



JunosE™ Software for E Series™ Broadband Services Routers

IP Services Configuration Guide

Release
12.3.x



Published: 2011-09-20

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Products made or sold by Juniper Networks or components thereof might be covered by one or more of the following patents that are owned by or licensed to Juniper Networks: U.S. Patent Nos. 5,473,599, 5,905,725, 5,909,440, 6,192,051, 6,333,650, 6,359,479, 6,406,312, 6,429,706, 6,459,579, 6,493,347, 6,538,518, 6,538,899, 6,552,918, 6,567,902, 6,578,186, and 6,590,785.

JunosE™ Software for E Series™ Broadband Services Routers IP Services Configuration Guide
Release 12.3.x
Copyright © 2011, Juniper Networks, Inc.
All rights reserved.

Revision History
October 2011—FRS JunosE 12.3.x

The information in this document is current as of the date listed in the revision history.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. The Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at <http://www.juniper.net/support/eula.html>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Abbreviated Table of Contents

	About the Documentation	xxi
Part 1	Chapters	
Chapter 1	Configuring Routing Policy	3
Chapter 2	Configuring NAT	61
Chapter 3	Configuring J-Flow Statistics	91
Chapter 4	Configuring BFD	107
Chapter 5	Configuring IPsec	119
Chapter 6	Configuring Dynamic IPsec Subscribers	169
Chapter 7	Configuring ANCP	185
Chapter 8	Configuring Digital Certificates	205
Chapter 9	Configuring IP Tunnels	237
Chapter 10	Configuring Dynamic IP Tunnels	251
Chapter 11	IP Reassembly for Tunnels	269
Chapter 12	Securing L2TP and IP Tunnels with IPsec	277
Chapter 13	Configuring the Mobile IP Home Agent	305
Part 2	Index	
	Index	323

Table of Contents

	About the Documentation	xxi
	E Series and JunosE Documentation and Release Notes	xxi
	Audience	xxi
	E Series and JunosE Text and Syntax Conventions	xxi
	Obtaining Documentation	xxiii
	Documentation Feedback	xxiii
	Requesting Technical Support	xxiii
	Self-Help Online Tools and Resources	xxiv
	Opening a Case with JTAC	xxiv
Part 1	Chapters	
Chapter 1	Configuring Routing Policy	3
	Overview	3
	Platform Considerations	4
	References	4
	Route Maps	4
	Route Map Configuration Example	5
	Multiple Values in a Match Entry	6
	Negating Match Clauses	7
	Matching a Community List Exactly	8
	Removing Community Lists from a Route Map	8
	Matching a Policy List	9
	Redistributing Access Routes	9
	Setting Multicast Bandwidths	9
	Match Policy Lists	19
	Access Lists	20
	Filtering Prefixes	20
	Configuration Example 1	21
	Configuration Example 2	21
	Configuration Example 3	22
	Filtering AS Paths	22
	Configuration Example 1	23
	Using Access Lists in a Route Map	24
	Configuration Example 1	24
	Using Access Lists for PIM Join Filters	29
	Clearing Access List Counters	30
	Creating Table Maps	30
	Using the Null Interface	32
	Prefix Lists	32
	Using a Prefix List	33

Chapter 2

Prefix Trees	35
Using a Prefix Tree	35
Community Lists	37
Extended Community Lists	40
Using Regular Expressions	42
AS-path Lists	42
Community Lists	43
Community Numbers	43
Metacharacters	43
Using Metacharacters as Literal Tokens	44
Regular Expression Examples	44
Managing the Routing Table	47
Troubleshooting Routing Policy	47
Monitoring Routing Policy	48
Configuring NAT	61
Overview	61
Platform Considerations	62
Module Requirements	62
References	62
NAT Configurations	63
Traditional NAT	63
Basic NAT	63
NAPT	63
Bidirectional NAT	64
Twice NAT	64
Network and Address Terms	64
Inside Local Addresses	65
Inside Global Addresses	65
Outside Local Addresses	65
Outside Global Addresses	65
Understanding Address Translation	65
Inside Source Translation	65
Outside Source Translation	66
Address Assignment Methods	66
Static Translations	66
Dynamic Translations	66
Order of Operations	66
Inside-to-Outside Translation	66
Outside-to-Inside Translation	67
PPTP and GRE Tunneling Through NAT	67
Packet Discard Rules	68
Before You Begin	68
Configuring a NAT License	68
Limiting Translation Entries	69
Specifying Inside and Outside Interfaces	69
Defining Static Address Translations	69
Creating Static Inside Source Translations	70
Creating Static Outside Source Translations	70

	Defining Dynamic Translations	71
	Creating Access List Rules	71
	Defining Address Pools	72
	Defining Dynamic Translation Rules	73
	Creating Dynamic Inside Source Translation Rules	74
	Creating Dynamic Outside Source Translation Rules	75
	Defining Translation Timeouts	75
	Clearing Dynamic Translations	76
	NAT Configuration Examples	77
	NAPT Example	77
	Bidirectional NAT Example	78
	Twice NAT Example	80
	Cross-VRF Example	81
	Tunnel Configuration Through NAT Examples	83
	Clients on an Inside Network	83
	Clients on an Outside Network	84
	GRE Flows Through NAT	84
	Monitoring NAT	85
	Displaying the NAT License Key	85
	Displaying Translation Statistics	85
	Displaying Translation Entries	87
	Displaying Address Pool Information	88
	Displaying Inside and Outside Rule Settings	89
Chapter 3	Configuring J-Flow Statistics	91
	Overview	91
	Interface Sampling	91
	Aggregation Caches	92
	Flow Collection	92
	Main Flow Cache Contents	92
	Cache Flow Export	93
	Aging Flows	93
	Operation with NAT	93
	Operation with High Availability	94
	Platform Considerations	94
	Before You Configure J-Flow Statistics	94
	Configuring Flow-Based Statistics Collection	94
	Enabling Flow-Based Statistics	95
	Enabling Flow-Based Statistics on an Interface	95
	Defining a Sampling Interval	95
	Setting Cache Size	97
	Defining Aging Timers	97
	Specifying the Activity Timer	97
	Specifying the Inactivity Timer	97
	Specifying Flow Export	98
	Configuring Aggregation Flow Caches	98
	Monitoring J-Flow Statistics	101
	Clearing J-Flow Statistics	101
	J-Flow show Commands	101

