click **Login** and the Main Menu page, shown in [Figure 8-2](#), appears.

Figure 8-2 Main Menu Page

Broadband Access Center for Cable Logout

User:admin Role:Administrator

Main Menu

[Configuration](#)
Use this page to view or change Broadband Access Center for Cable configuration settings.

[Devices](#)
Use this page to manage (add, delete or search) IP devices.

[Nodes](#)
Use this page to manage Nodes and Node types.

[Servers](#)
Use this page to view the status of the Broadband Access Center for Cable servers.

[Users](#)
Use this page to manage users.

CISCO SYSTEMS

129899

Logging Out

Complete this procedure to log out of BACC:

- Step 1** Click **Logout** located at the top right corner of any administrator page. A confirmation dialog appears.
- Step 2** Click **OK**. This returns you to the User Login page (see [Figure 8-1](#)).

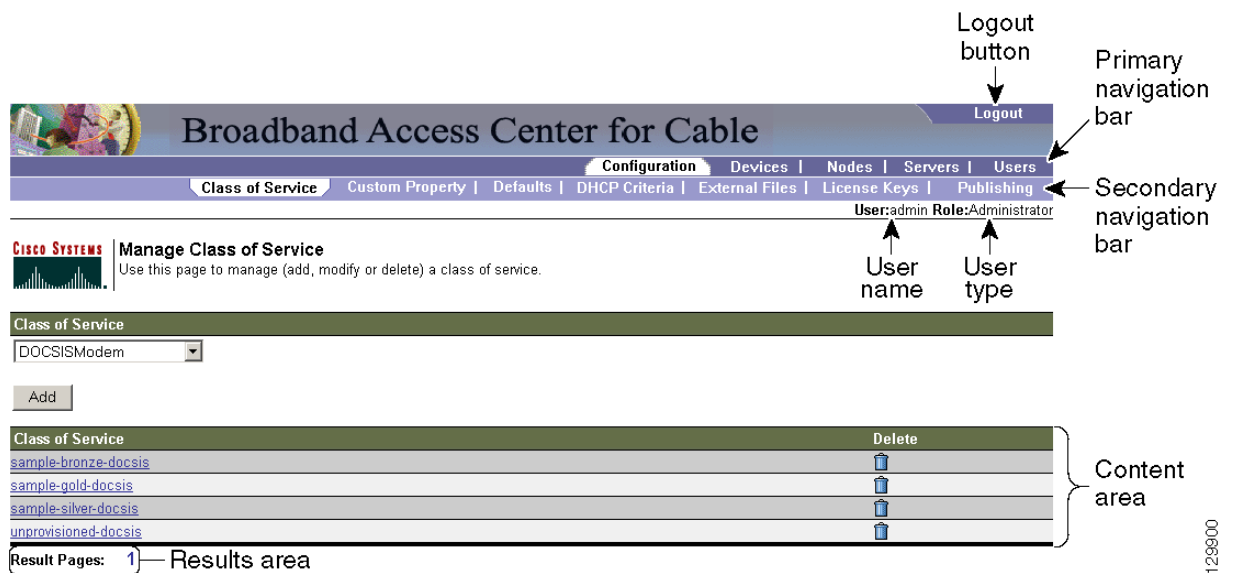
Administrator's User Interface

The BACC administrator user interface, as shown in [Figure 8-3](#), is divided into three separate areas:

- Banner area—Contains all menu options including: a **Logout** button, the Primary Navigation bar, and the Secondary Navigation bar.
- Content area—Contains all BACC data resulting from the functions performed from the Primary and Secondary Navigation bars.
- Results area—Indicates how many pages of information are available. This is particularly useful when reviewing the results of searches.

Some BACC administrator pages may also contain View, Delete, Submit, or Reset controls.

Figure 8-3 Administrator User Interface



129900

Selecting Navigation Bar Items

To select a Navigation bar item, click the desired Primary Navigation bar item, and then click the appropriate Secondary Navigation bar item. (See [Figure 8-3](#).) Note that the Users option has no secondary options to select from.

The Main Menu

You can select from these options on the Main menu:

- Configuration
- Devices
- Servers
- Nodes
- Users

After selecting one of these options, you can make further menu selections by clicking the desired menu item in the Primary Navigation bar. In addition, each Primary Navigation bar menu item displays unique options in the Secondary Navigation bar.

Configuration

Use the Configuration menu to view and modify various configuration settings for the overall system. These include:

- Class of service
- Custom properties
- Defaults (including technology defaults)
- DHCP criteria
- External files
- License keys
- Publishing

Devices

Use the Devices menu to manage and maintain devices, and perform searches for any device using these search formats:

- Class of service
- DHCP criteria
- Fully qualified domain name (FQDN)
- IP address
- MAC address
- Owner ID

Servers

Use the Server menu to list the current status of these items or for any selected component:

- DPEs—Device Provisioning Engine
- NRs—Network Registrar servers
- Provisioning Groups
- RDU—Regional Distribution Unit

Nodes

The Nodes menu lets you group devices into specific categories that can be searched and documented using log entries. that then be searched on and very detailed log entries made. BACC supports pre-defined Nodes that can be used to group devices. You can also create your own nodes as required.

Users

Depending on your user type, you can use this menu to add, modify, and delete users. This menu displays all users configured to use BACC and identifies their user types.

There are three types of BACC users: an Administrator, a Read/Write user, and a Read Only user. Each has different levels of access with unique permissions to ensure access control and the integrity of provisioning data.

The assigned user type appears near the top right corner of every administrative user interface screen.

Administrator

BACC recognizes only one administrator and allows this user to view, add, change, delete device data, and create other users. As an Administrator, you can also change other users permissions from Read/Write to Read Only, and vice versa. In addition, you have the ability to change the passwords of any other user type.

Read/Write User

As a Read/Write user, you can perform the same functions as the administrator although you cannot create other users nor can you change the user types of others or change their passwords. Read/Write users can change their own password.

Read Only User

As a Read Only user, you have basic access including the ability to change your password and to view, but not change, device data. You cannot perform any action that is considered disruptive. You cannot, for example, perform reset or regenerate configurations.

**Note**

During migration from an acceptable previous release to BACC 2.7, all migrated users are assigned Read/Write privileges.

Scrolling Backward and Forward

Several functions within BACC administrator can result in lengthy lists of information that cannot always be displayed on a single page. BACC allows you to view these pages one at a time through a group of controls located in the lower left corner of each page. When multiple pages of information are available, the number of pages are also identified. To access a specific page, click the appropriate page number. Click the left and right arrow controls to move backward or forward respectively, through the number of displayed pages.



Using the Broadband Access Center for Cable Administrator User Interface

This chapter describes the administration activities performed using the Broadband Access Center for Cable (BACC) administrator user interface. These activities mainly involve monitoring the actions of various BACC components including:

- [User Management, page 9-1](#)
- [Device Management, page 9-3](#)
- [Node Management, page 9-16](#)
- [Viewing Servers, page 9-18](#)



Note

The procedures described in this chapter are presented in a tutorial manner. Wherever possible, examples are included to illustrate the possible results of each procedure.

User Management

Managing users involves adding, modifying, and deleting users who administer BACC. This section contains instructions for managing BACC users including:

- [Adding a New User](#)
- [Modifying Users](#)
- [Deleting Users](#)



Note

You can add and delete users only if you are logged in as the Administrator. See the [“Users” section on page 8-6](#) for additional information on user types.

Adding a New User

Adding a new user is a simple process of entering the user's name and creating a password. However, while creating a new user you do have to determine which type of user it will be; a Read/Write user or a Read Only user. BACC comes with one Administrator user already created; you cannot create an Administrator as a new user.

To add a new user:

- Step 1** Click **Users**, from either the Main Menu or the Primary Navigation bar. The Manage Users page appears. (See [Figure 9-1](#).)

Figure 9-1 Example Manage Users Page

User	Description	Role	Delete
Ace Duffy	Assistant Administrator	Read Write	
admin	Administrator	Administrator	

Result Pages: 1

129901

- Step 2** Click **Add** to display the Add User page.
- Step 3** Enter the new user's username and a password.
- Step 4** Confirm the new user's password, and select whether the new user's role is to be read only or read/write. See the "[Users](#)" section on page 8-6 for complete descriptions of each user type.
- Step 5** Enter a short description of the new user.



Tip Use the description field to identify the user's job or position; something that identifies the unique aspects of the new user.

- Step 6** Click **Submit** when complete. The Manage Users page appears with the new user added.



Note The new user's password must be recorded and stored in a safe place. This helps to prevent the loss or theft of the password and possible unauthorized entry.

Modifying Users

Although any user type can modify their own password and user description, only the administrator can modify any other user's information. You can change the password, user type, and description.



Note Any nonadministrative user that is created cannot be used to modify or delete the admin user.

To modify user properties:

- Step 1** From either the Main Menu or the Primary Navigation bar, click **Users**. The Manage User page appears.
- Step 2** Click the desired user name to display the Modify User page for that user.
- Step 3** Make the necessary changes to the password, user type (provided that you are logged in as the Administrator), and the user's description.
- Step 4** Click **Submit**. The Manage Users page appears with the appropriately modified user information.

Deleting Users

Only the administrator has the ability to delete any other user that appears in the Manager Users page. To delete a user:

- Step 1** From either the Main menu or the Primary Navigation bar, click **Users**. The Manage User page appears.



Note The default user called **admin** cannot be deleted.

- Step 2** Click the **Delete** icon corresponding to the user you want to delete. the Delete User dialog box appears.
- Step 3** Click **OK** to delete the selected user. The Manage Users page appears without the deleted user.

Device Management

Use the Devices menu to provision and manage various devices. You can:

- Search for a specific device or for a group of devices that share criteria that you specify. See the [“Searching for Devices” section on page 9-6](#).
- Add, modify, or delete devices in the RDU database. See the:
 - [“Adding Devices” section on page 9-13](#)
 - [“Modifying Devices” section on page 9-13](#)
 - [“Deleting Devices” section on page 9-14](#)
- Regenerate device configurations. See the [“Regenerating Device Configurations” section on page 9-14](#).

- Relate and unrelate any device to a specific node. See the “[Relating and Unrelating Devices](#)” section on page 9-15.
- Reset, or reboot, any selected device. See the “[Resetting Devices](#)” section on page 9-15.
- Return a device configuration to its default condition without rebooting the device. See the “[Unregistering a Device](#)” section on page 9-15.

Manage Devices Page

The Manage Devices page appears whenever you click **Devices** in either the Main menu or the primary menu bar. This page, shown in [Figure 9-2](#), contains the fields and controls necessary to perform all device management functions.

Figure 9-2 Manage Devices Page

Broadband Access Center for Cable Logout

Configuration **Devices** Nodes | Servers | Users

User:admin Role:Administrator

Manage Devices
Use this page to manage (add, delete or search) devices, then to view the details of the device listed.

Search Type	MAC Address or MAC Address wildcard	Page Size	
MAC Address Search	*	25	Search

Add Delete Regenerate Relate Reset Unrelate Unregister

	Identifier	Device Type	Status	Details
<input type="checkbox"/>	1.6.00:00:00:00:00:00	DOCSISModem	Unprovisioned	Details
<input type="checkbox"/>	1.6.00:00:00:00:00:00.01	DOCSISModem	Unprovisioned	Details
<input type="checkbox"/>	1.6.00:00:00:00:00:00.aa	DOCSISModem	Unprovisioned	Details

Result Pages: 1

129902

The Manage Device page utilizes two separate but related areas to generate search results that let you perform many device management functions. These areas are the Search Type drop-down list, defining which search to perform, and search value field, which qualifies the search type. You can perform these searches:

- Class of Service search
- DHCP Criteria search
- FQDN search
- IP Address search
- MAC Address search
- Node search
- Owner ID search

Some searches that you can perform allow the use of a wildcard character (*) to enhance the search function. Using the FQDN as an example, you use these formats when using wildcard characters to search:

- modem10.cisco.com
- *.cisco.com
- *.com
- *

**Note**

Cisco does not recommend using the last wildcard search (*) in systems that support hundreds of thousands, or more, devices. This can return many thousands of search results, and use extensive system resources sufficient to impact performance.

In addition, a Page Size drop-down lets you limit the number of search results displayed per page. You can select 25, 50, or 75 results for display. If the number of results returned for a search exceeds the number selected, a screen prompt appears at the bottom left corner of the administrators user interface to let you move from page to page of results.

Device Management Controls

These buttons are located directly below the search function fields and are generally used in conjunction with the search function. For example, you might search for devices belonging to a specific group of devices in order to perform some sort of management function. The following buttons are available, although each management function may not be available depending on the search type used.

Add

The Add function lets you add a new device to the RDU database. See the [“Adding Devices” section on page 9-13](#) for the appropriate instructions.

Delete

The Delete button lets you delete any selected device from the RDU database. See the [“Deleting Devices” section on page 9-14](#) for the appropriate instructions.

Regenerate

Use the Regenerate function to update stale configurations. Device configurations can become stale when changes are made to either the class of service or DHCP criteria parameters. Device configurations are automatically regenerated whenever class of service or DHCP properties are changed or new criteria are designated. They can also occur when an external file is replaced. However, some configurations cannot be automatically regenerated, but must rely on manual regeneration using either the generationConfiguration() method or the administrator’s user interface.

Relate

The Relate function lets you associate a device (using its MAC address) with a specific node.

Reset

The Reset button automatically reboots the selected device.

Unrelate

This function cancels the relationship between and selected device and the node that the device is currently related to.

Unregister

The Unregister function lets you reset a device back to its defaults as if it had just booted on the network.

Identifier

Identifies all devices matching the search criteria. Each of the identifiers displayed has a link to another page from which you can modify the device.

Device Type

Displays a drop-down list that identifies the available device types. Available selections, as they appear on screen, include:

- ATA186
- ATA188
- CableHomeManData
- CableHomeManWan
- DOCSISmodem
- Computer
- PacketCableMTA

Status

Identifies whether or not the device is provisioned. A provisioned device has been registered using the application programming interface (API), or the administrative user interface, and has booted on the network.

Details

Displays all available details for the selected device. See the [“Viewing Device Details” section on page 9-8](#) for additional information.

Searching for Devices

You can search for device information in a number of different ways using BACC. Each search result that you generate also carries with it a View Details function. The details that are displayed are identical whichever search method you use.

To search for a device type, from the Manage Devices page, click the **Search Type** button and a drop-down list appears. Subsequent search pages contain screen components that may be unique to the search type selected.

When the number of search results is greater than the selected page size, paging controls appear in the lower left corner of the page. These let you scroll forward or backward one page at a time, or to select a specific page. Refer to the [“Scrolling Backward and Forward” section on page 8-7](#) for additional information.

**Note**

A maximum of 1000 results are returned for any query with a maximum of 75 results displayed per page. You can change the default maximum by modifying the `/adminui/maxReturned` property, in `<BACC_HOME>/rdu/conf/adminui.properties` file, and then running the **bprAgent restart jrun** command (located in the `/etc/init.d/` directory) to restart the BACC JRun component.

Search Types

You can search for specific devices using these functions:

- Registered Class of Service search—Searches using the class of service that a device has been provisioned with.
- Selected Class of Service search—Searches using the class of service selected by the RDU for a device that, for one reason or another, cannot retain its registered Class of Service.
- Related Class of Service search—Searches using both the registered and selected class of service.
- Registered DHCP Criteria search—Searches for devices that belong to certain DHCP criteria.
- Selected DHCP Criteria search—Searches using the DHCP criteria selected by the RDU for a device that, for one reason or another, cannot retain its registered DHCP Criteria.
- Related DHCP Criteria search—Searches using both the registered and selected DHCP criteria.

**Note**

Under normal circumstances the Related/ Selected Class of Service and Related/Selected DHCP Criteria should be identical. If they are not you should investigate the reason and modify the Selected Class of Service/DHCP Criteria to match the Related Class of Service/DHCP Criteria.

- Fully qualified domain name (FQDN) search—The fully qualified domain name (FQDN) search is useful when searching for devices that are identified through the FQDN assigned by the DNS Server, especially when the device MAC address is unknown.

For example, **www.cisco.com** is a fully qualified domain name. Where **www** identifies the host, **cisco** identifies the second level domain, and **.com** identifies the third level domain.

- IP address search—The IP Address Search returns all devices on the network that currently have the specified DHCP leased IP address.
- MAC address search— The MAC address search function is best used when you know the precise MAC address for a specific modem or when all devices with a specific vendor-prefix unambiguously identify the equipment vendor. Therefore, if you perform a MAC address search, you can identify, by the MAC address, the manufacturer and type of device. See the [“Troubleshooting Devices by MAC Address” section on page 12-52](#) for information on the effective use of this search criteria.

**Note**

The vendor-prefix is the first 3 octets of the MAC address. For example, for MAC address `1,6,aa:bb:cc:dd:ee:ff`, the vendor-prefix is `“aa:bb:cc”`.

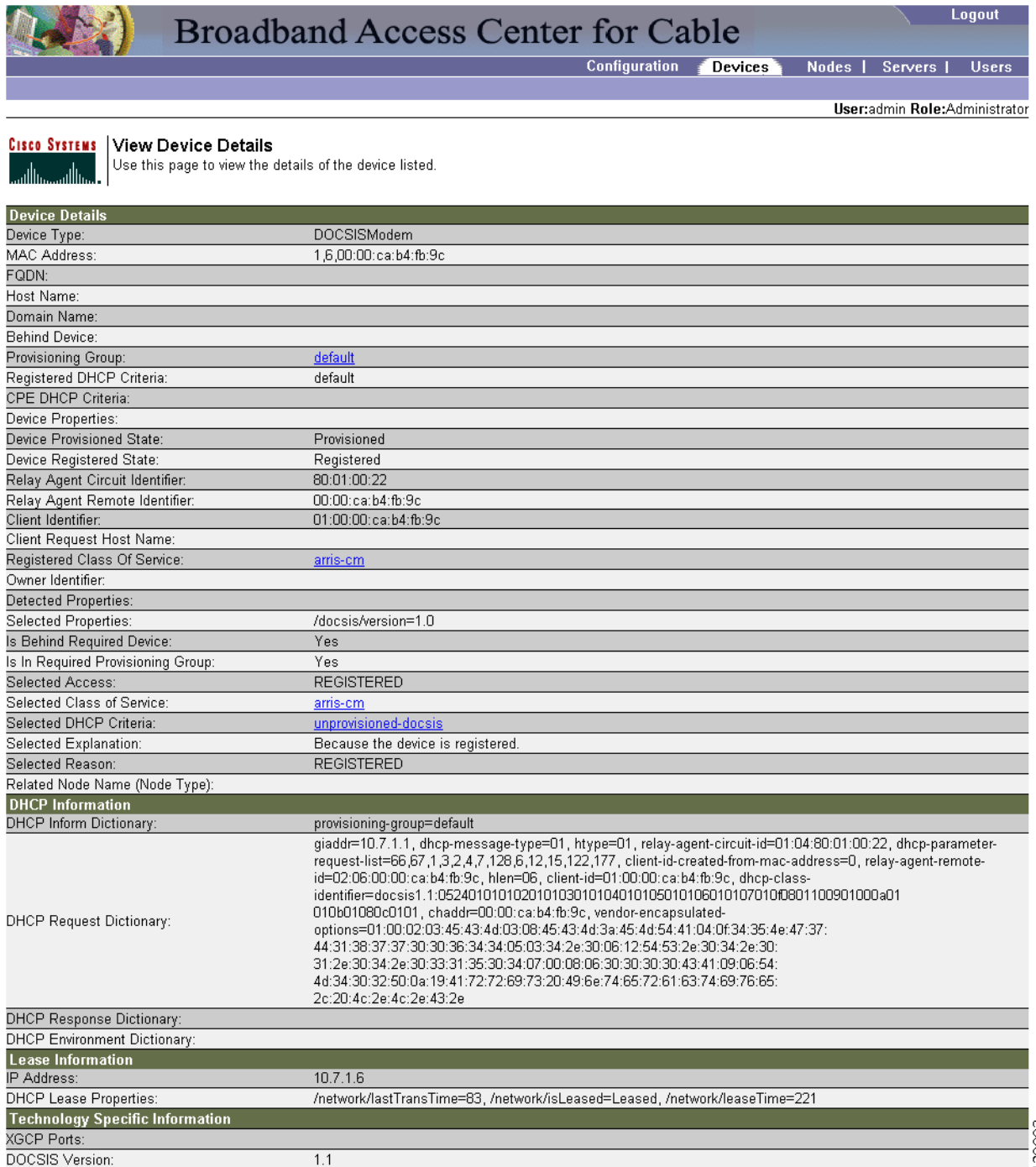
- Node search—The Node search function is used to search for nodes and node types that you have already created.
- Owner ID search— The owner ID can identify a device, it could identify the service subscriber’s account number, or anything else that uniquely identifies that device. Wildcard searching is not supported by this search function.

Viewing Device Details

You can view the details of any device identified in the search results. To view any device details, click the Details icon corresponding to the device you want to view and a View Device Details page appears. [Figure 9-3](#) provides an example View Device Details page.

The information that appears in the View Device Details page is largely dependent on the type of device you choose. However, [Figure 9-3](#) does identify the typical detail displayed for most devices.

Figure 9-3 Device Details Page



Broadband Access Center for Cable Logout

Configuration **Devices** Nodes | Servers | Users

User:admin Role:Administrator

CISCO SYSTEMS | **View Device Details**
Use this page to view the details of the device listed.

Device Details	
Device Type:	DOCSISModem
MAC Address:	1,6,00:00:ca:b4:fb:9c
FQDN:	
Host Name:	
Domain Name:	
Behind Device:	
Provisioning Group:	default
Registered DHCP Criteria:	default
CPE DHCP Criteria:	
Device Properties:	
Device Provisioned State:	Provisioned
Device Registered State:	Registered
Relay Agent Circuit Identifier:	80:01:00:22
Relay Agent Remote Identifier:	00:00:ca:b4:fb:9c
Client Identifier:	01:00:00:ca:b4:fb:9c
Client Request Host Name:	
Registered Class Of Service:	arris-cm
Owner Identifier:	
Detected Properties:	
Selected Properties:	/docsis/version=1.0
Is Behind Required Device:	Yes
Is In Required Provisioning Group:	Yes
Selected Access:	REGISTERED
Selected Class of Service:	arris-cm
Selected DHCP Criteria:	unprovisioned-docsis
Selected Explanation:	Because the device is registered.
Selected Reason:	REGISTERED
Related Node Name (Node Type):	
DHCP Information	
DHCP Inform Dictionary:	provisioning-group=default
DHCP Request Dictionary:	giaddr=10.7.1.1, dhcp-message-type=01, htype=01, relay-agent-circuit-id=01:04:80:01:00:22, dhcp-parameter-request-list=66,67,1,3,2,4,7,128,6,12,15,122,177, client-id-created-from-mac-address=0, relay-agent-remote-id=02:06:00:00:ca:b4:fb:9c, hlen=06, client-id=01:00:00:ca:b4:fb:9c, dhcp-class-identifier=docsis1.1:052401010201010301010401010501010601010701010801100901000a01010b01080c0101, chaddr=00:00:ca:b4:fb:9c, vendor-encapsulated-options=01:00:02:03:45:43:4d:03:08:45:43:4d:3a:45:4d:54:41:04:0f:34:35:4e:47:37:44:31:38:37:37:30:30:36:34:34:05:03:34:2e:30:06:12:54:53:2e:30:34:2e:30:31:2e:30:34:2e:30:33:31:35:30:34:07:00:08:06:30:30:30:30:43:41:09:06:54:4d:34:30:32:50:0a:19:41:72:72:69:73:20:49:6e:74:65:72:61:63:74:69:76:65:2c:20:4c:2e:4c:2e:43:2e
DHCP Response Dictionary:	
DHCP Environment Dictionary:	
Lease Information	
IP Address:	10.7.1.6
DHCP Lease Properties:	/network/lastTransTime=83, /network/isLeased=Leased, /network/leaseTime=221
Technology Specific Information	
XGCP Ports:	
DOCSIS Version:	1.1

Table 9-1 identifies the fields shown Figure 9-3.

Table 9-1 Device Details Page

Field or Button	Description
Device Details	
Device Type	Displays a drop-down list that identifies the available device types.
MAC Address	Identifies the devices MAC address.
FQDN	Identifies the fully qualified domain name for the selected device. For example, ACME.COM is a fully qualified domain name.
Host Name	Identifies the host. For example, from the FQDN description above, ACME is the host name.
Domain Name	Identifies the domain within which the host resides. For example, from the FQDN description above, .COM is the domain name.
Behind Device	Identifies the device that is behind this device.
Provisioning Group	Identifies the provisioning group to which the device is assigned.
DHCP Criteria	Identifies the DHCP criteria used. This is an active link that, if clicked, displays the appropriate Modify DHCP Criteria page.
CPE DHCP Criteria	Identifies the DHCP criteria used for customer premises equipment, when in the Promiscuous mode.
Device Properties	Identifies any properties, other than those displayed on this page, that can be set for this device. These are custom properties.
Device Provisioned State	Identifies whether the device is provisioned or not. A device is provisioned only when it is registered and has booted on the network.
Device Registered State	Identifies whether the device is registered or not.
Relay Agent Circuit Identifier	Identifies the relay agent circuit identifier of the device. This is equivalent to DHCP option 82, sub option 1.
Relay Agent Remote Identifier	Identifies the relay agent remote identifier of the device. This is the equivalent to the DHCP option 82, sub option 2.
Client Identifier	Identifies the client identification used by the device in its DHCP messages.
Client Request Host Name	Identifies the host name that the client requested in its DHCP messages.
Class of Service	Identifies the class of service assigned to this device.
Owner Identifier	Identifies the device. This may be a user ID, and account number, or may be blank.
Detected Properties	Identifies properties returned by the RDU device detection extension(s) when the device's configuration was generated.
Selected Properties	Identifies properties returned by the RDU service level selection extension(s) for the detected device type when the device's configuration was generated.
Is Behind Required Device	This returns "false" if the IPDeviceKeys.MUST_BE_BEHIND_DEVICE property has been used to establish a required relay agent device and the service level selection extension(s) determined this device did not boot behind the required relay agent.

Table 9-1 Device Details Page (continued)

Field or Button	Description
Is In Required Provisioning Group	This returns “false” if the IPDeviceKeys.MUST_BE_IN_PROV_GROUP property has been used to establish a required provisioning group and the service level selection extension(s) determined this device did not boot in the required provisioning group.
Selected Access	Identifies the access granted to the device by the service level selection extension(s): <ul style="list-style-type: none"> REGISTERED—Indicates the device was registered and met any requirements for access. PROMISCUOUS—Indicates the device’s provisioning will be based on policies assigned to it’ relay agent. DEFAULT—Indicates the device will be provisioned with the default access for its device type,. OTHER—Not used by the default extensions built into BACC and is provided for use by custom extensions.
Selected Class of Service	Identifies the name of the class of service used to generate the device’s configuration.
Selected DHCP Criteria	Identifies the name of the DHCP criteria used to generate the device’s configuration.
Selected Explanation	Provides a textual description of why the service level selection extension(s) selected the access they granted the device. For example, the device may have been granted default access because it did not boot in its required provisioning group.
Selected Reason	Identifies why the service level selection extension(s) selected the access they granted the device as an enumeration code. The possible values are: <ul style="list-style-type: none"> NOT_BEHIND_REQUIRED_DEVICE NOT_IN_REQUIRED_PROV_GROUP NOT_REGISTERED OTHER PROMISCUOUS_ACCESS_ENABLED REGISTERED RELAY_NOT_IN_REQUIRED_PROV_GROUP RELAY_NOT_REGISTERED Most of these indicate violations of requirements for granting registered or promiscuous access resulting in default access being granted.
Related Node Type	Identifies the node type to which this device is related. See the “ Node Management ” section on page 9-16 for additional information.

Table 9-1 Device Details Page (continued)

Field or Button	Description
DHCP Information	
DHCP Inform Dictionary	Identifies additional information that the Network Registrar extensions send to the RDU when requesting the generation of a configuration. This is for internal BACC use only.
DHCP Request Dictionary	Identifies the DHCPDISCOVER or DHCPREQUEST packet details that were sent from the Network Registrar extensions to the RDU when requesting the generation of a configuration.
DHCP Response Dictionary	This field is for internal BACC use only; it should always be empty.
DHCP Environment Dictionary	This field is for internal BACC use only; it should always be empty.
Lease Information	
IP Address	Identifies a device's IP address.
DHCP Lease Properties	Identifies the lease properties, this should always be empty.
Technology Specific Information	
Note	The technology-specific information identifies only data that is relevant for the technologies you are licensed to use.
XGCP Ports	Identifies the ports on which the gateway control protocol is active.
DOCSIS Version	Identifies the DOCSIS version currently in use.

Managing Devices

As mentioned earlier in this chapter, the Devices menu lets you create and manage how devices are maintained within the RDU database. Device management includes adding, deleting, and modifying devices as well as regenerating configurations, relating, and unregistering selected devices.

This section describes how to perform the various device management functions on new or existing devices. Several information fields appear consistently over all device management pages. These include:

- Device Type—When adding a device, this is a drop-down list that identifies the available device types you can create within BACC. Available selections, as they appear on screen, include:
 - ATA186
 - ATA188
 - CableHomeManData
 - CableHomeManWan
 - DOCSISmodem
 - Computer
 - PacketCableMTA

When modifying a device the device type can not be edited or changed.

- **MAC Address**—This is the MAC address of the device being added.

Enter the MAC address of the device being added in this field. When doing this, you must ensure that you enter the commas and colons in the appropriate positions. For example:

```
1,6,00:00:00:00:00:AE
```

- **Host Name**—Identifies the device's host. For example, from an FQDN of node.cisco.com, node is the host name.
- **Domain Name**—Identifies the domain within which the host resides. For example, from an FQDN of node.cisco.com, cisco.com is the domain name.
- **Owner Identifier**—Identifies the device using something other than the host name. This may be a user ID, an account number, or may be left blank.
- **Class of Service**—Specifies the class of service that the device is to be provisioned with.
- **DHCP Criteria**—Specifies the DHCP criteria that the device is to be provisioned with.

Depending on the page displayed, additional information may appear. Where appropriate, this additional information is identified in the following procedures.

Adding Devices

To add a device:

-
- Step 1** From the Manage Devices page, click **Add**. The Add Device page appears.
 - Step 2** Choose the device type and class of service, and complete the other fields on the page. In addition to the fields described earlier in this section, you can also add new values for existing property name/value pairs.
 - **Property Name**—Identifies the name of the custom or built-in device property.
 - **Property Value**—Identifies the value of the property.
 - Step 3** Click **Submit** to add the device, or **Reset** to clear all fields.
-



Note

To specify a CPE DHCP criteria for a DOCSIS modem, for use with promiscuous computers, you must specify the property `/provisioning/cpeDhcpCriteria`. The value must be a valid DHCP criteria.

Modifying Devices

To modify a device:

-
- Step 1** From the Manage Devices page, click the **Identifier** link corresponding to the desired device. The Modify Device page appears.
 - Step 2** Choose the desired parameter(s) from the applicable drop-down list(s) or enter the data in the correct field. You can modify any existing property name/value pairs by clicking **Add**, or delete any of them by clicking **Delete**.
 - Step 3** Click **Submit** to save the changes made to this device, or **Reset** to clear all fields.
-

Deleting Devices

Deleting devices is a simple process, but one that you should use carefully. To undo the delete, you must restore a previously backed up database or re-add the device.

**Note**

Refer to the [“Database Restore” section on page 7-5](#) for additional information if restoration of a backed up database becomes necessary.

To delete a device:

- Step 1** From the Devices page, locate the device that you want to delete. You can use one of the search types for this purpose.
- Step 2** Click the check box to the left of the desired device.
- Step 3** Click **Delete**. The device record stored in the RDU database is removed.

Regenerating Device Configurations

It is sometimes necessary to change many different class of service or DHCP criteria parameters. When this happens, existing device configurations become stale and require regeneration of the configuration. To eliminate the need to manually regenerate each configuration, and reduce the potential for introducing errors, BACC provides a configuration regeneration service (CRS) that you can use to automatically regenerate all device configurations.

Device configurations are automatically regenerated whenever:

- A class of service property is changed.
- A DHCP criteria property changes.
- An external file is replaced. This applies to files that are dynamic DOCSIS templates and are directly associated with a class of service.
- New default classes of service or DHCP criteria are designated.

In addition, some configurations cannot be automatically regenerated, but must rely on manual regeneration using either the `generationConfiguration()` method or the administrator’s user interface. Configurations that must be manually regenerated are those that become necessary whenever:

- A technology default is changed.
- The system defaults are changed.
- A file that is included within another DOCSIS template is changed.

**Note**

Regardless of how configurations are regenerated, they are not propagated to the devices until the device is rebooted.

Relating and Unrelating Devices

The concept of relating devices is somewhat similar to that of Class of Service or DHCP Criteria inasmuch as a device is related to a specific class of service or to a specific DHCP criteria. The significant difference is that the Class of Service and DHCP Criteria are considered to be predefined nodes and that you use nodes to group devices into arbitrary groups that you define.

In this context, the Relate function lets you associate a device, using its MAC address, to a specific node, which is in turn associated with a specific node type.

By relating a device to a specific node, an extraordinarily large volume of information is recorded for the device. This information can then be used to troubleshoot potential problems. [Table 9-2](#) identifies a possible work flow using the Relate and Unrelate functions.

Table 9-2 Example Relate/Unrelate Process

Step	Action
1.	Determine whether or not a problem exists and identify which device(s) are affected.
2.	Relate the device(s) to a node.
3.	Wait a few minutes to ensure that device traffic is passing, or perform a hard boot of the device.
4.	Open the rdu.log file in a word processing application and locate the entries for the appropriate devices MAC address.
5.	Identify, correct, test, and verify the problem.
6.	Unrelate the device from the node.

Resetting Devices

The Reset button lets you reboot any selected device.

Unregistering a Device

The unregister function lets you reset a device back to its defaults as if it had just booted on the network.



Note

If the device has never booted on the network, it will be deleted.

Node Management

Node management allows the creation, modification and deletion of nodes and node types. Within the context of BACC, node types can be considered as groups of nodes, while nodes themselves make up the node type.

Managing Node Types

Access the Manage Nodes page (shown in [Figure 9-4](#)) by selecting Nodes from either the main menu or the primary menu bar. Node Type is the default setting when this page appears.

Figure 9-4 Manage Nodes Page

The screenshot displays the 'Manage Nodes' page. At the top, there is a navigation bar with 'Configuration | Devices | **Nodes** | Servers | Users' and a 'Logout' button. Below this, the page title is 'Broadband Access Center for Cable'. The user information 'User:admin Role:Administrator' is shown in the top right. The main content area has a 'Manage Nodes' header with a sub-header 'Use this page to manage nodes and node types (add, delete, modify or search)'. A 'Search Type' dropdown is set to 'Node Type'. An 'Add' button is visible. Below is a table with three rows: 'EST Nodes', 'Node_ABC', and 'system'. Each row has a 'Delete' column with a trash icon. At the bottom left, it says 'Result Pages: 1'.

Node Types	Delete
EST Nodes	
Node_ABC	
system	

123906

Adding a Node Type

To add a new node type:

- Step 1** Click **Add** and the Add Node Type page appears.
- Step 2** Enter a name for the new node type.
- Step 3** Select the appropriate Property Name from the drop-down list and enter the required Property Value.
- Step 4** Click **Add** and the new node type appears. You can continue adding as many properties as required.
- Step 5** Click **Submit** when complete. The new node type is recorded in the RDU and the Manage Node Types page appears with the new node type added.

Modifying Node Types

To modify node type properties:

-
- Step 1** Click the desired node type and the Modify Node Type page appears.
 - Step 2** Make the necessary changes to the Property Name/Property Value pairs. If you need to delete a specific pair, click **Delete** next to the desired pair.
 - Step 3** Click **Submit** and the Manage Node page appears with the appropriately modified description.
-

Deleting Node Types

To delete node types:

-
- Step 1** In the Manager Node page click the **Delete** icon corresponding to the desired node type.
 - Step 2** In the Delete Node Type dialog box, click **OK** to delete the selected node type, or **Cancel** to return to the previous page.

The Manage Nodes page appears without the deleted Node Type.

Managing Nodes

You can create and modify nodes, and delete unwanted nodes.

Adding a New Node

To add a new node:

-
- Step 1** Select **Nodes** from the drop-down list on the Manage Nodes page.
 - Step 2** Click **Add** and the Add Node page appears.
 - Step 3** Enter the new node name and select the appropriate Node Type for this node.
Click **Submit** when complete and the Manage Node page appears with the new node added.
 - Step 4** Select the appropriate Property Name from the drop-down list and enter the required Property Value.
 - Step 5** Click **Add** to increase the number of applicable Property Name/Property Value pairs.
 - Step 6** Click **Submit** when complete. The new node is recorded in the RDU and the Manage Nodes page appears with the new node added.
-

Modifying a Node

To modify node properties:

-
- Step 1** Click the desired node and the Modify Nodes page appears.
 - Step 2** Make the necessary changes to the Property Name/Property Value pairs. If you need to delete a specific pair, click **Delete** next to the desired pair.
 - Step 3** Click **Submit** and the Manage Node page appears with the appropriately modified description.
-

Deleting Nodes

You can delete any node that appears in the Manager Node page by checking the box corresponding to the node being deleted and then click **Delete**. The node is immediately deleted.

Relating/Unrelating Node Types to Nodes

The relate and unrelate functions are used to establish a relationship between specific node types and nodes. To either relate or unrelate this relationship:

-
- Step 1** Click **Relate** or **Unrelate**, as desired, for the selected node. Either the Relate Nodes or Unrelate Node page appears.
 - Step 2** Select the appropriate **Node Type** from the drop-down list and select the group to which the node will be related/unrelated.
 - Step 3** Click **Submit** and the Manage Nodes page appears.
-

Viewing Node Details

Use the **Devices** icon to search for a specific node. Doing this displays the Node Search function on the Manage Devices page. From this page select the appropriate Node Type and enter the node name. You can use wildcard characters to locate a group of similarly named nodes. See the [“Searching for Devices” section on page 9-6](#) for additional information on search functions.

Viewing Servers

This section describes the BACC administrator view server pages:

- [Listing Device Provisioning Engines, page 9-19](#)
- [Listing Network Registrar Extension Points, page 9-21](#)
- [Listing Provisioning Groups, page 9-23](#)
- [Viewing Regional Distribution Unit Details, page 9-24](#)

Listing Device Provisioning Engines

The List Device Provisioning Engines page lets you monitor the list of all DPEs currently registered with the BACC database. Each DPE name displayed on this page is a link to another page that shows the details for that DPE. Click this link to display the details page, which is similar to [Figure 9-5](#).


Note

The RDU determines the names of the CNR extensions and DPEs by the interface they connect to the RDU on. That is, the name of the DPE or CNR extensions is what the RDU machine thinks it is.

Figure 9-5 View Device Provisioning Engine Details Page

The screenshot shows the 'View Device Provisioning Engines Details' page in the Broadband Access Center for Cable. The page header includes navigation tabs for Configuration, Devices, Nodes, Servers, and Users, with 'Servers' selected. Below the header, there are tabs for DPEs, NRs, Provisioning Groups, and RDU. The user is identified as 'admin' with the role of 'Administrator'. The main content area displays the following details:

Device Provisioning Engine Details	
Host Name:	test.cisco.com
Port:	49186
IP Address:	10.10.10.1
Primary Provisioning Group(s):	default
Secondary Provisioning Group(s):	
PacketCable Enabled:	No
CableHome Enabled:	Yes
Properties:	
Version:	BPR 2.7
UpTime:	12 day(s) 21 hour(s) 54 min(s) 49 sec(s)
State:	Ready
Cache Statistics	
Hits:	0
Misses:	0
Files:	0
Configurations:	2
TFTP Statistics	
Packets Received:	0
Packets Dropped:	0
Packets Successful:	0
Packets Failed:	0
Time Of Day Statistics	
Packets Received:	0
Packets Dropped:	0
Packets Successful:	0
Packets Failed:	0
PacketCable SNMP Statistics	
SNMP Informs Successful:	0
SNMP Sets Successful:	0
SNMP Configuration Informs Successful:	0
SNMP Configuration Informs Failed:	0
PacketCable MTA Statistics	
MTA AP Requests Received:	0
MTA AP Responses Sent:	0
PacketCable KDC Statistics	
KDC FQDN Requests Received:	0
KDC FQDN Responses Sent:	0

[Table 9-3](#) identifies the fields and buttons shown in [Figure 9-5](#).


Note

The type of DPE details displayed for a hardware or Solaris DPE are the same.

Table 9-3 View Device Provisioning Engine Details Page

Field or Button	Description
Device Provisioning Engine Details	
Host Name	Identifies the DPE host name.
Port	Identifies the DPE port number.
IP Address	Identifies the IP address of the DPE.
Primary Provisioning Group(s)	Identifies the primary provisioning groups that the selected DPE belongs to. This is an active link that, if clicked, displays the Provisioning Group Details page for that provisioning group.
Secondary Provisioning Group(s)	Identifies the secondary provisioning group (provided that this DPE belongs to a secondary provisioning group) that the selected DPE belongs to.
PacketCable Enabled	Identifies whether the PacketCable voice technology is enabled on this DPE.
CableHome Enabled	Identifies whether the home networking technology is enabled on this DPE.
Properties	Identifies which properties have been assigned to this DPE.
Version	Identifies the version of DPE software currently in use.
Up Time (in seconds)	Specifies the total amount of time that the DPE has been operational since its last period of down time.
State	Identifies whether the DPE is ready for operations. Note If this field reads Offline, up time, and all fields below it, are not displayed.
Cache Statistics	
Hits	Identifies the number of cache hits that occurred since the last time the DPE was started.
Misses	Identifies the number of cache misses that occurred since the last time the DPE was started.
Files	Identifies the number of cache files that are currently stored in the DPE.
Configurations	Identifies how many device configuration files are saved in cache.
TFTP Statistics	
Packets Received	Identifies the number of TFTP packets that were received by the selected DPE.
Packets Dropped	Identifies the number of TFTP packets that were dropped due to the DPE being overloaded.
Packets Successful	Identifies the number of TFTP packets that were transmitted successfully.
Packets Failed	Identifies the number of TFTP packets that failed during transmission.
Time of Day Statistics	

Table 9-3 View Device Provisioning Engine Details Page (continued)

Field or Button	Description
Packets Received	Identifies the number of Time of Day packets that were received by the selected DPE.
Packets Successful	Identifies the number of Time of Day packets that were transmitted successfully.
Packets Failed	Identifies the number of Time of Day packets that failed during transmission.
PacketCable SNMP Statistics	
SNMP Informs Successful	Identifies the number of inform requests that were successfully sent.
SNMP Sets Successful	Identifies the number of successful SNMP sets.
Configuration Informs Successful	Identifies the number of SNMP informs received from PacketCable MTAs indicating that they were successfully provisioned.
Configuration Informs Failed	Identifies the number of SNMP informs received from PacketCable MTAs indicating that they failed to be provisioned.

Listing Network Registrar Extension Points

The List Network Registrar Extension Points page lists the extension points for all Network Registrar servers that have been registered with the RDU, and are configured for use with BACC. Network Registrar servers automatically register with the RDU when those servers are started.

Each Network Registrar extension point displayed on this page is a link to a secondary page that displays details of that extension point. Click this link to display the details page, which is similar to [Figure 9-6](#).

Figure 9-6 View Cisco Network Registrar Details Page

The screenshot shows the 'Broadband Access Center for Cable' interface. At the top, there is a navigation menu with 'Logout' on the right and 'Configuration | Devices | Nodes | Servers | Users' in the center. Below this, there are sub-menus for 'DPEs', 'NRs', 'Provisioning Groups', and 'RDU'. The user is identified as 'admin' with the role of 'Administrator'.

The main content area is titled 'View Network Registrar Extension Point Details' and includes a sub-header: 'Use this page to view the current values for the Network Registrar extension point that you selected.'

The details are organized into three sections:

- Network Registrar Extension Point Details:**

Host Name:	bacdev3-dpe-4.cisco.com
IP Address:	10.86.147.16
Provisioning Group:	default
PacketCable Enabled:	Yes
CableHome Enabled:	Yes
Properties:	
Version:	BPR 2.7XF (bacc-27-S-csrc_main-200504070002)
UpTime:	1 day(s) 8 hour(s) 48 min(s) 47 sec(s)
State:	Ready
- Network Registrar Extension Point Statistics:**

Packets Received:	0
Packets Ignored:	0
Packets Dropped:	0
Packets Successful:	0
Packets Failed:	0
- Device Provisioning Engine(s) Details:**

DPE:	bacdev3-dpe-4.cisco.com (10.86.147.16)
Port:	49186
Type:	Primary Device Provisioning Engine
Status:	Ready

Table 9-4 identifies the fields and buttons shown in Figure 9-6.

Table 9-4 View Network Registrar Extension Point Details Page

Field or Button	Description
Network Registrar Extension Point Details	
Host Name	Displays the host name of the system running Network Registrar.
IP Address	Identifies the IP address of the Network Registrar.
Provisioning Group	Identifies the provisioning group for this Network Registrar servers. This is an active link that, if clicked, displays the Provisioning Group Details page for that provisioning group.
PacketCable Enabled	Identifies whether the PacketCable voice technology is enabled.
Properties	Identifies the properties that are applied to the selected Network Registrar.
Version	Identifies the extension point version currently in use.
Up Time	Identifies how long the Network Registrar extension point has been operational. This is indicated in hours, minutes, and seconds.
State	Identifies whether the DPE is ready for operations. Note If this field reads Offline, up time, and all fields below it, are not displayed.
Network Registrar Extension Point Statistics	

Table 9-4 View Network Registrar Extension Point Details Page (continued)

Field or Button	Description
Packets Received	Identifies the number of packets that were received.
Packets Ignored	Identifies the number of packets that were ignored.
Packets Dropped	Identifies the number of packets that were dropped.
Packets Successful	Identifies the number of packets that transferred successfully.
Packets Failed	Identifies the number of packets that failed to be transferred.
Device Provisioning Engine(s) Details	
DPE	Identifies the IP address of the device provisioning engine.
Port	Identifies the DPE port number.
Type	Identifies whether this DPE is a primary or secondary DPE.
Status	Identifies whether the DPE is operational.

Listing Provisioning Groups

The List Provisioning Groups page lets you monitor all current provisioning groups. Each provisioning group appearing in this list is a link to its own details page. Click this link to display the details page, which is similar to [Figure 9-7](#).

Figure 9-7 View Provisioning Group Details Page

Broadband Access Center for Cable Logout

Configuration | Devices | **Servers** | Users

DPEs | NRs | **Provisioning Groups** | RDU

CISCO SYSTEMS **View Provisioning Group Details**
Use this page to view the current values for the provisioning group that you selected.

Provisioning Group Details	
Name:	default
Primary Device Provisioning Engine(s):	bac-u5-26.cisco.com
Secondary Device Provisioning Engine(s):	
Network Registrar Extension Point(s):	bac-u5-26.cisco.com

101573

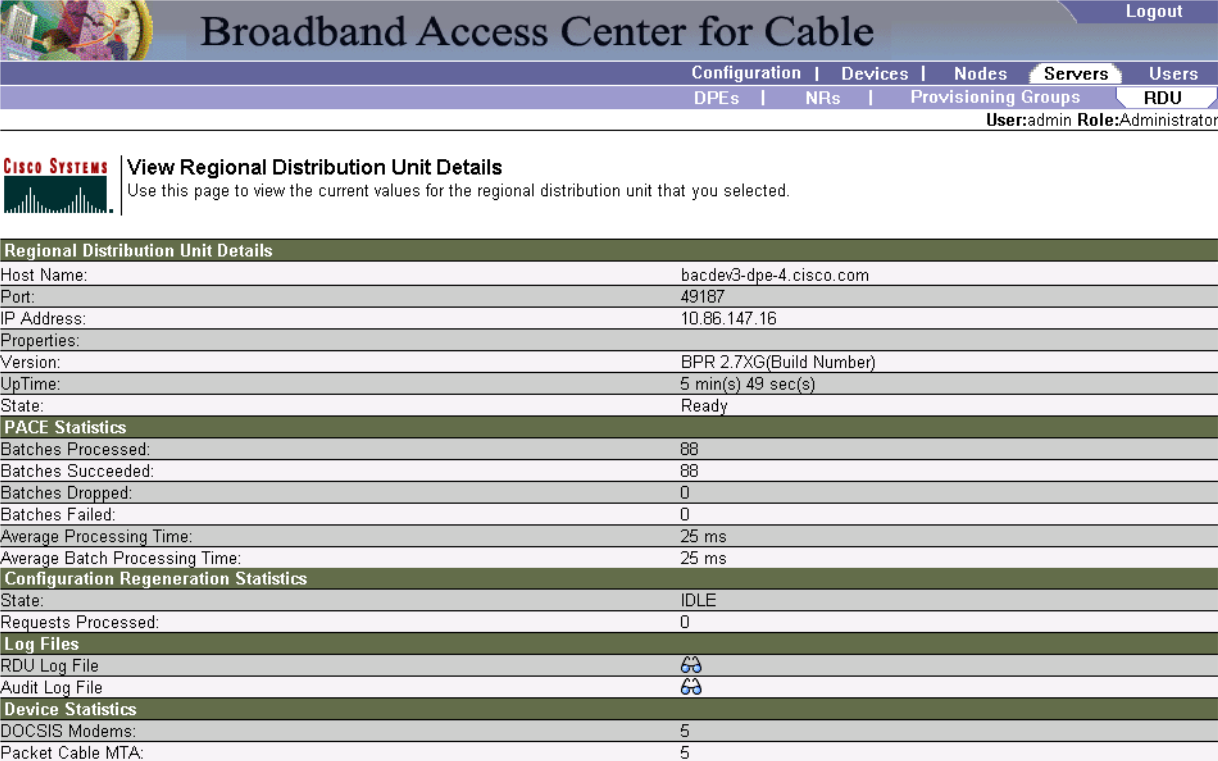
[Table 9-5](#) identifies the fields and buttons shown in [Figure 9-7](#). The fields described in [Table 9-5](#) may include active links that, if clicked, display the appropriate details page.

Table 9-5 View Provisioning Groups Details Page

Field or Button	Description
Name	Identifies the provisioning group name selected from the List Provisioning Groups page.
Primary Device Provisioning Engine(s)	Identifies the host names of the DPEs that are primary for this provisioning group.
Secondary Device Provisioning Engine(s)	Identifies the host names of the DPEs that are secondary for this provisioning group.
Network Registrar Extension Points	Identifies the host name of the Network Registrar server assigned to this provisioning group.

Viewing Regional Distribution Unit Details

The RDU option, from the Servers menu, displays details of the RDU currently in use. [Figure 9-8](#) illustrates a sample RDU details page.

Figure 9-8 View Regional Distribution Unit Details Page


The screenshot shows the following interface elements:

- Page Header:** Broadband Access Center for Cable, with navigation tabs for Configuration, Devices, Nodes, Servers (selected), and Users. Sub-tabs under Servers include DPEs, NRs, Provisioning Groups, and RDU (selected). User: admin, Role: Administrator.
- Section Header:** View Regional Distribution Unit Details. Description: Use this page to view the current values for the regional distribution unit that you selected.
- Regional Distribution Unit Details Table:**

Regional Distribution Unit Details	
Host Name:	bacdev3-dpe-4.cisco.com
Port:	49187
IP Address:	10.86.147.16
Properties:	
Version:	BPR 2.7XG(Build Number)
UpTime:	5 min(s) 49 sec(s)
State:	Ready
PACE Statistics	
Batches Processed:	88
Batches Succeeded:	88
Batches Dropped:	0
Batches Failed:	0
Average Processing Time:	25 ms
Average Batch Processing Time:	25 ms
Configuration Regeneration Statistics	
State:	IDLE
Requests Processed:	0
Log Files	
RDU Log File	63
Audit Log File	63
Device Statistics	
DOCSIS Modems:	5
Packet Cable MTA:	5

[Table 9-6](#) identifies the fields and buttons shown in [Figure 9-8](#).

Table 9-6 View RDU Details Page

Field or Button	Description
Regional Distribution Unit Details	
Host Name	Identifies the host name of the system that is running the regional distribution unit.
Port	Identifies the RDU listening port number. Although the default port number is 49187, the port number displayed in this field is the same as that entered as the Regional Distribution Unit Host/Port screen during installation.
IP Address	Identifies the IP address assigned to the RDU.
Properties	Identifies the RDU properties that are used for server configuration and control.
Version	Specifies the version of RDU software currently in use.
Up Time	Specifies the total amount of time that the RDU has been operational since its last period of down time.
State	Identifies whether the RDU is ready to respond to requests.
PACE Statistics	
Batches Processed	Identifies how many individual batches have been processed while the PACE engine has been operating.
Batches Succeeded	Identifies how many individual batches have been successfully processed while the PACE engine has been operating.
Batches Dropped	Identifies how many batches have been dropped while the PACE engine has been operating.
Batches Failed	Identifies how many batches have failed processing while the PACE engine has been operating.
Average Processing Time	Identifies the average time, in milliseconds, that it takes to process the batch excluding the time it spends in the queue if RDU is too busy.
Average Batch Processing Time	Identifies the average time, in milliseconds, that it takes to process the batch including the time it spends in the queue if RDU is too busy.
Configuration Regeneration Statistics	
State	Identifies whether the configuration regeneration service is ready to respond to requests.
Requests Processed	Identifies how many configuration regeneration requests have been processed.

Table 9-6 View RDU Details Page (continued)

Field or Button	Description
Device Statistics	
<p>The information presented in this area depends on the technologies licensed and configured. The values presented for each technology represent the number of devices identified in the RDU database. These devices may include:</p>	
<ul style="list-style-type: none"> • DOCSIS Modems • Computers • ATA 186 and 188 devices • PacketCable MTAs • CableHome WAN DATA/WAN MAN devices 	
Note	<p>The Device Statistics section appears only when the appropriate devices are present. If external Jar files are installed, detailed information appears immediately after the Device Statistics section.</p>

**Note**

For performance reasons, the RDU does not write updated database statistics to disk every time a device record is updated. Database statistics, such as device object counters shown in [Table 9-6](#), are maintained in memory and periodically written to the database. Consequently, the device statistics displayed in this page are approximate counts that may not exactly match the actual number of devices in the database.



Configuring Broadband Access Center for Cable

This chapter describes the Broadband Access Center for Cable (BACC) configuration activities that you perform using Configuration menu options:

- [Configuring the Class of Service, page 10-1](#)
- [Configuring Custom Properties, page 10-6](#)
- [Configuring Defaults, page 10-7](#)
- [Configuring DHCP Criteria, page 10-23](#)
- [Managing External Files, page 10-25](#)
- [Managing License Keys, page 10-29](#)
- [Managing Regional Distribution Unit Extensions, page 10-31](#)
- [Publishing Provisioning Data, page 10-32](#)

Configuring the Class of Service

Using the BACC administrator, you can configure the classes of service offered to your customers. For example, you can associate DOCSIS options with different DOCSIS classes of service. You use the BACC administrator user interface to add, modify, view, or delete any selected class of service. Start with the Manage Class of Service page, as shown in [Figure 10-1](#).

Figure 10-1 Manage Class of Service Page

Table 10-1 identifies the fields and buttons shown in Figure 10-1.

Table 10-1 Configure Class of Service Page

Field or Button	Description
Class of Service	<p>A drop-down list that identifies the technology classes of service that you can search for. Available selections, as they appear on screen, include:</p> <ul style="list-style-type: none"> • ATA 186 • ATA 188 • CableHomeManData • CableHomeManWan • DOCSISmodem • Computer • PacketCableMTA <p>Note Refer to the “Configuring Defaults” section on page 10-7, for additional information on these areas of technology.</p>
Add	Lets you add a new class of service.
Class of Service list	Displays the attributes of any selected class of service.
Delete	Lets you delete selected classes of service.

Adding a Class of Service

To add a specific class of service:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **Class of Service** from the Secondary Navigation bar.
 - Step 3** Click **Add**. The Add Class of Service page appears. This page identifies the various settings for the selected class of service.
 - Step 4** Enter the name of your new class of service.
 - Step 5** Choose a **Class of Service Type**.
 - Step 6** Enter a **Property Name** and **Property Value** in the appropriate fields.

For example:

Assume that you want to create a new class of service called Gold-Classic for DOCSIS modems. You might:

- a. Enter **Gold-Classic** as the Class of Service Name.
 - b. Choose **DOCSIS** from the service type drop-down list.
 - c. Choose the `/cos/docsis/file` property file name.
 - d. Enter **Gold-Classic.cm** in the Property Value field and then continue with the rest of this procedure.
- Step 7** Click **Add** to add the property to the defining class of service.
 - Step 8** Click **Submit** to finalize the process or **Reset** to return all fields to their previous setting. After submitting the class of service, the Manage Class of Service page appears to show the newly added class of service for that particular device type.



Note

Multiple Property Name:Property Value pairs could appear on this page. You use the **Delete** button to remove any unwanted pairs from the class of service.



Caution

When adding a DOCSISModem class of service, you must specify the `/cos/docsis/file` property with the value being the name of a previously added external file. This file is used when provisioning a DOCSIS device that has this class of service.

When adding a PacketCable class of service, you must specify the `/cos/packetCableMTA/file` property with the value being the name of a previously added external file. This file will be used when provisioning a Packetcable device that has this class of service.

When adding a CableHomeWanMan class of service, you must specify the `/cos/cableHomeWanMan/file` property with the value being the name of a previously added external file. This file will be used when provisioning a CableHomeWanMan device that has this class of service.

Table 10-2 identifies the fields and buttons shown in the Add Class of Service page.

Table 10-2 Add Class of Service Page

Field or Button	Description
Class of Service Name and Type	
Class of Service Name	Lets you enter the name of the new class of service.
Class of Service Type	A drop-down list that identifies the types of classes of service that you can select.
Property Name/Value	
Property Name	Specifies the appropriate property. You can select the desired property from the drop-down list.
Property Value	Specifies the value for the property name selected.
Add	Adds the new Property Name:Property Value pair to create the new class of service.
Submit	Activates or implements the changes you have made.
Reset	Returns all settings to their previous setting.

Modifying a Class of Service

You modify your classes of service by selecting the various properties and assigning appropriate property values. When creating a class of service for the first time you must select all the required properties and assign values to them. If you make a mistake, or your business requirements for a certain class of service change, you can either change the property value before submitting your previous changes or delete the Property Name:Property Value pair altogether.



Note

Subsequent device configurations will include the changes you implement here. All existing configurations are regenerated, although the devices on the network will not get the new configuration until they reboot.

To add, delete, or modify class of service properties:

- Step 1** Choose **Configuration** on the Primary Navigation bar.
- Step 2** Choose **Class of Service** from the Secondary Navigation bar.
- Step 3** Choose the class of service to be modified.
- Step 4** Click the link corresponding to the desired class of service. The Modify Class of Service page appears; note that the selected class of service name and type are displayed below the page description.
 - To add a new property to the selected class of service:
 - Select the first property that you want assigned to the selected class of service, from the Property Name drop-down and then, after entering the appropriate value for that property, click **Add**.
 - Repeat for any other properties you want to assign to the selected class of service.

- To delete a property for the selected class of service:
 - Locate the unwanted property in the list immediately above the Property Name drop-down.
 - Click **Delete**.
- To modify the value currently assigned to a property:
 - Delete the appropriate property as described above.
 - Add the same property back to the class of service while entering the new Property Value.

**Note**

If you delete a required property you must add it back, and select the appropriate value, before you submit the change.

- Step 5** Click **Submit** to make the modifications to the class of service. Each property added to a class of service, is displayed when you click **Submit**. After doing so, a confirmation page appears to regenerate the configuration for the devices with the selected Class of Service.
- Step 6** Click **OK** and the modified class of service will be available in the Manage Class of Service page.

Deleting a Class of Service

You can delete any existing Class of Service but, before you attempt to do so, you must ensure that there are no devices associated with that Class of Service.

**Tip**

Where there are large numbers of devices associated with a Class of Service to be deleted, use the BACC application programmers interface (API) to write a program to iterate through these devices to reassign another class of service to the devices.

If you try to delete a Class of Service with devices associated with it, this error message is displayed:

```
The following error(s) occurred while processing your request.
Error: Class Of Service [sample-COS] has devices associated with it, unable to delete
Please correct the error(s) and resubmit your request.
```

The specific class of Service is specified within the error message. In this example this is represented by *sample-COS*.

To delete a class of service:

- Step 1** Choose **Configuration** on the Primary Navigation bar.
- Step 2** Choose **Class of Service** from the Secondary Navigation bar.
- Step 3** Click **Delete** for any desired class of service, and a confirmation dialog box appears.

**Note**

You cannot delete the default 'unprovisioned-docsis' Class of Service.

- Step 4** Click **OK** to delete the file, or **Cancel** to return to the Manage Class of Service page. (See [Figure 10-1](#).)

**Note**

A class of service cannot be deleted if devices are associated with it or, if it is designated as the default class of service.

Configuring Custom Properties

Custom properties let you specify additional customizable device information to be stored in the RDU database. The Custom Property configuration page is found under the Configuration menu and you use this page to add or delete custom properties.

**Caution**

Although you can delete custom properties if they are currently in use, doing so could cause extreme difficulty to other areas where the properties are in use.

To configure custom properties:

- Step 1** Choose **Configuration** on the Primary Navigation bar.
- Step 2** Choose **Custom Property** from the Secondary Navigation bar and the Configure Custom Properties page appears.
 - To add a custom property:
 - Click **Add** on the Configure Custom Properties page, and the Add Custom Property page appears.
 - Enter the name of the new custom property.
 - Choose a custom property type from the drop-down list.
 - Click **Submit** when complete. After the property has been added to the administrative database, the Configure Custom Properties page appears.
 - To delete a custom property:
 - Identify the custom property to be deleted from the Configure Custom Properties page.
 - Click the **Delete** icon corresponding to the desired custom property, and the custom properties deletion dialog box appears.
 - Click **OK** to delete the custom property.
- Step 3** After clicking **Submit** or **OK**, and your custom property is added or deleted, the Configure Custom Properties page appears.

Configuring Defaults

The Defaults page, found under the Configuration option, lets you access the default settings for the overall system, including the regional distribution unit (RDU), Network Registration extensions, and all supported technologies.

Selecting Configuration Options

The procedure for configuring specific default types is identical. Complete this procedure to access the desired defaults page and then refer to the appropriate section within this chapter for a description of the various page components.

-
- Step 1** Choose **Configuration** on either the Primary Navigation bar or Main Menu page.
 - Step 2** Choose **Defaults** from the Secondary Navigation bar and the Configure Defaults page appears.
 - Step 3** Choose the desired default type from the list to the left of the screen. The appropriate defaults page appears.
-

ATA 186 Defaults

The Cisco ATA 186 is a handset-to-Ethernet adaptor that turns a traditional telephone into an Ethernet IP telephone. You can take advantage of the many IP telephony applications by connecting an existing analog telephone to this device.

The Configure ATA 186 Defaults page ([Figure 10-2](#)) displays a list of default values currently available to support the ATA 186.

Figure 10-2 Configure ATA 186 Defaults Page

Broadband Access Center for Cable Logout

[Configuration](#) | [Devices](#) | [Nodes](#) | [Servers](#) | [Users](#)
[Class of Service](#) | [Custom Property](#) | **Defaults** | [DHCP Criteria](#) | [External Files](#) | [License Keys](#) | [Publishing](#)

User:admin Role:Administrator

CISCO SYSTEMS | **Configure Defaults**
 Use this page to change the defaults.
 Fields marked with an "*" are required.

Defaults

- [ATA 186 Defaults](#)
- [ATA 188 Defaults](#)
- [CHWAN DATA Defaults](#)
- [CHWAN MAN Defaults](#)
- [Computer Defaults](#)
- [DOCSIS Defaults](#)
- [NR Defaults](#)
- [Packet Cable Defaults](#)
- [RDU Defaults](#)
- [System Defaults](#)
- [XGCP Defaults](#)

ATA 186 Defaults

Extension Point:

Disruption Extension Point:

Service Level Selection Extension Point:

Default Class of Service:

Default DHCP Criteria:

Automatic FQDN Generation: Enabled Disabled

129892

Table 10-3 identifies the fields and buttons shown in Figure 10-2. In many cases, the parameters that appear on this page also appear in other default pages.

Table 10-3 Configure ATA 186 Defaults Page

Field or Button	Description
Extension Point	Identifies the extension point to execute when generating a configuration for a device of this technology.
Disruption Extension Point	Identifies the extension point to be executed to disrupt a device of this technology.
Service Level Selection Extension Point	Identifies the extension used to determine the DHCP criteria and Class of Service required for a device.
Default Class of Service	Identifies the current default class of service for a specific device technology, in this example, ATA186. New, unrecognized devices of that technology type will be assigned to this class of service. Use the drop-down list to select a new default value.
Default DHCP Criteria	Identifies the current default DHCP criteria for a specific device technology, in this example, ATA186. New, unrecognized devices of that technology type will have this default DHCP criteria assigned. Use the drop-down list to select a new default value.

Table 10-3 Configure ATA 186 Defaults Page (continued)

Field or Button	Description (continued)
Automatic FQDN Generation	Automatically generates a host and domain name for the device. Two selectable options are available: <ul style="list-style-type: none"> • Enabled—Automatic generation of the FQDN is enabled. • Disabled—Automatic generation of the FQDN is disabled. Note See the “Automatic FQDN Generation” section on page 10-35 for additional information.
Submit	Activates the changes you have made. After the administrative database has been updated the Configure Defaults page will reflect the changes you have made.
Reset	Returns all settings to their previous setting.

ATA 188 Defaults

The Cisco ATA 188 interfaces regular telephones with IP-based ethernet telephony networks. The ATA 188 provides true, next-generation voice-over-IP (VoIP) terminations to support the needs of the enterprise, small-office environments, and emerging VoIP managed voice services and local services market.

The Configure ATA 188 Defaults page displays a list of default values currently available to support the ATA 188. The default parameters displayed for the ATA 188 are identical to those displayed for the ATA 186 although the values you select could be different.

CableHome WAN Defaults

There are two distinct CableHome WAN default screens; one for WAN-Data devices and one for WAN-Man devices.

WAN-Data devices, which are entirely dependent on their corresponding WAN-Man devices, are not unlike computers operating in the promiscuous mode, relative to their cable modems. A WAN-Data device is simply a MAC address and an IP address.

In either case, select the desired default from the displayed list to display the appropriate page. Each WAN default page contains the same fields and buttons, as identified in [Table 10-4](#).

Table 10-4 Configure WAN MAN /WAN Data Defaults Page

Field or Button	Description
Extension Point	Identifies the extension point to execute when generating a configuration for a WAN device.
Disruption Extension Point	Identifies the extension point to be executed to disrupt a WAN device.
Service Level Selection Extension Point	Identifies the extension used to determine the DHCP criteria and Class of Service required for a device.

Table 10-4 Configure WAN MAN /WAN Data Defaults Page (continued)

Field or Button	Description
Default Class of Service	Identifies the current default class of service for a WAN-Data. New, unrecognized WAN devices are assigned to this class of service. Use the drop-down list to select a new default value.
Default DHCP Criteria	Identifies the current default DHCP criteria for a specific device technology. New, unrecognized WAN devices are assigned this default DHCP criteria. Use the drop-down list to select a new default value.
Automatic FQDN Generation	Automatically generates a host and domain name for the device. Two selectable options are available: <ul style="list-style-type: none"> • Enabled—Automatic generation of the FQDN is enabled. • Disabled—Automated FQDN generation is disabled. <p>Note See the “Automatic FQDN Generation” section on page 10-35 for additional information.</p>

CableHome WAN Data Defaults

When you select the CableHome wide area network (WAN) Data defaults link, the CableHome Defaults page (see Figure 10-3) appears. Use this page to configure the WAN-Data device type.

Figure 10-3 Configure CableHome WAN-Data Defaults Page

The screenshot shows the 'Broadband Access Center for Cable' interface. The top navigation bar includes 'Logout', 'Configuration', 'Devices', 'Nodes', 'Servers', and 'Users'. Below this, a secondary navigation bar lists 'Class of Service', 'Custom Property', 'Defaults', 'DHCP Criteria', 'External Files', 'License Keys', and 'Publishing'. The user is logged in as 'admin' with the role of 'Administrator'.

The main content area is titled 'Configure Defaults' and includes a note: 'Use this page to change the defaults. Fields marked with an "*" are required.' A left sidebar lists various default configuration categories, with 'CH WAN DATA Defaults' selected.

The 'CableHome WAN DATA Defaults' configuration form contains the following fields:

- Extension Point:
- Disruption Extension Point:
- Service Level Selection Extension Point:
- Default Class of Service:
- Default DHCP Criteria:
- Automatic FQDN Generation: Enabled, Disabled

At the bottom of the form are 'Submit' and 'Reset' buttons. A vertical label '100003' is visible on the right side of the page.

CableHome WAN-Man Defaults

When you select the CableHome WAN-Man defaults link, the CableHome WAN-Man Defaults page (see Figure 10-4) appears. Use this page to configure the WAN-Man device type.

Figure 10-4 Configure CableHome WAN-Man Defaults Page

The screenshot displays the 'Configure Defaults' page for 'CableHome WAN MAN Defaults'. The page header includes the Cisco Systems logo and navigation tabs for Configuration, Devices, Nodes, Servers, and Users. The 'Defaults' tab is active, and the user is logged in as 'admin' with the role of 'Administrator'. The main content area contains the following configuration fields:

CableHome WAN MAN Defaults	
Extension Point:	<input type="text" value="com.cisco.csrc.extensions.CableHomeWanMan"/>
Disruption Extension Point:	<input type="text" value="com.cisco.csrc.extensions.CableHomeWanMan"/>
Service Level Selection Extension Point:	<input type="text" value="com.cisco.provisioning.cpe.extensions.builtin.sel"/>
Default Class of Service:	<input type="text" value="unprovisioned-computer"/>
Default DHCP Criteria:	<input type="text" value="unprovisioned-computer"/>
Automatic FQDN Generation:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled

At the bottom of the configuration area are 'Submit' and 'Reset' buttons. A sidebar on the left lists other default configuration pages, and the Cisco Systems logo is at the top left. The page number '129894' is visible on the right side.

Computer Defaults

The Computer Defaults page (Figure 10-5) displays a list of default values currently applied to the computers supported by BACC.

Figure 10-5 Configure Computer Defaults Page

The screenshot shows the BACC Configuration page. The top navigation bar includes 'Logout', 'Configuration', 'Devices', 'Nodes', 'Servers', and 'Users'. Below this, a secondary navigation bar lists 'Class of Service', 'Custom Property', 'Defaults', 'DHCP Criteria', 'External Files', 'License Keys', and 'Publishing'. The user information 'User:admin Role:Administrator' is displayed in the top right.

The main content area is titled 'Configure Defaults' and includes the instruction: 'Use this page to change the defaults. Fields marked with an "*" are required.' A left-hand menu lists various default categories, with 'Computer Defaults' selected.

The 'Computer Defaults' configuration form contains the following fields and controls:

- Extension Point:
- Disruption Extension Point:
- Service Level Selection Extension Point:
- Default Class of Service:
- Default DHCP Criteria:
- Automatic FQDN Generation: Enabled, Disabled

At the bottom of the form are 'Submit' and 'Reset' buttons. A vertical ID number '129895' is visible on the right side of the page.

Refer to Table 10-3 for the description of all fields and buttons appearing in Figure 10-5.



Note

Changes to the default Class of Service and default DHCP Criteria cause regeneration to occur. Any other changes made to this page will not affect the current devices.

DOCSIS Defaults

When the DOCSIS Defaults option is selected, the DOCSIS Defaults page appears. This page (Figure 10-6) displays a list of default DOCSIS values currently applied to cable modems supported by BACC.

Figure 10-6 Configure DOCSIS Defaults Page

Broadband Access Center for Cable Logout

Configuration | Devices | Nodes | Servers | Users
 Class of Service | Custom Property | **Defaults** | DHCP Criteria | External Files | License Keys | Publishing
 User:admin Role:Administrator

CISCO SYSTEMS **Configure Defaults**
 Use this page to change the defaults.
 Fields marked with an "*" are required.

Defaults

- [ATA 186 Defaults](#)
- [ATA 188 Defaults](#)
- [CH WAN DATA Defaults](#)
- [CH WAN MAN Defaults](#)
- [Computer Defaults](#)
- [DOCSIS Defaults](#)
- [NR Defaults](#)
- [Packet Cable Defaults](#)
- [RDU Defaults](#)
- [System Defaults](#)
- [XGCP Defaults](#)

DOCSIS Defaults

Extension Point:	<input type="text" value="com.cisco.csrc.extensions.DOCSISExtension"/>
Disruption Extension Point:	<input type="text" value="com.cisco.csrc.extensions.DOCSISDeviceDisrup"/>
Service Level Selection Extension Point:	<input type="text" value="com.cisco.provisioning.cpe.extensions.builtin.sel"/>
Default Class of Service:	<input type="text" value="unprovisioned-docsis"/>
Default DHCP Criteria:	<input type="text" value="unprovisioned-docsis"/>
TFTP Modem Address Option:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
TFTP Time Stamp Option:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
Automatic FQDN Generation:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
CMTS Shared Secret:	<input type="text" value="*****"/>
CMTS Default Docsis Version:	<input type="text" value="1.0"/>
Relay Agent IP Address to CMTS Version Mapping file:	<input type="text"/>

129896

Refer to Table 10-5 for the description of all fields and buttons appearing in Figure 10-6.

**Note**

Changes to the default Class of Service and default DHCP Criteria cause regeneration to occur. Changes to any TFTP option come into effect starting from the next TFTP transfer.

Table 10-5 identifies the fields and buttons that are unique to this defaults page.

Table 10-5 Configure DOCSIS Defaults Page

Field or Button	Description
Extension Point	Identifies the extension point to execute when generating a configuration for a DOCSIS device.
Disruption Extension Point	Identifies the extension point to be executed to disrupt a DOCSIS device.
Service Level Selection Extension Point	Identifies the extension used to determine the DHCP criteria and Class of Service required for a device.
Default Class of Service	Identifies the current default class of service for a device. New, unrecognized devices are assigned to this class of service. Use the drop-down list to select a new default value.
Default DHCP Criteria	Identifies the current default DHCP criteria for a specific device technology. New, unrecognized devices are assigned this default DHCP criteria. Use the drop-down list to select a new default value.
TFTP Modem Address Option	Identifies whether the TFTP modem address option is enabled.
TFTP Time Stamp	Identifies whether the TFTP server will issue a time stamp.
Automatic FQDN Generation	Automatically generates a host and domain name for the device. Two selectable options are available: <ul style="list-style-type: none"> Enabled—Automatic generation of the FQDN is enabled. Disabled—Automated FQDN generation is disabled. <p>Note See the “Automatic FQDN Generation” section on page 10-35 for additional information.</p>
CMTS Shared Secret	Identifies the character string that BACC uses in the calculation of the CMTS MIC in the configuration file. The CMTS uses it to authenticate the configuration file that a cable modem submits to the CMTS for authorization.
CMTS Default Docsis Version	Specifies the default DOCSIS version used by all CMTSs. If you do not enter a DOCSIS version in this field, it will default to version 1.0.
Relay Agent IP Address to CMTS Version Mapping file	Identifies the mapping file used by the CMTS. This file specifies the DOCSIS version that the CMTS will use.

**Note**

If you enable either or both of the TFTP options on this page, that appropriate TFTP information will be included in the TFTP file before it is sent to the DOCSIS cable modem.

Network Registrar Defaults

BACC provides Network Registrar (NR) extension points that allow BACC to pull information from an incoming DHCP packet(s) to detect a device's technology. They also let BACC respond to device DHCP requests with options that correspond to the configuration stored at the DPE.

When the NR Defaults option is selected, the NR Defaults page (see [Figure 10-7](#)) appears.

Figure 10-7 Configure Network Registrar Defaults Page

Refer to [Table 10-6](#) for the description of all fields and buttons appearing in [Figure 10-7](#).

Table 10-6 Configure Network Registrar Defaults Page

Field or Button	Description
NR Attributes from Request Dictionary (for 2.0 Extensions)	Identifies a comma-separated list of attributes pulled from the Network Registrar request dictionary, as strings, when sending a request to the RDU to generate a configuration for the current device. Note This property applies only to the BPR 2.0 Network Registrar extensions.
NR Attributes from Request Dictionary as Bytes (for 2.5 Extensions)	Identifies a comma-separated list of attributes pulled out of the Network Registrar request dictionary as bytes when sending a request to the RDU to generate a configuration for the current device. Note This property applies only to the BACC 2.5 (or higher) Network Registrar extensions.

Table 10-6 *Configure Network Registrar Defaults Page (continued)*

Field or Button	Description
NR Attributes from Request Directory as Strings (for 2.5 Extensions)	Identifies a comma-separated list of attributes pulled from the Network Registrar request dictionary as strings when sending a request to the RDU to generate a configuration for the current device. Note This property applies only to the BACC 2.5 (or higher) Network Registrar extensions.
NR Attributes from Environment Directory	Identifies a comma-separated list of attributes pulled out of the Network Registrar environment dictionary as strings when sending a request to the RDU to generate a configuration for the current device. Note This property applies to both BPR 2.0 and BACC 2.5 (or higher) Network Registrar extensions.
Submit	Activates or implements the changes you have made. After the administrative database has been updated to reflect the changes you make, modified changes appear in the Configure Defaults page.
Reset	Returns all settings to their previous setting.

**Note**

Changes made to this page do not take effect until the Network Registrar extensions are reloaded.

PacketCable Defaults

The PacketCable Defaults page identifies those defaults necessary to support the PacketCable voice technology. When selected the PacketCable Defaults page (see [Figure 10-8](#)) appears.

Figure 10-8 Configure PacketCable (Voice Technology) Defaults Page

CISCO SYSTEMS | **Configure Defaults**
Use this page to change the defaults.
Fields marked with an ** are required.

Defaults

- ATA 186 Defaults
- ATA 188 Defaults
- CHWAN DATA Defaults
- CHWAN MAN Defaults
- Computer Defaults
- DOCSIS Defaults
- NR Defaults
- Packet Cable Defaults
- RDU Defaults
- System Defaults
- XGCP Defaults

Packet Cable Defaults

Extension Point:

Disruption Extension Point:

Service Level Selection Extension Point:

Default Class of Service:

Default DHCP Criteria:

SNMP Set Timeout (secs):

MTA Provisioning Notification:

Automatic FQDN Generation: Enabled Disabled

129991

[Table 10-7](#) identifies the fields and buttons that are unique to this defaults page.

Table 10-7 Configure PacketCable (Voice Technology) Defaults Page

Field or Button	Description
Extension Point	Identifies the extension point to execute when generating a configuration for a device of this technology.
Disruption Extension Point	Identifies the extension point to be executed to disrupt a device of this technology.
Service Level Selection Extension Point	Identifies the extension used to determine what DHCP criteria and Class of Service required for a device.
Default Class of Service	Identifies the current default class of service for a device. New, unrecognized devices are assigned to this class of service. Use the drop-down list to select a new default value.

Table 10-7 *Configure PacketCable (Voice Technology) Defaults Page (continued)*

Field or Button	Description
Default DHCP Criteria	Identifies the current default DHCP criteria for a specific device technology. New, unrecognized devices are assigned this default DHCP criteria. Use the drop-down list to select a new default value.
SNMP Set Timeout	Identifies the SNMP set timeout in seconds.
MTA Provisioning Notification	Notification that an MTA event has taken place. An event occurs when the MTA sends its provisioning complete inform based on the selected choice. Options available include: <ul style="list-style-type: none"> • On Failure • On Success • During Provisioning • Always • Never
Automatic FQDN Generation	Identifies whether a fully qualified domain name (FQDN) will be generated.

RDU Defaults

When you select the RDU defaults link, the RDU Defaults page (see [Figure 10-9](#)) appears. Use this page to configure the RDU to communicate with Network Registrar. See the *Cisco CNS Network Registrar User's Guide* for additional information.

Figure 10-9 Configure RDU Defaults Page

The screenshot shows the 'Broadband Access Center for Cable' web interface. The top navigation bar includes 'Logout', 'Configuration', 'Devices', 'Nodes', 'Servers', and 'Users'. Below this, a secondary navigation bar lists 'Class of Service', 'Custom Property', 'Defaults', 'DHCP Criteria', 'External Files', 'License Keys', and 'Publishing'. The user is logged in as 'admin' with the role of 'Administrator'.

The main content area is titled 'Configure Defaults' and includes the instruction: 'Use this page to change the defaults. Fields marked with an "*" are required.' A left-hand sidebar lists various default configuration categories, with 'RDU Defaults' highlighted in red.

The 'RDU Defaults' configuration form contains the following fields and options:

Configuration Extension Point:	<input type="text" value="com.cisco.csrc.extensions.CommonExtension"/>
Device Detection Extension Point:	<input type="text" value="com.cisco.csrc.extensions.DeviceDetectionEP"/>
Publishing Extension Point:	<input type="text" value="com.cisco.support.extensions.publishing.Device"/>
Default Device Type For Device Detection:	<input type="text" value="None"/>
Extension Point Jar File Search Order:	<input type="text" value="changeloggers.jar,removetimeservers.jar"/>
CCM Server IP Address:	<input type="text"/>
CCM Server Port:	<input type="text" value="1244"/>
CCM Server User:	<input type="text" value="admin"/>
CCM Server Password:	<input type="text"/>
CCM Server Confirm Password:	<input type="text"/>
CCM Server:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
CCM Server Timeout (secs):	<input type="text" value="60"/>

At the bottom of the form are 'Submit' and 'Reset' buttons. A vertical ID number '129884' is visible on the right side of the page.

Table 10-8 describes all fields and buttons appearing in Figure 10-9.

Table 10-8 Configure RDU Defaults Page

Field or Button	Description
Configuration Extension Point	Identifies the common extension points executed before any other technology extension point is executed.
Device Detection Extension Point	Identifies the extension point used to determine a device's type (for example DOCSIS or computer) based on information pulled from the device's DHCP DISCOVER requests.
Publishing Extension Point	Identifies the extension point to be used for an RDU publishing plug-in. This is useful when you need to publish RDU data into another database.
Extension Point Jar File Search Order	Specifies the sequence in which the classes are searched in the Jar files that are listed in the preceding four fields.
CCM Server IP Address	Identifies the CCM server's IP address.
CCM Server Port	Identifies the CCM server port on which BACC communicates.
CCM Server User	Identifies the CCM server user name and is used in conjunction with the password fields.
CCM Server Password	Identifies the password used to authenticate the CCM Server User.
CCM Server Confirm Password	Authenticates the CCM Server Password.
CCM Server	Specifies whether the BACC interface to the CCM Server is enabled or disabled.
CCM Server Timeout (ms)	Specifies the length of time that BACC attempts to connect with the CCM Server until BACC declares the connection down.



Note

See the [Managing Regional Distribution Unit Extensions, page 10-31](#) for related information on RDU extension points.

System Defaults

When you select the Systems Defaults link, the System Defaults page (see [Figure 10-10](#)) appears.

Figure 10-10 System Defaults Page

Broadband Access Center for Cable Logout

Configuration | Devices | Nodes | Servers | Users
 Class of Service | Custom Property | **Defaults** | DHCP Criteria | External Files | License Keys | Publishing
 User:admin Role:Administrator

CISCO SYSTEMS **Configure Defaults**
 Use this page to change the defaults.
 Fields marked with an "*" are required.

Defaults

- [ATA 186 Defaults](#)
- [ATA 188 Defaults](#)
- [CHWAN DATA Defaults](#)
- [CHWAN MAN Defaults](#)
- [Computer Defaults](#)
- [DOCSIS Defaults](#)
- [NR Defaults](#)
- [Packet Cable Defaults](#)
- [RDU Defaults](#)
- [System Defaults](#)
- [XGCP Defaults](#)

System Defaults

SNMP Write Community String:

SNMP Read Community String:

Promiscuous Mode: Enabled
 Disabled

Default Provisioned Promiscuous DHCP Criteria:

Maximum Diagnostics Device Count:

MIB List:

Supplemental MIB List:

Excluded MIB Tokens:

Excluded Supplemental MIB Tokens:

129886



Note You can configure the default values using the BACC application program interface.

[Table 10-9](#) describes all fields and buttons appearing in [Figure 10-10](#).

Table 10-9 Configure System Defaults Page

Field or Button	Description
SNMP Write Community String	Identifies the default write community string for any device that may require SNMP information. The default write community string is private .

Table 10-9 Configure System Defaults Page (continued)

Field or Button	Description
SNMP Read Community String	Identifies the default read community string for any device that can read or access the SNMP MIB. The default read community string is public .
Promiscuous Mode	Identifies whether the Promiscuous mode is enabled. There are two options: <ul style="list-style-type: none"> • Enable—Enables the Promiscuous mode within BACC. • Disable—Disables the Promiscuous mode within BACC.
Default Provisioned Promiscuous DHCP Criteria	Identifies the default DHCP criteria used to provision a CPE in the Promiscuous mode, when the device that the CPE is behind does not have a CPE DHCP criteria specified.
Maximum Diagnostic Device Count	Identifies the maximum number of MAC addresses (devices) that you can troubleshoot at any one time.
MIB List	Identifies a list of MIBs used by the RDU that do not require restarting the RDU.
Supplemental MIB List	Identifies an extended list of MIBs used by the RDU.
Excluded MIB Tokens	Defines those key words, or tokens, that cannot be redefined by a MIB.
Excluded Supplemental MIB Tokens	Defines those additional key words, or tokens, that cannot be redefined by a MIB and do not appear in the Excluded MIB Tokens list.

Gateway (xGCP) Control Protocol Defaults

XGCP is a gateway control protocol that lets external call agents control gateways in a Voice over IP (VoIP) environment. The Configure XGCP Defaults page (Figure 10-11) displays a list of default values currently applied to the xGCP gateway devices supported by BACC.

Figure 10-11 Configure XGCP Page

Broadband Access Center for Cable Logout

Configuration | Devices | Nodes | Servers | Users

Class of Service | Custom Property | **Defaults** | DHCP Criteria | External Files | License Keys | Publishing

User:admin Role:Administrator

CISCO SYSTEMS | **Configure Defaults**
Use this page to change the defaults.
Fields marked with an * are required.

Defaults

- ATA 186 Defaults
- ATA 188 Defaults
- CHWAN DATA Defaults
- CHWAN MAN Defaults
- Computer Defaults
- DOCSIS Defaults
- NR Defaults
- Packet Cable Defaults
- RDU Defaults
- System Defaults
- XGCP Defaults

XGCP Defaults

Signalling Type:

Version Number:

Use old format for merit-dump string: Enabled Disabled

129862

Table 10-10 describes all fields and buttons appearing in Figure 10-11.

Table 10-10 Configure XGCP Defaults Page

Field or Button	Description
Signalling Type	Identifies the xGCP signalling type, such as: S, M, and so on.
Version Number	Identifies the xGCP version number in use.
Use old format for merit-dump string	Enables or disables the use of the old string format, which does not include the version number.



Note

Subsequent device configurations will include the changes you implement here. However, all existing configurations are not changed. To make the changes in any existing configuration, you must regenerate the configuration using the application programming interface (API).

Configuring DHCP Criteria

In BACC, DHCP criteria describe the specific criteria for a device when selecting a scope in Network Registrar. For example, a DHCP criteria called **provisioned-docsis** has an inclusion selection tag called **tagProvisioned**. The DHCP criteria is associated with a DOCSIS modem. When this modem requests an IP address from the Network Registrar, Network Registrar looks for scopes associated with the scope selection tag **tagProvisioned**.

To access the DHCP Criteria page:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **DHCP Criteria** from the Secondary Navigation bar and the Manage DHCP Criteria page appears.
-

Adding DHCP Criteria

To add a DHCP criteria:

-
- Step 1** Click **Add**, on the DHCP Criteria page, and the Add DHCP Criteria page appears.
 - Step 2** Enter the name of the DHCP criteria you want to create.
 - Step 3** Enter the DHCP Criteria client-class name.
 - Step 4** Enter the inclusion and exclusion selection tags.



Note

When creating new DHCP criteria, the client-class and Inclusion and Exclusion selection tag names you enter must be the exact names from within Network Registrar. Refer to the *Network Registrar User's Guide* and the *Network Registrar CLI Reference* for additional information about client-class and selection tags. You should specify either the Client Class, Inclusion Selection Tag or Exclusion Selection Tag names when creating a new DHCP criteria.

- Step 5** You can add or modify the properties that are added on the DHCP criteria. Enter or select a Property Name, or select an existing name, and enter or modify the appropriate Property Value.
 - Step 6** Click **Add** after changing or creating the property name-property value pair.
 - Step 7** Click **Submit**. After the DHCP criteria is successfully added in the RDU database, it will be visible in the Manage DHCP Criteria Page.
-

Modifying DHCP Criteria

To modify existing DHCP criteria:

- Step 1** On the Manage DHCP criteria page, click the DHCP criteria link that you want to modify and the Modify DHCP Criteria page appears.
 - Step 2** Make the desired changes to the client-class, inclusion and exclusion selection tags, and the property value settings.
 - Step 3** Click **Submit**. After successful modification of the DHCP criteria in the RDU Database, the Manage DHCP Criteria page appears.
-



Note

Subsequent device configurations will include the changes you implement here. All existing configurations are regenerated, although the devices on the network will not get the new configuration until they are rebooted.

Deleting DHCP Criteria

Deleting DHCP criteria using the administrator application does not delete the actual DHCP server configurations from the DHCP server. You must delete the DHCP server configurations manually. To delete an existing criteria:

- Step 1** Choose **Configuration** on the Primary Navigation bar.
- Step 2** Choose **DHCP Criteria** from the Secondary Navigation bar and the Manage DHCP Criteria page appears.
- Step 3** Click the **Delete** icon corresponding to the criteria you want to delete, and a deletion dialog box appears.
- Step 4** Click **OK** to delete the criteria or click **Cancel** to abort the operation. The Manage DHCP Criteria page appears.



Note

You can delete a DHCP criteria only if there are no devices associated with that criteria, and it is not designated as the default DHCP criteria. If a DHCP criteria has devices associated, you must associate a different DHCP criteria before deleting the criteria.

Managing External Files

Using the BACC administrative user interface, you can manage the TFTP server files or template files for dynamic generation for DOCSIS, PacketCable MTAs, and WAN-Man files, or software images for devices (see [Figure 10-12](#)). You can add, delete, replace, or export any file type, including:

- Template files—These are text files that contain either DOCSIS, PacketCable, or CableHome options and values that, when used in conjunction with a particular class of service, provide dynamic file generation.



Note Template files can be created in any text editor, but must have a `tmpl` file extension. Refer to the [“Developing Template Files”](#) section on page 12-1 for additional template information.

- Static configuration files—These files are used as a configuration file for a device. For example, a static configuration file called `gold.cm` would identify the gold DOCSIS class of service. BACC treats this file type like any other binary file.
- IOS images—These are images stored in firmware for a Cisco device. The Cisco device can upload the image to upgrade its functionality. BACC treats this file type like any other binary file.