Chapter 4	Configuring BFD	107
	Bidirectional Forwarding Detection Overview	107
	How BFD Works	108
	Negotiation of the BFD Liveness Detection Interval	108
	BFD Platform Considerations	110
	BFD References	110
	Configuring a BFD License	111
	BFD Version Support	111
	Configuring BFD	112
	Managing BFD Adaptive Timer Intervals	112
	Clearing BFD Sessions	113
	Monitoring BFD	114
	System Event Logs	114
	Viewing BFD Information	115
Chapter 5	Configuring IPSec	119
	Overview	119
	IPSec Terms and Acronyms	119
	Platform Considerations	121
	References	121
	IPSec Concepts	122
	Secure IP Interfaces	122
	RFC 2401 Compliance	123
	IPSec Protocol Stack	123
	Security Parameters	124
	Manual Versus Signaled Interfaces	125
	Operational Virtual Router	126
	Transport Virtual Router	126
	Perfect Forward Secrecy	128
	Lifetime	128
	Inbound and Outbound SAs	129
	Transform Sets	130
	Other Security Features	132
	IP Security Policies	132
	ESP Processing	132
	AH Processing	133
	IPSec Maximums Supported	133
	DPD and IPSec Tunnel Failover	133
	Tunnel Failover	134
	IKE Overview	134
	Main Mode and Aggressive Mode	135
	Aggressive Mode Negotiations	135
	IKE Policies	136
	Priority	136
	Encryption	137
	Hash Function	137
	Authentication Mode	137
	Diffie-Hellman Group	137
	Lifetime	138

	IKE SA Negotiation	138
	Generating Private and Public Key Pairs	138
	Configuration Tasks	139
	Configuring an IPSec License	139
	Configuring IPSec Parameters	139
	Creating an IPSec Tunnel	142
	Configuring DPD and IPSec Tunnel Failover	147
	Defining an IKE Policy	149
	Refreshing SAs	151
	Enabling Notification of Invalid Cookies	152
	Configuration Examples	153
	Configuration Notes	153
	Monitoring IPSec	160
	System Event Logs	161
	show Commands	161
Chapter 6	Configuring Dynamic IPSec Subscribers	169
	Overview	169
	Dynamic Connection Setup	169
	Dynamic Connection Teardown	170
	Dynamic IPSec Subscriber Recognition	170
	Licensing Requirements	170
	Inherited Subscriber Functionality	171
	Using IPSec Tunnel Profiles	171
	Relocating Tunnel Interfaces	172
	User Authentication	172
	Platform Considerations	172
	References	173
	Creating an IPSec Tunnel Profile	173
	Configuring IPSec Tunnel Profiles	174
	Limiting Interface Instantiations on Each Profile	174
	Specifying IKE Settings	174
	Setting the IKE Local Identity	174
	Setting the IKE Peer Identity	175
	Appending a Domain Suffix to a Username	176
	Overriding IPSec Local and Peer Identities for SA Negotiations	176
	Specifying an IP Profile for IP Interface Instantiations	177
	Defining the Server IP Address	177
	Specifying Local Networks	178
	Defining IPSec Security Association Lifetime Parameters	178
	Defining User Reauthentication Protocol Values	179
	Specifying IPSec Security Association Transforms	179
	Specifying IPSec Security Association PFS and DH Group Parameters	180
	Defining the Tunnel MTU	180
	Defining IKE Policy Rules for IPSec Tunnels	180
	Specifying a Virtual Router for an IKE Policy Rule	181
	Defining Aggressive Mode for an IKE Policy Rule	181

	Monitoring IPSec Tunnel Profiles	182
	System Event Logs	182
	show Commands	182
Chapter 7	Configuring ANCP	185
	Overview	185
	Access Topology Discovery	186
	Line Configuration	186
	Transactional Multicast	186
	OAM	187
	Retrieval of DSL Line Rate Parameters	187
	Learning the Partition ID from an Access Node	187
	Platform Considerations	187
	References	188
	Configuring ANCP	188
	Creating a Listening TCP Socket for ANCP	188
	Accessing L2C Configuration Mode for ANCP	188
	Defining the ANCP Session Timeout	189
	Learning the Access Node Partition ID	189
	Configuring ANCP Interfaces	189
	Configuring ANCP Neighbors	190
	Accessing L2C Neighbor Configuration Mode for ANCP	190
	Defining an ANCP Neighbor	191
	Limiting Discovery Table Entries	191
	Clearing ANCP Neighbors	192
	Configuring Topology Discovery	192
	Configuring ANCP for QoS Adaptive Mode	192
	Triggering ANCP Line Configuration	193
	Adjusting the Data Rate Reported by ANCP for DSL Lines	194
	Configuring Transactional Multicast for IGMP	194
	Creating an IGMP Session for ANCP	195
	ANCP IGMP Configuration Example	195
	Complete Configuration Example	196
	Triggering ANCP OAM	197
	Monitoring ANCP	198
Chapter 8	Configuring Digital Certificates	205
	Overview	205
	Digital Certificate Terms and Acronyms	205
	Platform Considerations	206
	References	206
	IKE Authentication with Digital Certificates	207
	Signature Authentication	207
	Generating Public/Private Key Pairs	208
	Obtaining a Root CA Certificate	208
	Obtaining a Public Key Certificate	209
	Offline Certificate Enrollment	209
	Online Certificate Enrollment	209
	Authenticating the Peer	210
	Verifying CRLs	210