Note [Figure 10-12](#) is displayed after clicking the Search button on the Manage External Files page.

Figure 10-12 Manage External Files Page

The screenshot shows the BACC administrative interface. At the top, there is a navigation bar with the following items: Configuration, Devices, Nodes, Servers, Users, Class of Service, Custom Property, Defaults, DHCP Criteria, External Files (selected), License Keys, and Publishing. The user is identified as 'admin' with the role of 'Administrator'. Below the navigation bar, there is a section titled 'View External Files' with the instruction 'Use this page to view an external file.' The main content area features a search bar labeled 'External File or External File wildcard' with an asterisk in the input field, a 'Page Size' dropdown set to '25', and a 'Search' button. Below the search bar are 'Delete' and 'Add' buttons. A table lists the external files with columns for checkboxes, file names, 'View' (with a magnifying glass icon), and 'Export' (with a document icon). The files listed are: bronze.cm, changeloggers.jar, gold.cm, removetimeservers.jar, unprov_packet_cable.bin, and unprov_wan_man.cfg. At the bottom left, it says 'Result Pages: 1'.

External Files	View	Export
<input type="checkbox"/> bronze.cm		
<input type="checkbox"/> changeloggers.jar		
<input type="checkbox"/> gold.cm		
<input type="checkbox"/> removetimeservers.jar		
<input type="checkbox"/> unprov_packet_cable.bin		
<input type="checkbox"/> unprov_wan_man.cfg		

129887

Table 10-11 identifies the fields and buttons shown in Figure 10-12.

Table 10-11 Manage External Files Page

Field or Button	Description
External Files	Identifies the filename. An asterisk (*) can be used as a wildcard character to allow searching for partial filenames. For example, you can enter *.cm to list all external files ending with the .cm extension. An example of an invalid wildcard is bronze*.
Page Size	Identifies the length of page to be displayed.
Search	Initiates the search for an external file with a name that matches the entry in the External Files field.
Delete	Removes any selected external file from the database.
Add	Adds a new file.
External Files list	Displays a list of external files that match the search criteria. Note The check boxes immediately to the left of any selected item in this list must be checked before it can be deleted.
View	Displays the details of the selected binary file.
Export	Exports any selected file to the client's computer.

Adding External Files

To add an existing external file:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **External Files** from the Secondary Navigation bar. The Manage External Files page appears.
 - Step 3** Click **Add** and the Add External Files page appears.
 - Step 4** Enter the **Source filename** and the **External filename**.



Note If you do not know the exact name of the source file, use the **Browse** function to navigate to the desired directory and select the file. By default, file sizes up to 12 MB are supported.

- Step 5** Click **Submit**. The Manage External Files page appears to indicate that the file has been added.
-

Viewing External Files

To view the contents of a DOCSIS or PacketCable voice technology external file:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **External Files** from the Secondary Navigation bar. The Manage External Files page appears.
 - Step 3** Search for the required file using the search field and appropriate wildcard characters.

Step 4 Click the **Details** icon corresponding to the DOCSIS, CableHome WAN-Man, and PacketCable MTA binary configuration files and a View Binary File Contents page appears. [Figure 10-13](#) identifies example binary file content and [Figure 10-14](#) identifies example Jar file content.

Figure 10-13 Example Binary File Content

Off	File Bytes	Option	Description	Value
0	030101	3	Network Access Control	On
3	041F	4	Class of Service	
5	010101	4.1	Class ID	1
8	02040001F400	4.2	Maximum Downstream Rate	128000 bits/sec
14	03040000F400	4.3	Maximum Upstream Rate	64000 bits/sec
20	040101	4.4	Upstream Channel Priority	1
23	050400000000	4.5	Guaranteed Minimum Upstream Channel Data Rate	0 bits/sec
29	06020640	4.6	Maximum Upstream Channel Transmit Burst	1600 bytes
33	070100	4.7	Class-of-Service Privacy Enable	Disabled
36	0B133082000F 060A2B060103 530102010701 020104	11	SNMP MIB Object	.iso.org.dod.internet.experim ntal.docsvDev.docsvDevMIBObjects .docsvDevNmAccessTable.docsvDevNm mAccessEntry.docsvDevNmAccessSt
57	0B1630820012 060A2B060103 530102010201 4004FFFFFFFF	11	SNMP MIB Object	.iso.org.dod.internet.experim ntal.docsvDev.docsvDevMIBObjects .docsvDevNmAccessTable.docsvDevNm mAccessEntry.docsvDevNmAccessIp

129889

Figure 10-14 Example Jar File Content

Extension Point JAR File Details

changeloggers.jar

Main attributes

Created-By	1.4.1_02 (Sun Microsystems Inc.)
Implementation-Title	"Change Loggers"
Implementation-Vendor	"Cisco Systems, Inc."
Implementation-Version	"1.0"
Manifest-Version	1.0
Name	com/cisco/support/extensions/publishing
Specification-Title	"Tracking Changes"
Specification-Vendor	"General Cable, Inc."
Specification-Version	"1.0"
com.cisco.support.extensions.publishing.DeviceChangeLogger.class attributes	
Implementation-Title	"Device Change Logger"
Implementation-Version	"2.0"

129890

Replacing External Files

To replace an existing external file:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **External Files** from the Secondary Navigation bar.
 - Step 3** Select the link that corresponds to the file you want to replace from the search output list. The Replace External Files page appears. Note that the selected filename already appears on this page.
 - Step 4** Enter the path and filename of the source file to be used as a replacement for the displayed external filename.



Note If you do not know the exact name or location of the source file, use the **Browse** function to navigate to the desired directory and select the file.

- Step 5** Click **Submit**. After submitting the replacement file, a confirmation page appears to indicate that, after replacement, BACC will regenerate configurations for the affected devices.
- Step 6** Click **OK** and the Manage External Files page appears.



Note All devices using this file through a class of service are regenerated after the replacement is finished.

Exporting External Files

You can copy external files to your local hard drive using the export function.



Note The procedure described below assumes that you are using Internet Explorer. This procedure is different if you are using Netscape Navigator.

To export a file:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **External Files** from the Secondary Navigation bar.
 - Step 3** Identify the external file that you want to export.
 - Step 4** Click the **Export** icon and you are prompted to either open the file or save it.
 - Step 5** Return to the BACC user interface.
-

Deleting External Files

Complete this procedure to delete an existing external file:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **External Files** from the Secondary Navigation bar.
 - Step 3** In the **External Files** field, enter the filename of the external file that you want to modify.
 - Step 4** Click **Search**. The appropriate file will appear in the External Files list.
 - Step 5** Choose the appropriate file or files.
 - Step 6** Click **Delete**.

**Caution**

Deleting a template file that is not directly linked to a class of service, but is referenced by another template file that is linked to a class of service, will cause the configuration regeneration service to fail.

**Note**

You cannot delete a file that has a class of service associated with it. You must remove the class of service association before proceeding. See the [“Configuring the Class of Service” section on page 10-1](#) for additional information.

Managing License Keys

Software licenses are used to activate specific features or to increase the functionality of your installation. Each license is available as either a permanent license or an evaluation license.

- **Permanent**—A permanent license is purchased for use in your network environment and activates the specific features for which it is intended.
- **Evaluation**—An evaluation license enables functionality for a specific amount of time after installation. You can upgrade an evaluation license to a permanent license by entering a new permanent license number.

**Caution**

Do not attempt to deploy into a fully operational network with an evaluation license key installed. Any provisioning done using an evaluation license will be disabled when that evaluation license expires.

When you upgrade from an evaluation license to a permanent license, you do not have to re-install the software. You can perform the upgrade directly from the BACC administrator user interface; you do not have to repeat the entire installation process.

The Manage License Keys page ([Figure 10-15](#)) displays a list of licenses that have been entered for your implementation. This BACC release supports both evaluation and permanent licenses for high-speed data (DOCSIS cable modems), PacketCable MTAs, ATAs, DPEs, CableHome WAN-Man and WAN-Data devices, and computers. The status of each available license is displayed as active, expired, not installed, or identifies the expiration date.

**Note**

You can upgrade your evaluation licenses to permanent status. You can also upgrade a permanent license to increase the number of authorized devices. When you reach the limit of your number of licensed devices you cannot provision new devices, but existing devices that are already provisioned continue to receive service.

Figure 10-15 Manage License Keys Page

Broadband Access Center for Cable Logout

Configuration | Devices | Nodes | Servers | Users
 Class of Service | Custom Property | Defaults | DHCP Criteria | External Files | **License Keys** | Publishing
 User:admin Role:Administrator

CISCO SYSTEMS **Manage License Keys**
 Use this page to manage your license keys for the BACC technologies.

Technology	License Key	Version	Type	Devices	Status
DPE	DPEPerm242005	2.0.0	Permanent	20	Installed on May 24, 2005
docsis	docsisPerm242005	2.0.0	Permanent	100000000	Installed on May 24, 2005
packetcable	packetcablePerm242005	2.0.0	Permanent	100000000	Installed on May 24, 2005

License Key:

129888

Adding and Modifying a License

To add, modify, or upgrade a license:

- Step 1** Choose **Configuration** on the Primary Navigation bar.
- Step 2** Choose **Licenses** from the Secondary Navigation bar.
- Step 3** Obtain your new license key from either your Cisco Systems, Inc. representative or the Cisco Technical Assistance Center (TAC) website. See the Preface in this guide for TAC contact information.
- Step 4** Enter the new license key in the License Key field.
- Step 5** Click **Add/Upgrade** to install the new license key. If you enter a permanent license key, it overwrites the corresponding evaluation key (if that key was installed). If you enter a license key (permanent or evaluation) for a new technology, it will appear in the technology list.

Managing Regional Distribution Unit Extensions

Creating a custom extension point is essentially a programming activity that can, when used in conjunction with the BACC administrator's user interface, allow you to expand the quantity of device types and technologies supported.

Managing extensions includes:

- [Writing a New Class, page 10-31](#)
- [Installing RDU Custom Extension Points, page 10-31](#)
- [Viewing RDU Extensions, page 10-32](#)

**Note**

You can specify multiple extension points by making them run one after another. You do this by specifying the extensions points in a comma-separated list.

Writing a New Class

This procedure is included to better illustrate the entire custom extension creation process. You can create many different types of extensions; for the purposes of this procedure the a new Publishing Extension Point is used.

To write the new class:

-
- Step 1** Create a Java source file for the custom publishing extension.
 - Step 2** Create a manifest file for the Jar file that will contain the extension class.
 - Step 3** Create the Jar file for the custom extension point. You can give the jar file any name you wish although the name given should be descriptive in nature and not be a duplicate of any other existing Jar file.
-

Installing RDU Custom Extension Points

After a Jar file is created, use the administrator's user interface to install it:

-
- Step 1** Use the [“Adding External Files” procedure on page 10-26](#) to add the new Jar file.

**Note**

Use the Browse function to locate the Jar file created in the [“Writing a New Class” procedure on page 10-31](#) and select this file as the Source File; leaving the External File Name blank assigns the same file name for both source and external. The external file name is what you will see through the Administrator's user interface.

- Step 2** Click **Submit**.

**Note**

An error message is generated if the class name does not exist within the jar file or if BACC detects any other errors.

- Step 3** Return to the RDU Defaults page and note that the newly added Jar file appears in the Extension Point Jar File Search Order field.
- Step 4** Enter the extension class name in the Publishing Extension Point field.
- Step 5** Click **Submit** to commit the changes to the RDU database.
- Step 6** View the RDU extensions to ensure that the correct extensions are loaded.

Viewing RDU Extensions

You can view the attributes of all RDU extensions directly from the View Regional Distribution Unit Details page. This page displays details on the installed extension Jar files and the loaded extension class files. See [Viewing Regional Distribution Unit Details, page 9-24](#).

Publishing Provisioning Data

BACC has the capability to publish the provisioning data it tracks to an external datastore in real time. To do this, a publishing plug-in must be developed to write the data to the desired datastore. The Manage Publishing page, shown in [Figure 10-16](#), identifies information such as the plug-in name, its current status (whether it is enabled or disabled), and switch to enable or disable it.

You can enable as many plug-ins as required by your implementation but care must be exercised because the use of publishing plug-ins can decrease system performance.



Note

BACC does not ship with any publishing plug-ins. You must create your own plug-ins and then manage them from this page. The plug-ins shown in [Figure 10-16](#) are for illustration only.

Figure 10-16 Manage Publishing Page

The screenshot shows the 'Manage Publishing' page in the Broadband Access Center for Cable. The page header includes 'Broadband Access Center for Cable' and a navigation menu with 'Configuration', 'Devices', 'Nodes', 'Servers', and 'Users'. The 'Publishing' tab is selected. Below the header, the page title is 'Manage Publishing' and the instruction is 'Use this page to manage (enable, disable or modify) publishing plug-ins.' The main content is a table with the following data:

Plug-In	Current Status	Enable/Disable Plug-in
TestPublisher	Enabled	[Disable plug-in]
TestPublisher	Disabled	[Enable plug-in]
TestPublisher	Enabled	[Disable plug-in]
TestPublisher	Enabled	[Disable plug-in]
TestPublisher	Enabled	[Disable plug-in]
TestPublisher	Enabled	[Disable plug-in]

129891

Publishing Datastore Changes

To publish changes to an external datastore:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **Publishing** from the Secondary Navigation bar. The Manage Publishing page appears as shown in [Figure 10-16](#). This page displays a list of all available database plug-ins and identifies the current status of each.
 - Step 3** Click on the appropriate status indicator to enable or disable the required plug-in. Note that as you click the status, it toggles from enabled to disabled. (See [Figure 10-16](#).)
-

Modifying Publishing Plug-In Settings

These settings are a convenient way for plug-in writers to store plug-in settings in the RDU for their respective datastore. To modify the publishing plug-in settings:

-
- Step 1** Choose **Configuration** on the Primary Navigation bar.
 - Step 2** Choose **Publishing** from the Secondary Navigation bar and the Manage Publishing page appears.
 - Step 3** Click the link corresponding to the plug-in you want to modify. The Modify Publishing Plug-Ins page appear.

[Table 10-12](#) identifies the fields shown in the Modify Publishing Plug-Ins page.

Table 10-12 Modifying Publishing Plug-ins Page

Field or Button	Description
Plug-In	Identifies publishing plug-in name.
Server	Identifies the server name on which the data store resides.
Port	Identifies the port number on which the data store resides.
IP Address	Identifies the IP address of the server on which the data store resides. This is usually specified when the server name is not used.
User	Identifies the user to allow access to the data stored.
Password	Identifies the user's password which allows access to the data stored.
Confirm Password	This is used to confirm the password entered above.

- Step 4** Enter the required values in the Server, Port, IP Address, User, Password, and Confirm Password fields. These are all required fields and you must supply this information before proceeding.
 - Step 5** Click **Submit** to make the changes to the selected plug-in, or click **Reset** to clear all fields on this page.
-

Configuring the SRV Record in the Network Registrar DNS Server

You must configure the Network Registrar DNS server to operate with the KDC. Refer to your Network Registrar documentation, and these instructions, to perform this configuration.



Note

It is recommended that you create a zone name that matches the desired realm name, and that the only DNS record in this special zone (other than the records required by the DNS server to maintain the zone) should be the SRV record for the realm. This example assumes that the desired Kerberos realm is voice.acme.com, and that all other KDC, Network Registrar, and DPE configuration has been performed. The FQDN of the KDC is assumed to be kdc.acme.com.

Step 1 Start the **nrcmd** CLI (usually located under /opt/nwreg2/local/usrbin), and enter your username and password.

Step 2 Enter this command to create a zone for the Kerberos realm:

```
nrcmd> zone voice.acme.com create primary <address of nameserver> hostmaster
```

Step 3 Enter this command to add the SRV record to the new zone:

```
nrcmd> zone voice.acme.com. addRR _kerberos._udp. srv 0 0 88 <address of KDC>
```

Step 4 Enter these commands to save and reload the DNS server:

```
nrcmd> save
nrcmd> dns reload
```

Configuring SNMPV3 Cloning on the RDU and DPE for Secure Communication with PacketCable MTAs

BACC lets you enable an external network manager for SNMPV3 access to MTA devices. Additionally, the RDU is capable of performing SNMPV3 operations in a specific MTA.

To enable this capability, set the security key material at the DPEs and RDU. After the key material has been set, the BACC API calls that are used to create cloned SNMPV3 entries are enabled.



Note

Enabling this capability will impact provisioning performance.

Creating the Key Material and Generating the Key

Creating the key material is a two-step process. First, run a script command on the RDU and then run a CLI command on the DPE.



Note

This shared secret is not the same shared secret as the CMTS or the BACC shared secrets.

To create the key material:

Step 1 From the <BACC_HOME>/rdu/bin directory, run this script on the RDU:

```
generateSharedSecret.sh <password>
```

Where the <password> is any password, 6 to 20 characters in length, that you create. This password is then used to generate a 46 byte key. This key is stored in a file, called keymaterial.txt, that is located in the <BACC_HOME>/rdu/conf directory.

Step 2 Run the **packetcable snmp key-material** DPE CLI command, with the <password> used in step 1 to generate that key, on all DPEs for which this voice technology is enabled.

This generates the same 46 byte key on the DPE and ensures that the RDU and DPE(s) are synchronized and can communicate with the MTA securely.

Automatic FQDN Generation

When configuring the PacketCable voice technology, a fully qualified domain name (FQDN) must reside in the BACC database for each voice device, whenever the KDC queries the registration server for that FQDN. The BACC automatic FQDN generation feature is not limited to use by any single voice technology; it can be used by any BACC technology.

Automatically Generated FQDN Format

An automatically generated FQDN in BACC follows this format:

```
prefix<htype>-<hlen>-aa-bb-cc-dd-ee-ffsuffix.domain
```

Where:

- prefix, suffix, and domain—Identify information set using either the BACC administrators user interface or the provisioning API.



Note

In the example FQDN used here, *prefix1,6,aa-bb-cc-dd-ee-ffsuffix* is the generated host name and *domain* is the domain name.

- 1,6,aa-bb-cc-dd-ee-ff—is the device MAC address

The entry of a prefix and suffix property is optional. If you do not specify these properties, and a host name is not specified during PacketCable MTA provisioning and, if neither the prefix nor suffix property is defined in the BACC property hierarchy, the device's MAC address followed by the domain name are used as the generated FQDN.

For example:

A device with the MAC address **1,6,aa:bb:cc:dd:ee:ff** will have this FQDN generated:

```
aa-bb-cc-dd-ee-ff.domain
```

Specifically, when configuring for PacketCable and many other technologies, the domain name property must also be configured. If you do not specify a domain name while provisioning a PacketCable MTA, the BACC property hierarchy is searched and, if it is not found, the MTA is not provisioned. However, if you do specify the domain name during MTA provisioning, that domain name will be used regardless of the domain name property that is specified in the BACC property hierarchy.

Properties for Automatically Generated FQDNs

Properties can be defined at any acceptable point in the BACC property hierarchy. You can use the System Defaults, Technology Defaults, DHCP Criteria, or Class of Service to accomplish this, and you can also do this at the device level.

FQDN Validation

There are a few things to consider when entering the information that is used to generate an FQDN. These include:

- Use valid alphanumeric characters only in the generated FQDN.
- Keep the length of each label (characters between the dots in the generated FQDN) to fewer than 63 characters.
- Do not allow the overall length of the generated FQDN to exceed 254 characters.



Note

The FQDN supports host and domain names as per RFC1035.

Sample Automatic FQDN Generation

This section provides an example of creating an automatically generated FQDN.

- Step 1** Choose the appropriate class of service, and set the /fqdn/domain property value to the DNS domain for all devices using this class of service. For the purposes of this example, assume that the domain in use is **pctest.com**, and that you want to provision a set of PacketCable devices into that domain.



Note

If a domain is not specified, devices in the class of service will not receive a DHCP configuration from BACC.

- Step 2** Click **Submit**.

In this example, a device with MAC address 1,6,aa:bb:cc:dd:ee:ff will yield an automatically generated FQDN of 1-6-aa-bb-cc-dd-ee-ff.pctest.com. Additionally, the Automatic FQDN Generation field should be enabled in the device's default configuration.



Configuring and Using the Sample User Interface

The Broadband Access Center for Cable (BACC) Sample User Interface (SUI) provides you with a sample user interface for both self- and pre-provisioning. The SUI can demonstrate the basic functionality of BACC in lab testing scenarios. In full BACC deployments, SUI functionality is expected to be provided by billing, OSS, and/or workflow applications.



Caution

The SUI is not intended to be used as a deployment vehicle. It is for demonstration only.

What is the Sample User Interface?

BACC provides a sample workflow application that manages the automated provisioning of devices on the network, and an administrator interface that gives basic functions to those who manage the accounts that are maintained in BACC.

In the SUI, data is managed in two distinct ways: registration of devices on the network and the accounting of results from those registrations. For example, pages that permit the complete self-provisioning of new cable modems using credit card information, must be capable of handling the automated billing for services together with device tracking. While the SUI does not track accounting information, it allows each device to be associated with an owner identifier (Owner ID). For example, the association with an Owner ID allows objects stored in BACC to be related to external objects such as billing account systems. In this sample workflow, the Owner ID is used as the account number, but is not actually related to any external data. The Owner ID associated on device objects in BACC can be any external string used to group devices.

The SUI uses shortcuts to provide an interface that can support functions such as accounting, without actually needing an external accounting entity. Accounting information is stored on the modem object as custom properties. When viewing an account, the modem is found using the Owner ID and then the account data is retrieved from the modem.

BACC supports two distinct methods of managing devices: the Standard mode and the Promiscuous mode.

- In the Standard mode, modems and computers are tracked individually.
- In the Promiscuous mode, only the modems are tracked regardless of how many computers may exist on the other side of the modem. When Promiscuous mode is enabled, computers receive access only if they are behind a provisioned modem.

Starting and Stopping the Sample User Interface

The Sample User Interface includes subscriber and administrator interfaces. You can change the subscriber interface flow to support: pre- or self- provisioning, tracking of customers, and tracking of devices being given access. Before you use the SUI, you should examine the `sampleui.properties` file. This file contains a variety of controls that specify the behavior of the interface. Refer to the [“Sample sampleui.properties File” section on page 11-9](#) to see the default `sampleui.properties` file.

You can open this file, and change its content to perform the functions you desire, using any text editor. After your changes are complete and saved, restart the SUI and all changes will take affect.

Run this command to start the SUI:

```
startSampleUI.sh
```

Run this command to stop the SUI:

```
stopSampleUI.sh
```



Note

Both these commands are located in the `<BACC_HOME>/rdu/sampleUI/bin` directory.

Sample User Interface Configuration Options

You can configure the SUI using the options described in this section. Modifying these options forces the SUI to behave in different flows. The intention of this is to represent the majority of your requirements. These options are either controlled by settings existing in BACC or are defined in the `sampleui.properties` file. See the [“Sample sampleui.properties File” section on page 11-9](#) for additional information.

Classes of Service

Classes of service are defined in the interfaces configuration file and also in the normal service definition within BACC. The classes of service within the SUI also reference the intended DHCP criteria to be used for the devices and have a description for presentation in the interface. For example, if you chose a class of service called Blue, the SUI could translate that into a BACC class of service called Gold and a DHCP criteria called residential-provisioned. When the SUI is started, it would attempt to verify that the referenced classes of service are already defined.

Promiscuous Mode

The Promiscuous mode is defined as the behavior involving how computers are tracked. When the Promiscuous mode is enabled, a computer automatically receives a provisioned configuration when it is plugged in behind a provisioned cable modem.

When the Promiscuous mode is enabled, computers are not asked for registration information. However, when this mode is disabled (a situation known as the Standard mode), the SUI lets users register their computers. This includes the optional selection of an Internet service provider (ISP) for each computer. This is maintained within BACC and can be accessed from the RDU Defaults page in the administrator’s user interface.

**Note**

You must restart the SUI after you change the Promiscuous mode.

Selecting an Internet Service Provider

You can select an Internet service provider (ISP) individually, for each computer registered with an account, whenever the Standard mode PC registration (non-Promiscuous) mode is used. Selecting an ISP has the same affect as choosing the DHCP criteria assigned to the computer. This setting is configured within the interface's configuration file. If there is a single ISP, the ISP-selection controls are bypassed when moving through the subscription interface.

Using the Technician Login

You can use the technician login to demonstrate a provisioning flow whereby a technician brings a cable modem to a customer's home and plugs it in. The Technician Login page appears in which you authenticate the technician in the system before proceeding with the provisioning of the cable modem on the network.

If authentication is disabled, but technician provisioning is enabled, the demonstration will be of a self-provisioning nature.

Administrative Access Levels

The administrators that can access the SUI administration interface are configured in the interface configuration file. You can use four types of administrators within the SUI.

- full—This type has complete access to create and delete accounts and manage devices in the interface.
- createonly—This type only has access to create new accounts using the interface.
- readonly—This type only has access to view accounts that have been created in the system.
- tech—This type only has access to log in through the technician interface. This is for auto-provisioning devices from the customer premises.

Subscriber Provisioning Examples

This section describes various work flows that are presented while using the SUI. Having the Promiscuous mode enabled or disabled has a significant effect on the behavior of the SUI. Consequently, the flows in this section are identified separately.

Standard Customer Premise Equipment Registration

This section describes provisioning activities when the system is in the standard, non-Promiscuous mode of operation. These provisioning activities are discussed in these sections:

- [Provisioning a New Cable Modem and a New Computer, page 11-4](#)
- [Provisioning a New Computer with an Existing Cable Modem, page 11-4](#)
- [Altering an Existing Computer ISP, page 11-5](#)

Provisioning a New Cable Modem and a New Computer

When a new modem and new computer are connected to a network, and the user brings a web browser online, the user is redirected to the provisioning interface.

-
- Step 1** The SUI checks the `sampleui.properties` file to determine if technician provisioning is enabled:
- If this feature is enabled, the SUI continues with the next step.
 - If it is disabled, an error page appears stating that the modem is not registered and the customer should contact the MSO to register their cable modem with the system.
- Step 2** The SUI checks the `sampleui.properties` file to determine if technician authentication is required:
- If this feature is not required, the modem registration screen appears, and you can enter account details to be registered with the system.
 - If the feature is required, the modem registration page appears, and you can enter your technician username and password, and the account details to be registered with the system.
- Step 3** A computer registration page appears. You can use this page to register the computer at the same time the modem is registered. This page also identifies that the modem registration has been successful.
- Step 4** The SUI checks the `sampleui.properties` file to determine if the optional ISP selection is enabled.
- If it is enabled, a drop-down list with available ISPs appears for you to select the appropriate ISP option.
 - If it is disabled, no ISP selection is displayed.
- Step 5** Click **Register This Computer** and a message appears stating that the computer has been successfully registered with the network.
-

Provisioning a New Computer with an Existing Cable Modem

When an existing modem and new computer are connected to the network and then the user brings a web browser online, the user is redirected to the provisioning interface.

-
- Step 1** The SUI displays the computer registration page. From here, you can register the computer on the network.
- Step 2** The SUI checks the `sampleui.properties` file to determine if the optional ISP selection is enabled.
- If it is enabled, a drop-down list with available ISPs appears for you to select the appropriate ISP option
 - If it is disabled, no ISP selection is displayed.

- Step 3** Click **Register This Computer** and a message appears stating that the computer has been successfully registered with the network.
-

Altering an Existing Computer ISP

When an existing modem and existing computer are connected to the network and then the user brings a web browser online, the user can browse the network. They then direct their browser to the provisioning interface.

- Step 1** The SUI determines whether or not the optional ISP selection feature is enabled.
- If it is enabled, a drop-down list with available ISPs appears for you to select the appropriate ISP option.
 - If it is disabled, no ISP selection is displayed and a message appears stating this computer is already registered on the system.
- Step 2** Click **Register This Computer** and a message appears stating that the computer has been successfully registered with the network.
-

Promiscuous Customer Premises Equipment Registration

This section describes equipment registration using the SUI. These provisioning activities are discussed in these sections:

- [Provisioning a New Cable Modem and a New Computer, page 11-5](#)
- [Provisioning an Existing Cable Modem and a New Computer, page 11-6](#)

Provisioning a New Cable Modem and a New Computer

When a new modem and new computer are connected to the network and then the user brings a web browser online, the user is redirected to the provisioning interface.

- Step 1** The SUI checks the `sampleui.properties` file to determine if technician provisioning is enabled:
- If this feature is enabled, the SUI continues with the next step.
 - If it is disabled, an error page appears stating that the modem is not registered and that the customer should contact their MSO to register their cable modem with the system.
- Step 2** The SUI checks the `sampleui.properties` file to determine if technician authentication is required:
- If this feature is not required, the modem registration screen appears for you to enter account details to be registered with the system.
 - If the feature is needed, the modem registration page appears for you to enter your technician username and password, and the account details to be registered with the system.
- Step 3** A message appears, stating that the modem and computer have been successfully registered with the network.
-

Provisioning an Existing Cable Modem and a New Computer

When an existing modem and new computer are connected to the network and then the user brings a web browser online, the user can browse the network. Once they can browse the network, they must direct their browser to the provisioning interface. After being directed, a message appears to indicate that the cable modem and computer are registered on the system.

Administrator Provisioning Examples

This section identifies some examples that illustrate the use of the SUI in performing account maintenance and account searches. The components of each SUI page that appear only when certain permissions have been assigned, are identified with an *if applicable* note appended to the end of the component name.

Searching for Accounts

This section explains how to perform account searches using the SUI. These search activities are discussed:

- [Searching by Account Number, page 11-6](#)
- [Searching by IP Address, page 11-6](#)
- [Searching by MAC Address, page 11-7](#)

Searching by Account Number

You can search for an account, using an account number, after logging in. You specify the account number to search for and, if found, the account details are displayed. If the account is not found, an error message appears stating that the account number does not exist in the system.

Searching by IP Address

You can search for an account, using an IP address, after logging in. You specify the IP address of a computer or modem currently provisioned by BACC and, if found, the owner is checked for the device to determine what account should be displayed.

If a valid account is found, full account details for the device are displayed. If a valid device is found, but not a valid account, the current device's MAC address and IP address appear at the bottom of the search page.

If the device cannot be found, an error is displayed indicating the search did not find a matching device.

Searching by MAC Address

You can search for an account, using a MAC address, after logging in. You specify the MAC address of a computer or modem currently provisioned by BACC and, if found, the owner is checked for the device to see what account should be displayed.

If a valid account is found, full account details for the device are displayed. If a valid device is found, but not a valid account, the current MAC address and IP address for the device are shown at the bottom of the search page.

If the device cannot be found, an error is displayed indicating the search did not find a matching device.

Account Maintenance

This section explains how to perform account maintenance using the SUI. These maintenance activities are discussed in these sections:

- [Registering a New Account, page 11-7](#)
- [Managing Classes of Service, page 11-7](#)
- [Managing Cable Modems, page 11-8](#)
- [Managing Computers, page 11-8](#)

Registering a New Account

You can use this workflow to register a new account, and modem, with the SUI. You must log into the system and then:

-
- Step 1** The SUI displays a page that shows the search options (if applicable) to choose from. This page also contains the **Create a new Account** button.
 - Step 2** Click **Create a new Account**.
 - Step 3** The SUI displays a page that lets you enter the account number, MAC address of the cable modem, and the class of service applicable to the account.
 - Step 4** After entering and submitting the desired information, the account is created.
 - Step 5** After the account is created, a new account creation page appears. This lets you rapidly enter multiple accounts into the system.
-

Managing Classes of Service

You can use this workflow to change the class of service on an account and disable the cable modem. You must log into the system and then:

-
- Step 1** The SUI displays a page that shows the search options (if applicable) to choose from. You can search for the account by using the account number, IP address, or MAC address as the search criteria.
 - Step 2** The SUI displays a page that contains all the information on the account, including the currently selected class of service, whether the cable modem is enabled, the account owner information, and the list of registered computers.

- Step 3** Select the appropriate class of service from the drop-down menu.
 - Step 4** Click **Update**.
 - Step 5** The SUI redisplay the same page containing all the account information. However, on this page the class of service will have been changed.
-

Managing Cable Modems

You can use this workflow to change the modem that is currently associated with an account. The account details such as user name can also be updated following this workflow. You must log into the system and then:

- Step 1** The SUI displays a page that shows the search options (if applicable) to choose from. You can search for the account by using the account number, IP address, or MAC address as the search criteria.
 - Step 2** The SUI displays a page that contains all the information on the account, including the currently selected class of service, whether the cable modem is enabled, the account owner information, and the list of registered computers.
 - Step 3** Enter the new MAC address for the account's cable modem.
 - Step 4** Click **Update**.
 - Step 5** The SUI redisplay the same page containing all the account information. However, on this page the MAC address will have been changed.
-

Managing Computers

You can use this workflow to unregister computers that were previously registered using the Subscriber portion of the SUI. This workflow applies only when Standard mode PC registration is used. You must log into the system and then:

- Step 1** The SUI displays a page that shows the search options (if applicable) to choose from. You can search for the account by using the account number, IP address, or MAC address as the search criteria.
 - Step 2** The SUI displays a page that contains all the information on the account, including the currently selected class of service, whether the cable modem is enabled, the account owner information, and the list of registered computers.
 - Step 3** Determine which computer you want to unregister and click the appropriate **Delete** button.
 - Step 4** The SUI redisplay the same page containing all the account information, however, the computer has been removed from the list.
-

Deleting an Account

You can use this workflow to delete an account that was registered using the SUI. You must log into the system and then:

-
- Step 1** The SUI displays a page that shows the search options (if applicable) to choose from. You can search for the account by using the account number, IP address, or MAC address as the search criteria.
 - Step 2** The SUI displays a page that contains all the information on the account, including the currently selected class of service, whether the cable modem is enabled, the account owner information, and the list of registered computers.
 - Step 3** Click **X**, at the Delete account field from the system prompt.
 - Step 4** The SUI displays the same page containing all the account information. A prompt appears, next to the Delete button, asking for confirmation before proceeding.
 - Step 5** Click **X**, at the Delete account field from the system prompt, again to confirm deletion of the account.
 - Step 6** The SUI displays the original search page showing that the account has been deleted.
-

Sample sampleui.properties File

This section identifies the contents of a sample sampleui.properties file. This properties file is located in the <BACC_HOME>/rdu/conf directory.

```
# (C) Copyright 2001-2003 by Cisco Systems, Inc.
# This program contains proprietary and confidential information.
# All rights reserved. This software shall not be used by an party
# except by prior written consent of Cisco Systems.
#
# DO NOT CHANGE. This is the version of the properties file which
# is used during execution and for system updates.
#
version=1.2
#####
# System connection information.
#####
#
# BPR RDU connection information
#
adminuser=admin
adminpass=admin
host=localhost
port=49187
#####
# Provisioning configuration parameters
```

```
#####
#
# Administrator access credentials
#
# 3 levels of access: full, createonly, and readonly
# full -- can create and read accounts
# createonly -- can only create new accounts
# readonly -- can only read accounts
# tech -- technician access for auto-provisioning modems
# (The default access level is 'full')
#
# The user.number must equal the number of user accounts
# being tracked. If the number is 4, there must exist
# entries for users 1-4.
#
user.number=4
user.1.name=admin
user.1.password=changeme
user.1.access=full
user.2.name=config
user.2.password=changeme
user.2.access=createonly
user.3.name=monitor
user.3.password=changeme
user.3.access=readonly
user.4.name=tech
user.4.password=changeme
user.4.access=tech
#
# Indicates if promiscuous-mode is enabled
#
# promiscuous-mode is a special mode of BPR in which computers are not
# tracked by the provisioning system. (true/false)
#
# This is controled by setting the RDU Defaults option:
#   ModemKeys.PROMISCUOUS_MODE_ENABLED
#
# Indicates if technician auto-provisioning is enabled
#
# This mode allows a technician to provision a new account in the field
# without requiring the MAC address to be pre-registered.
#
techprovisioning.enabled=true
#
```

```
# Indicates if technician username/password is required for modem
# registration
#
# If technician username/password is not required, then this
# demo can be used to simulate modem self-registration
#
techprovisioning.authentication=false
#
# Unprovisioned configuration (for disabled modems)
#
# The client class should match the unprovisioned client class configured
# in CNR. The service must be unique (i.e. NOT have the same name of
# any of the services specified below).
#
# These values are controlled by setting the DOCSIS Defaults options, use:
#     TechnologyDefaultsKeys.DOCSIS_DEFAULT_CLASS_OF_SERVICE
#     TechnologyDefaultsKeys.DOCSIS_DEFAULT_DHCP_CRITERIA
#     TechnologyDefaultsKeys.COMPUTER_DEFAULT_DHCP_CRITERIA
#
# Classes of Service for Modems
#
# The DHCP criterias specified here must match valid
# DHCP criterias specified in the RDU. If promiscuous
# mode is enabled, you must specify CPE DHCP criterias.
#
# The service.number must equal the number of services
# being tracked. If the number is 3, there must exist
# entries for services 1-3.
#
service.number=3
service.1.name=gold
service.1.title=1.5Mb/s Lightning Fast!
service.1.dhcpcriteria=ccProvisionedDOCSISModem
service.1.cpedhcpcriteria=provcpetagProvisionedPromiscuousCpe
service.2.name=silver
service.2.title=512Kb/s Power User
service.2.dhcpcriteria=ccProvisionedDOCSISModem
service.2.cpedhcpcriteria=provcpetagProvisionedPromiscuousCpe
service.3.name=bronze
service.3.title=64kb/s Economy Service
service.3.dhcpcriteria=ccProvisionedDOCSISModem
service.3.cpedhcpcriteria=provcpetagProvisionedPromiscuousCpe
#
# ISPs for Computers
```

```
#
# The computerisp.number must equal the number of ISPs
# available. If the number is 1, there must exist
# entries for ISPs 1-1.
#
computerisp.number=1
computerisp.1.name=msonet
computerisp.1.title=MSO.net Services
computerisp.1.dhcpcriteria=ccProvisionedComputer
#
# Default COS, DHCPCriteria and CPE DHCPCriteria
# that modem and computers are placed in when modem's
# access is disabled through administrator UI
#
# Appropriate DOCSISClassOfService and DHCPCriteria objects have
# to be pre-created in the RDU
#
# disabled.modem.cpedhcpcriteria=disabled-computer -- defined a
# DHCPCriteria which computer's behind the modem get when modem's
# access is disabled
#
disabled.modem.cos=disabled
disabled.modem.dhcpcriteria=disabled-modem
disabled.modem.cpedhcpcriteria=disabled-computer
```




Broadband Access Center for Cable Support Tools and Advanced Concepts

This chapter contains information on, and explains the use of, tools that help you maintain Broadband Access Center for Cable (BACC) as well as speed and improve the installation, deployment, and use of this product.

This chapter discusses these topics:

- [Developing Template Files, page 12-1](#)
- [Using the Configuration File Utility, page 12-25](#)
- [The RDU Log Level Tool, page 12-38](#)
- [Using the PKCert.sh Tool to Manage KDC Certificates, page 12-41](#)
- [Using the changeNRProperties.sh Tool, page 12-43](#)
- [Using the Keygen Tool, page 12-45](#)
- [Using the snmpAgentCfgUtil.sh Command, page 12-46](#)
- [Using the disk_monitor.sh Tool to Monitor Available Disk Space, page 12-51](#)
- [Troubleshooting Devices by MAC Address, page 12-52](#)



Note

This section contains many examples of tool use. In many cases, the tool filenames include a path specified as <BACC_HOME>. This indicates the default directory location.

Developing Template Files

BACC uses templates to help administrators deploy dynamic PacketCable, DOCSIS, and CableHome files. Using templates, you can create a template file in an easily readable format, and edit it quickly and simply. A template is an ASCII text file that represents the PacketCable, DOCSIS, or CableHome options and values used for generating a valid PacketCable, DOCSIS, or CableHome file. BACC uses the .tmpl file extension to identify template files. You must add template files to the RDU as an external file using either the administrator's user interface or the API, before any class of service can reference it.

When installing the BACC RDU component, several sample template files are copied to the <BACC_HOME>/rdu/samples directory.

Although all that you need to create or edit a template is a simple text editor, before attempting to create your own template file, you should thoroughly familiarize yourself with this information:

- BACC provisioning flows
- DOCSIS 1.0, 1.1, and 2.0 RFI specifications
- PacketCable 1.0 and 1.1 specifications
- MTA device provisioning specification
- CableHome 1.0 specification
- SNMP MIBs for cable devices (for example, DOCS-CABLE-DEVICE-MIB)

Template Grammar

A template is comprised of four different types of statements:

- [Comments, page 12-3](#)
- [Includes, page 12-3](#)
- [Options, page 12-3](#)
- [Instance Modifier, page 12-5](#)

Comments allow you to document your templates. Includes allow you to create building block templates to be used in other templates. Options allow you to specify the PacketCable, DOCSIS, or CableHome type length value (TLV) in a descriptive manner. [Table 12-1](#) describes the available template grammar options.

Table 12-1 Template Grammar

Option	Description
<comment>	::= #[ascii-string]
<include>	::= include “<filename.tpl>”
<option-description>	::= option <option-num> [instance <instance-num>] <option-value>
<option-num>	::= <unsigned-byte>[.<unsigned-byte>]*
<option-value>	::= <well-defined-value> <custom-value>
<well-defined-value>	::= <option-value-string>[,<option-value-string>]*
<custom-value>	::= <ascii-value> <hex-value> <ip-value> <snmp-value>
<ascii-value>	::= ascii <ascii-string>
<hex-value>	::= hex <hex-string>
<ip-value>	::= ip <ip-string>
<instance-num>	::= <unsigned integer>
<template>	::= <template-statement>*
<template-statement>	::= <comment> <include> <option-description>
<snmp-value>	::= snmp <snmpvar-oid>,<snmpvar-type>,<snmpvar-value>

Comments

Comments provide information only and are always located between the pound (#) symbol and the end of a line. [Example 12-1](#) shows example comment usage.

Example 12-1 Example Comment Usage

```
#
# Template for gold service
#

option 3 1 # enabling network access
```

Includes

Include files let you build a hierarchy of similar, but slightly different, templates. This is very useful for defining options that are common across many service classes without having to duplicate the options in several templates.

You can use multiple include statements in a single template although the location of the include statement in the template is significant; the contents of the include file are included wherever the include statement is found in the template. The included template must be added as an external file to the RDU before it can be used. The included file must not contain any location modifiers such as `../..` because the templates are stored without path information in the RDU database. Examples [12-2](#) and [12-3](#) illustrate both correct and incorrect usage of the include option.

Example 12-2 Correct Include Statement Usage

```
# Valid, including common options
include "common_options.tpl"
```

Example 12-3 Incorrect Include Statement Usage

```
# Invalid, using location modifier
include "../common_options.tpl"

# Invalid, using incorrect file suffix
include "common_options.common"

# Invalid, not using double quotes
include common_options.tpl
```

Options

PacketCable, DOCSIS, and CableHome configuration files consist of properly encoded option id-value pairs. Two forms of options are supported: defined and custom.

- Well-defined options require the option number and value. The value is encoded based on the encoding type of the option number.
- Custom options require the option number, explicit value encoding type, and the value.

When using compound options, for example, option 43, you can use the instance modifier to specify the TLV groupings. See the [“Instance Modifier” section on page 12-5](#) for additional information.

When specifying one of these well-defined options in a template, it is not necessary to specify a value encoding for the value. See the “[Encoding Types for Defined Options](#)” section on page 12-8 and the “[DOCSIS Option Support](#)” section on page 12-12 for additional information on these defined encoding types.

When specifying custom options (e.g. option 43), you must specify the encoding type for the option. The available encoding types are:

- **ASCII**— ASCII type encodes any given value as an ASCII string without a NULL terminator. If the value contains spaces, they must be double quoted.
- **hex**—The value must be valid hexadecimal and there must be exactly 2 characters for each octet. If 01 is specified as the value, then exactly one octet is used in the encoding. If 0001 is specified as the value, then exactly two octets are used in the encoding process.
- **IP address**—IP address type encodes any given value as 4 octets. For example, the IP address 10.10.10.1 is encoded as 0A0A0A01.

Use a comma to separate multi-valued options on a given line. Each value is treated as such, so you might have to double quote one of the values, but not the others. A good example of a multi-valued option is Option 11 (SNMP Varbind). See the “[SNMP Varbind](#)” section on page 12-5 for additional information.

When specifying compound options, there is no need to specify the top level option (for example option 4 when specifying option 4.1). Examples 12-4 and 12-5 illustrate both correct and incorrect usage of the option statement.

Example 12-4 Correct Option Statement Usage

```
# Valid, specifying the number for well known option 3
option 3 1

# Valid, specifying the number for option 4 sub-option 1
option 4.1 1

# Valid, specifying a vendor option as hex
option 43.200 hex 00000C

# Valid, specifying a vendor option as ascii
option 43.201 ascii "enable log"

# Valid, specifying a vendor option as IP
option 43.202 ip 10.4.2.1
```

Example 12-5 Incorrect Option Statement Usage

```
# Invalid, using hex with incorrect hex separator
option 43.200 hex 00.00.0C

# Invalid, not using double quotes when needed
option 43.201 ascii enable log

# Invalid, not specifying IP address correctly
option 43.202 ip 10-10-10-1

# Invalid, specifying the description for option "Network Access Control"
option "Network Access Control" 1

# Invalid, specifying top level option
option 4
```

Instance Modifier

The instance modifier is used to group compound options into specific individual Tag-Length-Values (TLVs). Examples 12-6 and 12-7 illustrate both correct and incorrect methods of creating separate TLVs. These are required to enable the IOS DOCSIS modem to interpret the IOS commands as two separate commands.