	File Extensions	211
	Certificate Chains	211
	IKE Authentication Using Public Keys Without Digital Certificates	212
	Configuration Tasks	212
	Public Key Format	213
	Configuring Digital Certificates Using the Offline Method	213
	Configuring Digital Certificates Using the Online Method	219
	Configuring Peer Public Keys Without Digital Certificates	224
	Monitoring Digital Certificates and Public Keys	228
Chapter 9	Configuring IP Tunnels	237
	Overview	237
	GRE Tunnels	237
	DVMRP Tunnels	238
	Platform Considerations	238
	Module Requirements	238
	ERX7xx Models, ERX14xx Models, and the ERX310 Router	238
	E120 Router and E320 Router	239
	Redundancy and Tunnel Distribution	239
	References	239
	Configuration Tasks	240
	Configuration Example	242
	Configuring IP Tunnels to Forward IP Frames	243
	Preventing Recursive Tunnels	244
	Creating Multicast VPNs Using GRE Tunnels	244
	Monitoring IP Tunnels	244
Chapter 10	Configuring Dynamic IP Tunnels	251
	Dynamic IP Tunnel Overview	251
	Data MDT for Multicast VPNs and Dynamic IP Tunnels	252
	Mobile IP and Dynamic IP Tunnels	252
	Combining Dynamic and Static IP Tunnels in the Same Chassis	253
	Changing and Removing Existing Dynamic IP Tunnels	253
	Platform Considerations	253
	Module Requirements	254
	ERX7xx Models, ERX14xx Models, and the ERX310 Router	254
	E120 Router and E320 Router	254
	Redundancy and Tunnel Distribution	255
	References	255
	Configuring a Destination Profile for Dynamic IP Tunnels	255
	Modifying the Default Destination Profile	255
	Modifying the Configuration of the Default Destination Profile	256
	Configuring a Destination Profile for GRE Tunnels	256
	Creating a Destination Profile for DVMRP Tunnels	257
	Monitoring Dynamic IP Tunnels	260

Chapter 11	IP Reassembly for Tunnels	269
	Overview	269
	Platform Considerations	270
	Module Requirements	270
	ERX7xx Models, ERX14xx Models, and the ERX310 Router	270
	E120 Router and E320 Router	271
	Configuring IP Reassembly	271
	Monitoring IP Reassembly	272
	Setting Statistics Baselines	272
	Displaying Statistics	273
Chapter 12	Securing L2TP and IP Tunnels with IPSec	277
	Overview	277
	Tunnel Creation	277
	IPSec Secured-Tunnel Maximums	278
	Platform Considerations	278
	Module Requirements	278
	References	278
	L2TP/IPSec Tunnels	279
	Setting Up the Secure L2TP Connection	280
	L2TP with IPSec Control and Data Frames	280
	Compatibility and Requirements	281
	Client Software Supported	281
	Interactions with NAT	281
	Interaction Between IPSec and PPP	281
	LNS Change of Port	282
	Group Preshared Key	282
	NAT Passthrough Mode	282
	NAT Traversal	282
	How NAT-T Works	283
	UDP Encapsulation	283
	UDP Statistics	284
	NAT Keepalive Messages	284
	Configuring and Monitoring NAT-T	285
	Single-Shot Tunnels	285
	Configuration Tasks for Client PC	286
	Configuration Tasks for E Series Routers	286
	Enabling IPSec Support for L2TP	287
	Configuring NAT-T	288
	Configuring Single-Shot Tunnels	289
	GRE/IPSec and DVMRP/IPSec Tunnels	290
	Setting Up the Secure GRE or DVMRP Connection	290
	Configuration Tasks	291
	Enabling IPSec Support for GRE and DVMRP Tunnels	291
	Configuring IPSec Transport Profiles	292
	Monitoring DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec Tunnels	296
	System Event Logs	296
	show Commands	296

Chapter 13	Configuring the Mobile IP Home Agent	305
	Mobile IP Overview	305
	Mobile IP Agent Discovery	306
	Mobile IP Registration	306
	Home Address Assignment	306
	Authentication	307
	AAA	307
	Subscriber Management	308
	Mobile IP Routing and Forwarding	308
	Mobile IP Platform Considerations	309
	Mobile IP References	309
	Before You Configure the Mobile IP Home Agent	309
	Configuring the Mobile IP Home Agent	310
	Monitoring the Mobile IP Home Agent	315
 Part 2	 Index	
	Index	323

List of Figures

Part 1	Chapters	
Chapter 1	Configuring Routing Policy	3
	Figure 1: Applying Route Maps to Routes	6
	Figure 2: Filtering with Access Lists	22
	Figure 3: Filtering with AS-Path Access Lists	23
	Figure 4: Route Map Filtering	24
	Figure 5: Community Lists	38
Chapter 2	Configuring NAT	61
	Figure 6: NAT Example	77
	Figure 7: Bidirectional NAT Example	79
	Figure 8: Twice NAT Example	80
	Figure 9: Cross-VRF Example	82
	Figure 10: PPTP Tunnels on an Inside Network	83
	Figure 11: PPTP Tunnels on an Outside Network	84
Chapter 5	Configuring IPSec	119
	Figure 12: IPSec Tunneling Stack	123
	Figure 13: IPSec Tunneling Packet Encapsulation	124
	Figure 14: IPSec Security Parameters in Relation to the Secure IP Interface	125
	Figure 15: Customer A's Corporate Frame Relay Network	153
	Figure 16: ISP-X Uses ERX Routers to Connect Corporate Offices over the Internet	154
	Figure 17: Connecting Customers Who Use Similar Address Schemes	156
Chapter 7	Configuring ANCP	185
	Figure 18: Using ANCP with an Access Node	195
Chapter 9	Configuring IP Tunnels	237
	Figure 19: IP Tunneling	237
	Figure 20: Transport and Tunnel Networks Using Different Routing Protocols	244
Chapter 11	IP Reassembly for Tunnels	269
	Figure 21: Tunneling Through an IP Network That Fragments Packets	270
Chapter 12	Securing L2TP and IP Tunnels with IPSec	277
	Figure 22: L2TP with IPSec Application	280
	Figure 23: L2TP/IPSec Connection	280
	Figure 24: L2TP Control Frame Encapsulated by IPSec	281
	Figure 25: L2TP Data Frame Encapsulated by IPSec	281
	Figure 26: L2TP Control Frame with NAT-T UDP Encapsulation	283
	Figure 27: L2TP Data Frame with NAT-T UDP Encapsulation	284

Figure 28: IKE Packet with NAT-T UDP Encapsulation	284
Figure 29: GRE/IPSec Connection	290

List of Tables

	About the Documentation	xxi
	Table 1: Notice Icons	xxii
	Table 2: Text and Syntax Conventions	xxii
Part 1	Chapters	
Chapter 1	Configuring Routing Policy	3
	Table 3: Match and Set Policy Values	31
	Table 4: Action Based on Well-Known Community Membership	37
	Table 5: Supported Regular Expression Metacharacters	43
	Table 6: Sample Regular Expressions	45
Chapter 4	Configuring BFD	107
	Table 7: Determining BFD Versions	111
Chapter 5	Configuring IPSec	119
	Table 8: IPSec Terms and Abbreviations	119
	Table 9: Security Parameters Used on Secure IP Interfaces	124
	Table 10: Security Parameters per IPSec Policy Type	126
	Table 11: Supported Transforms	131
	Table 12: Supported Security Transform Combinations	131
	Table 13: Initiator Proposals and Policy Rules	136
Chapter 8	Configuring Digital Certificates	205
	Table 14: Digital Certificate Terms and Acronyms	205
	Table 15: Outcome of IKE Phase 1 Negotiations	211
	Table 16: File Extensions (Offline Configuration)	211
Chapter 12	Securing L2TP and IP Tunnels with IPSec	277
	Table 17: Configuration and Monitoring Tasks for NAT-T	285
	Table 18: Differences in Handling Timeout Periods for L2TP/IPSec Tunnels	286

About the Documentation

- E Series and JunosE Documentation and Release Notes on page xxi
- Audience on page xxi
- E Series and JunosE Text and Syntax Conventions on page xxi
- Obtaining Documentation on page xxiii
- Documentation Feedback on page xxiii
- Requesting Technical Support on page xxiii

E Series and JunosE Documentation and Release Notes

For a list of related JunosE documentation, see
<http://www.juniper.net/techpubs/software/index.html>.

If the information in the latest release notes differs from the information in the documentation, follow the *JunosE Release Notes*.

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at
<http://www.juniper.net/techpubs/>.

Audience

This guide is intended for experienced system and network specialists working with Juniper Networks E Series Broadband Services Routers in an Internet access environment.

E Series and JunosE Text and Syntax Conventions

Table 1 on page xxii defines notice icons used in this documentation.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.

Table 2 on page xxii defines text and syntax conventions that we use throughout the E Series and JunosE documentation.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents commands and keywords in text.	<ul style="list-style-type: none"> Issue the clock source command. Specify the keyword exp-msg.
Bold text like this	Represents text that the user must type.	host1(config)#traffic class low-loss1
Fixed-width text like this	Represents information as displayed on your terminal's screen.	host1#show ip ospf 2 Routing Process OSPF 2 with Router ID 5.5.0.250 Router is an Area Border Router (ABR)
<i>Italic text like this</i>	<ul style="list-style-type: none"> Emphasizes words. Identifies variables. Identifies chapter, appendix, and book names. 	<ul style="list-style-type: none"> There are two levels of access: <i>user</i> and <i>privileged</i>. <i>clusterId</i>, <i>ipAddress</i>. <i>Appendix A, System Specifications</i>
Plus sign (+) linking key names	Indicates that you must press two or more keys simultaneously.	Press Ctrl + b.
Syntax Conventions in the Command Reference Guide		
Plain text like this	Represents keywords.	terminal length
<i>Italic text like this</i>	Represents variables.	<i>mask</i> , <i>accessListName</i>

Table 2: Text and Syntax Conventions (*continued*)

Convention	Description	Examples
(pipe symbol)	Represents a choice to select one keyword or variable to the left or to the right of this symbol. (The keyword or variable can be either optional or required.)	diagnostic line
[] (brackets)	Represent optional keywords or variables.	[internal external]
[]* (brackets and asterisk)	Represent optional keywords or variables that can be entered more than once.	[level1 level2 l1]*
{ } (braces)	Represent required keywords or variables.	{ permit deny } { in out } { clusterId ipAddress }

Obtaining Documentation

To obtain the most current version of all Juniper Networks technical documentation, see the Technical Documentation page on the Juniper Networks Web site at <http://www.juniper.net/>.

To download complete sets of technical documentation to create your own documentation CD-ROMs or DVD-ROMs, see the Portable Libraries page at

<http://www.juniper.net/techpubs/resources/index.html>

Copies of the Management Information Bases (MIBs) for a particular software release are available for download in the software image bundle from the Juniper Networks Web site at <http://www.juniper.net/>.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation to better meet your needs. Send your comments to techpubs-comments@juniper.net, or fill out the documentation feedback form at <https://www.juniper.net/cgi-bin/docbugreport/>. If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract,

or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf> .
- Product warranties—For product warranty information, visit <http://www.juniper.net/support/warranty/> .
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: <http://www.juniper.net/customers/support/>
- Search for known bugs: <http://www2.juniper.net/kb/>
- Find product documentation: <http://www.juniper.net/techpubs/>
- Find solutions and answer questions using our Knowledge Base: <http://kb.juniper.net/>
- Download the latest versions of software and review release notes: <http://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://www.juniper.net/alerts/>
- Join and participate in the Juniper Networks Community Forum: <http://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <http://www.juniper.net/cm/>

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: <https://tools.juniper.net/SerialNumberEntitlementSearch/>

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at <http://www.juniper.net/cm/> .
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see <http://www.juniper.net/support/requesting-support.html> .

PART 1

Chapters

- [Configuring Routing Policy on page 3](#)
- [Configuring NAT on page 61](#)
- [Configuring J-Flow Statistics on page 91](#)
- [Configuring BFD on page 107](#)
- [Configuring IPsec on page 119](#)
- [Configuring Dynamic IPsec Subscribers on page 169](#)
- [Configuring ANCP on page 185](#)
- [Configuring Digital Certificates on page 205](#)
- [Configuring IP Tunnels on page 237](#)
- [Configuring Dynamic IP Tunnels on page 251](#)
- [IP Reassembly for Tunnels on page 269](#)
- [Securing L2TP and IP Tunnels with IPsec on page 277](#)
- [Configuring the Mobile IP Home Agent on page 305](#)

CHAPTER 1

Configuring Routing Policy

This chapter provides information about configuring routing policy for your E Series router. It describes routing policy configuration in general as it might be used with various routing protocols, such as Border Gateway Protocol (BGP), Intermediate System to Intermediate System (IS-IS), Open Shortest Path First (OSPF), and Routing Information Protocol (RIP).