Example 12-6 Correct IOS Command Line Entries

```
# Valid, each IOS command gets its own TLV
option 43.8 instance 1 00-00-0C
option 43.131 instance 1 ascii "login"
option 43.8 instance 2 00-00-0C
option 43.131 instance 2 ascii "password cable"
```

Example 12-7 Incorrect IOS Command Line Entries

```
# Invalid, IOS commands are grouped into one TLV
option 43.8 00-00-0C
option 43.131 ascii "login"
option 43.131 ascii "password cable"

# Invalid, using instance on non-compound options
option 3 instance 1 1
```



Note The encoding type for option 43.8 is an organizationally unique identifier (OUI). Unlike that shown in Example 12-4, this only accepts an 00-00-0C format.

SNMP Varbind

You must use an object identifier (OID), when specifying DOCSIS option 11, PacketCable option 64, or CableHome option 28. The MIB that contains the OID must be in one of the following MIBs loaded by the RDU. You must specify as much of the OID as needed to uniquely identify it. You can use the name or the number of the OID. The RDU automatically loads these MIBs:

- SNMPv2-SMI
- SNMPv2-TC
- CISCO-SMI
- CISCO-TC
- SNMPv2-MIB
- RFC1213-MIB
- IANAifType-MIB
- IF-MIB

DOCSIS MIBs

These DOCSIS MIBs are loaded into the RDU:

- DOCS-IF-MIB
- DOCS-BPI-MIB

- CISCO-CABLE-SPECTRUM-MIB
- CISCO-DOCS-EXT-MIB
- SNMP-FRAMEWORK-MIB
- DOCS-CABLE-DEVICE-MIB
- DOCS-CABLE-DEVICE-MIB-OBSOLETE
- CISCO-CABLE-MODEM-MIB

Two versions of the DOCS-CABLE-DEVICE MIB are loaded into the RDU:

- DOCS-CABLE-DEVICE-MIB-OBSOLETE (experimental branch)
- DOCS-CABLE-DEVICE-MIB (mib2 branch)

A fully-qualified MIB OID (.experimental...) always uniquely identifies a MIB OID.

If you use a non-fully qualified MIB OID from DOCS-CABLE-DEVICE-MIB, it will always default to DOCS-CABLE-DEVICE-MIB and not DOCS-CABLE-DEVICE-MIB-OBSOLETE.

Examples 12-8 and 12-9 illustrate using a fully-qualified MIB OID and a non-fully qualified MIB OID.

Example 12-8 Fully Qualified MIB OID

```
# Valid, uniquely identifying an OID
option 11 .experimental.docsDev.docsDevMIBObjects.docsDevNmAccessTable.docsDevNmAccess
Entry.docsDevNmAccessStatus.1, Integer, 4
```

Example 12-9 Non-Fully Qualified MIB OID (Defaults to DOCS-CABLE-DEVICE-MIB)

```
# Valid, Non-Fully Qualified MIB OID.
option 11 .docsDevNmAccessStatus.1, Integer, 4
```

If no DOCSIS CMs in a deployment require DOCS-CABLE-DEVICE-MIB-OBSOLETE, you can always use the shorter form of the MIB OID.

PacketCable MIBs

These PacketCable (North American) MIBs are loaded into the RDU:

- CLAB-DEF-MIB
- PKTC-MTA-MIB
- PKTC-SIG-MIB
- PKTC-EVENT-MIB

CableHome MIBs

These CableHome MIBs are loaded into the RDU:

- CABH-CAP-MIB
- CABH-CDP-MIB
- CABH-CTP-MIB
- CABH-PS-DEV-MIB

- CABH-QOS-MIB
- CABH-SEC-MIB

These additional MIBs are needed but are not part of the BACC product:

- CABH-CTP-MIB needs RMON2-MIB, TOKEN-RING-RMON-MIB
- CABH-SEC-MIB needs DOCS-BPI2-MIB.

Macro Variables

Macro variables are specified as values in templates that let you specify device-specific option values. When a macro variable is encountered in the template, the properties hierarchy is searched for the macro variable name and the value of the variable is then substituted. The variable name is a custom property, which is predefined in the RDU. It must not contain any spaces.



Note

When you use custom properties for macro variables, you must use `DataType.STRING`.

After the custom property is defined, it can be used in this property hierarchy:

- System defaults
- Technology defaults, such as PacketCable, DOCSIS, or CableHome
- DHCP criteria properties
- Class of service properties
- Device properties

The template parser works bottom up when locating properties in the hierarchy (device first, then the class of service, and so on) and converts the template option syntax. The following syntax is supported for macro variables:

- `${var-name}`—This syntax is a straight substitution. If the variable is not found, the parser will generate an error.
- `${var-name, ignore}`—This syntax lets the template parser ignore this option if the variable value is not found in the properties hierarchy.
- `${var-name, default-value}`—This syntax provides a default value if the variable is not found in the properties hierarchy.

Examples 12-10 and 12-11 illustrate both correct and incorrect usage of option 11.

Example 12-10 Correct Macro Variables Usage

```
# Valid, using macro variable for max CPE's, straight substitution
option 18 ${MAX_CPES}

# Valid, using macro variable for max CPE's, ignore option if variable not found
# option 18 will not be defined in the DOCSIS configuration file if MAX_CPES
# is not found in the properties hierarchy
option 18 ${MAX_CPES, ignore}

# Valid, using macro variable for max CPE's with a default value
option 18 ${MAX_CPES, 1}
```

```

# Valid, using macro variable for vendor option
option 43.200 hex ${MACRO_VAR_HEX}

# Valid, using macro variable for vendor option
option 43.201 ascii ${MACRO_VAR_ASCII}

# Valid, using macro variable for vendor option
option 43.202 ip ${MACRO_VAR_IP}

# Valid, using macro variable in double quotes
option 18 "${MAX_CPES}"

# Valid, using macro variable within a value
option 43.131 ascii "hostname ${HOSTNAME}"

# Valid, using macro variables in multi-valued options
option 11 ${ACCESS_CONTROL_MIB,
.mib-2.docsDev.docsDevMIBObjects.docsDevNmAccessTable.docsDevNmAccessEntry.docsDevNmAccess
Control.1}, Integer, ${ACCESS_CONTROL_VAL, 3}

# Valid, using macro variable in an include statement
include "${EXTRA_TEMPLATE}"

# Valid, using macro variable in an include statement with a default value
include "${EXTRA_TEMPLATE, modem_reset.tpl}"

# Valid, using macro variable in an include statement with a default value
include "${EXTRA_TEMPLATE, modem_reset}.tpl"

# Valid, using macro variable in an include statement with an ignore clause
include "${MY_TEMPLATE, ignore}"

```

Example 12-11 Incorrect Macro Variables Usage

```

# Invalid, using macro variable as the option number
option ${MAX_CPES} 1

# Invalid, using macro variable with space in name
option 18 ${MAX CPES}

```

Encoding Types for Defined Options

Table 12-2 identifies the options with defined encoding types.

Table 12-2 Defined Option Encoding Types

Encoding	Input	Example
Boolean	0 for false and 1 for true.	0
Bytes	A series of hexadecimal octets. Each octet must be exactly 2 characters in length.	000102030405060708
IP Address	Four unsigned integer 8, dot (.) separated.	10.10.10.1
Multiple IP Addresses	Comma separated list of IP addresses.	10.11.12.13,10.11.12.14
MAC Address	Six hexadecimal octets colon (:) or dash (-) separated. Each octet must be exactly 2 characters in length. Colons and dashes must not be mixed.	00:01:02:03:04:05 or 00-01-02-03-04-05

Table 12-2 Defined Option Encoding Types (continued)

Encoding	Input	Example
MAC Address And Mask	Twelve octets colon (:) or dash (-) separated. Each octet must be exactly 2 characters in length. Colons and dashes must not be mixed. The first six octets represent the MAC address; the last six represent the mask for the MAC address.	00:01:02:03:04:05:06:07:08:09:0A:0B or 00-01-02-03-04-05-06-07-08-09-0A-0B
NVTASCII	An ASCII string. The encoded string will not be NULL terminated.	This is an ASCII string
OID	An SNMP OID string.	sysinfo.0
OIDCF	An SNMP OID string and an unsigned integer (0 or 1) comma separated.	sysinfo.0,1
OUI	Three hexadecimal octets colon (:) or dash (-) separated. Each octet must be exactly 2 characters in length.	00-00-0C
SNMPVarBind	An SNMP OID string, type, and value. Each of these is comma separated. Valid types are: <ul style="list-style-type: none"> • BITS • Counter • Counter32 • Counter64 • Gauge • Gauge32 • INTEGER • Integer32 • IpAddress • OCTETSTRING • OBJECTIDENTIFIER • Opaque • TimeTicks • Unsigned32 <p>Note The OCTETSTRING can be either a string that will be converted to hexadecimal notation without a trailing NULL, octet string for example, or hexadecimal notation contained in single quotes, 'aa:bb:cc' for example.</p>	.experimental.docsDev.docsDevMIBObjects · docsDevNmAccessTable.docsDevNmAccessEntry.docsDevNmAccessStatus.1, INTEGER, 4
Sub Type	One or two comma separated unsigned integer 8.	12 or 12,14

Table 12-2 Defined Option Encoding Types (continued)

Encoding	Input	Example
Unsigned integer 8	0 to 255	14
Unsigned integer 16	0 to 65535	1244
Unsigned integer 32	0 to 4294967295	3455335
Unsigned integer 8 and unsigned integer 16	One unsigned integer 8 and one unsigned integer 16, comma separated.	3,12324
Unsigned integer 8 pair	Two unsigned integer 8, comma separated.	1,3
Unsigned integer 8 triplet	Three unsigned integer 8, comma separated.	1,2,3
ZTASCII	An ASCII string. The encoded string will be NULL terminated.	This is an ASCII string

BITS Value Syntax

When using the BITS type, you must specify either the labels (“interval1 interval2 interval3”) or numeric bit location (“0 1 2”). Note that label values are 1-based and bit values are 0-based.

This is the syntax that uses the bit numbers:

```
option 11 .pktcSigDevR0Cadence.0,STRING,"0 1 2 3 4 5 6 7 8 9 10 11 12 13 14"
```

This is the syntax for the customer octet string (FFFE000000000000) that uses the labels:

```
option 11 .pktcSigDevR0Cadence.0,STRING,"interval1 interval2 interval3
interval4 interval5 interval6 interval7 interval8 interval9 interval10
interval11 interval12 interval13 interval14 interval15"
```

OCTETSTRING Syntax

The OCTETSTRING can be either a string that is converted to hexadecimal notation without a trailing NULL (for example, octet string), or hexadecimal notation contained within single quotes, (for example, 'aa:bb:cc').

Adding SNMP TLVs Without a MIB

You can add SNMP TLVs in dynamic configuration files (DOCSIS, PacketCable, CableHome) without requiring the MIB be loaded by the RDU. From within RDU configuration extensions, the functionality can be accessed with the DOCSISOptionFactory interface, using the following method:

```
public OptionValue createOptionValue(OptionSyntax syntax, String optionNumStr, String[]
optionValueList)
```

The public OptionSyntax.SNMP enumerated value can be used in the above method, in conjunction with the optionValueList containing the tuple: OID, Type, Value.

From RDU dynamic configuration templates, the following syntax is used to specify SNMP TLVs that are not validated against the RDU MIBs:

```
option <option-number> snmp <OID>, <Type>, <Value>
```

Examples:

```
# DOCS-CABLE-DEVICE-MIB:
option 11 snmp .docsDevNmAccessIp.1, IPADDRESS, 192.168.1.1

# Arris vendor specific SNMP TLV (OID numbers only, mix names/numbers)
option 11 snmp .1.3.6.1.4.1.4115.1.3.1.1.2.3.2.0, INTEGER, 6
option 11 snmp .enterprises.4115.1.3.1.1.2.3.2.0, INTEGER, 6

# NOTE: trailing colon required for single octet
option 11 snmp .1.3.6.1.2.1.69.1.2.1.6.3, STRING, 'c0:'
```

The allowed SNMP variable type names are:

IETF standard SMI Data Type	SNMP API name
Integer32	INTEGER
Integer (Enumerated)	INTEGER
Unsigned32	UNSIGNED32
Gauge32	GAUGE
Counter32	COUNTER
Counter64	COUNTER64
Timeticks	TIMETICKS
OCTET STRING	STRING
OBJECT IDENTIFIER	OBJID
IpAddress	IPADDRESS
BITS	STRING

For example, to specify a SMI Integer32 type, the following types are accepted (regardless of case sensitivity): Integer32, INTEGER.

For OCTET STRING type, all of the following types are accepted: OCTET STRING, OCTETSTRING, or STRING.

The custom SNMP TLV template option can be used to specify any SNMP TLV, including those that are present in the RDU MIBs. The custom SNMP TLV error checking is less stringent, and will not detect incorrect scalar/columnar references (for example, .0 vs. .n in OID names).

DOCSIS Option Support

Table 12-3 describes DOCSIS options and identifies the specific version support for each option.

Table 12-3 DOCSIS Options and Version Support

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
0	PAD	No length and no value	N/A	True	✓	✓	✓
1	Downstream Frequency	Unsigned integer 32	Multiples of 62500	False	✓	✓	✓
2	Upstream Channel ID	Unsigned integer 8	None	False	✓	✓	✓
3	Network Access Control	Boolean	None	False	✓	✓	✓
4	Class of Service	Compound	None	False	✓	✓	✓
4.1	Class ID	Unsigned integer 8	Between 1-16 inclusive	False	✓	✓	✓
4.2	Maximum Downstream Rate	Unsigned integer 32	None	False	✓	✓	✓
4.3	Maximum Upstream Rate	Unsigned integer 32	None	False	✓	✓	✓
4.4	Upstream Channel Priority	Unsigned integer 8	Less than 8	False	✓	✓	✓
4.5	Guaranteed Minimum Upstream Channel Data Rate	Unsigned integer 32	None	False	✓	✓	✓
4.6	Maximum Upstream Channel Transmit Burst	Unsigned integer 16	None	False	✓	✓	✓
4.7	Class-of-Service Privacy Enable	Boolean	None	False	✓	✓	✓
6	CM MIC Configuration Setting	Byte 16	None	False	✓	✓	✓
7	CMTS MIC Configuration Setting	Byte 16	None	False	✓	✓	✓
9	Software Upgrade Filename	NVTASCII	None	True	✓	✓	✓
10	SNMP Write-Access Control	OIDCF	None	True	✓	✓	✓
11	SNMP MIB Object	SNMPVarBind	None	True	✓	✓	✓
14	CPE Ethernet MAC Address	MAC Address	None	True	✓	✓	✓
15	Telephony Settings Option	NVTASCII	None	False	✓	✓	✓
15.2	Service Provider Name	NVTASCII	None	False	✓	✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
15.3	Telephone Number (1)	NVTASCII	None	False	✓	✓	✓
15.4	Telephone Number (2)	NVTASCII	None	False	✓	✓	✓
15.5	Telephone Number (3)	NVTASCII	None	False	✓	✓	✓
15.6	Connection Threshold	Unsigned integer 8	None	False	✓	✓	✓
15.7	Login Username	NVTASCII	None	False	✓	✓	✓
15.8	Login Password	NVTASCII	None	False	✓	✓	✓
15.9	DHCP Authentication	Boolean	None	False	✓	✓	✓
15.10	DHCP Server	IP Address	None	False	✓	✓	✓
15.11	RADIUS realm	NVTASCII	None	False	✓	✓	✓
15.12	PPPAAuthentication	Unsigned integer 8	None	False	✓	✓	✓
15.13	Demand Dial Inactivity Timer Threshold	Unsigned integer 8	None	False	✓	✓	✓
16	SNMP IP Address (No Longer Used)	IP Address	None	False	✓	✓	✓
17	Baseline Privacy Configuration Setting	Compound	None	False	✓	✓	✓
17.1	Authorize Wait Timeout	Unsigned integer 32	Between 1 and 30 inclusive	False	✓	✓	✓
17.2	Reauthorize Wait Timeout	Unsigned integer 32	Between 1 and 30 inclusive	False	✓	✓	✓
17.3	Authorization Grace Time	Unsigned integer 32	Between 1 and 1800 inclusive	False	✓		
17.3	Authorization Grace Time	Unsigned integer 32	Between 1 and 6047999 inclusive	False		✓	✓
17.4	Operational Wait Timeout	Unsigned integer 32	Between 1 and 10 inclusive	False	✓	✓	✓
17.5	Rekey Wait Timeout	Unsigned integer 32	Between 1 and 10 inclusive	False	✓	✓	✓
17.6	TEK Grace Time	Unsigned integer 32	Between 1 and 1800 inclusive	False	✓		
17.6	TEK Grace Time	Unsigned integer 32	Between 1 and 302399 inclusive	False		✓	✓
17.7	Authorize Reject Wait Timeout	Unsigned integer 32	Between 1 and 600 inclusive	False	✓	✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
17.8	SA Map Wait Timeout	Unsigned integer 32	Between 1 and 18006 inclusive	False		✓	✓
17.9	Maximum Clock Drift	Unsigned integer 32	Between 1 and 10 inclusive	False		✓	✓
18	Maximum Number of CPEs	Unsigned integer 32	Greater than 0	False	✓	✓	✓
19	TFTP Server Timestamp	Unsigned integer 32	None	False	✓	✓	✓
20	TFTP Server Provisioned Modem Address	IP Address	None	False	✓	✓	✓
21	Software Upgrade TFTP Server	IP Address N	None	False	✓	✓	✓
22	Upstream Packet Classification Encoding	Compound	None	True		✓	✓
22.1	Classifier Reference	Unsigned integer 8	Between 1 and 255 inclusive	False		✓	✓
22.2	Classifier Identifier	Unsigned integer 16	Between 1 and 65535 inclusive	False		✓	✓
22.3	Service Flow Reference	Unsigned integer 16	Between 1 and 65535 inclusive	False		✓	✓
22.4	Service Flow Identifier	Unsigned integer 32	Greater than 0	False		✓	✓
22.5	Rule Priority	Unsigned integer 8	None	False		✓	✓
22.6	Classifier Activation State	Boolean	None	False		✓	✓
22.7	Dynamic Service Change Action	Unsigned integer 8	Less than 3	False		✓	✓
22.8	Classifier Error Encodings	Compound	None	False		✓	✓
22.8.1	Error Parameter	Sub Type	None	False		✓	✓
22.8.2	Error Code	Unsigned integer 8	Less than 26	False		✓	✓
22.8.3	Error Message	ZTAASCII	None	False		✓	✓
22.9	IP Packet Classification Encodings	Compound	None	False		✓	✓
22.9.1	IP Type of Service Range and Mask	Unsigned integer 8 triplet	None	False		✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
22.9.2	IP Protocol	Unsigned integer 16	Less than 258	False		✓	✓
22.9.3	IP Source Address	IP Address	None	False		✓	✓
22.9.4	IP Source Mask	IP Address	None	False		✓	✓
22.9.5	IP Destination Address	IP Address	None	False		✓	✓
22.9.6	IP Destination Mask	IP Address	None	False		✓	✓
22.9.7	TCP/UDP Source Port Start	Unsigned integer 16	None	False		✓	✓
22.9.8	TCP/UDP Source Port End	Unsigned integer 16	None	False		✓	✓
22.9.9	TCP/UDP Destination Port Start	Unsigned integer 16	None	False		✓	✓
22.9.10	TCP/UDP Destination Port End	Unsigned integer 16	None	False		✓	✓
22.10	Ethernet LLC Packet Classification Encodings	Compound	None	False		✓	✓
22.10.1	Destination MAC Address	MAC Address and Mask	None	False		✓	✓
22.10.2	Source MAC Address	MAC Address	None	False		✓	✓
22.10.3	Ethertype/DSAP/MacType	Unsigned integer 8 and unsigned integer 16	None	False		✓	✓
22.11	IEEE 802.1P/Q Packet Classification Encodings	Compound	None	False		✓	✓
22.11.1	IEEE 802.1P User_Priority	Unsigned integer 8 pair	Less than 8	False		✓	✓
22.11.2	IEEE 802.1Q VLAN_ID	Unsigned integer 16	None	False		✓	✓
22.43	Vendor Specific Classifier Parameters	Compound	None	False		✓	✓
22.43.8	Vendor ID	OUI	None	False		✓	✓
23	Downstream Packet Classification Encoding	Compound	None	True		✓	✓
23.1	Classifier Reference	Unsigned integer 8	Between 1 and 255 inclusive	False		✓	✓
23.2	Classifier Identifier	Unsigned integer 16		False		✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
23.3	Service Flow Reference	Unsigned integer 16	Between 1 and 65535	False		✓	✓
23.4	Service Flow Identifier	Unsigned integer 32	Between 1 and 65535	False		✓	✓
23.5	Rule Priority	Unsigned integer 8	Greater than 0	False		✓	✓
23.6	Classifier Activation	Boolean	None	False		✓	✓
23.7	Dynamic Service Change Action	Unsigned integer 8	Less than 3	False		✓	✓
23.8	Classifier Error Encodings	Compound	None	False		✓	✓
23.8.1	Error Parameter	Sub Type	None	False		✓	✓
23.8.2	Error Code	Unsigned integer 8	Less than 26			✓	✓
23.8.3	Error Message	ZTASCII	None	False		✓	✓
23.9	IP Classification Encodings	Compound	None	False		✓	✓
23.9.1	IP Type of Service Range and Mask	Unsigned integer 8	None	False		✓	✓
23.9.2	IP Protocol	Unsigned integer 16	Less than 258	False		✓	✓
23.9.3	IP Source Address	IP Address	None	False		✓	✓
23.9.4	IP Source Mask	IP Address	None	False		✓	✓
23.9.5	IP Destination Address	IP Address	None	False		✓	✓
23.9.6	IP Destination Mask	IP Address	None	False		✓	✓
23.9.7	TCP/UDP Source Port Start	Unsigned integer 16	None	False		✓	✓
23.9.8	TCP/UDP Source Port End	Unsigned integer 16	None	False		✓	✓
23.9.9	TCP/UDP Destination Port Start	Unsigned integer 16	None	False		✓	✓
23.9.10	TCP/UDP Destination Port End	Unsigned integer 16	None	False		✓	✓
23.10	Ethernet LLC Packet Classification Encodings	Compound					✓
23.10.1	Destination MAC Address	MAC Address and Mask\	None	False		✓	✓
23.10.2	Source MAC Address	MAC Address	None	False		✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
23.10.3	Ethertype/DSAP/MacType	Unsigned integer 8 and unsigned integer 16	None	False		✓	✓
23.11	IEEE 802.1P/Q Packet Classification Encodings	Compound	None	False		✓	✓
23.11.1	IEEE 802.1P User_Priority	Unsigned integer 8 pair	Less than 8	False		✓	✓
23.11.2	IEEE 802.1Q VLAN_ID	Unsigned integer 16	None	False		✓	✓
23.43	Vendor Specific Classifier Parameters	Compound	None	False		✓	✓
23.43.8	Vendor ID	OUI	None	False		✓	✓
24	Upstream Service Flow Scheduling	Compound	None	True		✓	✓
24.1	Service Flow Reference	Unsigned integer 16	Greater than 0	False		✓	✓
24.3	Service Identifier	Unsigned integer 16	None	False		✓	✓
24.4	Service Class Name	ZTASCII	None	False		✓	✓
24.5	Service Flow Error Encodings	Compound	None	True		✓	✓
24.5.1	Errored Parameter	Unsigned integer 8	None	False		✓	✓
24.5.2	Error Code	Unsigned integer 8	Less than 26	False		✓	✓
24.5.3	Error Message	ZTASCII	None	False		✓	✓
24.6	Quality of Service Parameter Set Type	Unsigned integer 8	Less than 8	False		✓	✓
24.7	Traffic Priority	Unsigned integer 8	Less than 8	False		✓	✓
24.8	Upstream Maximum Sustained Traffic Rate	Unsigned integer 32	None	False		✓	✓
24.9	Maximum Traffic Burst	Unsigned integer 32	None	False		✓	✓
24.10	Minimum Reserved Traffic Rate	Unsigned integer 32	None	False		✓	✓
24.11	Assumed Minimum Reserved Rate Packet Size	Unsigned integer 16	None	False		✓	✓
24.12	Timeout for active QoS Parameters	Unsigned integer 16	None	False		✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
24.13	Timeout for Admitted QoS Parameters	Unsigned integer 16	None	False		✓	✓
24.14	Maximum Concatenated Burst	Unsigned integer 16	None	False		✓	✓
24.15	Service Flow Scheduling Type	Unsigned integer 8	Between 1-6 inclusive	False		✓	✓
24.16	Request/Transmission Policy	Unsigned integer 32	Less than 512	False		✓	✓
24.17	Nominal Polling Interval	Unsigned integer 32	None	False		✓	✓
24.18	Tolerated Poll Jitter	Unsigned integer 32	None	False		✓	✓
24.19	Unsolicited Grant Size	Unsigned integer 16	None	False		✓	✓
24.20	Nominal Grant Interval	Unsigned integer 32	None	False		✓	✓
24.21	Tolerated Grant Jitter	Unsigned integer 32	None	False		✓	✓
24.22	Grants per Interval	Unsigned integer 8	Less than 128	False		✓	✓
24.23	IP Type of Service Overwrite	Unsigned integer 8 pair	None	False		✓	✓
24.24	Unsolicited Grant Time Reference	Unsigned integer 32	None	False		✓	✓
24.43	Vendor Specific PHS Parameters	Compound	None	False		✓	✓
24.43.8	Vendor ID	OUI	None	False		✓	✓
25	Downstream Service Flow Scheduling	Compound	None	True		✓	✓
25.1	Service Flow Reference	Unsigned integer 16	Greater than 0	False		✓	✓
25.3	Service Identifier	Unsigned integer 16	None	False		✓	✓
25.4	Service Class Name	ZTASCII	None	False		✓	✓
25.5	Service Flow Error Encodings	Compound	None	True		✓	✓
25.5.1	Errored Parameter	Unsigned integer 8	None	False		✓	✓
25.5.2	Error Code	Unsigned integer 8	Less than 26	False		✓	✓
25.5.3	Error Message	ZTASCII	None	False		✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
25.6	Quality of Service Parameter Set Type	Unsigned integer 8	Less than 8	False		✓	✓
25.7	Traffic Priority	Unsigned integer 8	Less than 8	False		✓	✓
25.8	Downstream Maximum Sustained Traffic Rate	Unsigned integer 32	None	False		✓	✓
25.9	Maximum Traffic Burst	Unsigned integer 32	None	False		✓	✓
25.10	Minimum Reserved Traffic Rate	Unsigned integer 32	None	False		✓	✓
25.11	Assumed Minimum Reserved Rate Packet Size	Unsigned integer 16	None	False		✓	✓
25.12	Timeout for active QoS Parameters	Unsigned integer 16	None	False		✓	✓
25.13	Timeout for Admitted QoS Parameters	Unsigned integer 16	None	False		✓	✓
25.14	Maximum Downstream Latency	Unsigned integer 32	None	False		✓	✓
25.43	Vendor Specific PHS Parameters	Compound	None	False		✓	✓
25.43.8	Vendor ID	OUI	None	False		✓	✓
26	Payload Header Suppression	Compound	None	True		✓	✓
26.1	Classifier Reference	Unsigned integer 8	Greater than 0	False		✓	✓
26.2	Classifier Identifier	Unsigned integer 16	Greater than 0	False		✓	✓
26.3	Service Flow Reference	Unsigned integer 16	Greater than 0	False		✓	✓
26.4	Service Flow Identifier	Unsigned integer 32	Greater than 0	False		✓	✓
26.5	Dynamic Service Change Action	Unsigned integer 8	Less than 4	False		✓	✓
26.6	Payload Header Suppression Error Encodings	Compound	None	False		✓	✓
26.6.1	Errored Parameter	Unsigned integer 8	None	False		✓	✓
26.6.2	Error Code	Unsigned integer 8	Less than 26	False		✓	✓
26.6.3	Error Message	ZTASCII	None	False		✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
26.7	Payload Header Suppression Field (PHSF)	Bytes	None	False		✓	✓
26.8	Payload Header Suppression Index (PHSI)	Unsigned integer 8	Greater than 0	False		✓	✓
26.9	Payload Header Suppression Mask (PHSM)	Bytes	None	False		✓	✓
26.10	Payload Header Suppression Size (PHSS)	Unsigned integer 8	None	False		✓	✓
26.11	Payload Header Suppression Verification (PHSV)	Boolean	None	False		✓	✓
26.43	Vendor Specific PHS Parameters	Compound	None	False		✓	✓
26.43.8	Vendor ID	OUI	None	False		✓	✓
28	Maximum Number of Classifiers	Unsigned integer 16	None	False		✓	✓
29	Privacy Enable	Boolean	None	False		✓	✓
32	Manufacturer CVC	Bytes	None	False		✓	✓
33	Co-signer CVC	Bytes	None	False		✓	✓
34	SnmpV3 Kickstart Value	Compound	None	False		✓	✓
34.1	SnmpV3 Kickstart Security Name	NVTASCII	None	False		✓	✓
34.2	SnmpV3 Kickstart Manager Public Number	Bytes	None	False		✓	✓
35	Subscriber Management Control	Bytes	None	False		✓	✓
36	Subscriber Management CPE IP Table	Multiple IP Addresses	None	False		✓	✓
37	Subscriber Management Filter Groups	Bytes	None	False		✓	✓
38	Configuration File Element - docsisV3 Notification Receiver	Compound	None	False		✓	✓
38.1	IP Address of Trap Receiver	IP address	None	False		✓	✓
38.2	UDP Port Number of Trap Receiver	unsigned integer 16	None	False		✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
38.3	Type of Trap Sent by the PS	unsigned integer 8	None	False		✓	✓
38.4	Timeout	unsigned integer 32	None	False		✓	✓
38.5	Number of Retries When Sending an Inform After Sending the Inform First	unsigned integer 8	None	False		✓	✓
38.6	Notification Filtering Parameters	OID	None	False		✓	✓
38.7	Security Name to Use When Sending SNMP V3 Notification	NVTASCII	None	False		✓	✓
39	Enable 2.0 Mode	Enable/Disable	None	False			✓
40	Enable Test Mode	SubOptions	None	True			✓
41	Downstream Channel List	SubOptions	None	True			✓
41.1	Single Downstream Channel	SubOptions	None	True			✓
41.1.1	Single Downstream Channel Timeout	unsigned integer 16	None	False			✓
41.1.2	Single Downstream Channel Frequency	unsigned integer 32	None	False			✓
41.2	Downstream Frequency Range	SubOptions		True			✓
41.2.1	Downstream Frequency Range Timeout	unsigned integer 16	None	False			✓
41.2.2	Downstream Frequency Range Start	unsigned integer 32	Multiples of 62500	False			✓
41.2.3	Downstream Frequency Range End	unsigned integer 32	Multiples of 62500	False			✓
41.2.4	Downstream Frequency Range Step Size	unsigned integer 32	None	False			✓
41.3	Default Scanning	unsigned integer 32	None	True			✓
42	Multicast MAC Address	MAC Address	None	True			✓
43	Vendor-Specific Information	Compound	None	True	✓	✓	✓
43.1	Static Downstream Frequency	Unsigned integer 32	None	False	✓	✓	✓
43.2	Sync Loss Timeout	Unsigned integer 32	None	False	✓	✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
43.3	Update Boot Monitor Image	NVTASCII	None	False	✓	✓	✓
43.4	Power Backoff	Unsigned integer 16	None	False	✓	✓	✓
43.8	Vendor ID	OUI	None	False	✓	✓	✓
43.9	Update Factory System Image	Boolean	None	False	✓	✓	✓
43.10	Phone Lines	Unsigned integer 8	None	False	✓	✓	✓
43.11	IP Precedence Settings	Compound	None	True	✓	✓	✓
43.11.1	IP Precedence Value	Unsigned integer 8	None	False	✓	✓	✓
43.11.2	Rate Limit	Unsigned integer 32	None	False	✓	✓	✓
43.128	IOS Configuration Filename	NVTASCII	None	False	✓	✓	✓
43.129	IOS Config File Without Console Disable	NVTASCII	None	False	✓	✓	✓
43.131	IOS CLI Command	NVTASCII	None	True	✓	✓	✓
43.132	1.0 Plus Flow Encodings	Compound	None	False	✓	✓	✓
43.132.1	1.0 Plus Flow ID	Unsigned integer 8	None	False	✓	✓	✓
43.132.2	Class ID	Unsigned integer 8	None	False	✓	✓	✓
43.132.3	Unsolicited Grant Size	Unsigned integer 16	Between 1-65535 inclusive	False	✓	✓	✓
43.132.4	Nominal Grant Interval	Unsigned integer 32	Between 1-65535 inclusive	False	✓	✓	✓
43.132.5	Grants Per Interval	Unsigned integer 8	Between 0-127 inclusive	False	✓	✓	✓
43.132.6	Embedded Voice Calls	Unsigned integer 8	Between 0-127 inclusive	False	✓	✓	✓
43.132.7	Hold Queue Length	Unsigned integer 16	Between 0-4096 inclusive	False	✓	✓	✓
43.132.8	Fair Queue	Compound	None	False	✓	✓	✓
43.132.8.1	Congestive Discard Threshold	Unsigned integer 16	Between 1-4096 inclusive	False	✓	✓	✓

Table 12-3 DOCSIS Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	DOCSIS Version		
					1.0	1.1	2.0
43.132.8.2	Number of Dynamic Conversation Queues	Unsigned integer 16	Between 16-4096 inclusive	False	✓	✓	✓
43.132.8.3	Number of Reservable Conversation Queues	Unsigned integer 16	Between 0-1000 inclusive	False	✓	✓	✓
43.132.9	Custom Queue List Length	Unsigned integer 8	Between 1-16 inclusive	False	✓	✓	✓
43.132.10	Random Detection	Boolean	None	False	✓	✓	✓
43.132.11	Priority Group	Unsigned integer 8	Between 1-16 inclusive	False	✓	✓	✓
43.132.12	Service Policy File	NVTASCII	None	False	✓	✓	✓
43.132.13	Inactivity Timer	Unsigned integer 16	Between 1-10080 inclusive	False	✓	✓	✓
43.132.14	COS Tag	NVTASCII	None	False	✓	✓	✓
43.133	Downstream Sub Channel ID	Unsigned integer 8	Between 0-15 inclusive	False	✓	✓	✓
43.134	SU Tag	NVTASCII	None	False	✓	✓	✓
255	End-of-Data Marker	No length and no value	N/A	False	✓	✓	✓

PacketCable Option Support

Table 12-4 identifies the PacketCable 1.0 MTA options supported by BACC.

Table 12-4 PacketCable MTA 1.0 Options

Number	Description	Encoding	Validation	Multi-valued	PacketCable Version	
					1.0	1.1
11	SNMP MIB Object	SNMPVarBind with 1 byte length	None	True	✓	✓
38	SNMPv3 Notification Receiver	SubOptions	None	True	✓	✓
38.1	SNMPv3 Notification Receiver IP Address	IPAddress	None	False	✓	✓
38.2	SNMPv3 Notification Receiver UDP Port Number	Unsigned integer 16	None	False	✓	✓

Table 12-4 PacketCable MTA 1.0 Options (continued)

Number	Description	Encoding	Validation	Multi-valued	PacketCable Version	
					1.0	1.1
38.3	SNMPv3 Notification Receiver Trap Type	SNMPTrapType	From 1 to 5	False	✓	✓
38.4	SNMPv3 Notification Receiver Timeout	Unsigned integer 16	None	False	✓	✓
38.5	SNMPv3 Notification Receiver Retries	Unsigned integer 16	From 0 to 255	False	✓	✓
38.6	Notification Receiver Filtering Parameters	OID	None	False	✓	✓
38.7	Notification Receiver Security Name	NVTASCII	None	False	✓	✓
43	Vendor-Specific Information	SubOptions	None	True	✓	✓
43.8	Vendor ID	OUI	None	False	✓	✓
64	SNMP MIB Object	SNMPVarBind with 2 byte length	None	True	✓	✓
254	Telephony Config File Start/End	Unsigned integer 8	Must be 1 or 255	False	✓	✓

Non-Secure CableHome Option Support

Table 12-5 identifies the non-secure CableHome options supported by BACC.

Table 12-5 Non-secure CableHome Options and Version Support

Option Number	Description	Encoding	Validation	Multi-valued	CableHome Version
					1.0
0	PAD	No length and no value	None	True	✓
9	Software Upgrade Filename	NVTASCII	None	False	✓
10	SNMP Write-Access Control	OIDCF	None	True	✓
12	Modem IP Address	IP Address	None	False	✓
14	CPE Ethernet MAC Address	MACAddress	None	True	✓
21	Software Upgrade TFTP Server	IPAddress	None	False	✓
28	SNMP MIB Object	SNMPVarBind	None	True	✓
32	Manufacturer CVC	Bytes	None	False	✓
33	Co-signer CVC	Bytes	None	True	✓
34	SnmpV3 Kickstart Value	SubOptions	None	False	✓

Table 12-5 Non-secure CableHome Options and Version Support (continued)

Option Number	Description	Encoding	Validation	Multi-valued	CableHome Version
					1.0
34.1	SnmpV3 Kickstart Security Name	NVTASCII	None	False	✓
38	SNMPv3 Notification Receiver	SubOptions	None	True	✓
38.1	SNMPv3 Notification Receiver IP Address	IPAddress	None	False	✓
38.2	SNMPv3 Notification Receiver UDP Port Number	Unsigned integer 16	None	False	✓
38.3	SNMPv3 Notification Receiver Trap Type	SNMPTrapType	From 1 to5	False	✓
38.4	SNMPv3 Notification Receiver Timeout	Unsigned integer 16	None	False	✓
38.5	SNMPv3 Notification Receiver Retries	Unsigned integer 16	None	False	✓
38.6	Notification Receiver Filtering Parameters	OID	None	False	✓
38.7	Notification Receiver Security Name	NVTASCII	None	False	✓
43	Vendor-Specific Information	SubOptions	None	True	✓
43.1	Vendor ID	OUI	None	False	✓
53	PS MIC. A 20 octet SHA-1 hash of PS config file	Bytes	None	False	✓
255	End-of-Data Marker	No length and no value	None	False	✓

Using the Configuration File Utility

You use the configuration file utility to test, validate, and view PacketCable 1.0, DOCSIS 1.0/1.1/2.0, and CableHome template and configuration files. These activities are critical to successful deployment of individualized configuration files. See the “[Developing Template Files](#)” section on page 12-1 for more information on templates.

The configuration file utility is available only when the RDU is installed and the utility is installed in the <BACC_HOME>/rdu/bin directory.

Both the template file being encoded and the binary file being decoded must reside in the directory from which the configuration file utility is invoked.

All examples found in this section assume that the RDU is operating and that these conditions apply:

- The BACC application is installed in the home directory (/opt/CSCObr).
- The RDU login name is admin.
- The RDU login password is changeme.

**Note**

Some of the examples provided in this section have been truncated whenever the omitted formation is of no consequence to the example or its outcome. Instances where this occurs are identified by a series of periods (...) that are located just prior to the example summary.

This section discusses these topics:

- [Parsing a Local Template File, page 12-28](#)
- [Parsing an External Template File, page 12-29](#)
- [Specifying Macro Variables at the Command Line, page 12-32](#)
- [Specifying a Device for Macro Variables, page 12-33](#)
- [Specifying Output to a Binary File, page 12-35](#)
- [Viewing a Local Binary File, page 12-36](#)
- [Viewing an External Binary File, page 12-37](#)

Running the Configuration File Utility

In subsequent procedures and examples, the phrase “run the configuration file utility” means to enter the **runCfgUtil.sh** command from the directory specified. To run the configuration file utility, run this command from the home directory:

```
runCfgUtil.sh (options)
```

Where the available (*options*) include:

- **-c <secret>**—Specifies the CMTS shared secret when parsing a DOCSIS template file. To specify the default shared secret, enter **-c cisco**.
- **-cablehome**—Identifies the input file as a CableHome portal service configuration file. Do not use this with either the **-docsis** or **-pkt** options.
- **-d**—Decodes the binary input file. Do not use this with the **-e** option.
- **-docsis**—Specifies the input file is a DOCSIS configuration file. Do not use this default with the **-pkt** option.
- **-e**—Encodes the template input file. Do not use this default with the **-d** option.
- **-g**—Generates a template file from either a docsis, packetcable or cablehome binary file.
- **-h<host:port>**—Specifies the host and port number. The default port number is 49187.
- **-i <device id>**—Specifies the device to use when parsing macro variables. For example, if your device ID is 1,6,00:00:00:00:00:01, enter **-i 1,6,00:00:00:00:00:01**. When using this option, you must also use the **-u** and **-p** options, respectively, to specify the username and password. Do not use this with the **-m** option.
- **-l <filename>**—Identifies the input file as being on the local file system. For example, if your input file is called any_file, enter **-l any_file**. Do not use this with the **-r** option.
- **-loc**—Specifies the PacketCable locale, na (North America) or euro (Europe). The default is na. If the MTA is euro-MTA, then the locale should be set to euro.
- **-m <macros>**—Specifies key value pairs for macro variables. The format is key=value. If you require multiple macro variables, use a double comma separator between the key value pairs, for example, **key_1=value_1,,key_2=value_2**. Do not use this with the **-i** option.

- **-p <password>**—Specifies the password to use when connecting to the RDU. For example, if your password is 123456, enter **-p 123456**.
- **-o <filename>**—Saves parsed template file as a binary file. For example, if you want the output to be found in a file call `op_file`, enter **-o op_file**.
- **-pkt**—Identifies the input file as a PacketCable MTA configuration file. Do not use this with the **-docsis** option.
- **-r <filename>**—Identifies the input file as an external file that has been added to the RDU. For example, if your file is called `file25` enter **-r file25**. When using this option you must also use the **-u** and **-p** options, respectively, to specify the username and password. Do not use this with the **-l** option.
- **-s**—Displays the parsed template or the contents of the binary file in a human readable format.
- **-t**—Specifies the PacketCable encoding type, secure or basic (default is secure).
- **-u <username>**—Specifies the username to use when connecting to the RDU. For example, if your username is `admin`, enter **-u admin**.
- **-v <version>**—Specifies the DOCSIS version being used. For example, if you are using DOCSIS 1.1, enter **-v 1.1**. If you do not specify the version number, the command defaults to use DOCSIS 2.0.

**Note**

The configuration file utility does not include option 19 (TFTP server timestamp) and option 20 (TFTP server provisioned modem address) in the template file, however the BACC TFTP mixing does. Also, options 6 (CM MIC) and 7 (CMTS MIC) are both automatically inserted into the encoded template file. Therefore, you do not have to specify these message integrity checks (MIC).

Using the Configuration File Utility

To use the configuration file utility to test BACC templates:

- Step 1** Develop the template as described in the [“Developing Template Files” section on page 12-1](#).
- Step 2** Run the configuration file utility on the local file system. If the template contains macro variables, perform these operations in the order specified:
 - a. Test with command line substitution.
 - b. Test with a device that has been added to your RDU.
- Step 3** Add the template (and any included templates that are used) to the RDU.
- Step 4** Run the configuration file utility as an external file. If the template contains macro variables, perform these operations in the order specified:
 - a. Test with command line substitution.
 - b. Test with a device that has been added to your RDU.
- Step 5** Configure a class of service to use the template.

Converting a Binary File Into a Template File

Usage Guidelines

Use the **runCfgUtil.sh** command to convert binary configuration memory files into template files. BACC dynamic configuration generation is based on templates that are created. Automatically converting existing, tested, binary files to template files speeds the process and reduces the possibility of introducing errors.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command to convert an existing binary file into a template file.

```
runCfgUtil.sh -g -l <binary_file> -o <template_file>
```

Where:

- **-g**—specifies that a template file needs to be generated from an input binary file
- **-l <binary_file>**—specifies the local input file, including the path name. In all cases, the input binary file name will have a *.cm file extension; bronze.cm for example.
- **-o <template_file>**—specifies the output template file, including the path name. In all cases, the input binary file name will have a *.tmpl file extension; test.tmpl for example.

Parsing a Local Template File

Usage Guidelines

Use the **runCfgUtil.sh** command to parse external template files.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command to parse a local template file:

```
runCfgUtil.sh -pkt -l <file>
```

Where:

- **-pkt**—identifies the input file as a packetcable MTA file
- **-l**—specifies the input file is on the local file system
- **<file>**—identifies the input template file being parsed

Parsing the File

To parse a template file that is on the local file system:

Step 1 Change directory to /opt/CSCObpr/rdu/samples/packet_cable.

Step 2 Select a template file to use.



Note This example uses an existing template file called unprov_packet_cable.tmpl. The -pkt option is used because this is a PacketCable MTA template.

Step 3 Run the configuration file utility using this command:

```
runCfgUtil.sh -pkt -l unprov_packet_cable.tpl
```

Where:

- `-pkt`—identifies the input file as a packetcable MTA file
- `-l`—specifies the input file is on the local file system
- `unprov_packet_cable.tpl`—identifies the input template file being parsed

After running the utility, results similar to these should appear:

```
Off      File Bytes      Option      Description      Value
0        FE0101          254         Telephony Config File Start/End 1
3        0B153013060E   11          SNMP MIB Object      .iso.org.dod.internet.
                2B06010401A30B                private.enterprises.ca
                0202010101                bleLabs.clabProject.cl
                0700020102                abProjPacketCable.pktc
                                MtaMib.pktcMtaMibObjec
                                ts .pktcMtaDevBase.
                                pktcMtaDevEnable
                                d.0, INTEGER, false(2)
.....

0 error(s), 0 warning(s) detected. Parsing of unprov_packet_cable.tpl was successful.
The file unprov_packet_cable.tpl was parsed successfully in 434 ms.
The parser initialization time was 92 ms.
The parser parse time was 342 ms.
```

Parsing an External Template File

Usage Guidelines

Use the **runCfgUtil.sh** command to parse external template files.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command to parse an external template file:

```
runCfgUtil.sh -r <file> -u <username> -p <password>
```

Where:

- `-r <file>`—identifies the input file as an external file that has been added to the RDU
- `-u <username>`—specifies the username to use when connecting to the RDU
- `-p <password>`— specifies the password to use when connecting to the RDU

Parsing the File

To parse a template file that has been added to the RDU:

- Step 1** Change directory to `/opt/CSCObpr/rdu/samples/docsis`.
- Step 2** Select a template file to use.