This chapter contains the following sections:

- [Overview on page 3](#)
- [Platform Considerations on page 4](#)
- [References on page 4](#)
- [Route Maps on page 4](#)
- [Match Policy Lists on page 19](#)
- [Access Lists on page 20](#)
- [Using the Null Interface on page 32](#)
- [Prefix Lists on page 32](#)
- [Prefix Trees on page 35](#)
- [Community Lists on page 37](#)
- [Using Regular Expressions on page 42](#)
- [Managing the Routing Table on page 47](#)
- [Troubleshooting Routing Policy on page 47](#)
- [Monitoring Routing Policy on page 48](#)

Overview

Routing policy determines how the system handles the routes it receives from and sends to neighboring routers. In many cases, routing policy consists of the following:

- Filtering routes
- Accepting certain routes
- Accepting and modifying other routes

- Rejecting some routes
- Determining the routing protocol used to distribute the routes

You can think of routing policy as a way to control the flow of routes into and out of the router.

The decision about which routes to accept from and advertise to various neighbors has an important impact on the traffic that crosses a network. Routing policy is used to enforce business agreements between two or more Internet service providers (ISPs) concerning the amount and type of traffic that is allowed to pass between them.

You can use one or more of the following mechanisms to configure routing policy:

- [“Route Maps” on page 4](#)
- [“Match Policy Lists” on page 19](#)
- [“Access Lists” on page 20](#)
- [“Prefix Lists” on page 32](#)
- [“Prefix Trees” on page 35](#)
- [“Community Lists” on page 37](#)

Platform Considerations

Configuring routing policies is supported on all E Series routers.

For information about the modules supported on E Series routers:

- See the *ERX Module Guide* for modules supported on ERX7xx models, ERX14xx models, and the Juniper Networks ERX310 Broadband Services Router.
- See the *E120 and E320 Module Guide* for modules supported on the Juniper Networks E120 and E320 Broadband Services Routers.

References

For more information about the protocols discussed in this chapter, see their respective chapters in this guide and other guides within the JunosE documentation set, and to the References sections within those chapters.

Route Maps

You can use route maps to control and modify routing information and to define conditions for redistributing routes between routing domains. You can apply route maps to inbound, outbound, or redistribution routes. A route map consists of *match* clauses and *set* clauses.

Match clauses specify the attribute values that determine whether a route matches the route map. A route that has the same attribute values passes the match condition. Routes that pass all the match conditions match the route map. You issue **match** commands to define the match conditions for a route map. You can specify the match conditions in

any order. If you do not specify *any* match conditions in a route map, that route map matches *all* routes.

Set clauses define how the attributes are modified for matching routes. The set conditions apply only to routes that pass all the match conditions (or a route map with no match conditions). When a route passes all the match conditions, the router software applies all set conditions. You issue **set** commands to define the set conditions for a route map.

You assign a unique string called the map tag to identify each route map. You can have multiple *instances* of a route map, where each instance consists of a different group of clauses. Each instance is identified by a sequence number. When you apply a route map, the routing protocol evaluates routes against the instance of the route map with the lowest sequence number. If the routes pass all the match conditions specified in the lowest-numbered instance, and if all **set** commands are successfully applied, no other instance of the route map is considered. However, any routes that do not pass all the match conditions are evaluated against the next instance of the route map. For example, suppose you create two instances of route map boston5, one with sequence number 10 and one with sequence number 25. When you apply boston5, routes are evaluated first against instance 10; any that do not match are evaluated against instance 25.

When you apply a route map, you specify the **permit** or **deny** keyword:

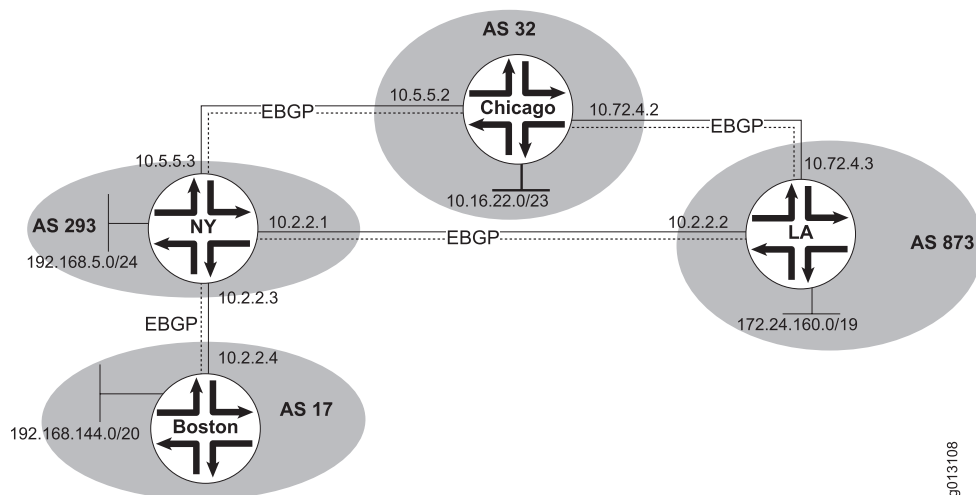
- If you specify the **permit** keyword, routes that match the route map are accepted, forwarded, or redistributed. Routes that do not match the route map are rejected or blocked.
- If you specify the **deny** keyword, routes that match the route map are rejected or blocked. Routes that do not match the route map are accepted, forwarded, or redistributed.

A route map must have at least one match clause or one set clause. If you have no match clauses, all routes match the route map, and the set conditions apply to all routes. If you have no set clauses, no action is taken other than that specified by the **permit** or **deny** keyword.

Route Map Configuration Example

Consider the network structure shown in [Figure 1 on page 6](#). Suppose you do not want router Boston to receive any routes that originate in or pass through router Chicago.