**Note**

This example uses an existing template file called `unprov.tmpl`. The `-docsis` option is used because a DOCSIS template is being used.

Step 3 Run the configuration file utility using this command:

```
runCfgUtil.sh -r <unprov.tmpl> -u <admin> -p <changeme> -docsis
```

Where:

- `unprov.tmpl`—identifies the input file
- `admin`—identifies the user name
- `changeme`—identifies the password
- `-docsis`—identifies the file as a DOCSIS template

After running the utility, results similar to these should appear:

**Note**

The results shown here are for illustration only and have been truncated for brevity.

Off	File Bytes	Option	Description	Value
0	030101	3	Network Access Control	On
3	041F	4	Class of Service	
5	010101	4.1	Class ID	1
8	02040000FA00	4.2	Maximum Downstream Rate	128000 bits/sec
14	03040000FA00	4.3	Maximum Upstream Rate	64000 bits/sec
20	040101	4.4	Upstream Channel Priority	1
.				
252	06108506547F C9152B44DB95 5420843EF6FE	6	CM MIC Configuration Setting	8506547FC9152B44 DB955420843EF6FE
270	0710644B675B 70B7BD3E09AC 210F794A1E8F	7	CMTS MIC Configuration Setting	644B675B70B7BD3E 09AC210F794A1E8F
288	FF	255	End-of-Data Marker	
289	00	0	PAD	
290	00	0	PAD	
291	00	0	PAD	

```
0 error(s), 0 warning(s) detected. Parsing of unprov.tmpl was successful.
The file unprov.tmpl was parsed successfully in 375 ms.
The parser initialization time was 63 ms.
The parser parse time was 312 ms.
```

Parsing a Template File and Adding a User-Specified Shared Secret

Usage Guidelines

Use the **runCfgUtil.sh** command to parse a template file and add a shared secret that you specify.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command to parse a template file and add a shared secret:

```
runCfgUtil.sh -e -docsis -l <file> -c <secret>
```

Where:

- **-e**—identifies the encode option.
- **-docsis**—Identifies the input file as a DOCSIS template file.
- **-l**—Specifies the input file is on the local file system.
- **<file>**—Identifies the input template file being parsed.
- **-c <secret>**—Specifies the CMTS shared secret when parsing a DOCSIS template file. The default shared secret is **-c cisco**.

Parsing the File

To parse a locally saved template file, and set a user specified shared secret:

Step 1 Change directory to `/opt/CSCObpr/rdu/samples/docsis`.

Step 2 Select a template file to parse.



Note

This example uses an existing template file called `unprov.tmpl`. The `-docsis` option is used because this is a DOCSIS template.

Step 3 Run the configuration file utility using this command:

```
runCfgUtil.sh -e -docsis -l unprov.tmpl -c shared
```

Where:

- **-e**—identifies the encode option.
- **-docsis**—Identifies the input file as a DOCSIS template file.
- **-l**—Specifies the input file is on the local file system. The locally saved file used in this example is `unprov.tmp`.
- **-c**—Indicates that a shared secret is being set and `shared` identifies that new shared secret.

After running the utility, results similar to these should appear:

Off	File Bytes	Option	Description	Value
0	030100	3	Network Access Control	Off
3	041F	4	Class of Service	
5	010101	4.1	Class ID	1

```

8      02040001F400    4.2      Maximum Downstream Rate      128000 bits/sec
14     03040000FA00    4.3      Maximum Upstream Rate        64000 bits/sec
20     040101          4.4      Upstream Channel Priority     1
. . . . .
. . . . .
252    06108506547F    6         CM MIC Configuration Setting  8506547FC9152B44
      C9152B44DB95
      5420843EF6FE
      DB955420843EF6FE
270    0710644B675B    7         CMTS MIC Configuration Setting 644B675B70B7BD3E
      70B7BD3E09AC
      210F794A1E8F
      09AC210F794A1E8F
288    FF             255      End-of-Data Marker
289    00             0         PAD
290    00             0         PAD
291    00             0         PAD

```

```

0 error(s), 0 warning(s) detected. Parsing of unprov.tmpl was successful.
The file unprov.tmpl was parsed successfully in 375 ms.
The parser initialization time was 63 ms.
The parser parse time was 312 ms.

```

Specifying Macro Variables at the Command Line

Usage Guidelines Use the **runCfgUtil.sh** command to specify macro variables.

Syntax Description You must use this syntax when using the **runCfgUtil.sh** command when specifying macro variables at the command line:

```
runCfgUtil.sh -e -l <file> -m <"macros">
```

Where:

- **-e**—identifies the encode option.
- **-l**—Specifies the input file is on the local file system.
- **<file>**—Identifies the input template file being parsed.
- **-m**—Specifies the macro variables to be substituted when parsing a template.
- **<"macros">**—Identifies the desired macros. When multiple macro variables are required, insert a double comma separator between each macro.

To specify values for macro variables at the command line:

-
- Step 1** Change directory to /opt/CSCObpr/rdu/samples/templates.
- Step 2** Select a template file to use.
- Step 3** Identify the macro variables in the template. In this example, the macro variables are macro1 (option 3) and macro11 (option 4.2).
- Step 4** Identify the values for the macro variables. The value for macro1 will be set to 1, and the value for macro11 to 64000.
- Step 5** Run the configuration file utility using this command:

```
runCfgUtil.sh -e -l macro.tmpl -m "macro1=1,,macro11=64000"
```

Where:

- -e—identifies the encode option.
- -unprov.tmpl—Identifies the input file.
- macro1=1,,macro11=64000—Identifies the key value pairs for macro variables. Since multiple macro variables are necessary, a double comma separator is inserted between the key value pairs.

After running the utility, results similar to these should appear:

Off	File Bytes	Option	Description	Value
0	030101	3	Network Access Control	On
3	041F	4	Class of Service	
5	010101	4.1	Class ID	1
8	02040000FA00	4.2	Maximum Downstream Rate	64000 bits/sec
14	03040000FA00	4.3	Maximum Upstream Rate	64000 bits/sec
20	040101	4.4	Upstream Channel Priority	1

.....

```
0 error(s), 0 warning(s) detected. Parsing of macro.tmpl was successful.
The file macro.tmpl was parsed successfully in 854 ms.
The parser initialization time was 76 ms.
The parser parse time was 778 ms.
```

Specifying a Device for Macro Variables

Usage Guidelines

Use the **runCfgUtil.sh** command to specify a device for macro variables.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command when specifying a device for macro variables:

```
runCfgUtil.sh -e -l <file> -i <MAC> -u <username> -p <password>
```

Where:

- -e—identifies the encode option
- -l—specifies the input file is on the local file system
- <file>—identifies the input template file being parsed
- -i—specifies the device to use when parsing macro variables
- <MAC>—identifies the MAC address of the device
- -u <username>—specifies the username to use when connecting to the RDU
- -p <password>— specifies the password to use when connecting to the RDU

To specify a device to be used for macro variable substitution:

-
- Step 1** Change directory to /opt/CSCObpr/rdu/samples/templates.
- Step 2** Select a template file to use. This example will use the existing template file, macro.tmpl.
- Step 3** Identify the macro variables in the template. In this example, the macro variables are macro1 (option 3) and macro11 (option 4.2).
- Step 4** Identify the device to use. This example will assume that the device exists in the RDU and has the macro variables set as properties. The value for macro1 will be set to 1, and the value for macro11 to 64000.
- Step 5** Run the configuration file utility using this command:

```
runCfgUtil.sh -l macro.tmpl -i "1,6,00:01:02:03:04:05" -u admin -p changeme
```

Where:

- macro.tmpl—Identifies the input file.
- 1,6,00:01:02:03:04:05—Identifies the MAC address of the device. The MAC address used here is for example purposes only.
- admin—Identifies the default username being used in this example.
- changeme—identifies the default password being used in this example

After running the utility, results similar to these should appear:

Off	File Bytes	Option	Description	Value
0	030101	3	Network Access Control	On
3	041F	4	Class of Service	
5	010101	4.1	Class ID	1
8	02040000FA00	4.2	Maximum Downstream Rate	64000 bits/sec
14	03040000FA00	4.3	Maximum Upstream Rate	64000 bits/sec
20	040101	4.4	Upstream Channel Priority	1

.....

```
0 error(s), 0 warning(s) detected. Parsing of macro.tmpl was successful.
The file macro.tmpl was parsed successfully in 823 ms.
The parser initialization time was 102 ms.
The parser parse time was 803 ms.
```

Specifying Output to a Binary File

Usage Guidelines

Use the **runCfgUtil.sh** command to specify a device for macro variables.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command when specifying output to a binary file:

```
runCfgUtil.sh -l <input_file> -o <output_file>
```

Where:

- **-l**—specifies the input file is on the local file system
- **<input_file>**—identifies the input template file being parsed
- **-o**—specifies that file name where the binary contents of the parsed template file are stored
- **<output_file>**—specifies the output filename to be used

To specify the output from parsing a template to a binary file:

-
- Step 1** Change directory to `/opt/CSCObpr/rdu/samples/templates`.
 - Step 2** Select a template file to use.
 - Step 3** Identify the name of the output file. This example will use `unprov.cm`.
 - Step 4** Run the configuration file utility using this command:

```
runCfgUtil.sh -l unprov.tmpl -o unprov.cm
```

Where:

- `unprov.tmpl`—identifies the existing template file being parsed into a binary file
- `unprov.cm`—specifies the output filename to be used

After running the utility, results similar to these should appear:

```
# /opt/CSCObpr/rdu/bin/runCfgUtil.sh -l unprov.tmpl -o unprov.cm
Broadband Provisioning Registrar Configuration Utility
Version: 2.7
```

```
0 error(s), 0 warning(s) detected. Parsing of unprov.tmpl was successful.
The file unprov.tmpl was parsed successfully in 595 ms.
The parser initialization time was 262 ms.
The parser parse time was 333 ms.
```

Viewing a Local Binary File

Usage Guidelines

Use the **runCfgUtil.sh** command to view a local binary file.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command to view a local binary file:

```
runCfgUtil.sh -d -l <file>
```

Where:

- **-d**—specifies that the command is going to decode a binary input file for viewing
- **-l**—identifies that the input file is located on the local file system
- **<file>**—identifies the existing binary input file to be viewed

To view a binary file that is on the local file system:

Step 1 Change directory to `/opt/CSCObpr/rdu/samples/packet_cable`.

Step 2 Select a binary file to view.

Step 3 Run the configuration file utility using this command:

```
runCfgUtil.sh -d -l unprov_packet_cable.bin
```

Where:

- **unprov_packet_cable.bin**—identifies the existing binary input file to be viewed

After running the utility, results similar to these should appear:

Off	File Bytes	Option	Description	Value
0	FE0101	254	Telephony Config File Start/End	1
3	0B153013060E 2B06010401A30B 02020101010700 020102	11	SNMP MIB Object	.iso.org.dod.internet. private.enterprises.ca bleLabs.clabProject. clabProjPacketCa ble.pktcMtaMib.pktcMta MibObjects .pktcMtaDevBase. pktcMtaDevEnable d.0, INTEGER, fals e(2)
.				

```

270      0710644B675B   7          CMTS MIC Configuration Setting   644B675B70B7BD3E
      70B7BD3E09AC                                     09AC210F794A1E8F
      210F794A1E8F

288      FF              255        End-of-Data Marker

289      00              0          PAD

290      00              0          PAD

291      00              0          PAD

0 error(s), 0 warning(s) detected. Parsing of unprov.tmpl was successful.
The file unprov.tmpl was parsed successfully in 375 ms.
The parser initialization time was 63 ms.
The parser parse time was 312 ms.

```

Activating PacketCable BASIC Flow

Usage Guidelines

Use the **runCfgUtil.sh** command to support the generation and insertion of the PacketCable BASIC Flow integrity hash into a BASIC flow static configuration file.

Syntax Description

You must use this syntax when using the **runCfgUtil.sh** command to support the PacketCable BASIC Flow.

```
runCfgUtil.sh -t <basic/secure>
```

Where:

- basic—This option calculates and inserts a PacketCable BASIC flow integrity hash into an MTA static configuration file.
- secure—This option does not insert the PacketCable BASIC flow integrity hash into an MTA static configuration file. This is the default setting.

The RDU Log Level Tool

You use the RDU log level tool to change the current log level of the RDU (on a local computer) from the command line. Although this tool is only available in an RDU installation, it is located in the <BACC_HOME>/rdu/bin directory. [Table 12-6](#) identifies the available log levels and the types of message written to the log file when enabled.

Table 12-6 Logging Levels

Log Level	Description
Emergency	System unstable.
Alert	Immediate action needed.
Critical	Critical conditions exist.
Error	Error conditions exist.

Table 12-6 Logging Levels (continued)

Log Level	Description
Warning	Warning conditions exist.
Notification	This is a normal, but significant, condition.
Information	This is used only for informational messages.

Cisco recommends that you keep the RDU logging level at the Warning level to help maintain a steady operations state. The Information level is recommended if you need to maintain steady state performance during debug operations. You should however, exercise caution when running with the Information level set because this creates a great number of log entries which in itself can adversely impact performance.

Using the RDU Log Level Tool

All examples assume that the user name for the RDU is admin, the password for the RDU is changeme, and the RDU is running.

Enter this command to run the RDU log level tool:

```
setLogLevel.sh [0..6] [-help] [-show] [-default] [-debug]
```

Where:

- *-[0..6]*—Identifies the logging level to be used. These are the available levels:
 - 0—This is the emergency level. It sets the logging function to save all emergency messages.
 - 1—This is the alert level. It sets the logging function to save all activities requiring immediate action and those of a more severe nature.
 - 2—This is the critical level. It sets the logging function to save all unusual conditions and those of a more severe nature.
 - 3—This is the error level. It sets the logging function to save all error messages and those of a more severe nature.
 - 4—This is the warning level. It sets the logging function to save all emergency messages and those of a more severe nature.
 - 5—This is the notification level. It sets the logging function to save all notification level messages and those of a more severe nature.
 - 6—This is the info level. It sets the logging function to save all available logging messages.
- *-help*—Displays help for the tool.
- *-show*—Displays the current log level set for the RDU server.
- *-default*—Sets the RDU to the installation default level 5 (notification).
- *-debug*— Sets an interactive mode to enable or disable tracing categories for RDU server.



Note

You should only enable the debug settings that the Cisco support staff recommends.

You can also use this tool to perform these functions:

- [Setting the RDU Log Level, page 12-40](#)
- [Viewing the RDU's Current Log Level, page 12-40](#)

Setting the RDU Log Level

You can use this tool to change the logging level from one value to any other value. The following example illustrates how to set the RDU logging level to the warning level, as indicated by the number 4 in the `setLogLevel.sh` command. The actual log level set is not important for the procedure, it can be interchanged as required.

To set the RDU logging level:

Step 1 Change directory to `<BACC_HOME>/rdu/bin`.

Step 2 Run the RDU log level tool using this command:

```
setLogLevel.sh 4
```

This prompt appears:

```
Please type RDU username:
```

Step 3 Enter the RDU username at the prompt. In this example, the default username (admin) is used.

```
Please type RDU username: admin
```

This prompt appears:

```
Please type RDU password:
```

Step 4 Enter the RDU password for the RDU at the prompt. In this example, the default password (changeme) is used.

```
Please type RDU password: changeme
```

This message appears to notify you that the log level has been changed. In this example, the level was 5, for notification, and is now 4, for warning.

```
RDU Log level was changed from 5 (notification) to 4 (warning).
```

Viewing the RDU's Current Log Level

You can use this tool to view the RDU log and determine which logging level is configured before attempting to change the value. This procedure illustrates how to use the tool to view the RDU's current logging level, and assumes that you have a computer already connected to the RDU.

To view the RDU's current logging level:

Step 1 Change directory to `<BACC_HOME>/rdu/bin`.

Step 2 Run this command:

```
setLogLevel.sh -show
```

This prompt appears:

```
Please type RDU username:
```

Step 3 Enter the RDU username (admin) and press **Enter**.

```
Please type RDU username: admin
```


This prompt appears:

```
Please type RDU password:
```

Step 4 Enter the RDU password (changeme) and press **Enter**.

```
Please type RDU password: changeme
```

This message appears:

```
The logging is currently set at level: 4 (warning)
```

```
All tracing is currently disabled.
```

Using the PKCert.sh Tool to Manage KDC Certificates

The PKCert tool is used to install, and then manage, the KDC certificates that are required by the KDC for its operation. This tool takes the CableLabs service provider certificates that you obtain and converts them into the series of certificate files, similar to those shown below, that the KDC needs to operate.

- Cablelabs_Service_Provider_Root.cer
- Service_Provider.cer
- Local_System.cer
- KDC.cer

This tool also allows you to verify certificate chains and copy and rename a certificate chain to the names required by the KDC.



Note

This tool is available for use only when the KDC component is installed.

Running the PKCert Tool

Run the PKCert tool by executing this command, the default location of which is in the <BACC__HOME>/kdc directory:

```
PKCert.sh [function] [option]
```

Where:

- [function]—identifies which function will be performed. For example, enter one of these switches to choose the appropriate function:
 - -c—Creates a KDC certificate. See the “[Creating a KDC Certificate](#)” section on page 12-42 for additional information.
 - -v—Verifies and normalizes the PacketCable certificate set. See the “[Validating the KDC Certificates](#)” section on page 12-43 for additional information.



Note

If you encounter difficulty using any of these options, you can specify a -? option to display all available help information on your computer screen.

- *[option]*—Implements optional functions that are dependent on the function selected above.

When you run the PKCert command, it will print a list of all errors encountered while performing the requested activities. You can use this printout to troubleshoot any problems that may have occurred.

Creating a KDC Certificate

Enter this command, from the /opt/CSCObpr/kdc directory, to create the KDC certificate:

```
PKCert.sh -s <dir> -d <dir> -c <cert> -e -r <realm> -a <name> -k <keyFile> [-n <serial>] [-o]
```

Where:

- -a <name>—specifies the DNS name of KDC
- -c <Cert File>—uses the service provider certificate (DER encoded)
- -d <directory>—specifies the destination directory
- -e—identifies the certificate as being a Euro-PacketCable certificate
- -k <Key File>—uses the service provider private key (DER encoded)
- -n <Serial#>—set the certificate serial number
- -o—overwrite existing files
- -r <Realm>—specifies the Kerberos realm for KDC certificate
- -s <directory>—specifies the source directory

When a new certificate is created and installed, the new certificate identifies the realm in the subject alternate name field. The new certificate is unique to its current environment in that it contains:

- The KDC realm
- The DNS name associated with this KDC that the MTA will use.

For example:

```
PKCert.sh -c "-s . \
-d /opt/CSCObpr/kdc/solaris/packetcable/certificates \
-k CLCerts/Test_LSCA_privkey.der \
-c CLCerts/Test_LSCA.cer \
-r PCTEST.CISCO.COM \
-n 100 \
-a kdc.pctest.cisco.com \
-o"
```

Using this command creates the files /opt/CSCObpr/kdc/solaris/packetcable/certificates/KDC.cer and /opt/CSCObpr/kdc/solaris/packetcable/certificates/KDC_private_key.pkcs8. The KDC certificate will have a realm set to PCTEST.CISCO.COM, a serial number set to 100, and the KDC server's FQDN is set to kdc.pctest.cisco.com.



Note

A console message is displayed after the successful completion of the command indicating that the file /opt/CSCObpr/kdc/solaris/packetcable/certificates/KDC_private_key.pkcs8 must be copied to /opt/CSCObpr/kdc/solaris/KDC_private_key.pkcs8. The command line option **-o** tells the utility that it should overwrite any pre-existing files.

Validating the KDC Certificates

This command examines all files in the source directory specified and attempts to identify them as X.509 certificates. If legitimate X.509 certificates are found, the files are properly renamed and copied to the destination directory. An error is generated whenever more than one legitimate chain of certificates for a particular purpose (service provider or device) is identified. If this occurs, you must remove the extra certificate from the source directory and run the command again.



Note

Usage instructions for PKCert are displayed on the screen when you enter the **PKCert.sh -v -?** command.

Enter this command, from the /opt/CSCObpr/kdc directory, to validate the KDC certificate:

```
PKCert.sh -v -s <dir> -d <dir> -o -r <dir>
```

Where:

- -s <directory>—specifies the source directory
- -d <directory>—specifies the destination directory
- -o—overwrites any existing files
- -r <directory>—specifies the reference certificate directory

Verification is performed against reference certificates built into this package. If the '-d' option is specified then certificate(s) are installed in the target directory with name normalization.

For example:

```
PKCert.sh -v \
"-s . \
-d /opt/CSCObpr/kdc/solaris/packetcable/certificates \
-o"
```

Using the changeNRProperties.sh Tool

The BACC installation program establishes values for configuration properties used by BACC extensions that are incorporated into the Network Registrar DHCP server. You use the **changeNRProperties.sh** command, which is found in the <BACC_HOME>/cnr_ep/bin directory, to change key configuration properties.

Invoking the script without any parameters will display a help message listing the properties that can be set.

To run this command:

Step 1 Change directory to <BACC_HOME>/cnr_ep/bin.

Step 2 Run the changeNRProperties.sh command:

```
changeNRProperties.sh <options>
```

Where *<options>* can include:

- **-help**—Displays this help message. The **-help** option must be used exclusively. Do not use this in conjunction with any other option.
- **-e <enabled|disabled>**—Sets the PacketCable enable property. Enter **-e enabled** to enable the property, and **-e disabled** to disable it.
- **-d**—Displays the current properties. The **-d** option must be used exclusively. Do not use this in conjunction with any other option.
- **-s <secret>**—Identifies the BACC shared secret. For example, enter **-s secret**, if the shared secret is the word *secret*.
- **-f <fqdn>**—Identifies the RDU's FQDN. For example, enter **-f rdu.cisco.com**, if you use rdu.cisco.com as the fully qualified domain name.
- **-p <port>**—Identifies the RDU port you want to use. For example, enter **-p 49187**, if you wanted to use port number 49187.
- **-r <realm>**—Identifies the PacketCable realm. For example, enter **-r CISCO.COM**, if your PacketCable realm is CISCO.COM.



Note The realm must be entered in uppercase letters.

- **-g <prov group>**—Identifies the provisioning group. For example, enter **-g group1**, if you are using provisioning group called group1.
- **-t <00|01>**—Identifies whether or not the PacketCable TGT is set to off or on. For example, enter **-t 00** to set this to off, or **-t 01** to set it to on.
- **-a <ip>**—Identifies the PacketCable primary DHCP server address. For example, enter **-a 10.10.10.2** if the IP address of your primary DHCP server is 10.10.10.2.
- **-b <ip>**—Identifies the PacketCable secondary DHCP server address. For example, enter **-b 10.10.10.4**, if the IP address of your secondary DHCP server is 10.10.10.4. You can also enter **-b null** to set a null value, if appropriate.
- **-y <ip>**—Identifies the PacketCable primary DNS server address. For example, enter **-y 10.10.10.6**, if the IP address of the PacketCable primary DNS server is 10.10.10.6.
- **-z <ip>**—Identifies the PacketCable secondary DNS server address. For example, enter **-z 10.10.10.8**, if the IP address of your secondary DNS server is 10.10.10.8. You can also enter **-z null** to set a null value, if appropriate.

Step 3 Restart the DHCP server.

Examples

This is an example of changing the Network Registrar extensions with the NR Extensions Properties tool:

```
# /opt/CSC0bpr/cnr_ep_bin/changeNRProperties.sh -g primary1
Current NR Properties:
RDU Port: 49187
RDU FQDN: rdu.acme.com
Provisioning Group: primary1
Shared Secret: fggTaLg0XwKRs
PacketCable Enable: enabled
PacketCable TGT: 01
```

```
PacketCable Realm: ACME.COM
PacketCable Primary DHCP Server: 192.168.1.2
PacketCable Secondary DHCP Server: NOT SET
PacketCable Primary DNS Server: 192.168.1.2
PacketCable Secondary DNS Server: NOT SET
```

**Note**

You must restart your NR DHCP server for the changes to take effect.

This is an example of viewing the current properties:

```
# /opt/CSCObpr/cnr_ep_bin/changeNRProperties.sh -d
Current NR Properties:
RDU Port: 49187
RDU FQDN: rdu.acme.com
Provisioning Group: primary1
Shared Secret: fggTaLg0XwKRs
PacketCable Enable: enabled
PacketCable TGT: 01
PacketCable Realm: ACME.COM
PacketCable Primary DHCP Server: 192.168.1.2
PacketCable Secondary DHCP Server: NOT SET
PacketCable Primary DNS Server: 192.168.1.2
PacketCable Secondary DNS Server: NOT SET
```

Using the Keygen Tool

The keygen tool is used to generate PacketCable service keys. The service keys are symmetric triple data encryption standard (triple DES or 3DES) keys (shared-secret) required for KDC communications. The KDC server requires service keys for each of the DPE's provisioning FQDNs.

The KDC server reads the service keys on startup. Any modification to the service keys requires the KDC server to be restarted. Any changes made to the DPE provisioning FQDN through the DPE CLI requires a corresponding change to the KDC service key filename. This applies since the KDC service key uses the DPE provisioning FQDN as part of its filename.

This tool, which is located in the <BACC_HOME>/kdc directory, uses command-line arguments for the DPE provisioning FQDN, realm name, and a password, and generates the service key files.

Syntax Description

Use this syntax when using the Keygen tool:

```
keygen [options] <fqdn> <realm> <password>
```

Where options include:

- **-?**—Displays this usage message and exits the command.
- **-v, -version**—Displays the version of this tool and exits the command.
- **-q, -quiet**—Implements a quiet mode whereby no output is created.
- **-c, -cms**—Creates a service key for the CMS system.
- **<fqdn>**—Identifies the DPE's fully qualified domain name and is a required entry.
- **<realm>**—Identifies the Kerberos realm and is a required entry.
- **<password>**—Specifies the password to be used. This is also a required field. The password must be between 6 and 20 characters in length. For example:

Three service key files are written in KDC keys directory using this filename syntax:

```
mtafqdnmap, <fqdn>@<REALM>
mtaprovsrvr, <fqdn>@<REALM>
krbtgt, <REALM>@<REALM>
```

The service key file always contains a version field of 0x0000.

Examples

This example illustrates how to generate service keys used for KDC to DPE communications. Enter this command:

```
bash-2.05b$ keygen dpe.cisco.com CISCO.COM changeme
```

When this command is implemented, these KDC service keys are written to the <BACC_HOME>/kdc/solaris/keys directory:

```
mtafqdnmap, dpe.cisco.com@CISCO.COM
mtaprovsrvr, dpe.cisco.com@CISCO.COM
krbtgt, CISCO.COM@CISCO.COM
```



Note The KDC must be restarted before the new keys are recognized.

Use this BPR agent command to restart the KDC:

```
/etc/init.d/bprAgent restart kdc
```

This example illustrates the generation of a CMS service key. Enter this command:

```
bash-2.05b$ keygen -c cms-fqdn.com CMS-REALM-NAME changeme
```

When this command is implemented, this CMS service key is written to the <BACC_HOME>/kdc/solaris/keys directory.

```
cms, cms-fqdn.com@CMS-REALM-NAME
```



Note When running this tool, enter the same password used for the PacketCable registration, kdc-service-key command. See the *Cisco Broadband Access Center for Cable Command Line Reference* for additional information.

Using the snmpAgentCfgUtil.sh Command

You can use the **snmpAgentCfgUtil.sh** command to manage the SNMP agent installed on a Solaris computer. Using this command, which is located in the <BACC_HOME>/snmp/bin directory, you can add (or remove) your host to a list of other hosts that receive SNMP notifications, and start and stop the SNMP agent process. This command should be run from the local directory.



Note The default port number of an SNMP agent running on a Solaris computer, is 8001.

You can use the RDU SNMP agent to:

- [Adding a Host, page 12-47](#)
- [Deleting a Host, page 12-47](#)

- [Adding an SNMP Agent Community, page 12-48](#)
- [Deleting an SNMP Agent Community, page 12-48](#)
- [Starting the SNMP Agent, page 12-49](#)
- [Stopping the SNMP Agent, page 12-49](#)
- [Changing the SNMP Agent Location, page 12-50](#)
- [Setting Up SNMP Contacts, page 12-50](#)
- [Listing SNMP Agent Settings, page 12-50](#)

Adding a Host

This command adds the host address to the list of hosts that receive SNMP notifications from the SNMP agent.

Syntax Description

To add a host, enter:

```
> snmpAgentCfgUtil.sh add host <host-addr> community <community> [udp-port <port>]
```

Where:

- *<host-addr>*—specifies either the IP address or FQDN of the host that you want to add to the list of hosts
- *<community>*—specifies the community (read or write) to be used while sending SNMP notifications
- *<port>*—identifies the UDP port used for sending the SNMP notifications

Examples

This example shows how to use the `snmpAgentCfgUtil.sh` command to add a host:

```
> snmpAgentCfgUtil.sh add host test.cisco.com community trapCommunity udp-port 162
OK
Please restart [stop and start] SNMP agent.
```

Deleting a Host

You can remove a host from the list of those receiving SNMP notifications from the SNMP agent.

Syntax Description

To delete a host, enter:

```
snmpAgentCfgUtil.sh delete host <host-addr>
```

Where:

- *<host-addr>*—Specifies the IP address of the host that you want to delete from the list of hosts.

Examples

This example shows how to use the `snmpAgentCfgUtil.sh` command to delete a host:

```
> ./snmpAgentCfgUtil.sh delete host test.cisco.com
OK
Please restart [stop and start] SNMP agent.
```

Adding an SNMP Agent Community

You can add an SNMP community string to allow access to the SNMP agent.

Syntax Description

To add the community string, enter:

```
snmpAgentCfgUtil.sh add community string [ro | rw]
```

Where:

- `<string>`—Identifies the SNMP community.
- `<ro>`—Assigns a read only (ro) community string. Only `get` requests (queries) can be performed. The ro community string allows `get` requests, but no `set` operations. The NMS and the managed device must reference the same community string.
- `<rw>`—Assigns a read write (rw) community string. SNMP applications require read-write access for `set` operations. The rw community string enables write access to OID values.

**Note**

The default ro and rw community strings are `bprread` and `bprwrite` respectively. Cisco recommends that you change these values before deploying BACC.

Examples

This example shows how to add an SNMP agent community string:

```
> snmpAgentCfgUtil.sh add community fsda54 ro
OK
Please restart [stop and start] SNMP agent.
```

Deleting an SNMP Agent Community

You can delete an SNMP community string to prevent access to the SNMP agent.

Syntax Description

To delete the community string, enter:

```
snmpAgentCfgUtil.sh delete community string [ro | rw]
```

Where:

- `<string>`—identifies the SNMP community
- `<ro>`—assigns a read only (ro) community string
- `<rw>`—assigns a read write (rw) community string

**Note**

See the “Adding an SNMP Agent Community” section on page 12-48 for additional information on the ro and rw community strings.

Examples

This example shows how to delete an SNMP agent community string:

```
> snmpAgentCfgUtil.sh delete community fsda54 ro
OK
Please restart [stop and start] SNMP agent.
```

Starting the SNMP Agent

This command starts the SNMP agent process on any Solaris computer having BACC already installed.

Syntax Description

To start the SNMP agent process, enter:

```
snmpAgentCfgUtil.sh start
```

Stopping the SNMP Agent

This command stops the SNMP agent process on any Solaris computer having BACC already installed.

Syntax Description

To stop the SNMP agent process, enter:

```
snmpAgentCfgUtil.sh stop
```

Configuring an SNMP Agent Listening Port

This command specifies the port number that the SNMP agent will listen to. The default port number used by RDU SNMP agent is 8001.

Syntax Description

To establish a port for the SNMP agent to listen to, enter:

```
snmpAgentCfgUtil.sh udp-port <port>
```

Where <port> identifies the port number that the SNMP agent will listen to.

Changing the SNMP Agent Location

This command lets you enter a string of text that you can use to indicate the location of the device running the SNMP agent. This could, for example, be used to identify the physical location of the device. You can enter anything that you like provided that the location is less than 255 characters long.

Syntax Description To enter the location of the SNMP agent, enter:

```
snmpAgentCfgUtil.sh location <location>
```

Where *<location>* is the character string identifying the agents location.

Examples In this example, the physical location of the SNMP agent is in an equipment rack identified as rack 5D:

```
snmpAgentCfgUtil.sh location "equipment rack 5D"
```

Setting Up SNMP Contacts

This command lets you enter a string of text that you can use to identify the contact person for the SNMP agent, together with information on how to contact this person. This could, for example, be used to identify a specific person including that person's telephone number. You can enter anything that you like up to 255 characters long.

Syntax Description Use the following syntax to enter the specific SNMP agent contact information:

```
snmpAgentCfgUtil.sh contact <contact-info>
```

Where *<contact-info>* is the character string identifying the individual to contact concerning the SNMP agent.

Examples In this example, the contact name is Ace Duffy and the telephone extension is 1234:

```
snmpAgentCfgUtil.sh contact "Ace Duffy - ext 1234"
```

Listing SNMP Agent Settings

Usage Guidelines This command lets you display all current SNMP settings.

Syntax Description To display all current SNMP settings, enter:

```
snmpAgentCfgUtil.sh show
```

Examples

This sample output could be displayed after running this command.

```
# ./snmpAgentCfgUtil.sh show
Location : Washington_1
Contact : John
Port Number : 8001
Notification Type : trap
Notification Recipient Table :
[ Host IP address, Community, UDP Port ]
[ 10.10.10.1, public , 162 ]
Access Control Table :
Read Only Communities
baccread
Read Write Communities
baccwrite
```

Specifying SNMP Notification Types

Usage Guidelines

This command lets you specify which types of notifications (traps or informs) will be sent from the SNMP agent. By default, traps are sent although you can set this to send SNMP informs instead.

Syntax Description

Use this syntax to run this command:

```
snmpAgentCfgUtil.sh inform [retries timeout] |trap
```

Where the parameter is the back off timeout between retries.

Using the `disk_monitor.sh` Tool to Monitor Available Disk Space

Monitoring available disk space is an important system administration task. You can use a number of custom written scripts or commercially available tools to do so.

The `disk_monitor.sh` command, located in the `<BACC_HOME>/rdu/sample/tools` directory, sets threshold values for one or more file systems. When these thresholds are surpassed, an alert is generated through the Solaris syslog facility, at 60-second intervals, until additional disk space is available.



Note

Cisco recommends that, at a minimum, you use the `disk_monitor.sh` script to monitor the `<BACC_DATA>` and `<BACC_DBLOG>` directories.

To monitor available disk space:

```
# ./disk_monitor.sh (file system-directory) (x)
```

Where:

- *(file system-directory)*—identifies any directory in a file system to monitor.
- *(x)*—identifies the percentage threshold applied to the specified file system.

Examples

Assume that you want to be notified when a file system (/var/CSCObpr for example) with database logs reaches 80% of its capacity. Enter the command using this syntax:

```
# ./disk_monitor.sh /var/CSCObpr 80
```

When the database logs disk space reaches 80% capacity, an alert is sent to the syslog file:

```
Dec 7 8:16:03 perf-u80-1 BPR: [ID 702911 local6.warning] File system /var/bpr usage is 81%
(threshold is 80%)
```

Troubleshooting Devices by MAC Address

You can use node management to troubleshoot a specific device, or a group of devices without turning logging on, and without searching through log files for device- or group-specific information.

BACC maintains a list of devices, based on MAC address, that contains detailed device information for troubleshooting problems. Troubleshooting information is stored centrally at the RDU and is maintained on a per-device basis; neither the DPE or Network Registrar extensions store this data. Rather, they forward this information to the RDU which, upon receiving information writes it to the appropriate device log file. If the connection from either the DPE or Network Registrar extension to the RDU is lost, this information is discarded until that connection is restored.

**Note**

Any modifications to this list, such as the addition of a new device or group, take place immediately; there is no need to reboot the device.

The DPE maps all device MAC addresses to its IP address mapping for that device, and the Network Registrar extensions send the IP update to the DPE whenever the extensions determine that device troubleshooting is enabled.

You can manage the device tracking list using the administrator's GUI to configure the number of devices that can be tracked at any given time; the default value is 100 devices. The tracking feature is considered to be turned off until a device is listed. See [Device Management, page 9-3](#) for more information.

**Caution**

Additional memory and disk space is required whenever this feature is used. As the number of tracked devices increases, so does the amount of memory and disk space that is required to support the number of logs that are created.



Alert and Error Messages

This appendix identifies all alert and error messages that Broadband Access Center for Cable (BACC) generates.

Alert Messages

BACC generates alerts through the UNIX syslog service. Syslog is a client-server protocol that manages the logging of information on UNIX. BACC syslog alerts are not a logging service. They provide a notification that a problem exists, but do not necessarily define the specific cause of the problem. This information might be found in the appropriate BACC log files.

Message Format

This is the format of alert messages generated by BACC:

XXX-#-###: <Message>

Where:

- *XXX*—Identifies the facility code. These can include:
 - RDU (regional distribution unit)
 - DPE (device provisioning engine)
 - AGENT (rduSnmpAgent or dpeSnmpAgent)
 - NR_EP (Network Registrar extension points)
 - KDC
- *#*—Identifies the severity level in use. There are three levels of alerts: 1, which is alert, 3 which is error, and 6 which identifies informational messages.
- *###*—Identifies the numeric error code as described in the following sections.
- <Message>—Identifies the alert text or message.

RDU Alerts

Whenever an RDU syslog alert is sent, additional details (if any) can be found in log file <BACC_DATA>/rdu/logs/rdu.log. [Table A-1](#) identifies the RDU alerts.

Table A-1 RDU Alerts

Alert	Description
RDU-1-101: RDU ran out of disk space	This alert indicates that the hard drive partition used by the RDU server has run out of space. You can remove or compress some of the log files. See Chapter 12, “Broadband Access Center for Cable Support Tools and Advanced Concepts,” for additional information on upgrading the disk.
RDU-1-103: RDU ran out of memory	This alert tells you that the RDU has run out of memory.
RDU-1-111: Evaluation key for technology [<i><technology_name></i>] expired	This alert appears whenever an evaluation key for the technology specified expires. You must contact Cisco sales or TAC for a new license key.
RDU-1-115: You have used [<i><percent></i>]% of available [<i><technology_name></i>] licenses.	This alert identifies the quantity of licences used (in percentage) out of the total number of allowable licenses. This alert starts to appear when you reach 80% of the license capacity.
BPR-RDU-4-1140: DNS took [X] seconds for lookup of address [10.0.0.1/test.com]; Check DNS configuration and health of servers	This alert is used to indicate that BACC performance is slowing down. The alert is generated whenever IP Address loop up takes more than 60 seconds.

Solaris DPE Alerts

Whenever a DPE syslog alert is sent, you can find additional details in the DPE logs.

You can use the **show log** command to access the DPE logs. See *Cisco Broadband Access Center for Cable Command Line Reference* for additional information.

Some DPE errors are also propagated to the RDU server log files. You can find these in the <BACC_DATA>/rdu/logs/rdu.log file. [Table A-2](#) identifies the Solaris DPE alerts.

Table A-2 DPE Alerts

Alert	Description
DPE-1-102: DPE ran out of disk space	<p>This alert notifies you that the DPE hard drive is full. There are three different actions you can perform:</p> <ol style="list-style-type: none"> a. Clear out any excess support bundles that may reside on the disk. You can do this by moving those support bundles to another machine and then running the clear bundles command from the DPE CLI. b. Run the clear logs command from the DPE CLI, to clear more disk space. c. As a last resort, run the clear cache command from the DPE CLI. This will remove any cache files and force the DPE to resynchronize with the RDU server.
DPE-1-104: DPE ran out of memory	<p>This alarm indicates that the DPE process has run out of memory.</p> <p>Determine how many device configurations are on the DPE; the more device configurations that exist, the more memory is used. The way to reduce device configurations is to limit the number of provisioning groups, either primary or secondary, that the DPE serves.</p>
DPE-1-109: Failed to connect to RDU	<p>This alert indicates that the RDU cannot be contacted. You must:</p> <ol style="list-style-type: none"> a. Verify that the DPE network is configured and connected correctly. b. Check that the DPE is configured to connect to the proper RDU, and that the connecting port is configured properly, using the dpe rdu-server command. c. Check that the RDU process is running on the correct server and listening on the correct port. The DPE attempts to reconnect to the RDU process every few seconds until a connection is established.

Agent Alerts

Whenever a syslog alert is sent by the watchdog agent process, you can find error details (if any) in the <BACC_DATA>/agent/logs/agent_console.log file and the log files corresponding to the specific component mentioned in the alert (if any). For example, if you receive an alert similar to *The [rdu] unexpectedly terminated*, you would check the RDU server log file (<BACC_DATA>/rdu/logs/rdu.log) for additional information. [Table A-3](#) identifies the agent alerts.

Table A-3 Agent Alerts

Alert	Description
AGENT-3-9001: Failed to start the [<i><component></i>]	This alert indicates that the process has failed to start the specified component.
AGENT-3-9002: The [<i><component></i>] unexpectedly terminated	This alert indicates that the specified component, monitored by the agent process, has unexpectedly failed.
AGENT-3-9003: Failed to stop the [<i><component></i>]	This alert indicates that a component did not stop when the agent attempted to stop it gracefully.
AGENT-6-9004: The [<i><component></i>] has started	This alert is generated any time a component is successfully started by the agent.
AGENT-6-9005: The [<i><component></i>] has stopped	This alert is generated any time a component has been successfully stopped through the watchdog agent. This message is for informational purposes only.

The [*<component>*] variable presented in the agent alerts list shown in [Table A-3](#) represents any of these component values:

- rdu
- dpe
- jrun
- rduSnmpAgent
- dpeSnmpAgent
- kdc

Network Registrar Extension Point Alerts

Whenever a BAC Network Registrar extension point syslog alert is sent, you can find additional details in the Network Registrars log file.

Table A-4 Network Registrar Extension Point Alerts

Alert	Description
NR_EP-1-106: Failed to connect to RDU	<p>This alert notifies you that the Network Registrar server cannot connect to the RDU. You should verify that the RDU process is running and, if it is not already running, start the RDU.</p> <p>If the RDU is running, use the Network Registrar computer to ping the RDU. If you are unable to ping the RDU, fix the routing tables or other communication parameters, between the two devices.</p> <p>If this alert is frequently repeated, you may have an unstable connection between the two hosts. Use generally accepted network troubleshooting techniques to improve the connectivity between the two hosts.</p>
NR_EP-1-107: Failed to connect to any DPEs	<p>This alert notifies you that the Network Registrar extension cannot connect to the DPEs.</p> <p>Check that there are DPEs in the provisioning group for each Network Registrar extension. If not, change the Network Registrar provisioning group to one that has DPEs available. If DPEs are in the provisioning group, ensure that the Network Registrar extension has registered with the RDU, if it has not, it will not recognize any of the DPEs.</p> <p>If, after completing the check, the alert continues, check that there is network connectivity between the Network Registrar extension and the DPEs in the provisioning group.</p> <p>If this alert is frequently repeated, you may have an unstable connection between the two hosts. Use generally accepted network troubleshooting techniques to improve the connectivity between the two hosts.</p>
NR_EP-6-108: The BACC NR extensions have started	This alert notifies you that the Network Registrar extensions have been started.
NR_EP-6-109: The BACC NR extensions have stopped	This alert notifies you that the Network Registrar extensions have been stopped.

Table A-4 Network Registrar Extension Point Alerts (continued)

Alert	Description
NR_EP-6-110: Registered with RDU [<i>address and port</i>]	This alert notifies you that the Network Registrar extensions have been registered with the RDU. The [<i>address and port</i>] identifies the address of the RDU that has registered the Network Registrar extensions.

RDU Error Messages with CCM

The RDU error messages described in this section are accompanied by corrective actions that you perform to remedy specified problems. These error messages are located in the rdu.log file, and are a direct result of the Lease Reservation feature and consequently, if you are not using the Network Registrar Regional CCM feature these error messages should not appear.

BACC 2.7 generates three types of lease reservation related error messages, including:

- Failure to add a reservation request failure. These messages are preceded by the following information which identifies several aspects of the error:

```
rdu.cisco.com: 2005 02 09 13:25:11 EST: %BPR-RDU-3-1146: PACE-13: Failed to add reservation [10.10.10.1] for [1,6,03:03:14:00:00:21] using client-class [unprovisioned-docsis] and selection-tag [NULL]; [AddReservation: FORWARD_FAILED]
```

- Failure to remove a reservation request failure. These messages are preceded by the following information which identifies several aspects of the error:

```
rdu.cisco.com: 2005 02 09 13:25:11 EST: %BPR-RDU-3-1147: Failed to remove reservation [10.10.10.1] for [1,6,03:03:14:00:00:21];
```

- Selection-criteria exclusion tag errors. These messages are preceded by the following information which identifies several aspects of the error:

```
rdu.cisco.com: 2005 02 09 13:25:11 EST: %BPR-RDU-4-1145: The use of selection-criteria exclusion tags [black] is not allowed when adding a lease reservation. They will be ignored.
```

Where, from the preceding examples:

- rdu.cisco.com:—identifies the RDU FQDN
- 2005 02 09 13:25:11 EST:—Identifies the date and time that the error took place.
- %BPR-RDU-#-####:—Identifies the RDU error number (#) and error message ID (####).
- [10.10.10.1]—Identifies the IP address being reserved.
- 1,6,03:03:14:00:00:21]—Identifies the MAC address that is attempting to reserve an IP address. Note that the second example illustrates BACC's support of variable length MAC addresses.