Figure 1: Applying Route Maps to Routes



You can use a route map to filter routes based on the autonomous system (AS) path to accomplish this goal. Use the following commands to configure router NY:

```
host1(config)#router bgp 293
host1(config-router)#network 192.168.5.0 mask 255.255.255.0
host1(config-router)#neighbor 10.5.5.2 remote-as 32
host1(config-router)#neighbor 10.2.2.2 remote-as 873
host1(config-router)#neighbor 10.2.2.4 remote-as 17
host1(config-router)#neighbor 10.2.2.4 route-map block1 out
host1(config-router)#exit
host1(config)#ip as-path access-list boston deny _32_
host1(config)#route-map block1 deny 1
host1(config-route-map)#match as-path boston
```

Multiple Values in a Match Entry

You can specify more than one value in each match entry of a route map by using any of the following **match** commands:

match as-path	match ipv6 next-hop
match community	match ipv6 route-source
match distance	match level
match extcommunity-list	match metric
match ip address	match policy-list
match ip next-hop	match route-type
match ipv6 address	match tag

A clause with multiple values matches a route that has any of the values; that is, the multiple values are logical ORed.

```
host1(config-route-map)#match ip address lisbon madrid
host1(config-route-map)#match as-path 10 20 30
```

You can also issue successive **match** commands to add new values to a route map entry for any of the commands listed above.

```
host1(config-route-map)#match ip address boston
host1(config-route-map)#match ip address newyork
```

This method is equivalent to issuing the following single command:

```
host1(config-route-map)#match ip address boston newyork
```

You cannot specify multiple values for the **match metric-type** command, because it has only two acceptable values, which are mutually exclusive. Specifying both values has the same effect as not specifying a metric type at all; specifying the same value more than once has no meaning.

Negating Match Clauses

If you specify a value when you negate a **match** command configured in a route map, only that value for the match entry is deleted. The routing software deletes the entire match entry only if the entry contains no other values. In some earlier releases, any value specified with a **no match** command was ignored, and the entire match entry was deleted. This change applies to all **match** commands configured in a route map.

For example, consider the following match entry to route map miami:

```
host1(config)#ip community-list corporate5 permit 32 463 21
host1(config)#ip community-list dade2 permit 41 53 22
host1(config)#route-map miami permit 1
host1(config-route-map)#match community corporate5 dade2
host1(config-route-map)#exit
host1(config)#exit
host1#show route-map
route-map miami, permit, sequence 10
Match clauses:
match community corporate5 dade2
```

In earlier releases, issuing a command like the following to remove a community (see [“Community Lists” on page 37](#)) not specified in the entry deleted the whole entry, but now nothing happens:

```
host1(config-route-map)#no match community southbeach
host1(config-route-map)#exit
host1(config)#exit
host1#show route-map
route-map miami, permit, sequence 10
Match clauses:
match community corporate5 dade2
```

If you instead issue the following commands, the specified value is deleted:

```
host1(config-route-map)#no match community dade2
```

```
host1(config-route-map)#exit
host1(config)#exit
host1#show route-map
route-map miami, permit, sequence 10
Match clauses:
match community corporate5
```

Issue either of the following commands to delete the entire match community entry:

```
host1(config-route-map)#no match community
host1(config-route-map)#no match community corporate5 dade2
```

Matching a Community List Exactly

You can use the **exact-match** keyword for the **match community** command to specify that a match exists only for the exact community numbers specified in the community list. The **exact-match** keyword applies only to a standard community list—that is, one not specified by a regular expression. You cannot use the **exact-match** keyword with a community list that is specified by a regular expression.

Consider the following example:

```
host1(config)#ip community-list 1 permit 100 200 300
host1(config)#exit
host1#show ip community-list
Community standard list 1
permit 0:100 0:200 0:300
host1(config)#route-map example1 permit 10
host1(config-route-map)#match community 1 exact-match
host1(config)#exit
host1#show route-map example1
route-map example, permit, sequence 10
Match clauses:
community (community-list filter): 1 exact-match
```

The route map *example1* permits a route only if the route contains community 100 and community 200 and community 300 and no additional communities.

If you do not specify the **exact-match** option, the route map also permits a match on a route that contains additional communities. For example, a route that contains communities 100, 200, 300, 400, and 450 matches.

Removing Community Lists from a Route Map

You can use the **set comm-list delete** command to remove the specified community list from routes matching the route map, provided that you created the community list with a single community number per list entry. For example, you cannot remove the community lists 231:10 and 231:20 with the **set comm-list delete** command if you created them with the following command:

```
host1(config)#ip community list 1 permit 231:10 231:20
```

You can, however, remove the lists with the **set comm-list delete** command if you created them separately with the following commands:

```
host1(config)#ip community list 1 permit 231:10
```



```
host1(config)#ip community list 1 permit 231:20
```

Matching a Policy List

You can use the **match policy-list** command to reference a policy list within the route map. Policy lists are like route maps, but they contain only match clauses and no set clauses. You can create a policy list to contain a group of match clauses once, referencing the list in any number of route maps and avoiding the task of having to reenter the match clauses separately into each route map.

For more information about creating IP policy lists, see [“Match Policy Lists” on page 19](#).

Redistributing Access Routes

Access-internal routes, such as DHCP and AAA/PPP host routes, are host routes to directly connected clients. Access routes, also known as AAA framed routes, are sourced by AAA.

The following example shows how you might redistribute access-internal routes and access routes by matching on a tag:

1. Configure route map tagtest to match tag 30.

```
host1(config)#route-map tagtest
host1(config-route-map)#match tag 30
```

2. Configure redistribution into BGP of the access-internal routes and access routes with route map tagtest.

```
host1(config)#router bgp 405
host1(config-router)#redistribute access route-map tagtest
host1(config-router)#redistribute access-internal route-map tagtest
```

Setting Multicast Bandwidths

You can use the **set admission-bandwidth** command to set a multicast bandwidth for admission control. Admission control is performed for the join and mapped interface when the OIF is added to the mroute.

You can use the **set qos-bandwidth** command to set a multicast bandwidth for QoS control. The QoS adjustment is made to the join interface when the OIF is added to the mroute.



NOTE: Both the admission bandwidth and QoS bandwidth are a constant bit rate.