Each of these examples contain a generalized description of the error as well as additional the error message that is returned from Network Registrar.

The following sections describe error messages generated by BACC:

[OBJECT_EXISTS]

A user is trying to make reservation [11.100.14.21] for mac [1,6,03:03:14:00:00:21], but the reservation request failed due to duplicate ip used (ip already being reserved by another device in the database).

Corrective Action:

Use a different IP address or free up the desired IP address.

[RemoveReservation: NOT_FOUND]

A user is trying to remove a reservation [11.100.0.103] for mac device [1,1,35]; the request failed due to non-existence reservation in CCM database. In other words, CCM can't find the reservation exist in the database.

Corrective Action:

Verify that the reservation does exist in the CCM database.

[AddReservation: INVALID_PARENT]

This means that CCM did not find any scope for the IP address of the reservation. CCM local has to find a scope that could contain the reservation. CCM displays this error message when reserved IP is out of scope. You will get this error message if you try to add a reservation to a nonexisting subnet or scope.

Corrective Action:

Recheck your DHCP server configuration (scope, subnet, and so on).

[AddReservation: INVALID_SECOND_PARENT]

This should only be returned by a local CCM. It means that no scope exist in the configuration could contain the reservation. Recheck your DHCP server configuration (scope, subnet, and so on).

You will get this error if you try to add a reservation to subnet that exists at the Regional cluster but no such scope/subnet exists at the local cluster(s):

Or CCM can't find a scope that both contains the reserved IP address and match clientclass/selection-tags criteria.

Corrective Action:

Recheck your DHCP server configuration (scope, subnet, clientclass, selection-tags, and so on).

[AddReservation: FORWARD_FAILED]

CCM regional was not able to forward the reservation request to any local CCM. This happens when the CCM can't find a scope that both contains the reserved IP address and match clientclass/selection-tags criteria.

The following two use cases illustrate solutions to potential causes of this error:

Example Case #1

Assume that there are three scopes with these attributes/tags; scope A is red and gray, scope B is blue and red, and scope C is blue and green. If the AddReservation API call specifies the inclusion of red and gray tags, but the scope that contains the IP address requested is scope B.

The outcome of this situation is that the reservation request fails and this error is logged into the rdu.log file.

Corrective Action

Use the same setup or configuration, but modify the AddReservation API call to specify the inclusion only red tags, rather than red and gray as described above. This should be successful and any scope that contains the IP address, and matches all of the selection tags, will be used.



Note

In this example, it is expected that the device gets the reservation from scope B.

Example Case #2

Assume that there are three scopes with these attributes/tags; scope A is red, scope B is blue, and scope C is green. Also assume that the AddReservation API call specifies a client class of test1, Network Registrar defines test1 with the green selection criteria, and the IP being requested is contained in the scope B.

The outcome of this situation is that the reservation request fails and this error is logged into the rdu.log file.

Corrective Action

Use the same setup or configuration, but reconfigure the test1 clientclass in Network Registrar with selection criteria blue. This should successfully correct the problem so that any scope containing IP address, and matching all of the selection tags, is used.

[AddReservation: AX_ETIME]

This indicates that a reservation request timeout has occurred. This could be due to the CCM being overloaded, or busy with tasks and requests, and was not able process the reservation request.

Corrective Action

Use the administrator's user interface to configure a longer timeout value. See the [“RDU Defaults” section on page 10-19](#) for information on submitting reservation requests.

[AddReservation: INVALID_OBJECT]

The reservation itself was invalid somehow: the MAC address was invalid, the IP address was missing, or it supplied an invalid client-class name or selection tags.

Corrective Action:

Check the reservation requests: the MAC address, IP address, client-class, selection tags, etc.

Selection-criteria exclusion tags will be ignored

The use of selection-criteria exclusion tags [black] is not allowed when adding a lease reservation. They will be ignored.

Corrective Action:

You should not configure selection-criteria exclusion tags in the DHCPCriteria since the use of selection-criteria exclusion tags is not allowed.

[AX_EIO]

This indicates that the connection, or session, session between the RDU and the CCM is broken.

Corrective Action:

There is no user interaction required when this error occurs. The RDU will automatically establish another connection to CCM.

[AX_EPIPE]

This indicates that the connection, or session, session between the RDU and the CCM is broken.

Corrective Action:

There is no user interaction required when this error occurs. The RDU will automatically establish another connection to CCM.



PacketCable DHCP Options to BACC Properties Mapping

This appendix identifies the mapping of BACC properties to the PacketCable DHCP options used for PacketCable provisioning.

The minimum required set of these properties is configured, during installation, in the `<BACC_HOME>/cnr_ep/conf/cnr_ep.properties` file. This file is located on the Network Registrar host. The set of properties defined in `cnr_ep.properties` is applied to all PacketCable voice technology devices in the provisioning group. Like other BACC properties, you can also set these properties on a device or a class of service. Setting them at the RDU, using either the administrator's user interface or the API, overrides the corresponding values set in the `cnr_ep.properties` file.



Note

See [“Using the changeNRProperties.sh Tool” section on page 12-43](#) for information on changing key these configuration properties.

BACC supports both PacketCable DHCP option 122 (as specified in RFC 3495 and 3594) and the deprecated PacketCable DHCP option 177. BACC does not ignore DHCP requests when it cannot populate option 122 and/or 177 content. Whatever option 122/177 content is available is populated and the decision to ignore the option is left to the EMTA.

When BACC receives a DHCP request asking for both option 122 and 177, BACC will ignore the request for option 177 and populate only option 122 content.



Caution

There should be only one instance of each property in `<BACC_HOME>/cnr_ep/conf/cnr_ep.properties`.

Option 122 and BACC Property Comparison

Table B-1 identifies the BACC properties, as they apply to the definition of option 122 in RFC-3495 and RFC-3594.

Table B-1 DHCP Option 122 to BACC Property Comparison

DHCP Option	Type	BACC Property Name
6	IP addr	/ccc/dns/primary
6	IP addr	/ccc/dns/secondary
122.1	IP addr	/ccc/dhcp/primary
122.2	IP addr	/ccc/dhcp/secondary
122.3	FQDN	/ccc/prov/fqdn Note Option 122.3 is automatically filled by BACC, consequently, do not set this property manually.
122.4	Integer	/ccc/kerb/auth/backoff/nomTimeout /ccc/kerb/auth/backoff/maxTimeout /ccc/kerb/auth/backoff/maxRetries
122.5	Integer	/ccc/kerb/app/backoff/nomTimeout /ccc/kerb/app/backoff/maxTimeout /ccc/kerb/app/backoff/maxRetries
122.6	String	/ccc/kerb/realm
122.7	Boolean	/ccc/tgt
122.8	Integer	/ccc/prov/timer
122.9	Integer	/ccc/security/ticket/invalidation



Caution

If any of /ccc/kerb/auth/backoff/nomTimeout, /ccc/kerb/auth/backoff/maxTimeout, or /ccc/kerb/auth/backoff/maxRetries are defined, they must all be defined. Similarly, if any of /ccc/kerb/app/backoff/nomTimeout, /ccc/kerb/app/backoff/maxTimeout, or /ccc/kerb/app/backoff/maxRetries are defined, they must all be defined.

Option 177 and BACC Property Comparison

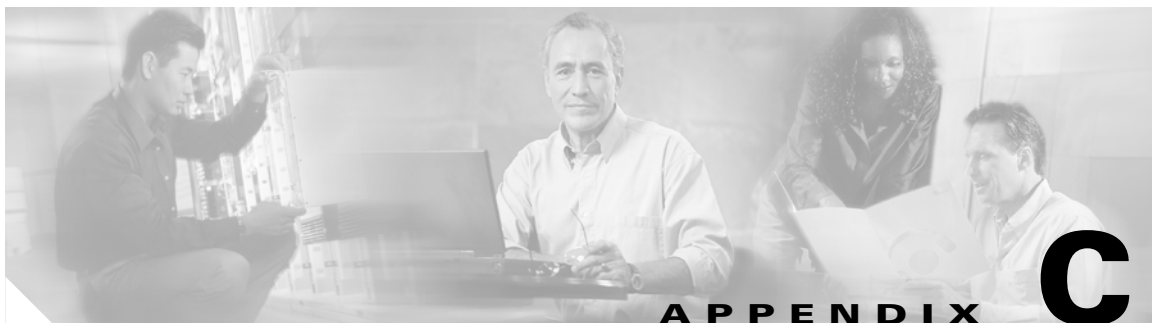
In accordance with PacketCable compliance wave 26, option 177 is deprecated, and option 122 is now the preferred MTA provisioning option. For legacy devices that still support option 177, Table B-2 identifies the BACC properties, as they apply to the definition of option 177

Table B-2 DHCP Option 177 to BACC Property Comparison

Option 177	Type	BACC Property Names
177.1	ip addr	/pkcbl/dhcp/primary
177.2	ip addr	/pkcbl/dhcp/secondary

Table B-2 DHCP Option 177 to BACC Property Comparison (continued)

Option 177	Type	BACC Property Names
177.3	fqdn	/pktcbl/snmp/entity/fqdn
177.4	ip addr	/pktcbl/dns/primary
177.5	ip addr	/pktcbl/dns/secondary
177.6	string	/pktcbl/snmp/realm
177.7	boolean	/pktcbl/snmp/tgt
177.8	integer	/pktcbl/provisioning/timer
177.10	integer	/pktcbl/kerberos/authentication/backoff/nomTimeout /pktcbl/kerberos/authentication/backoff/maxTimeout /pktcbl/kerberos/authentication/backoff/maxRetries
177.11	integer	/pktcbl/kerberos/application/backoff/nomTimeout /pktcbl/kerberos/application/backoff/maxTimeout /pktcbl/kerberos/application/backoff/maxRetries
177.12	integer	/pktcbl/snmp/kerberos/ticket/invalidation



Broadband Access Center for Cable Application Programming Interface Use Cases

This appendix presents a series of the most common provisioning API use cases, including pseudo-code segments that can be used to model typical service provider workflows. The pseudo code used in the use cases resembles java though it is not intended for direct compilation. Please refer to the BACC 2.7 API Javadoc for more details and sample Java code segments explaining individual API calls and features.

These use cases are directly related to device (and/or service) provisioning. Many administrative operations, such as managing class of services, DHCP criteria, and licenses are not addressed here. It is highly recommended to go through the API javadoc for more details on the related API calls. You can also use the administrator's user interface to perform most of these activities.

Use Cases

This appendix includes these use cases:

- [Self-Provisioned Modem and Computer in Fixed Standard Mode, page C-2](#)
- [Adding a New Computer in Fixed Standard Mode, page C-5](#)
- [Disabling a Subscriber, page C-7](#)
- [Pre-Provisioning Modems/Self-Provisioned Computers, page C-9](#)
- [Modifying an Existing Modem, page C-10](#)
- [Unregistering and Deleting a Subscriber's Devices, page C-11](#)
- [Self-Provisioning First-Time Activation in Promiscuous Mode, page C-14](#)
- [Bulk Provisioning 100 Modems in Promiscuous Mode, page C-16](#)
- [Pre-Provisioning First-Time Activation in Promiscuous Mode, page C-18](#)
- [Replacing an Existing Modem, page C-19](#)
- [Adding a Second Computer in Promiscuous Mode, page C-20](#)
- [Self-Provisioning First-Time Activation with NAT, page C-20](#)
- [Adding a New Computer Behind a Modem with NAT, page C-22](#)
- [Move Device to Another DHCP Scope, page C-23](#)
- [Log Device Deletions Using Events, page C-24](#)
- [Monitoring an RDU Connection Using Events, page C-25](#)

- [Logging Batch Completions Using Events, page C-26](#)
- [Getting Detailed Device Information, page C-26](#)
- [Searching Using the Default Class of Service, page C-27](#)
- [Retrieving Devices Matching a Vendor Prefix, page C-29](#)
- [Pre-Provisioning PacketCable eMTA, page C-31](#)
- [SNMP Cloning on PacketCable eMTA, page C-33](#)
- [Incremental Provisioning of PacketCable eMTA, page C-34](#)
- [Pre-Provisioning DOCSIS Modems with Dynamic Configuration Files, page C-36](#)
- [Optimistic Locking, page C-38](#)
- [Temporarily Throttling a Subscriber's Bandwidth, page C-40](#)
- [Pre-Provisioning CableHome WAN-Man, page C-41](#)
- [CableHome with Firewall Configuration, page C-42](#)
- [Retrieving Device Capabilities for CableHome WAN-Man, page C-44](#)
- [Self-Provisioning CableHome WAN-Man, page C-46](#)
- [Lease Reservation Use Cases, page C-48](#)

Self-Provisioned Modem and Computer in Fixed Standard Mode

The subscriber has a computer installed in a single dwelling unit and has purchased a DOCSIS cable modem. The computer has a web browser installed.

Desired Outcome

Use this workflow to bring a new unprovisioned DOCSIS cable modem and computer online with the appropriate level of service.

-
- Step 1** The subscriber purchases and installs a DOCSIS cable modem at home and connects a computer to it.
 - Step 2** The subscriber powers on the modem and the computer, and BACC gives the modem restricted access. The computer and modem are assigned IP addresses from restricted access pools.
 - Step 3** The subscriber starts a web browser, and a spoofing DNS server points the browser to a service provider's registration server (for example, an OSS user interface or a mediator).
 - Step 4** The subscriber uses the service provider's user interface to complete the steps required for registration, including selecting class of service.

Step 5 The service provider's user interface passes the subscriber's information, such as the selected class of service and computer IP address, to BACC, which then registers the subscriber's modem and computer.

```
// First we query the computer's information to find the modem's
// MAC address. We use the computers IP address (the web browser
// received this when the subscriber opened the service providers
// web interface)

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a query
// NO_CONFIRMATION is the confirmation mode because we are not
// attempting to reset the device

LeaseResults computerLease =

    getAllForIPAddress("10.0.14.38");
    // ipAddress: restricted access.

// Derive the modem MAC address from the computer's network
// information. The 1,6, is a standard prefix for an Ethernet
// device. The fully qualified MAC address is required by BACC

String modemMACAddress = "1,6," +
    computerLease.getSingleLease().get(RELAY_AGENT_REMOTE_ID);

// Now let's provision the modem and the computer in the same
// batch. This can be done because the activation mode of this
// batch is NO_ACTIVATION. More than one device can be operated
// on in a batch if the activation mode does not lead to more
// than one device being reset. NO_ACTIVATION will generate a
// configuration for the devices. However it will not attempt
// to reset the devices. The configuration can be generated
// because the devices have booted. NO_CONFIRMATION is the
// confirmation mode because we are not attempting to reset
// the modem. We do not want to reset the modem here because
// we want to notify the user to reset their computer. If we
// reset the modem in this batch, we will not be able to notify
// the user of anything until the modem has come back online.
// To add a DOCSIS modem:

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

add(
    DeviceType: DOCSIS,
        // deviceType: DOCSIS
    modemMACAddress,
        // macAddress: derived from computer lease
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "Silver",
```

```

        // ClassOfService
        "provisionedCM",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
        null
        // properties: not used
    );
    // properties: not used
String computerMACAddress = computerLease.
    getSingleLease().get(DeviceDetailsKeys.MAC_ADDRESS);

// Create a Map for the computer's properties
Map properties;

// Setting the property IPDeviceKeys.MUST_BE_BEHIND_DEVICE
// on the computer ensures that when the computer boots it
// will only receive its provisioned access when it is behind
// the given device. If it is not behind the given device,
// it will receive default access (unprovisioned). This makes
// the computer "fixed" behind the specified modem.
properties.put(IPDeviceKeys.MUST_BE_BEHIND_DEVICE,
    modemMACAddress);

add(
    DeviceType.COMPUTER,
        // deviceType: COMPUTER
    computerMACAddress,
        // macAddress: derived from computer lease
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    null,
        // classOfService : get the default COS
    "provisionedCPE",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    properties
        // properties:
    );

```

Step 6 The user interface prompts the subscriber to reboot the computer.

- Step 7** The provisioning client calls *performOperation(DeviceOperation.RESET, modemMACAddress, null)* to reboot the modem and gives the modem provisioned access.

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION);

// AUTOMATIC is the activation mode because we are attempting
// to reset the modem so that it receives its new class of service.
// NO_CONFIRMATION is the confirmation mode because we don't
// want the batch to fail if we can't reset the modem. The user
// may have power cycled the modem when they rebooted their computer.
// Send a batch to reset the modem now that the user has been
// notified to reboot their computer.

performOperation(
    DeviceOperation.RESET,
        //deviceOperation: Reset operation
    modemMACAddress,
        // macAddress:Modem's MAC address
    null
        // properties: not used
);
```

- Step 8** After rebooting, the computer receives a new IP address, and both cable modem and computer are now provisioned devices. The computer has access to the Internet through the service provider's network.
-

Adding a New Computer in Fixed Standard Mode

A multiple system operator (MSO) lets a subscriber have two computers behind a cable modem. The subscriber has one computer already registered and then brings home a laptop from work and wants access. The subscriber installs a hub and connects the laptop to it.

Desired Outcome

Use this workflow to bring a new unprovisioned computer online with a previously provisioned cable modem so that the new computer has the appropriate level of service.

- Step 1** The subscriber powers on the new computer and BACC gives it restricted access.
- Step 2** The subscriber starts a web browser on the new computer and a spoofing DNS server points it to the service provider's registration server (for example, an OSS user interface or a mediator).
- Step 3** The subscriber uses the service provider's user interface to complete the steps required to add a new computer.

Step 4 The service provider's user interface passes the subscriber's information, such as the selected class of service and computer IP address, to BACC, which then registers the subscriber's modem and computer.

```
// First we query the computer's lease information to its
// MAC address. We use the computers IP address (the web browser
// received this when the subscriber opened the service providers
// web interface)

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a query
// NO_CONFIRMATION is the confirmation mode because we are not
// attempting to reset the device.

LeaseResults computerLease =
    getAllForIPAddress("10.0.14.39");
    // ipAddress: restricted access.

String computerMACAddress = computerLease.
    getSingleLease().get(DeviceDetailsKeys.MAC_ADDRESS);

// We have the MAC address now. Let's add the computer to BACC.

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION will generate a configuration for the computer.
// However it will not attempt to reset the computer (this
// can not be done). The configuration can be generated because
// the device has booted. NO_CONFIRMATION is the confirmation
// mode because we are not attempting to reset the modem.

Map properties; // Map containing the properties for the computer

// Setting the property IPDeviceKeys.MUST_BE_BEHIND_DEVICE on
// the computer ensures that when the computer boots, it will
// only receive its provisioned access when it is behind the
// given device. If it is not behind the given device, it
// will receive default access (unprovisioned) and hence the
// fixed mode.

properties.put(IPDeviceKeys.MUST_BE_BEHIND_DEVICE, modemMACAddress);

// The IPDeviceKeys.MUST_BE_IN_PROV_GROUP property ensures that
// the computer will receive its provisioned access only when
// it's brought up in the specified provisioning group. This prevents
// the computer (and/or the modem) from moving from one locality to
// to another locality.

properties.put(IPDeviceKeys.MUST_BE_IN_PROV_GROUP, "bostonProvGroup");

add(
    DeviceType.COMPUTER,
        // deviceType: COMPUTER
    computerMACAddress,
        // macAddress: derived from computer lease
    null,
        // hostName: not used in this example
    null,
```



```

        // domainName: not used in this example
"0123-45-6789",
        // ownerID: here, account number from billing system
null,
        // classOfService: get the default COS
"provisionedCPE",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
properties
        // properties:
);

```

- Step 5** The user interface prompts the subscriber to reboot the new computer so that BACC can give the computer its registered service level.
- Step 6** The computer is now a provisioned device with access to the appropriate level of service.
-

Disabling a Subscriber

A service provider needs to disable a subscriber from accessing the Internet due to recurring non-payment.

Desired Outcome

Use this workflow to disable an operational cable modem and computer, so that the devices temporarily restrict Internet access for the user. Additionally, this use case can redirect the user's browser to a special page that could announce:

```
You haven't paid your bill so your Internet access has been disabled.
```

- Step 1** The service provider's application uses a provisioning client program to request a list of all of the subscriber's devices from BACC. The service provider's application then uses a provisioning client to individually disable or restrict each of the subscriber's devices.

```

// The service provider's application uses a provisioning client
// to get a list of all of the subscriber's devices.

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a query.
// NO_CONFIRMATION is the confirmation mode because we are not
// attempting to reset the devices.

getAllForOwnerID(

"0123-45-6789"); // Query all the devices for this account number
// For each device in the retrieved list:

```

```

// We are only interested in the modems. If we disable network
// access of the modems, we disable network access for all the

// subscriber's devices. If we are using roaming standard mode,
// we may also change the computer's network access (DHCPCriteria)
// because the subscriber may still be able to access the network
// from a different modem.

DeviceLoop:
{
    if (Device.deviceType == DOCSIS_MODEM)
    {
        get-new-batch(AUTOMATIC, NO_CONFIRMATION)

        // AUTOMATIC is the activation mode because we are
        // attempting to reset the modem so that becomes
        // disabled. NO_CONFIRMATION is the confirmation mode
        // because we do not want the batch to fail if we cannot
        // reset the modem. If the modem is off, it will
        // be disabled when it is turned back on.

        // Let's change the COS of the device so that it will restrict
        // the bandwidth usage of the modem.

        changeClassOfService(
            Device.MAC_ADDRESS,
            // macAddress: unique identifier for this modem
            "DisabledCOS");
            // newClassOfService: restricts bandwidth usage

        // Let's change the DHCP Criteria so that the modem will get an IP
        // address from a disabled CNR scope. This scope also points to
        // a spoofing DNS server so that the subscriber gets a restricted
        // access page.

        changeDHCPCriteria(
            Device.MAC_ADDRESS,
            // macAddress: unique identifier for this modem
            "DisabledDHCPCriteria");
            // newDHCPCriteria: disables Internet access
        }
    }
}
// end DeviceLoop

```

**Note**

You may need to consider the impact on the CPEs behind the modem when defining the characteristics of DisabledCOS and resetting the modem. This is especially important if you have voice end points behind the modem, because disrupting the cable modem might affect the telephone conversation in progress at that time.

The subscriber is now disabled.

Pre-Provisioning Modems/Self-Provisioned Computers

A new subscriber contacts the service provider and requests service. The subscriber has a computer installed in a single dwelling unit. The service provider pre-provisions all its cable modems in bulk.

Desired Outcome

Use this workflow to bring a pre-provisioned cable modem, and an unprovisioned computer, online in the roaming standard mode. This must be done so that both devices have the appropriate level of service and are registered.

-
- Step 1** The service provider chooses a subscriber username and password for the billing system.
 - Step 2** The service provider selects services that the subscriber can access.
 - Step 3** The service provider's field technician installs the physical cable to the new subscriber's house and installs the pre-provisioned device, connecting it to the subscriber's computer.
 - Step 4** The technician turns on the modem and BACC gives it a provisioned IP address.
 - Step 5** The technician turns on the computer and BACC gives it a private IP address.
 - Step 6** The technician starts a browser application on the computer and points the browser to the service provider's user interface.
 - Step 7** The technician accesses the service provider's user interface to complete the steps required for registering the computer behind the provisioned cable modem.

```
// To provision a computer:
// First we query the computer's lease information to its
// MAC address. We use the computers IP address (the web browser
// received this when the subscriber opened the service provider's
// web interface)

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a query
// NO_CONFIRMATION is the confirmation mode because we are not
// attempting to reset the device

LeaseResults computerLease =
    getAllForIPAddress("10.0.14.38");
    // ipAddress:

String computerMACAddress = computerLease.
    getSingleLease().get(DeviceDetailsKeys.MAC_ADDRESS);

// MSO admin UI calls the provisioning API to provision a computer.

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION will generate a new configuration for the computer
// however it will not attempt to reset it.
// NO_CONFIRMATION is the confirmation mode because we are not
// attempting to reset the computer because this cannot be done.

add(
    DeviceType.COMPUTER,
        // deviceType: Computer
    computerMACAddress,
```

```

        // macAddress: derived from computer lease
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    null,
        // ClassOfService : get the default COS
    "provisionedCPE",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    null
        // properties: not used
);

```

**Note**

The IPDeviceKeys.MUST_BE_BEHIND_DEVICE property is not set on the computer and this allows roaming from behind one cable modem to another.

Step 8

The technician restarts the computer and the computer receives a new provisioned IP address. The cable modem and the computer are now both provisioned devices. The computer has access to the Internet through the service provider's network.

Modifying an Existing Modem

A service provider's subscriber currently has a level of service known as **Silver** and has decided to upgrade to **Gold** service. The subscriber has a computer installed at home.

**Note**

The intent of this use case is to show how to modify a device. You can apply this example to devices provisioned in modes other than roaming standard.

Desired Outcome

Use this workflow to modify an existing modem's class of service and pass that change of service to the service provider's external systems.

Step 1

The subscriber phones the service provider and requests to have service upgraded. The service provider uses its user interface to change the class of service from **Silver** to **Gold**.

Step 2 The service provider's application makes these API calls in BACC:

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION)

// AUTOMATIC will generate a configuration for the device
// and will attempt to reset the device
// The configuration will be able to be generated because
// the device has booted. NO_CONFIRMATION is the confirmation
// mode because we don't want the batch to fail if the reset failed.
// This use case is a perfect example of the different
// confirmation modes that could be used instead of
// NO_CONFIRMATION. These confirmation modes give you
// a method to test whether a configuration was taken by
// a device. Also, these modes will take more time because
// the batch has to wait for the modem to reset.

changeClassOfService(
    "1,6,00:11:22:33:44:55",
        // macAddress: unique identifier for this modem
    "Gold"
        // newClassOfService
);
```

Step 3 The subscriber can now access the service provider's network with the *Gold* service.

Unregistering and Deleting a Subscriber's Devices

A service provider needs to delete a subscriber who has discontinued service.

Desired Outcome

Use this workflow to permanently remove all the subscriber's devices from the service provider's network.

Step 1 The service provider's user interface discontinues service to the subscriber.

Step 2 The service provider's application uses a provisioning client program to request a list of all the subscriber's devices from BACC, and unregisters and resets each device so that it is brought down to the default (unprovisioned) service level.



Note

If the device specified as the parameter to the "unregister" API is already in unregistered state then the status code from the API call will be set to `CommandStatusCodes .CMD_ERROR_DEVICE_UNREGISTER_UNREGISTERED_ERROR`. This is normal/expected behavior.

```
// MSO admin UI calls the provisioning API to get a list of
// all the subscriber's devices.

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)
```

```

// NO_ACTIVATION is the activation mode because this is a query.
// NO_CONFIRMATION is the confirmation mode because we are
// not attempting to reset the devices.

getAllForOwnerID(
    "0123-45-6789");
    // query all the devices for this account number

// We need to unregister all the devices behind each modem(s) or else the
// unregister call for that modem will fail.

// for each computer in the retrieved list:
DeviceLoop:
{

    if (Device.deviceType == COMPUTER)
    {

        get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

        // NO_ACTIVATION is the activation mode because we can't
        // to reset Computers.
        // NO_CONFIRMATION is the confirmation mode because
        // the unregister call will not reset the devices.

        unregister(
            Device.MAC_ADDRESS,
            // macAddress: unique identifier for this device
        );

    }

}

// for each modem in the retrieved list:
DeviceLoop:
{

    if (Device.deviceType == DOCSIS_MODEM)
    {

        get-new-batch(AUTOMATIC, NO_CONFIRMATION)

        // AUTOMATIC is the activation mode because we want
        // to reset as we unregister the device.

        unregister(
            Device.MAC_ADDRESS,
            // macAddress: unique identifier for this device
        );

    }

}

// end DeviceLoop.

```

**Note**

The next step is optional as some service providers prefer to keep the cable modem in the database unless it is broken. This step needs to be run only if the devices need to be deleted from the database.

Step 3 The service provider's application uses a provisioning client program to delete each of the subscriber's remaining devices individually from the database.

```
// MSO admin UI calls the provisioning API to get a list of
// all the subscriber's devices.

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a query.
// NO_CONFIRMATION is the confirmation mode because we are
// not attempting to reset the devices.

getAllForOwnerID(
    "0123-45-6789");
    // query all the devices for this account number
// for each device in the retrieved list:

DeviceLoop:
{
    // get a new batch for each modem being deleted

    if (Device.deviceType == DOCSIS_MODEM)
    {
        get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

        // NO_ACTIVATION is the activation mode because we don't want
        // to reset as we are deleting the device.
        // NO_CONFIRMATION is the confirmation mode because
        // the delete call will not reset the devices.

        delete(
            Device.MAC_ADDRESS,
            // macAddress: unique identifier for this device
            true
            // deleteDevicesBehind: deletes CPEs behind this modem.
        );
    }
}
//end DeviceLoop.
```

Self-Provisioning First-Time Activation in Promiscuous Mode

The subscriber has a computer (with a browser application) installed in a single dwelling unit and has purchased a DOCSIS cable modem.

Desired Outcome

Use this workflow to bring a new unprovisioned DOCSIS cable modem and computer online with the appropriate level of service.

-
- Step 1** The subscriber purchases a DOCSIS cable modem and installs it at home.
 - Step 2** The subscriber powers on the modem, and BACC gives it restricted access.
 - Step 3** The subscriber starts a browser application on the computer and a spoofing DNS server points the browser to the service provider's registration server (for example, an OSS user interface or a mediator).
 - Step 4** The subscriber uses the service provider's user interface to complete the steps required for registration, including selecting a class of service.

The service provider's user interface passes the subscriber's information to BACC, including the selected classes of service and computer IP address. The subscriber's cable modem and computer are then registered with BACC.

```
get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a
// query. NO_CONFIRMATION is the confirmation mode because
// we are not attempting to reset the device.
// First we query the computer's information to find the
// modems MAC address.
// We use the computer's IP address (the web browser
// received this when the subscriber opened the service
// provider's web interface). We also assume that "bostonProvGroup"
// is the provisioning group used in that locality.

List provGroupList;
provGroupList = provGroupList.add("bostonProvGroup");
Map computerLease = getAllForIPAddress(
    "10.0.14.38",
    // ipAddress: restricted access computer lease
    provGroupList
    // provGroups: List containing provgroup)

// Derive the modem MAC address from the computer's network
// information. The 1,6, is a standard prefix for an Ethernet
// device. The fully qualified MAC address is required by BACC

String modemMACAddress = "1,6," +
    computerLease.GetSingleLease().get(RELAY_AGENT_REMOTE_ID);

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// Now let's provision the modem
// NO_ACTIVATION will generate a configuration for the
// modem, however it will not attempt to reset it
```



```

// The configuration will be able to be generated because
// the modem has booted.
// NO_CONFIRMATION is the confirmation mode because we
// are not attempting to reset the modem
// Create a Map for the properties of the modem

Map properties;
// Set the property ModemKeys.PROMISCUOUS_MODE_ENABLED
// to enable promiscuous mode on modem

properties.put(ModemKeys.PROMISCUOUS_MODE_ENABLED, "enabled");

properties.put(ModemKeys.CPE_DHCP_CRITERIA, "provisionedCPE");

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    modemMACAddress,
        // macAddress: derived from computer lease
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "Silver",
        // ClassOfService
    "provisionedCM",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    properties
        // properties:
);

```

Step 5 The user interface prompts the subscriber to reboot the computer.

Step 6 The provisioning client calls *performOperation(...)* to reboot the modem and gives the modem provisioned access.

```

get-new-batch(AUTOMATIC, NO_CONFIRMATION)

// AUTOMATIC is the activation mode because we are attempting
// to reset the modem so that it receives the new class of service.
// NO_CONFIRMATION is the confirmation mode because we don't want
// the batch to fail if we can't reset the modem. The user might
// have power cycled the modem when they rebooted their computer

// send a batch to reset the modem now that the user has been
// notified to reboot their computer

performOperation(
    DeviceOperation.RESET,
        //deviceOperation: Reset operation

```

```
modemMACaddress,  
    // macAddress:Modem's MAC address  
null  
    // properties : not used  
);
```

- Step 7** When the computer is rebooted, it receives a new IP address. The cable modem is now a provisioned device. The computer is not registered with BACC, but it gains access to the Internet through the service provider's network. Computers that are online behind promiscuous modems are still available using the provisioning API.
-

Bulk Provisioning 100 Modems in Promiscuous Mode

A service provider wants to pre-provision 100 cable modems for distribution by a customer service representative at a service kiosk.

Desired Outcome

Use this workflow to distribute modem data for all modems to new subscribers. The customer service representative has a list of modems available for assignment.

- Step 1** The cable modem's MAC address data for new or recycled cable modems is collected into a list at the service provider's loading dock.
- Step 2** Modems that are assigned to a particular kiosk are bulk-loaded into BACC and are flagged with the identifier for that kiosk.

Step 3 When the modems are distributed to new subscribers at the kiosk, the customer service representative enters new service parameters, and changes the Owner ID field on the modem to reflect the new subscriber's account number.

```
// get a single batch for bulk load or 100 modems

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)
// The activation mode for this batch should be NO_ACTIVATION.
// NO_ACTIVATION should be used in this situation because no
// network information exists for the devices because they
// have not booted yet. A configuration can't be generated if no
// network information is present. And because the devices
// have not booted, they are not online and therefore cannot
// be reset. NO_CONFIRMATION is the confirmation mode because
// we are not attempting to reset the devices.
// Create a Map for the properties of the modem

Map properties;

// Set the property ModemKeys.PROMISCUOUS_MODE_ENABLED to
// enable promiscuous mode on modem.
// This could be done at a system level if promiscuous mode
// is your default provisioning mode.
properties.put(ModemKeys.PROMISCUOUS_MODE_ENABLED, "enabled")

// The ModemKeys.CPE_DHCP_CRITERIA is used to specify the DHCP
// Criteria to be used while selecting IP address scopes for
// CPEs behind this modem in the promiscuous mode.

properties.put(ModemKeys.CPE_DHCP_CRITERIA, "provisionedCPE");

// for each modem MAC-address in list:

ModemLoop:
{
    add(
        DeviceType.DOCSIS,
            // deviceType: DOCSIS
        modemMACAddress,
            // macAddress: derived from computer lease
        null,
            // hostName: not used in this example
        null,
            // domainName: not used in this example
        "0123-45-6789",
            // ownerID: here, account number from billing system
        "Silver",
            // ClassOfService
        "provisionedCM",
            // DHCP Criteria: Network Registrar uses this to
            // select a modem lease granting provisioned IP address
        properties
            // properties:
    );
}
```

```
)
//end ModemLoop.
```

Pre-Provisioning First-Time Activation in Promiscuous Mode

A new subscriber contacts the service provider and requests service. The subscriber has a computer installed in a single dwelling unit.

Desired Outcome

Use this workflow to bring a new unprovisioned cable modem and computer online with the appropriate level of service.

-
- Step 1** The service provider chooses a subscriber username and password for the billing system.
 - Step 2** The service provider selects the services that the subscriber can access.
 - Step 3** The service provider registers the device using its own user interface.
 - Step 4** The service provider's user interface passes information, such as the modem's MAC address and the class of service to BACC. Additionally, the modem gets a CPE DHCP criteria setting that lets Network Registrar select a provisioned address for any computers to be connected behind the modem. The new modem is then registered with BACC.
 - Step 5** The service provider's field technician installs the physical cable to the new subscriber's house and installs the pre-provisioned device, connecting it to the subscriber's computer.

```
// MSO admin UI calls the provisioning API to pre-provision
// an HSD modem.

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// The activation mode for this batch should be NO_ACTIVATION.
// NO_ACTIVATION should be used in this situation because no
// network information exists for the modem because it has not
// booted. A configuration cannot be generated if no network
// information is present. And because the modem has not booted,
// it is not online and therefore cannot be reset.
// NO_CONFIRMATION is the confirmation mode because we are not
// attempting to reset the modem.
// Create a map for the properties of the modem.

Map properties;

// Set the property ModemKeys.PROMISCUOUS_MODE_ENABLED to enable
// promiscuous mode on modem

properties.put(ModemKeys.PROMISCUOUS_MODE_ENABLED, "enabled")

properties.put(ModemKeys.CPE_DHCP_CRITERIA, "provisionedCPE");

add(
    DeviceType.DOCSIS,
    // deviceType: DOCSIS
```

```

"1,6,00:11:22:33:44:55",
    // macAddress: derived from computer lease
null,
    // hostName: not used in this example
null,
    // domainName: not used in this example
"0123-45-6789",
    // ownerID: here, account number from billing system
"Silver",
    // ClassOfService
"provisionedCM",
    // DHCP Criteria: Network Registrar uses this to
    // select a modem lease granting provisioned IP address
properties
    // properties:
);

```

Step 6 The technician powers on the cable modem and BACC gives it provisioned access.

Step 7 The technician powers on the computer and BACC gives it provisioned access. The cable modem and the computer are now both provisioned devices. The computer has access to the Internet through the service provider's network.

Replacing an Existing Modem

A service provider wants to replace a broken modem.



Note

If the computer has the option restricting roaming from one modem to another, and the modem is replaced, the computer's MAC Address for the modem must also be changed.

Desired Outcome

Use this workflow to physically replace an existing cable modem with a new modem without changing the level of service provided to the subscriber.

Step 1 The service provider changes the MAC address of the existing modem to that of the new modem.

```

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because we will
// not be able to reset as the new modem has not booted
// on the network.
// NO_CONFIRMATION is the confirmation mode because we are
// not trying to reset the modem
// To change the MAC address of a DOCSIS modem:

changeMACAddress (

```

```

"1,6,00:11:22:33:44:55"
    // old macAddress: unique identifier for the old modem
"1,6,11:22:33:44:55:66"
    /// new macAddress: unique identifier for the new modem
);

```

- Step 2** The service provider replaces the cable modem and turns it on. The computer must also be turned on.
- Step 3** The cable modem is now a fully provisioned device with the appropriate level of service, as is the computer behind the cable modem.
-

Adding a Second Computer in Promiscuous Mode

A subscriber wishes to connect a second computer behind an installed cable modem.

Desired Outcome

Use this workflow to ensure that the subscriber's selected service permits the connection of multiple CPEs, and that the subscriber has network access from both connected computers.



Note

This case does not require calls to the provisioning API.

- Step 1** The subscriber connects a second computer behind the cable modem.
- Step 2** The subscriber turns on the computer.
- Step 3** If the subscriber's selected service permits connecting multiple CPEs, BACC gives the second computer access to the Internet.
-

Self-Provisioning First-Time Activation with NAT

A university has purchased a DOCSIS cable modem with network address translation (NAT) and DHCP capability. The five occupants of the unit each have a computer installed with a browser application.

Desired Outcome

Use this workflow to bring a new unprovisioned cable modem (with NAT) and the computers behind it online with the appropriate level of service.

- Step 1** The subscriber purchases a cable modem with NAT and DHCP capability and installs it in a multiple dwelling unit.
- Step 2** The subscriber turns on the modem and BACC gives it restricted access.
- Step 3** The subscriber connects a laptop computer to the cable modem, and the DHCP server in the modem provides an IP address to the laptop.

- Step 4** The subscriber starts a browser application on the computer and a spoofing DNS server points the browser to the service provider's registration server (for example, an OSS user interface or a mediator).
- Step 5** The subscriber uses the service provider's user interface to complete the steps required for cable modem registration of the modem. The registration user interface detects that the modem is using NAT and registers the modem, making sure that the modem gets a class of service that is compatible with NAT.

```
get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a query.
// NO_CONFIRMATION is the confirmation mode because we are not
// attempting to reset the device.
// First we query the computer's information to find the modems
// MAC address.

// With NAT, the computer is never seen by the Network Registrar
// DHCP server. Instead, the modem has translated the IP address
// it assigned to the computer, so the web browser sees the
// modem's IP address. When the lease data is examined, the MAC
// address for the device and the relay-agent-remote-id are the
// same, indicating that this is an unprovisioned modem device,
// which is therefore presumed to be performing NAT.

LeaseResults modemLease =
    getAllForIPAddress("10.0.14.38");
    // ipAddress: restricted access.

String modemMACAddress = modemLease.
    getSingleLease().get(DeviceDetailsKeys.MAC_ADDRESS);
// MSO client registration program then calls the provisioning
// API to provision the NAT modem (with an appropriate class of
// service for NAT).

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION will generate a configuration for the modem
// however it will not attempt to reset it.
// The configuration will be able to be generated because the
// modem has booted. NO_CONFIRMATION is the confirmation mode
// because we are not attempting to reset the modem

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    modemMACAddress,
        // macAddress: derived from its lease
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "Silver",
        // ClassOfService
    "provisionedCM",
```

```

        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    null

        // properties: not used
    );

```

Step 6 The user interface prompts the subscriber to reboot the computer.

Step 7 The provisioning client calls *performOperation(...)* to reboot the modem and gives the modem provisioned access.

```

get-new-batch(AUTOMATIC, NO_CONFIRMATION)

// AUTOMATIC is the activation mode because we are attempting
// to reset the modem so that it receives its new class of service.
// NO_CONFIRMATION is the confirmation mode because we don't want
// the batch to fail if we can't reset the modem. The user might
// have power cycled the modem when they rebooted their computer
// send a batch to reset the modem now that the user has been
// notified to reboot their computer

performOperation(
    DeviceOperation.RESET,
        //deviceOperation: Reset operation
    "1,6,00:11:22:33:44:55",
        // macAddress:Modem's MAC address
    null
        // properties : not used
    );

```

Step 8 The cable modem is now fully provisioned and the computers behind it have full access to the service provider's network.



Note

Certain cable modems with NAT may require you to reboot the computer to get the new class of service settings. If the cable modem and NAT device are separate devices, the NAT device must also be registered similarly to registering a computer.

Adding a New Computer Behind a Modem with NAT

The landlord of an apartment building has four tenants sharing a modem and accessing the service provider's network. The landlord wants to provide Internet access to a new tenant, sharing the building's modem. The modem has NAT and DHCP capability. The new tenant has a computer connected to the modem.

Desired Outcome

Use this workflow to bring a new unprovisioned computer online with a previously provisioned cable modem so that the new computer has the appropriate level of service.



Note This case does not require calls to the provisioning API.

Step 1 The subscriber turns on the computer.

Step 2 The computer is now a provisioned device with access to the appropriate level of service.



Note The provisioned NAT modem hides the computers behind it from the network.

Move Device to Another DHCP Scope

A service provider is renumbering its network causing a registered cable modem to require an IP address from a different Network Registrar scope.

Desired Outcome

A provisioning client changes the DHCP criteria, and the cable modem receives an IP address from the corresponding DHCP scope.

Step 1 Change the DOCSIS modem's DHCP criteria to "newmodemCriteria".

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION);  
// AUTOMATIC is the Activation mode because we are attempting  
// to reset the modem so that a phone line is disabled  
// NO_CONFIRMATION is the Confirmation mode because we don't  
// want the batch to fail if we can't reset the modem.  
  
// This use case assumes that the DOCSIS modem has been  
// previously added to the database  
  
changeDHCPCriteria(  
    "1,6,ff:00:ee:11:dd:22"  
    // Modem's MAC address or FQDN  
    "newmodemCriteria"  
);
```

Step 2 The modem gets an IP address from the scope targeted by "newmodemCriteria."

Log Device Deletions Using Events

A service provider has multiple provisioning clients and wants to log device deletions.

Desired Outcome

When any provisioning client deletes a device, the provisioning client logs an event in one place.

- Step 1** Create a listener for the device deletion event. This class must extend the *DeviceAdapter* abstract class or, alternatively, implement the *DeviceListener* interface. This class must also override the *deletedDevice(DeviceEvent ev)* method in order to log the event.

```
public DeviceDeletionLogger
    extends DeviceAdapter
    //Extend the DeviceAdapter class.
{
    public void deletedDevice(DeviceEvent ev)
        //Override deletedDevice.
    {
        logDeviceDeletion(ev.getDeviceID());
        //Log the deletion.
    }
}
```

- Step 2** Register the listener and the qualifier for the events using the *PACEConnection* interface.

```
DeviceDeletionLogger deviceDeletionLogger =
    new DeviceDeletionLogger();
    // Modem's MAC address or FQDN
    "newmodemCriteria"

qualifier = new DeviceEventQualifier();
// We are interested only in device deletion.
qualifier.setDeletedDevice ();
// Add device listener using PACEConnection
connection.addDeviceListener(deviceDeletionLogger, qualifier
);
```

- Step 3** When a device is deleted from the system, the event is generated, and the listener is notified.

Monitoring an RDU Connection Using Events

A service provider is running a single provisioning client and wants notification if the connection between the provisioning client and the RDU breaks.

Desired Outcome

Use this workflow to have the event interface notify the service provider if the connection breaks.

-
- Step 1** Create a listener for the messaging event. This class must extend the `MessagingAdapter` abstract class or, alternatively, implement the `MessagingListener` interface. This class must override the `connectionStopped(MessagingEvent ev)` method.

```
// Extend the service provider's Java program using the
// provisioning client to receive Messaging events.
public MessagingNotifier
    extends MessagingAdapter
    //Extend the MessagingAdapter class.
{
    public void connectionStopped(MessagingEvent ev)
        //Override connectionStopped.
    {
        doNotification(ev.getAddress(), ev.getPort());
        //Do the notification.
    }
}
```

- Step 2** Register the listener and the qualifier for the events using the `PACEConnection` interface.

```
MessagingQualifier qualifier =
    new MessagingQualifier();
qualifier.setAllPersistentConnectionsDown();
MessagingNotifier messagingNotifier = new MessagingNotifier();
connection.addMessagingListener(messagingNotifier, qualifier
);
```

- Step 3** If a connection breaks, the event is generated, and the listener is notified.
-

Logging Batch Completions Using Events

A service provider has multiple provisioning clients and wants to log batch completions.

Desired Outcome

When any provisioning client completes a batch, an event is logged in one place.

-
- Step 1** Create a listener for the event. This class must extend the *BatchAdapter* abstract class or implement the *BatchListener* interface. This class must override the *completion(BatchEvent ev)* method in order to log the event.

```
public BatchCompletionLogger
    extends BatchAdapter
    //Extend the BatchAdapterclass.
{
    public void completion(BatchEvent ev)
        //Override completion.
    {
        logBatchCompletion(ev.BatchStatus().getBatchID());
        //Log the completion.
    }
}
```

- Step 2** Register the listener and the qualifier for the events using the *PACEConnection* interface.

```
BatchCompletionLogger batchCompletionLogger =
    new BatchCompletionLogger();
Qualify All qualifier = new Qualify All();
connection.addBatchListener(batchCompletionLogger , qualifier
);
```

- Step 3** When a batch completes, the event is generated, and the listener is notified.
-

Getting Detailed Device Information

A service provider wants to allow an administrator to view detailed information for a particular device.

Desired Outcome

The service provider's administrative application displays all known details about a given device, including MAC address, lease information, provisioned status of the device, and the device type (if known).

-
- Step 1** The administrator enters the MAC address for the device being queried into the service provider's administrative user interface.

Step 2 BACC queries the embedded database for the device details.

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION);

// MSO admin UI calls the provisioning API to query the details
// for the requested device. Query may be performed based on MAC
// address or IP address, depending on what is known about the
// device.
Map deviceDetails =
    getDetails(
        "1,6,00:11:22:33:44:55",
        // macORFqdn: unique identifier for the device
        true
        // needLeaseInfo: yes we need it
    );
```

Step 3 The service provider's application presents a page of device data details, which can display everything that is known about the requested device. If the device was connected to the service provider's network, this data includes lease information (for example, IP address and relay agent identifier). The data indicates whether the device was provisioned, and if it was, the data also includes the device type.

```
// extract device detail data from the map
String deviceType = (String)deviceDetails.get(DEVICE_TYPE);
String macAddress = (String)deviceDetails.get(MAC_ADDRESS);
String ipAddress = (String)deviceDetails.get(IP_ADDRESS);
String relayAgentID = (String)deviceDetails.get(RELAY_AGENT_ID);
Boolean isProvisioned = (Boolean)deviceDetails.get(IS_PROVISIONED);
// The admin UI now formats and prints the detail data to a view page
```

Searching Using the Default Class of Service

A service provider wants to allow an administrator to view data for all modems with the *default* class of service for DOCSIS device type.

Desired Outcome

The service provider's administrative application returns a list of DOCSIS devices with default class of service.

Step 1 The administrator selects the search option in service provider's administrative user interface.

Step 2 BACC queries the embedded database for a list of all MAC addresses for the devices that match the requested default class of service.

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION);

// Create a MACAddressSearchType to indicate search by
// default class of service.
MACAddressSearchType mst =
    new MACAddressSearchType.getByDefaultClassOfService(
```

```
                DeviceType.DOCSIS);

// Create a MACAddressSearchFilter to get 20 devices
// at a time. This indicates that we have a page size of 20.

MACAddressSearchFilter mySearchFilter =
    new MACAddressSearchFilter(
        mst,
        // type: MAC address search type
        false,
        // isInclusive:
        20
        // maximumReturned:
    );

// MAC address to start the search from.
String startMac = null;

// A list containing the MAC addresses returned from
// search.
List deviceList = null;

// If the size of deviceList is equal to 20 then
// there may be devices matching the search criteria.
while ((deviceList == null) ||
        (deviceList.size() == 20))
{
    // Use the provisioning API call to search BACC
    // database. The search starts from the MAC address
    // "startMac". If "startMac" is null then the search
    // starts from the very beginning of the index.

    deviceList = search (mySearchFilter, startMac);

    // See Step 3 for the definition of processMACAddressList

    startMac = processMACAddressList(deviceList);
}
}
```

- Step 3** The service provider's application requests details on these devices from BACC, and presents a page of device data. For each device, the code provides for display of the device type, MAC address, client class, and provisioned status of the device, one device per line.

```
processMACAddressList (List deviceList)
{
    Iterator iter = deviceList.iterator();

    String startMac = null;

    while (iter.hasNext())
    {
        startMac = (String) iter.next();
        // Get details for this device
        Map detailMap = getDetails (
            startMac,
            // MAC of current device
            true
            // yes, we need lease info
        );

        // extract device detail data from each map
        // this can be used for displaying in UI
        String deviceType = (String)detailMap.get (DEVICE_TYPE);
        String macAddress = (String)detailMap.get (MAC_ADDRESS);
        String clientClass = (String)detailMap.get (CLIENT_CLASS);
        Boolean isProvisioned = (Boolean)detailMap.get (IS_PROVISIONED);
        // format and print above data in output line
    }

    // We return the last Mac address in the list
    // so that the next search can be started from here

    return startMAC;
}
```

Retrieving Devices Matching a Vendor Prefix

A service provider wants to allow an administrator to view data for all devices matching a particular vendor prefix.

Desired Outcome

The service provider's administrative application returns a list of devices matching the requested vendor prefix.

- Step 1** The administrator enters the sub-string matching the desired vendor prefix into the service provider's administrative user interface.

Step 2 BACC queries the embedded database for a list of all MAC addresses for the devices that match the requested vendor prefix.

```
// Create a MACAddressPattern corresponding to the requested
// vendor prefix

MACAddressPattern pattern =
    new MACAddressPattern(
        "1,6,ff:00:ee:*",
        // macAddressPattern: the requested vendor prefix
    );

// Create a MACAddressSearchType to indicate search by
// MAC address pattern

MACAddressSearchType mst =
    new MACAddressSearchType.getDevices(pattern);

// Create a MACAddressSearchFilter to get 20 devices
// at a time. This indicates that we have a page size of 20.

MACAddressSearchFilter mySearchFilter =
    new MACAddressSearchFilter(
        mst,
        // type: MAC address search type
        false,
        // isInclusive:
        20
        // maximumReturned:
    );

// MAC address to start the search from.
String startMac = null;

// A list containing the MAC addresses returned from
// search.
List deviceList = null;

// If the size of deviceList is equal to 20 then
// there may be devices matching the search criteria.

while ((deviceList == null) ||

    (deviceList.size() == 20))
{
    // Use the provisioning API call to search BACC
    // database. The search starts from the MAC address
    // "startMac". If "startMac" is null then the search
    // starts from the very beginning of the index.

    deviceList = search (mySearchFilter, startMac);

    // See Step 3 for the definition of processMACAddressList

    startMac = processMACAddressList(deviceList);
}
}
```


Step 3 The service provider's application requests details on these devices from BACC, and presents a page of device data. For each device, the code displays the device type, MAC address, client class, and provisioned status of the device. One device is identified per line.

```
processMACAddressList (List deviceList)
{
    Iterator iter = deviceList.iterator();

    String startMac = null;

    while (iter.hasNext())
    {
        startMac = (String) iter.next();
        // Get details for this device
        Map detailMap = getDetails (
            startMac,
            // MAC of current device
            true
            // yes, we need lease info
        ) ;

        // extract device detail data from each map
        // this can be used for displaying in UI
        String deviceType = (String)detailMap.get (DEVICE_TYPE);
        String macAddress = (String)detailMap.get (MAC_ADDRESS);
        String clientClass = (String)detailMap.get (CLIENT_CLASS);
        Boolean isProvisioned = (Boolean)detailMap.get (IS_PROVISIONED);
        // format and print above data in output line
    }

    // We return the last Mac address in the list
    // so that the next search can be started from here

    return startMac;
}
```

Pre-Provisioning PacketCable eMTA

A new customer contacts a service provider to order PacketCable voice service. The customer expects to receive a provisioned embedded MTA.

Desired Outcome

Use this workflow to pre-provision an embedded MTA so that the modem MTA component has the appropriate level of service when brought online.



Note

This use case skips the call agent provisioning that is required for making telephone calls from eMTAs.

Step 1 The service provider chooses a subscriber username and password for the billing system.

- Step 2** The service provider chooses the appropriate class of service and DHCP criteria for the modem component and adds it to BACC.

```
get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// Let's provision the modem and the MTA component in the same
// batch. This can be done because the activation mode of this
// batch is NO_ACTIVATION. More than one device can be operated
// on in a batch if the activation mode does not lead to more
// than one device being reset.
// To add a DOCSIS modem:

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    "1,6,01:02:03:04:05:06",
        // macAddress: scanned from the label
    null
        // hostName: not used in this example
    null
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "Silver",
        // classOfService
    "provisionedCM",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    null
        // properties: not used
);
```

- Step 3** The service provider chooses the appropriate class of service and DHCP criteria for the MTA component and adds it to BACC.

```
// Continuation of the batch in Step2
// To add the MTA component:

add(
    DeviceType.PACKET_CABLE_MTA,
        // deviceType: PACKET_CABLE_MTA
    "1,6,01:02:03:04:05:07",
        // macAddress: scanned from the label
    null,
        // hostName: not used in this example, will be auto generated
    null,
        // domainName: not used in this example, will be auto generated.
        // The FqdnKeys.AUTO_FQDN_DOMAIN property must be set somewhere in the
        // property hierarchy.
    "0123-45-6789",
```

```

        // ownerID: here, account number from billing system
        "Silver",
        // ClassOfService
        "provisionedMTA",
        // DHCP Criteria: Network Registrar uses this to
        // select an MTA lease granting provisioned IP address
        null
        // properties: not used
    );

```

Step 4 The embedded MTA gets shipped to the customer.

Step 5 The customer brings the embedded MTA online and makes telephone calls using it.

SNMP Cloning on PacketCable eMTA

A customer has an SNMP Element Manager that wishes to gain access to a PacketCable eMTA.

Desired Outcome

An external Element Manager is granted secure SNMPv3 access to the PacketCable eMTA.



Note

Changes made to RW MIB variables are not permanent and are not updated in the BACC configuration for the eMTA. The information written into the eMTA MIB is lost the next time the MTA powers down or resets.

Step 1 Call the provisioning API method, `performOperation()`, passing in the MAC address of the MTA and the username of the new user to create on the MTA. This will be the username used in subsequent SNMP calls by the Element Manager.

```

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION);

// NO_ACTIVATION is the activation mode because we don't want to
// reset the device.
// NO_CONFIRMATION is the confirmation mode because we are
// not attempting to reset the device.

// The goal here is to create a new user on the MTA indicated
// by the MAC address. The other parameter needed here is the new
// user name, which is passed in the Map.
// Create a map that contains one element - the name of
// the new user to be created on the MTA
HashMap map = new HashMap();
map.put( SNMPPropertyKeys.CLONING_USERNAME, "newUser" );
// The first param is the actual device operation to perform.
performOperation(
    DeviceOperation.ENABLE_SNMPV3_ACCESS,
    // deviceOperation : ENABLE_SNMPV3_ACCESS
    "1,6,00:00:00:00:00:99",

```

```

        // macORFqdn : MAC Address of the modem
    map
        // parameters: operation specific parameters
    );

```

- Step 2** The provisioning API attempts to perform an SNMPv3 cloning operation to create an entry on the MTA for the new user passed in Step 1. The keys used in the new user entry row are a function of two passwords defined within BACC. These passwords will be made available to the customer and the RDU command passes these passwords (the auth and priv password) through a key localization algorithm to create an auth and priv key. These are stored, along with the new user, in the eMTA's user table.



Note The auth and priv passwords mentioned in this step may be changed by setting `SNMPPropertyKeys.CLONING_AUTH_PASSWORD (/snmp/cloning/auth/password)` and `SNMPPropertyKeys.CLONING_PRIV_PASSWORD (/snmp/cloning/priv/password)` properties respectively in the `rdu.properties` configuration file.

- Step 3** The customer issues SNMPv3 request using above specified username, passwords, and key localization algorithm to allow for secure communication with the MTA.

Incremental Provisioning of PacketCable eMTA

A customer has a PacketCable eMTA in service with its first line (end point) enabled. The customer wants to enable the second telephone line (end point) on the eMTA and connect a telephone to it.

Desired Outcome

The customer should be able to connect a telephone to the second line (end point) on the eMTA and successfully make phone calls from it without any service interruption.



Note In order to use the second line on the eMTA, the Call Agent need to be configured accordingly. This use case does not address provisioning of call agents.

- Step 1** The service provider's application invokes the BACC API to change the class of service of the eMTA. The new class of service supports two end points on the eMTA. This change in class of service does not take affect until the eMTA is reset. Disrupting the eMTA is not desirable; therefore, incremental provisioning is undertaken in the next step.

```

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because we don't want to
// reset the device.
// NO_CONFIRMATION is the Confirmation mode because we are not
// disrupting the device.

changeClassOfService(
    "1,6,ff:00:ee:11:dd:22" // eMTA's MAC address or FQDN

```

```
, "twoLineEnabledCOS" // This COS supports two lines.
);
```

Step 2 The service provider's application uses the BACC incremental update feature to set SNMP objects on the eMTA and thereby enabling the service without disrupting the eMTA.

```
// The goal here is to enable a second phone line, assuming one
// phone line is currently enabled. We will be adding a new
// row to the pktcNcsEndPntConfigTable.

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because we don't want to
// reset the device.
// NO_CONFIRMATION is the confirmation mode because we are
// not attempting to reset the device.

// Create a map containing one element - the list of SNMP
// variables to set on the MTA
HashMap map = new HashMap();

// Create an SnmpVarList to hold SNMP varbinds
SnmpVarList list = new SnmpVarList();

// An SnmpVariable represents an oid/value/type triple.

// pktcNcsEndPntConfigTable is indexed by the IfNumber, which in this
// case we will assume is interface number 12 (this is the last
// number in each of the oids below.

// The first variable represents the creation of a new row in
// pktcNcsEndPntConfigTable we are setting the RowStatus
// column (column number 26). The value of 4 indicates that
// a new row is to be created in the active state.
SnmpVariable variable = new SnmpVariable(
    ".1.3.6.1.4.1.4491.2.2.2.1.2.1.1.26.12",
    "4",
    SnmpType.INTEGER );
list.add( variable );

// The next variable represents the call agent id for this new
// interface, which we'll assume is 'test.com'
SnmpVariable variable = new SnmpVariable(
    ".1.3.6.1.4.1.4491.2.2.2.1.2.1.1.1.12",
    "test.com",
    SnmpType.STRING );
list.add( variable );

// The final variable represents the call agent port
SnmpVariable variable = new SnmpVariable(
    ".1.3.6.1.4.1.4491.2.2.2.1.2.1.1.2.12",
    "2728",
    SnmpType.INTEGER );
```

```
list.add( variable );

// Add the SNMP variable list to the Map to use in the API call
map.put( SNMPPPropertyKeys.SNMPVAR_LIST, list );

// Invoke the BACC API to do incremental update on the eMTA.
performOperation( DeviceOperation.INCREMENTAL_UPDATE // device operation
    , "1,6,00:00:00:00:00:99" // MAC Address
    , map // Parameters for the operation
    );
```

- Step 3** The eMTA is enabled to use the second telephone line. The eMTA continues to receive the same service, after being reset, since the class of service was changed in Step 1.
-

Pre-Provisioning DOCSIS Modems with Dynamic Configuration Files

A new customer contacts a service provider to order a DOCSIS modem with high-speed *Gold* data service for two CPEs behind it.

Desired Outcome

Use this workflow to pre-provision a DOCSIS modem with a class of service that uses DOCSIS templates. The dynamic configuration file generated from the templates is used while the modem comes online.

- Step 1** The service provider chooses a subscriber username and password for the billing system.
-

Step 2 The service provider chooses Gold class of service, and the appropriate DHCP criteria, and then adds the cable modem in to BACC.

```
get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

Map properties;

// Set the property ModemKeys.PROMISCUOUS_MODE_ENABLED to enable
// promiscuous mode on modem

properties.put(ModemKeys.PROMISCUOUS_MODE_ENABLED, "enabled")

// No CPE DHCP Criteria is specified.
// The CPEs behind the modem will use the default provisioned
// promiscuous CPE DHCP criteria specified in the system defaults.

// This custom property corresponds to a macro variable in the
// DOCSIS template for "gold" class of service indicating the
// maximum number of CPEs allowed behind this modem. We set it
// to two CPEs from this customer.

properties.put("docsis-max-cpes", "2");

// To add a DOCSIS modem:

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    "1,6,01:02:03:04:05:06",
        // macAddress: scanned from the label
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "gold",
        // classOfService:
    "provisionedCM",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    properties
        // properties:
);
```

Step 3 The cable modem is shipped to the customer.

Step 4 The customer brings the cable modem online and connects the computers behind it.

Optimistic Locking

An instance of the service provider application needs to ensure that it is not overwriting the changes made by another instance of the same application.

Desired Outcome

Use this workflow to demonstrate the optimistic locking capabilities provided by the BACC API.



Note

Locking of objects is done in multi-user systems to preserve integrity of changes, so that one person's changes do not accidentally get overwritten by another. With optimistic locking, you write your program assuming that any commit has a chance to fail if at least one of the objects being committed was changed by someone else since you began the transaction.

Step 1 The service representative selects the search option in the service provider's user interface and enters the cable modem's MAC address.

Step 2 BACC queries the embedded database, gets the details of the device and the MSO user interface displays the information.

```
// MSO admin UI calls the provisioning API to query the details
// for the requested device.
Map deviceDetails =
    getDetails(
        "1,6,00:11:22:33:44:55",
        // macORFqdn: unique identifier for the device
        true
        // needLeaseInfo: yes we need it
    );

// extract device detail data from the map
String deviceType = (String)deviceDetails.get(DEVICE_TYPE);
String macAddress = (String)deviceDetails.get(MAC_ADDRESS);
String ipAddress = (String)deviceDetails.get(IP_ADDRESS);
String relayAgentID = (String)deviceDetails.get(RELAY_AGENT_ID);

Boolean isProvisioned = (Boolean)deviceDetails.get(IS_PROVISIONED);
// service provider admin UI displays this information.

// Let's save the OID_REVISION_NUMBER property so that we can see in
// step 3.
String oidRevisionNumber = (String)deviceDetails.get(OID_REVISION_NUMBER);
```


Step 3 The service representative attempts to change the class of service and the DHCP criteria of the modem using the user interface. This in turn invokes the BACC API.

```
// We need a reference to Batch instance so that ensureConsistency()
// method can be invoked on it.
Batch batch = conn.newBatch() ;

List oidList = new ArrayList();
// Add the oid-rev number saved from step 2 to the list
oidList.add(oidRevisionNumber);

// Sends a list of OID revision numbers to validate before processing the
// batch. This ensures that the objects specified have not been modified
// since they were last retrieved.
batch.ensureConsistency(oidList);

    batch.changeClassOfService (
        "1,6,00:11:22:33:44:55",
        // macORFqdn: unique identifier for the device.
        "gold"
        // newCOSName : Class of service name.
    )

batch.changeDHCPCriteria (
    "1,6,00:11:22:33:44:55",
    // macORFqdn: unique identifier for the device.
    "specialDHCPCriteria"
    // newDHCPCriteria : New DHCP Criteria.
)

// This batch fails with BatchStatusCodes.BATCH_NOT_CONSISTENT,
// in case if the device is updated by another client in the mean time.
// If there is a conflict occurs, then the service provider client
// is responsible for resolving the conflict by querying the database
// again and then applying changes appropriately.
```

Step 4 The user is ready to receive Gold class of service with appropriate DHCP criteria.

Temporarily Throttling a Subscriber's Bandwidth

An MSO has a service that allows a subscriber to download only 10 Mbyte of data per month. Once the subscriber reaches that limit their downstream bandwidth is turned down from 1 Mbyte to 56 K. When the month is over they are moved back up to 1 Mbyte.


Note

You may want to consider changing upstream bandwidth as well, since peer-to-peer users and users who run websites tend to have very heavy upload bandwidth.

Desired Outcome

Use this workflow to move subscribers up and down in bandwidth according to their terms of agreement.

- Step 1** The MSO has a rate tracking system, such as *NetFlow*, which keeps track of each customer's usage by MAC address. Initially a customer is provisioned at the *Gold* class of service level with 1 Mbyte downstream.
- Step 2** When the rate tracking software determines that a subscriber has reached the 10 Mbyte limit it notifies the OSS. The OSS then makes a call into the BACC API to change the subscriber's class of service from *Gold* to *Gold-throttled*.

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION)

// AUTOMATIC is the activation mode because we are
// attempting to reset the modem so that it
// receives low bandwidth service.
// NO_CONFIRMATION is the confirmation mode
// because we do not want the batch to fail if we cannot
// reset the modem. If the modem is off, when it will
// be disabled when it is turned back on.

// Let's change the COS of the device so that it restricts
// bandwidth usage of the modem.
    changeClassOfService(
        Device.MAC_ADDRESS,
        // macAddress: unique identifier for this modem
        "Gold-throttled");
        // newClassOfService: restricts bandwidth usage to 56k
```

- Step 3** At the end of the billing period, the OSS calls the BACC API to change the subscriber's class of service back to *Gold*.

Pre-Provisioning CableHome WAN-Man

A new customer contacts a service provider to order home networking service. The customer expects a provisioned CableHome device to be shipped to him.

Desired Outcome

Use this workflow to pre-provision a CableHome device so that the cable modem and WAN-Man components on it will have appropriate level of service when brought online.

Step 1 The service provider chooses a subscriber username and password for the billing system.

Step 2 The service provider chooses the appropriate class of service and the DHCP criteria for the modem component, then adds it to BACC.

```
get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// Let's provision the modem and the WAN-Man component in the same
// batch.
// To add a DOCSIS modem:

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    "1,6,01:02:03:04:05:06",
        // macAddress: scanned from the label
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "Silver",
        // classOfService
    "provisionedCM",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    null
        // properties: not used
);
```

Step 3 The service provider chooses the appropriate class of service and DHCP criteria for the WAN-Man component, then adds it to BACC.

```
// Continuation of the batch in Step2
// To add the WAN-Man component:

add(
    DeviceType.CABLEHOME_WAN_MAN,
        // deviceType: CABLEHOME_WAN_MAN
    "1,6,01:02:03:04:05:07",
```

```

        // macAddress: scanned from the label
    null,
        // hostName: not used in this example.
    null,
        // domainName: not used in this example.
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "silverWanMan",
        // ClassOfService
    "provisionedWanMan",
        // DHCP Criteria: Network Registrar uses this to
        // select an MTA lease granting provisioned IP address
    null
        // properties: not used
    );

```

Step 4 The CableHome device gets shipped to the customer.

Step 5 The customer brings the CableHome device online.

CableHome with Firewall Configuration

A customer contacts a service provider to order a home networking service with the firewall feature enabled. The customer expects to receive a provisioned CableHome device.

Desired Outcome

Use this workflow to pre-provision a CableHome device so that the cable modem and the WAN-Man components on it, have the appropriate level of service when brought online.

Step 1 The service provider chooses a subscriber username and password for the billing system.

Step 2 The service provider chooses the appropriate class of service and DHCP criteria for the cable modem component, then adds it to BACC.

```

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// Let's provision the modem and the WAN-Man component in the same
// batch.
// To add a DOCSIS modem:

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    "1,6,01:02:03:04:05:06",
        // macAddress: scanned from the label
    null,
        // hostName: not used in this example

```

```

null,
    // domainName: not used in this example
"0123-45-6789",
    // ownerID: here, account number from billing system
"Silver",
    // classOfService
"provisionedCM",
    // DHCP Criteria: Network Registrar uses this to
    // select a modem lease granting provisioned IP address
null
    // properties: not used
);

```

Step 3 The service provider chooses the appropriate class of service and DHCP criteria for the WAN-Man component and adds it to BACC.

```

// Continuation of the batch in Step2
// To add the WAN-Man component:

// Create a Map to contain WanMan's properties
Map properties;

// The fire wall configuration for the Wan Man component is specified
// using the CableHomeKeys.CABLEHOME_WAN_MAN_FIREWALL_FILE property.
// This use case assumes that the firewall configuration file named
// "firewall_file.cfg" is already present in the RDU database and the
// firewall configuration is enabled in the Wan Man configuration file
// specified with the corresponding class of service.

properties.put(CableHomeKeys.CABLEHOME_WAN_MAN_FIREWALL_FILE,
"firewall_file.cfg");

add(
    DeviceType.CABLEHOME_WAN_MAN,
        // deviceType: CABLEHOME_WAN_MAN
"1,6,01:02:03:04:05:07",
        // macAddress: scanned from the label
null,
        // hostName: not used in this example.
null,
        // domainName: not used in this example.
"0123-45-6789",
        // ownerID: here, account number from billing system
"silverWanMan",
        // ClassOfService
"provisionedWanMan",
        // DHCP Criteria: Network Registrar uses this to
        // select an MTA lease granting provisioned IP address
properties

```

```

        // properties: contains the firewall config file
    );

```

- Step 4** The CableHome device gets shipped to the customer.
 - Step 5** The customer brings the CableHome device online and the cable modem and the WAN-Man component get provisioned IP addresses and proper configuration files.
-

Retrieving Device Capabilities for CableHome WAN-Man

A service provider wants to allow an administrator to view capabilities information for a CableHome WAN-Man device.

Desired Outcome

The service provider's administrative application displays all known details about a given CableHome WAN-Man component, including MAC address, lease information, provisioned status and the device capabilities information.

- Step 1** The administrator enters the MAC address of the WAN-Man being queried into the service provider's user interface.
- Step 2** BACC queries the embedded database for details of the device identified using the MAC address entered.

```

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION);

// MSO admin UI calls the provisioning API to query the details
// for the requested device.

Map deviceDetails =
    getDetails(
        1,6,00:11:22:33:44:55", "
        // macORFqdn: unique identifier for the device
        true
        // needLeaseInfo: yes we need it
    );

```

- Step 3** The service provider's application then presents a page of device data details, which can display everything that is known about the requested device. If the device was connected to the service provider's network, this data includes lease information, such as the IP address or the relay agent identifier. This data indicates whether the device is provisioned. If it is provisioned, the data also includes the device type and device capabilities information.


```
// extract device details information from the map
String deviceType      = (String)  deviceDetails.get(DeviceDetailsKeys.DEVICE_TYPE);
String macAddress      = (String)  deviceDetails.get(DeviceDetailsKeys.MAC_ADDRESS);
String ipAddress      = (String)  deviceDetails.get(DeviceDetailsKeys.IP_ADDRESS);
String relayAgentID   = (String)  deviceDetails.get(
                                   DeviceDetailsKeys.RELAY_AGENT_ID);
Boolean isProvisioned = (Boolean) deviceDetails.get(
                                   DeviceDetailsKeys.IS_PROVISIONED);
String formation      = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.FORMATION);
String deviceList     = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.DEVICE_LIST);
String serNum         = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.SERIAL_NUMBER);
String hwVer          = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.HARDWARE_VERSION);
String swVer          = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.SOFTWARE_VERSION);
String brVer          = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.BOOT_ROM_VERSION);
String vendorOui      = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.VENDOR_OUI);
String modelNum       = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.MODEL_NUMBER);
String vendorNum      = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.VENDOR_NAME);
String sysDesc        = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.SYSTEM_DESCRIPTION);
String fwVer          = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.FIRMWARE_VERSION);
String fwVer          = (String)  deviceDetails.get(
                                   IPDeviceCapabilities.FIREWALL_VERSION);
// The admin UI now formats and prints the detail data to a view page
```

Self-Provisioning CableHome WAN-Man

A subscriber has a computer with a browser application installed in a single dwelling unit and has purchased an embedded CableHome device.

Desired Outcome

Use this workflow to bring a new unprovisioned embedded CableHome device online with the appropriate level of service, and give the subscriber Internet access from computers connected to the embedded CableHome device.

-
- Step 1** The subscriber purchases an embedded CableHome device and installs it at home.
- Step 2** The subscriber powers on the embedded CableHome device. BACC gives the embedded cable modem restricted access, allowing 2 CPEs: one for the CableHome WAN-Man and the other for the computer.
-
-  **Note** This use case assumes an unprovisioned DOCSIS modem allows two CPEs behind it. Until configured to do otherwise, BACC supports only a single device behind an unprovisioned DOCSIS modem. You can change this behavior by defining an appropriate class of service that supports two CPEs and then using it as the default class of service for DOCSIS devices.
-
- Step 3** BACC configures the CableHome WAN-Man, including IP connectivity and downloading the default CableHome boot file. The default CableHome boot file configures the CableHome device in passthrough mode. The CableHome device is still unprovisioned.
- Step 4** The subscriber connects the computer to the CableHome device. The computer gets an unprovisioned (restricted) IP address. The subscriber starts a browser application on the computer. A spoofing DNS server points the browser to the service provider's registration server (for example, an OSS user interface or a mediator).
- Step 5** The subscriber uses the service provider's user interface to complete the steps required for cable modem registration, including selecting a class of service. The subscriber also selects a CableHome class of service.
- Step 6** The service provider's user interface passes the subscriber's information to BACC, including the selected classes of service for cable modem and CableHome, and computer IP address. The subscriber is then registered with BACC.

```
get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// NO_ACTIVATION is the activation mode because this is a
// query. NO_CONFIRMATION is the confirmation mode because
// we are not attempting to reset the device.
// First we query the computer's information to find the
// modems MAC address.
// We use the computers IP address (the web browser
// received this when the subscriber opened the service
// providers web interface). We also assume that "bostonProvGroup"
// is the provisioning group used in that locality.

List provGroupList;
provGroupList = provGroupList.add("bostonProvGroup");
Map computerLease = getAllForIPAddress(
    "10.0.14.38",
    // ipAddress: restricted access computer lease
```



```

        provGroupList
        // provGroups: List containing provgroup)
// Derive the modem MAC address from the computer's network
// information. The 1,6, is a standard prefix for an Ethernet
// device. The fully qualified MAC address is required by BPR
String modemMACAddress = "1,6," +

        computerLease.getSingleLease().get(RELAY_AGENT_REMOTE_ID);

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

// Now let's provision the modem
// NO_ACTIVATION will generate a configuration for the
// modem however it will not attempt to reset it
// The configuration will be able to be generated because
// the modem has booted.
// NO_CONFIRMATION is the confirmation mode because we
// are not attempting to reset the modem
// Create a Map for the properties of the modem

Map properties;
// Set the property ModemKeys.PROMISCUOUS_MODE_ENABLED
// to enable promiscuous mode on modem

properties.put(ModemKeys.PROMISCUOUS_MODE_ENABLED, "enabled");

properties.put(ModemKeys.CPE_DHCP_CRITERIA, "provisionedCPE");

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    modemMACAddress,
        // macAddress: derived from computer lease
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
    "Silver",
        // ClassOfService
    "provisionedCM",
        // DHCP Criteria: Network Registrar uses this to
        // select a modem lease granting provisioned IP address
    properties
        // properties:
);

```

Step 7 The user interface prompts the subscriber to reboot the computer.

Step 8 The provisioning client calls `performOperation(...)` to reboot the modem and gives the modem provisioned access.

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION)

// AUTOMATIC is the activation mode because we are attempting
// to reset the modem so that it receives its new class of service
// NO_CONFIRMATION is the confirmation mode because we don't want
// the batch to fail if we can't reset the modem. The user might
// have power cycled the modem when they rebooted their computer
// send a batch to reset the modem now that the user has been
// notified to reboot their computer

performOperation(
    DeviceOperation.RESET,
        //deviceOperation: Reset operation
    modemMACaddress,
        // macAddress:Modem's MAC address
    null
        // properties : not used
);
```

Step 9 When the computer is rebooted, it receives a new IP address from the CableHome device's DHCP server. The cable modem and the CableHome device are now both provisioned. Now the subscriber can connect a number of computers to the Ethernet ports of the CableHome device and they have access to the Internet.



Note

If the configuration file supplied to the WAN-Man component enables the WAN-Data component on the box, it will get provisioned in the promiscuous mode. This assumes that the promiscuous mode is enabled at the technology defaults level for the `DeviceType.CABLEHOME_WAN_DATA` device type.

Lease Reservation Use Cases

This section describes use cases specifically related to the use of the lease reservation feature. Standard administrative operations, such as managing class of service, DHCP criteria; licenses, and so on, are not addressed here. The lease reservation use cases include:

- [Bringing a Device Online Using a Service Provider IP Address, page C-49](#)
- [Removing and Recreating a Reservation, page C-50](#)
- [Assigning a New Device with an Old Device's IP Address, page C-51](#)
- [Removing a Reservation and Assigning a New IP Address, page C-52](#)
- [Rebooting a Device with the Same IP Address, page C-53](#)
- [Removing a Device from BACC, page C-54](#)
- [A Submitted Batch Fails when BACC Uses CCM, page C-55](#)
- [A Submitted Batch Fails when BACC does not Use CCM, page C-55](#)

API Calls Affected by the Lease Reservation Feature

The implementation of these API calls supports the lease reservation feature:

- `IPDevice.add(DeviceType DeviceType, String macAddress, String hostName, String domainName, String ownerID, String cosName, String dhcpCriteria, Map Properties)`
- `IPDevice.changeProperties(String macORFqdn, Map newPropToAdd, List propToDelete)`
- `IPDevice.changeMACAddress(String macORFqdn, String newMAC)`
- `IPDevice.delete(String macORFqdn, Boolean deleteDevicesBehind)`

Bringing a Device Online Using a Service Provider IP Address

When a device is added to the BACC system datastore, the service provider configures that device with the specific IP address for the device using a BACC property (`IPDeviceKeys.IP_RESERVATION`).

Desired Outcome

Bring a device online with the exact IP address set in the property by the service provider. The granted IP address is reserved (lease reservation) for that device.

-
- Step 1** The service provider adds a new device to the BACC system. The service provider configures the new device with a specific IP address 10.10.10.1.
- Step 2** The service provider's user interface passes device information to BACC, such as the MAC address, FQDN, and classes of service. The BACC property (`IPDeviceKeys.IP_RESERVATION`) is used to reserve a specific IP address (10.10.10.1) for this device.

```
Map properties;

// Set the property IPDeviceKeys.IP_RESERVATION to a specific
// IP address

properties.put( IPDeviceKeys.IP_RESERVATION, "10.10.10.1");

// To add a DOCSIS modem:

get-new-batch(NO_ACTIVATION, NO_CONFIRMATION)

add(
    DeviceType.DOCSIS,
        // deviceType: DOCSIS
    "1,6,01:02:03:04:05:06",
        // macAddress: scanned from the label
    null,
        // hostName: not used in this example
    null,
        // domainName: not used in this example
    "0123-45-6789",
        // ownerID: here, account number from billing system
```

```

"gold",
    // classOfService:
"provisionedCM",
    // DHCP Criteria: Network Registrar uses this to
    // select a modem lease granting provisioned IP address
properties
    // properties:
);

```

- Step 3** The new device is then registered with BACC. The reservation of 10.10.10.1 is also created for MAC address “01:02:03:04:05:06” in the BACC/Network Registrar system.
- Step 4** When the device is booted, it receives the exact IP address (10.10.10.1) set in the property by the service provider.

Rollback occurs if needed when the API command results in an error during processing or the change failed to commit to the RDU database. The command implementation removes the lease reservation of 10.10.10.1 for MAC address “01:02:03:04:05:06” from the BACC/Network Registrar if it was made during command processing.

Removing and Recreating a Reservation

After a device is registered to the BACC system with a reserved IP address, the service provider reassigns the device with a different IP address using a BACC property (IPDeviceKeys.IP_RESERVATION).

Desired Outcome

The device is rebooted with the exact IP address set in the property by the service provider. The reservation will be removed and recreated. The previously assigned IP address is unreserved for that device; and the new IP address is granted and reserved (lease reservation) for that device.

- Step 1** A device is registered to BACC with a reserved IP address of 10.10.10.1.
- Step 2** The service provider’s application makes these API calls in BACC to change the reserved IP to 10.10.10.5.

```

get-new-batch(AUTOMATIC, NO_CONFIRMATION);
// AUTOMATIC is the Activation mode because we are attempting
// to reset the device
// NO_CONFIRMATION is the Confirmation mode because we don't
// want the batch to fail if we can't reset the modem.

// This use case assumes that the DOCSIS modem has been
// previously added to the database

Map properties;

```

```

// Set the property IPDeviceKeys.IP_RESERVATION to a specific
// IP address

properties.put( IPDeviceKeys.IP_RESERVATION, "10.10.10.5");
// To reassign a different IP to:

    changeProperties(
        "1,6,01:02:03:04:05:06",
        // macAdd
        properties, null
    );

```

Step 3 The reservation of 10.10.10.1 for MAC address “01:02:03:04:05:06” is removed from the BACC/Network Registrar system. The reservation of 10.10.10.5 for MAC address “01:02:03:04:05:06” is created in the BACC/Network Registrar system.

Step 4 When the device is rebooted as the result of device disruption by PACE disruptor (AUTOMATIC is used in the Activation mode), it receives the exact IP address (10.10.10.5) set in the property by the service provider.

Rollback occurs if needed when the API command results in an error during processing or the change failed to commit to the RDU database. The command implementation attempts to roll the device back to the prior working configuration. In this case, reservation of 10.10.10.1 for MAC address “01:02:03:04:05:06” will be re-added to the BACC/Network Registrar system if it was removed; the reservation of 10.10.10.5 for MAC address “01:02:03:04:05:06” will be removed from BACC/Network Registrar system if it was created during command processing.

Assigning a New Device with an Old Device’s IP Address

After a device is registered to the BACC system with a reserved IP address, the service provider needs to replace the broken device with a device with a new MAC address.

Desired Outcome

The new device is booted with the same IP address granted to the broken old device. The reservation will be removed and recreated, as if the IP address of the device changed.

Step 1 A device is registered to BACC with a reserved IP address of 10.10.10.5.

Step 2 The service provider changes the MAC address of the existing device (“01:02:03:04:05:06”) to that of the new device (“01:02:03:04:05:07”) in the BACC system.

```

get-new-batch(AUTOMATIC, NO_CONFIRMATION);
// NO_ACTIVATION is the activation mode because we will
// not be able to reset as the new device has not booted
// on the network.
// NO_CONFIRMATION is the confirmation mode because we are
// not trying to reset the device

```

```
// To change the MAC address of a device:

changeMACAddress (
    "1,6,01:02:03:04:05:06",
        // old macAddress: unique identifier for the old device
    "1,6,01:02:03:04:05:07"
        //// new macAddress: unique identifier for the new device
);
```

- Step 3** The reservation of 10.10.10.5 for MAC address “01:02:03:04:05:06” is removed from the BACC/Network Registrar system. The reservation of 10.10.10.5 for MAC address “01:02:03:04:05:07” is created in the BACC/Network Registrar system.
- Step 4** When the device with MAC address “01:02:03:04:05:07” is turned on, it receives the same IP address (10.10.10.5) granted to the broken old device.

Rollback occurs if needed when the API command results in an error during processing or the change failed to commit to the RDU database. The command implementation attempts to roll the device back to the prior working configuration. In this case, the reservation of 10.10.10.5 for MAC address “01:02:03:04:05:06” will be re-added to BACC/Network Registrar system if it was removed, the reservation of 10.10.10.5 for MAC address “01:02:03:04:05:07” will be removed from BACC/Network Registrar system if it was created during command processing.

Removing a Reservation and Assigning a New IP Address

After a device is registered to the BACC system with a reserved IP address, the service provider needs to remove the reservation of the specific IP address since the device no longer needs a reserved IP assignment.

Desired Outcome

The reservation will be removed from the system and Network Registrar/BACC selects the next available IP address in the appropriate IP pool based on the selected DHCP criteria and assigns it to the device.

- Step 1** A device is registered to BACC with a reserved IP address of 10.10.10.5.
- Step 2** The service provider's application makes these API calls in BACC to remove the reservation.

```
get-new-batch(AUTOMATIC, NO_CONFIRMATION);
// AUTOMATIC is the Activation mode because we are attempting
// to reset the device
// NO_CONFIRMATION is the Confirmation mode because we don't
// want the batch to fail if we can't reset the modem.

// Add the property IPDeviceKeys.IP_RESERVATION the list to be removed

list.add( IPDeviceKeys.IP_RESERVATION);

// To reassign a different IP to:

changeProperties(
```

```

        "1,6,01:02:03:04:05:07",
        // macAdd
        null, list
    );

```

- Step 3** The reservation of 10.10.10.5 for MAC address "1,6,01:02:03:04:05:07" is removed from the system.
- Step 4** When the device is rebooted as the result of device disruption (AUTOMATIC is used in the Activation mode), dynamic address assignment occurs. Network Registrar/BACC selects the next available IP address in an appropriated IP address pool based on the selected DHCP criteria set on the device and grants it to the device.

Rollback occurs if needed when the API command results in an error during processing or the change failed to commit to the RDU database. The command implementation attempts to roll the device back to the prior working configuration. In this case, the reservation of 10.10.10.5 for MAC address "01:02:03:04:05:07" will be re-added from the BACC/Network Registrar system if it was removed during command processing.

Rebooting a Device with the Same IP Address

After a device is registered to the BACC system with a provisioned IP address (a dynamic IP assignment, not a reserved IP) granted by BACC/Network Registrar, the service provider reassigns the device with a different IP address using a BACC property (IPDeviceKeys.IP_RESERVATION).

Desired Outcome

The device is rebooted with the exact IP address set in the property (IPDeviceKeys.IP_RESERVATION) by the service provider. The granted new IP address is now reserved (lease reservation) for that device.

- Step 1** A device is registered to BACC with dynamic IP address (Network Registrar/BACC selected an IP address in an appropriated IP pool based on selected DHCP criteria on the device and granted it to the device).
- Step 2** The service provider's application makes these API calls in BACC to reassign the IP address.

```

get-new-batch(AUTOMATIC, NO_CONFIRMATION);
// AUTOMATIC is the Activation mode because we are attempting
// to reset the device
// NO_CONFIRMATION is the Confirmation mode because we don't
// want the batch to fail if we can't reset the modem.

Map properties;

// Set the property IPDeviceKeys.IP_RESERVATION to a specific
// IP address

properties.put( IPDeviceKeys.IP_RESERVATION, "10.10.10.1");

// To reassign a different IP to:

changeProperties(

    "1,6,01:02:03:04:05:08",

```

```

        // macAdd
        properties, null

    ):

```

Step 3 The reservation of 10.10.10.1 for Mac address "01:02:03:04:05:08" is made with BACC/Network Registrar.

Step 4 When the device is rebooted as the result of device disruption (AUTOMATIC is used in the Activation mode), it receives the exact IP address (10.10.10.1) set in the property by the service provider.

Rollback occurs if needed when the API command results in an error during processing or the change failed to commit to the RDU database. The command implementation attempts to roll the device back to the prior working configuration. The reservation of 10.10.10.1 for MAC address "01:02:03:04:05:08" will be removed from BACC/Network Registrar system if it was added during command processing. When the device reboots, BACC/Network Registrar selects the next available IP address in the appropriate IP pool based on the selected DHCP criteria and grants it to the device.

Removing a Device from BACC

A service provider needs to remove the subscriber's device with a reserved IP from the BACC system.

Desired Outcome

Permanently remove the subscriber's device from the BACC system. The granted IP address is unreserved for that device.

Step 1 A device is registered to BACC with a reserved IP address of 10.10.10.5.

Step 2 The service provider uses its own user interface to remove the device from the BACC system. The service provider's user interface, acting as a BACC client, passes the information to BACC. BACC updates the device information and removes the device.