For more information about multicast admission control or QoS adjustment, see *Configuring IPv4 Multicast* or *chapter Configuring IPv6 Multicast* in *JunosE Multicast Routing Configuration Guide*.

match as-path

- Use to match an AS-path access list.
- The implemented weight is based on the first matched AS path.
- Example

```
host1(config-route-map)#match as-path pathlist5
```

- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match as-path.

match community

- Use to match a community list.
- This command supports inbound and outbound route maps.
- Example

```
host1(config-route-map)#match community comm5
```

- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match community.

match distance

- Use to match any routes being redistributed out of the routing table that have the specified administrative distance.
- Distance is used to determine the relative preference between routes to the same prefix in order to pick the best route to that prefix in the routing table. Distance has no meaning in any other circumstance, and any attempt to match distance fails.
- Example

```
host1(config-route-map)#match distance 25
```

- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match distance.

match extcommunity

- Use to match an extended community list in a route map.
- You can specify one or more extended community list names in a match clause. If you specify more than one extended community list, the lists are logically ORed.
- Example

```
host1(config-route-map)#match extcommunity topeka10
```

- Use the **no** version to remove the match clause from a route map or a specified value from the match clause.
- See match extcommunity.

match ip address

- Use to match any route that has a destination network number that is permitted by an access list, a prefix list, or a prefix tree, or that performs policy routing on packets.
- Example

```
host1(config-route-map)#match ip address prefix-tree boston
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match ip address.

match ip next-hop

- Use to match any routes that have a next-hop router address passed by the specified access list, prefix list, or prefix tree.
- Example

```
host1(config-route-map)#match ip next-hop 5 acl_192_54_24_1
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match ip next-hop.

match ipv6 address

- Use to match any routes that have a destination network number address that is permitted by the specified prefix list.
- Example

```
host1(config-route-map)#match ipv6 address prefix-list boston
```
- Use the **no** version to delete all address match clauses from a route map unless you specify a prefix list, in which case only that prefix list match is removed from the route map.
- See match ipv6 address.

match ipv6 next-hop

- Use to match any routes that have a next-hop router address passed by the specified prefix list.
- Example

```
host1(config-route-map)#match ipv6 next-hop prefix-list next1
```
- Use the **no** version to delete all next-hop match clauses from a route map unless you specify a prefix list, in which case only that prefix list match is removed from the route map.
- See match ipv6 next-hop.

match ipv6 route-source

- Use to match any routes that are advertised from addresses contained in the specified prefix list.
- Example

```
host1(config-route-map)#match ipv6 route-source prefix-list source
```
- Use the **no** version to delete all route-source match clauses from a route map unless you specify a prefix list, in which case only that prefix list match is removed from the route map.
- See match ipv6 route-source.

match level

- Use to match routes for the specified level.
- Example

```
host1(config-route-map)#match level level-1
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match level.

match metric

- Use to match a route for the specified metric value.
- Example

```
host1(config-route-map)#match metric 10
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match metric.

match metric-type

- Use to match a route for the specified metric type.
- Example

```
host1(config-route-map)#match metric-type external
```
- Use the **no** version to delete the match clause from a route map.
- See match metric-type.

match policy-list

- Use to reference a policy list that has the specified name.
- Example

```
host1(config-route-map)#match policy-list list1
```
- Use the **no** version to remove the match clause from a route map.
- See match policy-list.

match route-type

- Use to match a route for the specified route type.
- Example

```
host1(config-route-map)#match route-type level-1
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match route-type.

match-set summary prefix-tree

- Use to specify the prefix tree that summarizes routes for a particular route map.
- Use the **ip prefix-tree** command to set the conditions of the prefix tree, including which routes to summarize and how many bits of the network address to preserve.
- Example

```
host1(config-route-map)#match-set summary prefix-tree boston
```
- Use the **no** version to disable the use of the prefix tree by the route map.
- See match-set summary prefix-tree.

match tag

- Use to match the tag value of the destination routing protocol.
- Example

```
host1(config)#route-map 1
host1(config-route-map)#match tag 25
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match tag.

route-map

- Use to define the conditions for redistributing routes from one routing protocol to another, and for filtering or modifying updates sent to or received from peers.
- Each **route-map** command has a list of **match** and **set** commands associated with it. That is, the route map itself consists of a set of clauses; each clause (also called an entry) consists of a **match** or **set** command:
 - **match** commands specify the match criteria, the conditions under which redistribution is allowed for the current route map.
 - **set** commands specify the set actions, the redistribution actions to perform if the criteria enforced by the **match** commands is met.
- You can specify match and set clauses to modify attributes of redistributed routes.
- Use route maps when you want to have detailed control over how routes are redistributed between routing processes.

- You specify the destination routing protocol with the **router** command.
- You specify the source routing protocol with the **redistribute** command.

- Example

```
host1(config)#route-map nyc1 permit 10
host1(config-route-map)#match ip address list1
host1(config-route-map)#set metric-type internal
```

- Use the **no** version to delete the route map.
- See route-map.

set as-path prepend

- Use to modify an AS path for BGP routes by prepending one or more AS numbers or a list of AS numbers to the path list.
- The only global BGP metric available to influence the best path selection is the AS path length. By varying the length of the AS path, a BGP device can influence the best path selection by a peer farther away.

- Example

```
host1(config-route-map)#set as-path prepend list list10
```

- Use the **no** version to delete the set clause from a route map.
- See set as-path prepend.

set automatic-tag

- Use to automatically compute the tag value of the destination routing protocol.

- Example

```
host1(config-route-map)#set automatic-tag
```

- Use the **no** version to delete the set clause from a route map.
- See set automatic-tag.

set comm-list delete

- Use to remove communities specified by the community list from the community attribute of routes that match the route map.
- You can use this command to delete communities only if the community list was created with a single community per list entry, as the following sample configuration for router host1 shows:

```
host1(config)#ip community-list 1 permit 231:10
host1(config)#ip community-list 1 permit 231:20
host1(config)#router bgp 45
host1(config-router)#neighbor 10.6.2.5 remote-as 5
host1(config-router)#neighbor 10.6.2.5 route-map indelete in
host1(config-router)#route-map indelete permit 10
host1(config-route-map)#set comm-list 1 delete
```

Router host1 receives the same route from 10.6.2.5 and applies the indelete route map. BGP compares each list entry with the community attribute. A match is found for the list entry 231:10, and this community is deleted from the community attribute. Similarly, a match is found for the list entry of 231:20, and this community is deleted from the community attribute.