```

delete(
    "1,6,01:02:03:04:05:07",
    // macAdd
    // deleteDevicesBehind res: unique identifier for this device
    true: deletes CPEs behind this modem.
):

```

Step 3 The reservation of 10.10.10.5 for MAC address "01:02:03:04:05:07" is also removed from BACC/Network Registrar system.

Rollback occurs if needed when the API command results in an error during processing or the change failed to commit to the RDU database. The command implementation attempts to roll the device back to the prior working configuration. The reservation of 10.10.10.5 for MAC address "01:02:03:04:05:07" will be re-added from the BACC/Network Registrar system if it was removed during command processing.

A Submitted Batch Fails when BACC Uses CCM

BACC is configured to use CCM. The OSS submits an API request to add or remove a reservation, but CCM is unavailable (connection down, CCM is incorrectly configured, CCM License issue, and so on).

Desired Outcome:

When a submitted batch fails, no reservation should be added or removed in BACC/Network Registrar; a pre-defined, well-known error code is correctly returned.

-
- Step 1** The OSS builds and submits a batch containing the API calls listed in the [“API Calls Affected by the Lease Reservation Feature”](#) section on page C-49 for adding or removing reservations.
- Step 2** The RDU receives the batch and processes it. During the processing, the RDU (via the API commands listed in the [“API Calls Affected by the Lease Reservation Feature”](#) section on page C-49) makes an external CCM call for lease reservation related tasks.
- Step 3** CCM is unavailable and an error occurs. Return status code from the API call will be set to `CommandStatusCodes .CMD_ERROR_CCM_UNREACHABLE`.
-

A Submitted Batch Fails when BACC does not Use CCM

BACC is **not** configured to use CCM. The OSS submits an API request to add or remove a reservation.

Desired Outcome

Submitted batch failed. No reservation is added/removed in the BACC/Network Registrar system; a pre-defined, well-known error code is correctly returned.

-
- Step 1** The OSS builds and submits a batch containing the API calls listed in the [“API Calls Affected by the Lease Reservation Feature”](#) section on page C-49 for adding or removing reservations.
- Step 2** The RDU receives the batch and processes it. During the processing, the RDU (via the API commands listed in the [“API Calls Affected by the Lease Reservation Feature”](#) section on page C-49) detects that CCM is not configured for the lease reservation feature.
- Step 3** An error occurs. Return status code from the API call will be set to `CommandStatusCodes .CMD_ERROR_CCM_NOT_CONFIGURED`.
-



A

- active logs** These database log files contain data that has not yet been written into the database. It is important to keep active log files until they become redundant. *See also* redundant logs and removable logs.
- agent** A watchdog agent is a daemon process that is used to monitor, stop, start and restart BAC component processes such as the RDU, JRun, and the SNMP agent.
- alert** A syslog or SNMP message notifying an operator or administrator of a network problem.
- API** Application programming interface. Specification of function-call conventions that defines an interface to a service.
- audit logs** A log file containing a summary of major changes in the RDU database. This includes changes to system defaults, technology defaults, DHCP criteria, and classes of service.

B

- bandwidth** Difference between the highest and lowest frequencies available for network signals. The term is also used to describe the rated throughput capacity of a given network medium or protocol.
- BACC** An integrated solution for data-over-cable service providers to configure and manage broadband modems, and enable and administer subscriber self-registration and activation. BACC is a scalable product capable of supporting millions of devices.
- broadband** Transmission system that multiplexes multiple independent signals onto one cable. In Telecommunications terminology; any channel having a bandwidth greater than a voice-grade channel (4 kHz). In LAN terminology; a co-axial cable on which analog signaling is used.
- Broadband Access Center for Cable** *See* BACC.

C

- cable modem termination system** *See* CMTS.
- caching** Form of replication in which information learned during a previous transaction is used to process later transactions.

client-class	A Network Registrar feature that provides differentiated services to users that are connected to a common network. The client-class is used in the BACC DHCP criteria to provide differentiated DHCP services to devices.
CMTS	Cable modem termination system. A CMTS is a component that exchanges digital signals with cable modems on a cable network. The CMTS is usually located in the cable provider's local office.
CMTS shared secret	<i>See</i> shared secret.
configuration file	A file containing configuration parameters for the device to be provisioned.
configuration generation	The process of combining a set of attributes to be delivered to a device using either DHCP, TFTP, or SNMP.
CPE	Customer premises equipment. Terminating equipment, such as telephones, computers, and modems, supplied and installed at a customer location.
customer premises equipment	<i>See</i> CPE.

D

device provisioning engine	<i>See</i> DPE.
DOCSIS Shared Secret	Shared secret for communication between DOCSIS devices in a BACC deployment.
Data Over Cable Service Interface Specification	<i>See</i> DOCSIS.
DOCSIS	Data over cable service interface specification. DOCSIS defines functionality in cable modems involved in high-speed data distribution over cable television system networks.
DPE	Device provisioning engine. The DPE caches device information to ensure BACC scalability and handles configuration requests including downloading configuration files to devices.
DSTB	Digital set-top box. A device that enables a television to become a user interface to the Internet and to receive and decode digital television signals.
Dynamic Configuration File	A dynamically created configuration file that uses template files to provide greater flexibility and security in the provisioning process.

F

FQDN	Fully qualified domain name. FQDN is the full name of a system, rather than just its hostname. For example, cisco is a hostname and www.cisco.com is an FQDN.
fully qualified domain name	<i>See</i> FQDN.

I

IP address An IP address is a 32-bit number that identifies each sender or receiver of information that is sent in packets across the Internet.

K

Key Distribution Center *See* KDC.

KDC A key distribution center that implements limited Kerberos functionality. Used in the provisioning of PacketCable MTAs.

M

MAC address Standardized data link layer address that is required for every port or device that connects to a LAN. Other devices in the network use these addresses to locate specific ports in the network and to create and update routing tables and data structures. MAC addresses are 6 bytes long and are controlled by IEEE. Also known as hardware address, MAC-layer address, or physical address. Compare with *network address*.

Media Terminal Adapter *See* MTA.

MSO Multiple system operator. A company that operates more than one cable TV or broadband system.

MTA Equipment at the customer end of a broadband (packetcable) network.

multiple service operator *See* MSO.

N

NAT Network address translation. Mechanism for reducing the need for globally unique IP addresses. NAT allows an organization with addresses that are not globally unique to connect to the Internet by translating those addresses into globally routeable address space. This is also known as Network Address Translation.

network address translation *See* NAT.

network administrator Person responsible for operation, maintenance, and management of a network. *See also* network operator.

network operator Person who routinely monitors and controls a network, performing such tasks as reviewing and responding to alarms, monitoring throughput, configuring new circuits, and resolving problems. *See also* network administrator.

Network Time Protocol	<i>See</i> NTP.
NR	Cisco Network Registrar. A software product that provides IP addresses, configuration parameters, and DNS names to DOCSIS cable modems and PCs, based on network and service policies.
NTP	Network Time Protocol (NTP). The NTP is a protocol designed to synchronize server clocks over a network.

P

provisioning API	A series of BACC functions that programs can use to make the operating system perform various functions.
provisioning groups	Groupings of DPE and DHCP servers, based on either network topology or geography.
publishing	Publishing provisioning information to an external datastore in real time. Publishing plug-ins must be developed to write data to a datastore.

R

RDU	Regional distribution unit. The RDU is the primary server in the BACC provisioning system. It manages generation of device configurations, processes all API requests, and manages the BACC system.
realm	The logical network served by a single Kerberos database and a set of Key Distribution Centers.
realm names	By convention, realm names are generally all uppercase letters, to differentiate the realm from the Internet domain. <i>See</i> realm.
redundancy	In internetworking, the duplication of devices, services, or connections so that, in the event of a failure, the redundant devices, services, or connections can perform the work of those that failed.
redundant logs	Database log files become redundant once its data has been written into the database. <i>See also</i> active logs and removable logs.
removable logs	Database log files become removable after either being backed up, or when the complete database that contains data for this log file has been backed up. <i>See also</i> active logs and redundant logs.

S

selection tags	Selection tags associated with Network Registrar scopes. These define the clients and client-classes associated with a scope.
-----------------------	---

- shared secret** A character string used to provide secure communication between two servers or devices.
- static configuration files** These files are used as a configuration file for a device. For example, a static configuration file called gold.cm would identify the gold DOCSIS class of service. BACC treats this file type like any other binary file.

T

- template files** Text files that contain DOCSIS or PacketCable MTA options and values that, when used in conjunction with a DOCSIS or PacketCable MTA class of service, provide dynamic file generation.
- TFTP** Trivial File Transfer Protocol. Simplified version of file transfer protocol (FTP) that allows files to be transferred from one computer to another over a network.
- TLV** Type-Length-Value. A tuple within a DOCSIS or PacketCable configuration file.
- trivial file transfer protocol** *See* TFTP.
- tuple** In programming languages, a tuple is an ordered set of values. Common uses for the tuple as a data type are: for passing a string of parameters from one program to another, or to represent a set of value attributes in a relational database.
- Type Length Value** *See* TLV.

U

- uBr** Universal Broadband Router (such as the Cisco 7246 or 7223), which is the Cisco router implementation of a DOCSIS CMTS.

V

- Voice over IP** *See* VoIP.
- VoIP** Voice over IP. VoIP is the ability to make telephone calls and send faxes over IP-based data networks with a suitable quality of service (QoS) and superior cost/benefit.

X

- XGCP** A Gateway Control Protocol used to pass data between networks. This includes that M (for Media) GCP and S (Simple) GCP.



A

- administrator's user interface [8-1 to 8-7](#)
 - accessing [8-1](#)
 - BACC architecture, and [2-19](#)
 - logging in [8-1](#)
 - logging out [8-3](#)
 - navigating [8-4](#)
 - Configuration menu [8-5](#)
 - Devices menu [8-5](#)
 - main menu [8-5](#)
 - menu bar [8-4](#)
 - Nodes menu [8-6](#)
 - Server menu [8-6](#)
 - Users menu [8-6](#)
 - scrolling [8-7](#)
- administrator provisioning examples [11-6 to 11-9](#)
 - account maintenance [11-7](#)
 - cable modems, managing [11-8](#)
 - classes of service, managing [11-7](#)
 - computers, managing [11-8](#)
 - deleting an account [11-9](#)
 - new accounts, registering [11-7](#)
 - accounts, searching for [11-6](#)
 - by account number [11-6](#)
 - by IP address [11-6](#)
 - by MAC address [11-7](#)
- advanced concepts (see tools and advanced concepts)
- agents, BACC architecture and [2-14 to 2-16](#)
 - agent, definition [GL-1](#)
 - agent alerts (see agent alerts)
 - BACC agent [2-15](#)
 - command line [2-16](#)
 - monitored processes [2-15](#)
 - DPE SNMP agent
 - MIB support [2-14](#)
 - SNMP agent [2-14](#)
- alert messages [A-1 to A-6](#)
 - agent alerts [A-4](#)
 - AGENT-3-9001 [A-4](#)
 - AGENT-3-9002 [A-4](#)
 - AGENT-3-9003 [A-4](#)
 - AGENT-6-9004 [A-4](#)
 - AGENT-6-9005 [A-4](#)
 - alerts, definition [GL-1](#)
 - message format [A-1](#)
 - Network Registrar extension point alerts [A-5](#)
 - NR_EP-1-106 [A-5](#)
 - NR_EP-1-107 [A-5](#)
 - NR_EP-6-108 [A-5](#)
 - NR_EP-6-109 [A-5](#)
 - NR_EP-6-110 [A-6](#)
 - RDU alerts [A-2](#)
 - BPR-RDU-4-1140 [A-2](#)
 - RDU-1-101 [A-2](#)
 - RDU-1-103 [A-2](#)
 - RDU-1-111 [A-2](#)
 - RDU-1-115 [A-2](#)
 - Solaris DPE alerts [A-2](#)
 - DPE-1-102 [A-3](#)
 - DPE-1-104 [A-3](#)
 - DPE-1-109 [A-3](#)

- architecture [2-1 to 2-19](#)
 - about [2-1 to 2-2](#)
 - administrator's interface [2-19](#)
 - agents [2-14 to 2-16](#)
 - BACC agent [2-15](#)
 - SNMP agent [2-14](#)
 - DPEs [2-4 to ??](#)
 - device types [2-4](#)
 - DSS, and [2-7](#)
 - license keys [2-5](#)
 - server assignments [2-7](#)
 - TACACS+, and DPE authentication [2-6](#)
 - TFTP server, and [2-7](#)
 - KDC [2-10 to 2-13](#)
 - certificates [2-12](#)
 - default KDC properties [2-10](#)
 - EuroPacketCable support [2-12](#)
 - licenses [2-12](#)
 - multiple-realm support [2-13](#)
 - logging [2-16 to 2-19](#)
 - dpe.log file, viewing [2-19](#)
 - DPE logs [2-17](#)
 - MIBs [2-13 to 2-14](#)
 - Network Registrar [2-9 to 2-10](#)
 - DHCP, and [2-9](#)
 - DNS, and [2-9](#)
 - lease reservation [2-10](#)
 - provisioning groups [2-8](#)
 - RDU s [2-3 to 2-4](#)
 - configuration generation [2-3](#)
 - service-level selection [2-3](#)
 - unit failover [2-4](#)
 - registration modes [2-2](#)
 - mixed mode [2-2](#)
 - promiscuous mode [2-2](#)
 - roaming mode [2-2](#)
 - standard mode [2-2](#)
 - sample user interface [2-19](#)
- ATA 186 defaults, configuring [10-7](#)
 - ATA 188 defaults, configuring [10-9](#)
 - audience for this document [xix](#)
 - audit logs, definition [GL-1](#)
-
- ## B
- BACC (BAC), definition [GL-1](#)
 - backup and recovery of database [7-4 to 7-6](#)
-
- ## C
- CableHome
 - configuration [6-1 to 6-3](#)
 - Device Provisioning Engine [6-3](#)
 - Network Registrar [6-2](#)
 - non-secure provisioning flow [6-1 to 6-2](#)
 - RDU [6-2](#)
 - provisioning
 - checklist (nonsecure) [3-14](#)
 - support [1-2](#)
 - WAN defaults, configuring [10-9](#)
 - WAN-Man defaults, configuring [10-11](#)
 - CableLabs Service Provider certificate trust hierarchy [5-18](#)
 - operational ancillary certificates [5-21](#)
 - Delivery Function (DF) certificate [5-21](#)
 - key distribution center (KDC) certificate [5-21](#)
 - PacketCable Server certificates [5-22](#)
 - root certificate [5-19](#)
 - Service Provider CA certificate [5-19](#)
 - cable modem states, troubleshooting [4-10 to 4-32](#)
 - DHCP - init(d) [4-18](#)
 - DHCP - init(i) [4-20](#)
 - Offline [4-11](#)
 - Online [4-10](#)
 - Option File Transfer Started-init(o) [4-25](#)
 - Ranging Process - init(r1),init(r2), and init(rc) [4-16](#)
 - Registration - reject (c) [4-32](#)
 - Registration - reject (m) [4-31](#)

- Reject(pk) and Reject(pt) [4-30](#)
- TOD exchange- init(t) [4-23](#)
- cautions
 - regarding
 - BACC_HOME/kdc directory [2-12](#)
 - BACC DHCP option settings [3-6](#)
 - cnr_ep.properties file [B-1](#)
 - custom properties, deleting [10-6](#)
 - disk space requirement figures [7-3](#)
 - DOCSISModem class of service, adding [10-3](#)
 - DSS, multiple, within one provisioning group [2-7](#)
 - KDC certificates, missing [2-12](#)
 - kerb auth nomTimeout, maxTimeout, and maxRetries [B-2](#)
 - network deployment with an evaluation license key [10-29](#)
 - sample user interface [2-19, 11-1](#)
 - template files, deleting [10-29](#)
 - troubleshooting devices by MAC address [12-52](#)
 - uBR7246 devices and debug commands [4-8, 4-19](#)
 - significance of [xxi](#)
- certificate revocation [5-24](#)
- certificate trust hierarchy for PacketCable [5-14 to 5-27](#)
 - MTA device certificate [5-17](#)
 - MTA device certificate hierarchy [5-15](#)
 - MTA Manufacturer Certificate [5-16](#)
 - MTA root certificate [5-16](#)
- validation [5-15](#)
- classes of service
 - configuring [10-1 to 10-6](#)
 - adding a class [10-3](#)
 - deleting a class [10-5](#)
 - modifying a class [10-4](#)
 - managing, administrator provisioning example [11-7](#)
 - SUI configuring sample [11-2](#)
- client-class, definition [GL-2](#)
- CMTS (cable modem termination system)
 - definition [GL-2](#)
 - shared secret, definition [GL-5](#)
- code verification certificate hierarchy [5-24](#)
 - CableLabs code verification CA certificate [5-25](#)
 - CableLabs code verification root CA certificate [5-25](#)
 - certificate revocation lists for CVCs [5-27](#)
 - common CVC requirements [5-24](#)
 - manufacturer code verification certificate [5-26](#)
 - service provider code verification certificate [5-26](#)
- configuration file, definition [GL-2](#)
- Configuration File state, and DOCSIS configuration [4-9](#)
- configuration file utility, using [12-25 to 12-38](#)
 - binary file, external, viewing [12-37](#)
 - binary file, local, viewing [12-36](#)
 - binary file output, specifying [12-35](#)
 - binary files, converting into template files [12-28](#)
 - macro variables, specifying a device for [12-33](#)
 - macro variables, specifying through CLI [12-32](#)
 - PacketCable BASIC flow, activating [12-38](#)
 - parsing external template files [12-29](#)
 - parsing local template files [12-28](#)
 - parsing template files and adding a shared secret [12-31](#)
 - running [12-26](#)
 - using [12-27](#)
- configuration generation, definition [GL-2](#)
- Configuration menu, about [8-5](#)
- configuration workflows and checklists [3-1 to 3-14](#)
 - component workflows [3-1 to 3-6](#)
 - DPE checklist [3-2 to 3-4](#)
 - Network Registrar checklist [3-6](#)
 - RDU checklist [3-2](#)
 - Solaris DPE checklist [3-4 to 3-5](#)
- technology workflows [3-7 to 3-14](#)
 - CableHome provisioning (nonsecure) [3-14](#)
 - DOCSIS checklist [3-7](#)
 - PacketCable checklist [3-8 to 3-14](#)
- configuring DOCSIS
 - network layer and above
 - DHCP state [4-7](#)
 - ToD state [4-8](#)

- configuring BACC [10-1 to 10-36](#)
 - class of service [10-1 to 10-6](#)
 - adding a class [10-3](#)
 - deleting [10-5](#)
 - modifying [10-4](#)
 - custom properties [10-6](#)
 - defaults [10-7 to 10-23](#)
 - ATA 186 defaults [10-7](#)
 - ATA 188 defaults [10-9](#)
 - CableHome WAN defaults [10-9](#)
 - computer defaults [10-12](#)
 - configuration options, selecting [10-7](#)
 - DOCSIS defaults [10-13](#)
 - Network Registrar defaults [10-15](#)
 - PacketCable defaults [10-17](#)
 - DHCP criteria [10-23 to 10-24](#)
 - adding criteria [10-23](#)
 - deleting criteria [10-24](#)
 - modifying criteria [10-24](#)
 - external files, managing [10-25 to 10-29](#)
 - adding files [10-26](#)
 - deleting files [10-29](#)
 - exporting files [10-28](#)
 - replacing files [10-28](#)
 - viewing files [10-26](#)
 - FQDN, automatic generation [10-35 to 10-36](#)
 - format [10-35](#)
 - properties [10-36](#)
 - sample [10-36](#)
 - validation [10-36](#)
 - license keys, managing [10-29 to 10-30](#)
 - adding a license [10-30](#)
 - modifying a license [10-30](#)
 - provisioning data, publishing [10-32 to 10-33](#)
 - Datastore changes [10-33](#)
 - plug-in settings, modifying [10-33](#)
 - RDU unit extensions, managing [10-31 to 10-32](#)
 - custom extension points, installing [10-31](#)
 - new class, writing [10-31](#)
 - viewing [10-32](#)
 - SNMPV3 cloning on the RDU and DPE [10-34 to 10-35](#)
 - key generation [10-34](#)
 - key material [10-34](#)
 - SRV Record in Network Registrar DNS server [10-34](#)
- configuring CableHome [6-1 to 6-3](#)
 - DPE [6-3](#)
 - Network Registrar [6-2](#)
 - RDU [6-2](#)
- configuring DOCSIS
 - network layer and above
 - Configuration File state [4-9](#)
 - Establish Privacy state [4-9](#)
 - Operational state [4-10](#)
 - Registration state [4-9](#)
 - Security Association state [4-8](#)
 - workflow
 - checklist [3-7](#)
 - full [4-2 to 4-4](#)
- configuring Network Registrar [6-2](#)
 - defaults [10-15](#)
 - SRV record in DNS server [10-34](#)
 - workflow checklist [3-6](#)
- configuring PacketCable [5-1 to 5-27](#)
 - certificate trust hierarchies [5-14 to 5-27](#)
 - CableLabs service provider certificate hierarchy [5-18](#)
 - certificate revocation [5-24](#)
 - certificate validation [5-15](#)
 - code verification certificate hierarchy [5-24](#)
 - MTA device certificate [5-17](#)
 - MTA hierarchy [5-15](#)
 - EuroPacketCable [5-5 to ??](#)
 - PacketCable BASIC [5-4](#)
 - SNMP v2C notifications [5-5](#)
 - TLV 38 and MIB support [5-5](#)
 - PacketCable Secure [5-1](#)
 - troubleshooting EMTA provisioning [5-6 to 5-9](#)
 - components [5-6](#)
 - key variables [5-8](#)

- troubleshooting scenarios [5-10 to 5-13](#)
- troubleshooting tools [5-9 to 5-10](#)
 - Ethereal, SnifferPro, or other [5-10](#)
 - logs [5-10](#)
- configuring SUI [11-1 to 11-12](#)
 - administrator provisioning examples [11-6 to 11-9](#)
 - account maintenance [11-7](#)
 - accounts, searching for [11-6](#)
 - sample configuration options [11-2 to 11-3](#)
 - administrative access levels [11-3](#)
 - classes of service [11-2](#)
 - ISP, selecting [11-3](#)
 - Promiscuous mode [11-2](#)
 - technician login, using [11-3](#)
 - sample sampleui.properties file [11-9 to 11-12](#)
 - subscriber provisioning examples [11-3 to 11-6](#)
 - promiscuous customer premise equipment registration [11-5](#)
 - standard customer premise equipment registration [11-4](#)
- customer premises equipment, definition [GL-2](#)

D

- database management [7-1 to 7-7](#)
 - backup and recovery [7-4 to 7-6](#)
 - backing up [7-4](#)
 - recovering [7-5](#)
 - restoring [7-5](#)
 - disk space requirements [7-3](#)
 - caution regarding [7-3](#)
 - handling out [7-3](#)
 - failure resiliency, understanding [7-1](#)
 - files [7-1 to 7-3](#)
 - automatic log management [7-2](#)
 - DB_VERSION [7-3](#)
 - history log [7-3](#)
 - storage [7-2](#)
 - transaction log [7-2](#)

- location, changing [7-6 to 7-7](#)
- RDU database migration [7-7](#)
- device management [9-3 to 9-15](#)
 - about [9-12](#)
 - adding devices [9-13](#)
 - controls [9-5](#)
 - deleting Devices [9-14](#)
 - device configurations, regenerating [9-14](#)
 - device details, viewing [9-8](#)
 - finding devices [9-6](#)
 - Manage Devices page [9-4](#)
 - modifying devices [9-13](#)
 - relating and unrelating devices [9-15](#)
 - resetting devices [9-15](#)
 - searching for devices [9-7](#)
 - unregistering devices [9-15](#)

Devices menu, about [8-5](#)

DHCP

- criteria defaults, configuring [10-23 to 10-24](#)
 - adding criteria [10-23](#)
 - deleting criteria [10-24](#)
 - modifying criteria [10-24](#)
- Network Registrar, and [2-9](#)
- state, and DOCSIS configuration [4-7](#)
- disk_monitor.sh tool [12-51 to 12-52](#)
- disk space, monitoring [12-51 to 12-52](#)
- DNS (Domain Name System), Network Registrar, and [2-9](#)
- DOCSIS (Data Over Cable Service Interface Specification)
 - configuring [4-1 to 4-32](#)
 - network layer and above [4-7 to 4-10](#)
 - workflow, checklist [3-7](#)
 - workflow, full [4-2 to 4-4](#)
 - definition [GL-2](#)
 - features [4-5 to 4-6](#)
 - DOCSIS 1.0, 1.1, 2.0 support [4-6](#)
 - dynamic configuration TLVs [4-5](#)
 - high-speed data support [1-2](#)

- shared secret, definition [GL-5](#)
- troubleshooting DOCSIS networks [4-6](#)
- documentation [xxi](#)
 - related to this product [xxi](#)
 - audience [xix](#)
 - organization [xx](#)
 - typographical conventions used in [xxi](#)
- DPE (Device Provisioning Engine)
 - about [2-4 to ??](#)
 - BACC architecture, and
 - dpe.log file, viewing [2-19](#)
 - logs [2-17](#)
 - configuring [6-3](#)
 - definition [GL-2](#)
 - device types [2-4](#)
 - hardware DPE (Device Provisioning Engine) [2-5](#)
 - Solaris DPEs [2-5](#)
 - dpe.log file, viewing [2-19](#)
 - DSS, and [2-7](#)
 - resetting DSS [2-8](#)
 - hardware DPE checklist [3-2 to 3-4](#)
 - license keys [2-5](#)
 - listing
 - currently registered DPEs [9-19](#)
 - servers [8-6](#)
 - server assignments [2-7](#)
 - SNMPV3 cloning, configuring [10-34 to 10-35](#)
 - key generation [10-34](#)
 - key material [10-34](#)
 - Solaris DPE alerts [A-2](#)
 - DPE-1-102 [A-3](#)
 - DPE-1-109 [A-3](#)
 - SPE-1-104 [A-3](#)
 - Solaris DPE checklist [3-4 to 3-5](#)
 - TACACS+, and DPE authentication [2-6](#)
 - client settings [2-6](#)
 - privilege levels [2-6](#)
 - TFTP server, and [2-7](#)

- DSS (DOCSIS shared secret), and DPEs
 - about [2-7](#)
 - resetting DSS [2-8](#)
- DSTB, definition [GL-2](#)
- Dynamic Configuration File, definition [GL-2](#)

E

- EMTA provisioning for PacketCable,
 - troubleshooting [5-6 to 5-9](#)
- components [5-6](#)
 - call management server [5-8](#)
 - DHCP Server [5-7](#)
 - DNS server [5-7](#)
 - embedded MTA [5-7](#)
 - Key Distribution Center [5-8](#)
 - PacketCable Provisioning Server [5-8](#)
- key variables [5-8](#)
 - certificates [5-8](#)
 - MTA configuration file [5-9](#)
 - scope selection tag(s) [5-9](#)
- error messages, RDU [A-6 to A-9](#)
 - AddReservation: AX_ETIME [A-8](#)
 - AddReservation: FORWARD_FAILED [A-8](#)
 - AddReservation: INVALID_OBJECT [A-9](#)
 - AddReservation: INVALID_PARENT [A-7](#)
 - AddReservation: INVALID_SECOND_PARENT [A-7](#)
 - AX_EIO [A-9](#)
 - AX_EPIPE [A-9](#)
 - OBJECT_EXISTS [A-7](#)
 - RemoveReservation: NOT_FOUND [A-7](#)
 - selection-criteria exclusion tags ignored [A-9](#)
- Establish Privacy state, and DOCSIS configuration [4-9](#)
- Ethereal, packet capture tool [5-10](#)
- EuroPacketCable
 - configuration [5-5 to ??](#)
 - support, and KDC [2-12](#)
 - voice services support [1-2](#)

external files, managing [10-25 to 10-29](#)

- adding [10-26](#)
- deleting [10-29](#)
- exporting [10-28](#)
- replacing [10-28](#)
- viewing [10-26](#)

F

features of BACC, overview [1-1](#)

FQDN (fully qualified domain name)

- automatic generation [10-35 to 10-36](#)
 - format [10-35](#)
 - properties [10-36](#)
 - sample [10-36](#)
 - validation [10-36](#)
- definition [GL-2](#)

G

GUI (see administrator's user interface)

I

IP address, definition [GL-3](#)

ISP, selecting [11-3](#)

K

KDC (Key Distribution Center)

- BACC architecture, and [2-10 to 2-13](#)
 - certificates [2-12](#)
 - default KDC properties [2-10](#)
 - EuroPacketCable support [2-12](#)
 - licenses [2-12](#)
 - multiple-realm support [2-13](#)
- certificates, managing [12-41 to 12-43](#)
 - creating [12-42](#)

- running the PKCert tool [12-41](#)
- validating [12-43](#)
- definition [GL-3](#)

Keygen tool [12-45 to 12-46](#)

L

license keys, managing [10-29 to 10-30](#)

- adding a license [10-30](#)
- modifying a license [10-30](#)

logging

- active logs, definition [GL-1](#)
- BACC architecture, and [2-16 to 2-19](#)
 - dpe.log file, viewing [2-19](#)
 - DPE logs [2-17](#)
 - log levels, configuring [2-18](#)
 - log levels and structures [2-17](#)
 - RDU logs [2-17](#)
- redundant logs, definition [GL-4](#)
- removable logs, definition [GL-4](#)

logging into BACC [8-1](#)

M

MAC address, definition [GL-3](#)

MIBs

- BACC architecture, and [2-13 to 2-14](#)
- CableHome, and SNMP Varbind [12-6](#)
- DOCSIS, and SNMP Varbind [12-5](#)
- DPE SNMP agent, and MIB support [2-14](#)
- EuroPacketCable, and PacketCable voice configuration [5-6](#)
- PacketCable, and SNMP Varbind [12-6](#)
- TLV 38, and MIB support [5-5](#)

MSO, definition [GL-3](#)

MTA (Media Terminal Adapter)

- definition [GL-3](#)
- manufacturer CVC, and DOCSIS [5-18](#)

multiple service operator, definition [GL-3](#)

N

NAT (network address translation), definition [GL-3](#)

network administrator, definition [GL-3](#)

network operator, definition [GL-3](#)

Network Registrar

about [2-9 to 2-10](#)

DHCP, and [2-9](#)

DNS, and [2-9](#)

lease reservation [2-10](#)

architecture [2-9 to 2-10](#)

configuring [6-2](#)

defaults, configuring [10-15](#)

DHCP, and [2-9](#)

DNS, and [2-9](#)

DNS server SRV record, configuring [10-34](#)

extension point alerts [A-5](#)

NR_EP-1-106 [A-5](#)

NR_EP-1-107 [A-5](#)

NR_EP-6-108 [A-5](#)

NR_EP-6-109 [A-5](#)

NR_EP-6-110 [A-6](#)

extension points, listing [9-21](#)

lease reservation [2-10](#)

servers, listing [8-6](#)

SRV record in DNS server, configuring [10-34](#)

workflow checklist [3-6](#)

Network Time Protocol, definition [GL-4](#)

nodes, managing [9-16 to 9-18](#)

about [9-17](#)

adding [9-17](#)

deleting [9-18](#)

details, viewing [9-18](#)

modifying [9-18](#)

Nodes menu, about [8-6](#)

node types [9-16](#)

adding [9-16](#)

deleting [9-17](#)

modifying [9-17](#)

relating and unrelating node types to nodes [9-18](#)

NR, definition [GL-4](#)

NRProperties.sh tool [12-43 to 12-45](#)

NTP, definition [GL-4](#)

O

Online state

Online, Online(d), Online(pk), Online(pt) states [4-27](#)

troubleshooting [4-10](#)

Operational state, and DOCSIS configuration [4-10](#)

overviews

BACC [1-1 to 1-2](#)

features and benefits [1-1](#)

technologies supported [1-1](#)

P

PacketCable

BACC properties, mapping to DHCP options [B-1 to B-3](#)

option 122 and BACC property comparison [B-2](#)

option 177 and BACC property comparison [B-2](#)

defaults, configuring [10-17](#)

RDU defaults [10-19](#)

System defaults [10-21](#)

xGCP gateway control protocol defaults [10-22](#)

MTAs, SNMPV3 cloning, and [10-34 to 10-35](#)

key generation [10-34](#)

key material [10-34](#)

voice services support

EuroPacketCable [1-2](#)

non-secure [1-2](#)

standard [1-2](#)

workflow checklists [3-8 to 3-14](#)

EuroPacketCable [3-12](#)

non-secure PacketCable [3-10](#)

PacketCable [3-8](#)

- PacketCable voice configuration [5-1 to 5-27](#)
 - certificate trust hierarchy [5-14 to 5-27](#)
 - CableLabs Service Provider [5-18](#)
 - code verification [5-24](#)
 - MTA device certificate [5-17](#)
 - MTA device certificate hierarchy [5-15](#)
 - revocation [5-24](#)
 - validation [5-15](#)
 - EMTA provisioning, troubleshooting [5-6 to 5-9](#)
 - components [5-6](#)
 - key variables [5-8](#)
 - EuroPacketCable [5-5 to ??](#)
 - EuroPacketCable MIBs [5-6](#)
 - PacketCable BASIC [5-4 to 5-5](#)
 - SNMP v2C notifications [5-5](#)
 - TLV 38 and MIB support [5-5](#)
 - PacketCable Secure [5-1 to 5-4](#)
 - PacketCable flows [5-1](#)
 - troubleshooting scenarios [5-10 to 5-13](#)
 - troubleshooting tools [5-9 to 5-10](#)
 - Ethereal, SnifferPro, and other [5-10](#)
 - logs [5-10](#)
- PKCert.sh tool [12-41 to 12-43](#)
 - KDC certificate, creating [12-42](#)
 - KDC certificates, validating [12-43](#)
 - running [12-41](#)
- provisioning API, definition [GL-4](#)
- provisioning data, publishing [10-32 to 10-33](#)
 - Datastore changes [10-33](#)
 - plug-in settings, modifying [10-33](#)
- provisioning groups
 - about [2-8](#)
 - definition [GL-4](#)
 - listing [8-6](#)
- publishing, definition [GL-4](#)

R

- RDU (Regional Distribution Units)
 - about [2-3 to 2-4](#)
 - configuration generation [2-3](#)
 - service-level selection [2-3](#)
 - unit failover [2-4](#)
 - alerts [A-2](#)
 - BPR-RDU-4-1140 [A-2](#)
 - RDU-1-101 [A-2](#)
 - RDU-1-103 [A-2](#)
 - RDU-1-111 [A-2](#)
 - RDU-1-115 [A-2](#)
 - configuring, and CableHome [6-2](#)
 - WAN-Data [6-3](#)
 - WAN-Man [6-2](#)
 - database migration [7-7](#)
 - definition [GL-4](#)
 - details, viewing [9-24](#)
 - error messages [A-6 to A-9](#)
 - AddReservation: AX_ETIME [A-8](#)
 - AddReservation:FORWARD_FAILED [A-8](#)
 - AddReservation: INVALID_OBJECT [A-9](#)
 - AddReservation: INVALID_PARENT [A-7](#)
 - AddReservation:
 - INVALID_SECOND_PARENT [A-7](#)
 - AX_EIO [A-9](#)
 - AX_EPIPE [A-9](#)
 - OBJECT_EXISTS [A-7](#)
 - RemoveReservation: NOT_FOUND [A-7](#)
 - selection-criteria exclusion tags ignored [A-9](#)
 - listing [8-6](#)
 - log level tool [12-38 to 12-41](#)
 - current log level, viewing [12-40](#)
 - setting [12-40](#)
 - using [12-39](#)
 - logs [2-17](#)

- SNMPV3 cloning, configuring [10-34 to 10-35](#)
 - key generation [10-34](#)
 - key material [10-34](#)
 - unit extensions, managing [10-31 to 10-32](#)
 - custom extension points, installing [10-31](#)
 - new class, writing [10-31](#)
 - viewing [10-32](#)
 - workflow checklist [3-2](#)
 - realm, definition [GL-4](#)
 - realm names, definition [GL-4](#)
 - redundancy, definition [GL-4](#)
 - redundant logs, definition [GL-4](#)
 - Registration state, and DOCSIS configuration [4-9](#)
 - removable logs, definition [GL-4](#)
-
- S**
- Security Association state, and DOCSIS configuration [4-8](#)
 - selection tags, definition [GL-4](#)
 - Server menu, about [8-6](#)
 - servers, viewing [9-18 to 9-26](#)
 - DPEs, listing [9-19](#)
 - provisioning groups, listing [9-23](#)
 - RDU unit details [9-24](#)
 - service classes (see classes of service)
 - shared secret
 - configuration file utility, and [12-31](#)
 - definition [GL-5](#)
 - DSS (DOCSIS Shared Secret)
 - about [2-7](#)
 - DPEs, and [2-7](#)
 - resetting [2-8](#)
 - SnifferPro, packet capture tool [5-10](#)
 - SNMP
 - cloning on PacketCable eMTA (use case) [C-33](#)
 - SNMP agent
 - community, adding [12-48](#)
 - community, deleting [12-48](#)
 - location, changing [12-50](#)
 - settings, listing [12-50](#)
 - starting [12-49](#)
 - stopping [12-49](#)
 - snmpAgentCfgUtil.sh Command [12-46 to ??](#)
 - hosts, adding [12-47](#)
 - hosts, deleting [12-47](#)
 - SNMP contacts, setting up new [12-50](#)
 - SNMP listening port, identifying [12-49](#)
 - SNMP notification types, specifying [12-51](#)
 - SNMPV3 cloning, configuring on RDU and DPE [10-34 to 10-35](#)
 - key generation [10-34](#)
 - key material [10-34](#)
 - SRV record in Network Registrar DNS server, configuring [10-34](#)
 - static configuration files, definition [GL-5](#)
 - subscriber provisioning examples [11-3 to 11-6](#)
 - promiscuous customer premise equipment registration [11-5](#)
 - existing cable modem and new computer [11-6](#)
 - new cable modem and new computer [11-5](#)
 - standard customer premise equipment registration [11-4](#)
 - existing cable modem and new computer [11-4](#)
 - existing computer ISP, altering [11-5](#)
 - new cable modem and new computer [11-4](#)
 - SUI (Sample User Interface)
 - about [11-1](#)
 - BACC architecture, and [2-19](#)
 - configuring [11-1 to 11-12](#)
 - administrative access levels [11-3](#)
 - classes of service [11-2](#)
 - ISP, selecting [11-3](#)
 - promiscuous mode [11-2](#)
 - sample configuration options [11-2 to 11-3](#)
 - starting and stopping [11-2](#)
 - technician login, using [11-3](#)
 - sampleui.properties file sample [11-9 to 11-12](#)

T

- template files, developing [12-1 to 12-24](#)
 - definition options, encoding types for [12-8](#)
 - BITS value syntax [12-10](#)
 - OCTETSTRING syntax [12-10](#)
 - DOCSIS option support [12-12](#)
 - grammar [12-2](#)
 - comments [12-3](#)
 - Include files [12-3](#)
 - instance modifier [12-5](#)
 - options [12-3](#)
 - macro variables [12-7](#)
 - PacketCable option support [12-23](#)
 - SNMP Varbind [12-5](#)
 - CableHome MIBs [12-6](#)
 - DOCSIS MIBs [12-5](#)
 - PacketCable MIBs [12-6](#)
 - template file definition [GL-5](#)
- TFTP, definition [GL-5](#)
- TLV, definition [GL-5](#)
- ToD state, establishing for DOCSIS configuration [4-8](#)
- tools and advanced concepts [12-1 to 12-52](#)
 - configuration file utility [12-25 to 12-38](#)
 - binary file, external, viewing [12-37](#)
 - binary file, local, viewing [12-36](#)
 - binary file output, specifying [12-35](#)
 - binary files, converting into template files [12-28](#)
 - macro variables, specifying a device for [12-33](#)
 - macro variables, specifying through CLI [12-32](#)
 - PacketCable BASIC flow, activating [12-38](#)
 - parsing external template files [12-29](#)
 - parsing local template files [12-28](#)
 - running [12-26](#)
 - using [12-27](#)
 - disk_monitor.sh tool [12-51 to 12-52](#)
 - Keyget tool [12-45 to 12-46](#)
 - non-secure CableHome option support [12-24 to 12-25](#)
 - NRProperties.sh tool [12-43 to 12-45](#)
 - PKCert.sh Tool [12-41 to 12-43](#)
 - KDC certificate, creating [12-42](#)
 - KDC certificates, validating [12-43](#)
 - running [12-41](#)
 - RDU log level tool [12-38 to 12-41](#)
 - current log level, viewing [12-40](#)
 - setting [12-40](#)
 - using [12-39](#)
 - snmpAgentCfgUtil.sh command [12-46 to ??](#)
 - hosts, adding [12-47](#)
 - hosts, deleting [12-47](#)
 - SNMP agent, starting [12-49](#)
 - SNMP agent, stopping [12-49](#)
 - SNMP agent community, adding [12-48](#)
 - SNMP agent community, deleting [12-48](#)
 - SNMP agent location, changing [12-50](#)
 - SNMP agent settings, listing [12-50](#)
 - SNMP contacts, setting up new [12-50](#)
 - SNMP listening port, identifying [12-49](#)
 - SNMP notification types, specifying [12-51](#)
- template files, developing [12-1 to 12-24](#)
 - definition options, encoding types for [12-8](#)
 - DOCSIS option support [12-12](#)
 - grammar [12-2](#)
 - macro variables [12-7](#)
 - PacketCable option support [12-23](#)
 - SNMP Varbind [12-5](#)
- troubleshooting devices by MAC address [12-52](#)
- trivial file transfer protocol, definition [GL-5](#)
- troubleshooting
 - alert messages [A-1 to A-6](#)
 - agent alerts [A-4](#)
 - message format [A-1](#)
 - RDU alerts [A-2](#)
 - Solaris DPE alerts [A-2](#)
 - cable modem states [4-10 to 4-32](#)
 - DHCP - init(d) [4-18](#)
 - DHCP - init(i) [4-20](#)
 - Offline [4-11](#)

- Online [4-10](#)
 - Options File Transfer Started-init(o) [4-25](#)
 - Ranging Process - init(r1),init(r2), and init(rc) [4-16](#)
 - Registration - reject (c) [4-32](#)
 - Registration - reject (m) [4-31](#)
 - Reject(pk) and Reject(pt) [4-30](#)
 - TOD exchange- (init(t)) [4-23](#)
 - devices, by MAC address [12-52](#)
 - DOCSIS networks [4-6](#)
 - EMTA provisioning for PacketCable [5-6 to 5-9](#)
 - components [5-6](#)
 - key variables [5-8](#)
 - scenarios for PacketCable voice configuration [5-10 to 5-13](#)
 - tools for PacketCable voice configuration [5-9 to 5-10](#)
 - Ethereal, SnifferPro, or other [5-10](#)
 - logs [5-10](#)
 - tuple, definition [GL-5](#)
 - Type Length Value, definition [GL-5](#)
 - typographical conventions in this document [xxi](#)
-
- ## U
- uBr, definition [GL-5](#)
 - uBR7246 devices and debug commands, cautions regarding [4-8, 4-19](#)
 - use cases
 - about use cases [C-1 to C-2](#)
 - adding
 - new computer behind a modem with NAT [C-22](#)
 - new computer in fixed standard mode [C-5](#)
 - second computer in promiscuous mode [C-20](#)
 - bulk provisioning 100 modems in Promiscuous mode [C-16](#)
 - CableHome with firewall configuration [C-42](#)
 - disabling a subscriber [C-7](#)
 - getting detailed device information [C-26](#)
 - incremental provisioning of PacketCable eMTA [C-34](#)
 - Lease Reservation use cases [C-48](#)
 - API calls affected by lease reservation [C-49](#)
 - assigning a new device with an old device's IP address [C-51](#)
 - bringing a device online using a service provider IP address [C-49](#)
 - rebooting a device with the same IP address [C-53](#)
 - removing a device from BACC [C-54](#)
 - removing and re-creating a reservation [C-50](#)
 - removing a reservation and assigning a new IP address [C-52](#)
 - submitted batch fails when BACC does not use CCM [C-55](#)
 - submitted batch fails when BACC uses CCM [C-55](#)
 - logging
 - batch completions using events [C-26](#)
 - device deletions using events [C-24](#)
 - modifying an existing modem [C-10](#)
 - monitoring an RDU connection using events [C-25](#)
 - moving a device to another DHCP scope [C-23](#)
 - optimistic locking [C-38](#)
 - pre-provisioning
 - Cable Home Wan-Man [C-41](#)
 - DOCSIS modems with dynamic configuration files [C-36](#)
 - first time activation in Promiscuous mode [C-18](#)
 - modems and self-provisioned computers [C-9](#)
 - PacketCable eMTA [C-31](#)
 - replacing an existing modem [C-19](#)
 - retrieving
 - device capabilities for Cable Home Wan-Man [C-44](#)
 - devices matching a vendor prefix [C-29](#)
 - searching using the default class of service [C-27](#)
 - self-provisioning
 - Cable Home Wan-Man [C-46](#)
 - first time activation in Promiscuous mode [C-14](#)
 - first time activation with NAT [C-20](#)
 - modem and computer in fixed standard mode [C-2](#)
 - SNMP cloning on PacketCable eMTA [C-33](#)
 - subscriber bandwidth, temporarily throttling [C-40](#)
 - unregistering and deleting a subscriber's devices [C-11](#)

- users, managing [9-1 to 9-3](#)
 - adding [9-2](#)
 - deleting [9-3](#)
 - modifying [9-3](#)
 - Users menu, about [8-6](#)
 - Administrator [8-6](#)
 - Read Only User [8-6](#)
 - Read Write User [8-6](#)
 - using BACC [9-1 to 9-26](#)
 - devices, managing [9-3 to 9-15](#)
 - about [9-12](#)
 - adding devices [9-13](#)
 - controls [9-5](#)
 - deleting devices [9-14](#)
 - device configurations, regenerating [9-14](#)
 - device details, viewing [9-8](#)
 - finding devices [9-6](#)
 - Manage Devices page [9-4](#)
 - modifying devices [9-13](#)
 - relating and unrelating devices [9-15](#)
 - resetting devices [9-15](#)
 - unregistering devices [9-15](#)
 - nodes, managing [9-16 to 9-18](#)
 - adding [9-17](#)
 - deleting [9-18](#)
 - details, viewing [9-18](#)
 - modifying [9-18](#)
 - relating and unrelating node types to nodes [9-18](#)
 - node types, managing [9-16](#)
 - adding a node type [9-16](#)
 - deleting node types [9-17](#)
 - modifying node types [9-17](#)
 - relating and unrelating node types to nodes [9-18](#)
 - servers, viewing [9-18 to 9-26](#)
 - DPEs, listing [9-19](#)
 - Network Registrar Extension Points, listing [9-21](#)
 - provisioning groups, listing [9-23](#)
 - RDU unit details [9-24](#)
 - user management [9-1 to 9-3](#)
 - adding a new user [9-2](#)
 - deleting users [9-3](#)
 - modifying users [9-3](#)
-
- V**
- voice services support [1-2](#)
 - EuroPacketCable [1-2](#)
 - PacketCable [1-2](#)
 - PacketCable, non-secure [1-2](#)
 - VoIP, definition [GL-5](#)
-
- X**
- XGCP, definition [GL-5](#)