- Example

```
host1(config-route-map)#set comm-list 1 delete
```

- Use the **no** version to delete the set clause from a route map.
- See set comm-list delete.

set community

- Use to set the community attribute in BGP updates.
- You can specify a community list number in the range 1–4294967295, or in the new community format of AA:NN, or one of the following well-known communities:
 - **local-asr**—Prevents advertisement outside the local AS
 - **no-advertise**—Prevents advertisement to any peer
 - **no-export**—Prevents advertisement beyond the BGP confederation boundary
- Alternatively, you can use the **list** keyword to specify the name of a community list that you previously created with the **ip community-list** command.
- Example

```
host1(config-route-map)#set community no-advertise
```

- Use the **none** keyword to remove the community attribute from a route.
- Use the **no** version to delete the set clause from a route map.
- See set community.

set dampening

- Use to enable BGP route flap dampening only on routes that pass the match clauses of, and are redistributed by, a particular route map.
- BGP creates a dampening parameter block for each unique set of dampening parameters—such as suppress threshold, reuse threshold, and so on—used by BGP. For example, if you have a route map that sets the dampening parameters to one set of values for some routes and to another set of values for the remaining routes, BGP uses and stores two dampening parameter blocks, one for each set.
- Example

```
host1(config-route-map)#set dampening 5 1000 1500 45 15
```

- Use the **no** version to delete the set clause from a route map.
- See set dampening.

set distance

- Use to set the administrative distance on routes being installed into the routing table that match the route map.
- Distance establishes preference between routes to the same prefix to identify the best route to that prefix. Setting distance in any other circumstance has no effect.
- Example

```
host1(config-route-map)#set distance 5
```
- Use the **no** version to delete the set clause from a route map.
- See set distance.

set extcommunity

- Use to set the extended community attributes in a route map for BGP updates.
- You can specify a site-of-origin (**soo**) extended community and a route target (**rt**) extended community at the same time in a set clause without overwriting the other.
- Example

```
host1(config-route-map)#set extcommunity rt 10.10.10.2:325
```
- Use the **no** version to delete the set clause from a route map.
- See set extcommunity.

set ip next-hop

- Use to set the next hop attribute of a route that matches a route map.
- You can specify an IP address or an interface as the next hop.
- Use the **peer-address** keyword to have the following effect:
 - On outbound route maps, disables the next-hop calculation by setting the next hop to the IP address of the BGP device
 - On inbound route maps, overrides any third-party next-hop configuration by setting the next hop to the IP address of the peer
- Example

```
host1(config-route-map)#set ip next-hop 192.56.32.1
```
- Use the **no** version to delete the set clause from a route map.
- See set ip next-hop.

set ipv6 next-hop

- Use to set the next hop attribute of a route that matches a route map.
- You can specify an IPv6 address or an interface as the next hop.
- Example

```
host1(config-route-map)#set ipv6 next-hop 1::1
```


- Use the **no** version to delete the set clause from a route map.
- See set ipv6 next-hop.

set level

- Use to specify where to import routes when all of a route map's match criteria are met.
- Example

```
host1(config-route-map)#set level level-2
```
- Use the **no** version to delete the set clause from a route map.
- See set level.

set local-preference

- Use to specify a preference value for the AS path.
- Example

```
host1(config-route-map)#set local-preference 200
```
- Use the **no** version to delete the set clause from a route map.
- See set local-preference.

set metric

- Use to set the metric value (for BGP, the MED) for a route.
- To establish an absolute metric, do not enter a plus or minus sign before the metric value.
- Example

```
host1(config-route-map)#set metric 10
```
- To establish a relative metric, specify a plus or minus sign immediately preceding the metric value. The value is added to or subtracted from the metric of any routes matching the route map. The relative metric value range is 0–4294967295.
- Example

```
host1(config-route-map)#set metric -25
```
- You cannot use both an absolute metric and a relative metric within the same route map sequence. Setting either metric overrides any previously configured value.
- Use the **no** version to delete the set clause from a route map.
- See set metric.

set metric-type

- Use to set the metric type for a route.
- For BGP, this command affects BGP behavior only in outbound route maps and has no effect on other types of route maps. If the route map contains both a **set metric-type** and a **set metric** clause, the **set metric** clause takes precedence. If you specify the **internal** metric type in a BGP outbound route map, BGP sets the MED of the advertised

routes to the IGP cost of the next hop of the advertised route. If the cost of the next hop changes, BGP is not forced to readvertise the route.

- For BGP, you can specify the following metrics:
 - **external**—Reverts to the normal BGP rules for propagating the MED; this is the BGP default
 - **internal**—Sets the MED of a received route that is being propagated to an external peer equal to the IGP cost of the indirect next hop
- For IS-IS, you can specify the following metrics:
 - **external**—Considers only the metric of the route itself is considered for comparison
 - **internal**—Considers both the metric of the route and the cost to the router that advertised the route are considered for comparison; this is the IS-IS default
- For OSPF, you can specify the following metrics:
 - **1**—Sets the cost of the external routes so that it is equal to the sum of all internal costs and the external cost
 - **2**—Sets the cost of the external routes so that it is equal to the external cost alone; this is the OSPF default
- Example
`host1(config-route-map)#set metric-type internal`
- Use the **no** version to delete the set clause from a route map.
- See set metric-type.

set origin

- Use to set the BGP origin of the advertised route.
- Example
`host1(config-route-map)#set origin egp`
- Use the **no** version to delete the set clause from a route map.
- See set origin.

set route-class

- Use to set the route class value. The route-class attribute enables you to associate a route class with incoming packets based on the destination or source address of the packet. For example, you can associate different route classes with different VPN services, while using the route classes to classify packets for quality of service (QoS).
- Example
`host1(config-route-map)#set route-class 50`
- Use the **no** version to delete the set clause from a route map.
- See set route-class.

set route-type

- Use to set the routes of the specified type (internal, internal-intra, internal-inter, or external).
- Example

```
host1(config-route-map)#set route-type internal
```
- Use the **no** version to delete the set clause from a route map.
- See set route-type.

set tag

- Use to set the tag value of the destination routing protocol.
- Example

```
host1(config-route-map)#set tag 23
```
- Use the **no** version to delete the set clause from a route map.
- See set tag.

set weight

- Use to specify the BGP weight for the routing table.
- The weights assigned with the **set weight** command in a route map override the weights assigned using the **neighbor weight** and **neighbor filter-list weight** commands.
- Example

```
host1(config-route-map)#set weight 200
```
- Use the **no** version to delete the set clause from a route map.
- See set weight.

Match Policy Lists

Match policy lists are very similar to route maps. However, unlike route maps, match policy lists can contain only match clauses and no set clauses.

You create match policy lists and then reference those lists within route maps. Because match policy lists share the same match clauses with route maps, they function the same way, and you can use the same match commands when you create your match policy lists.



NOTE: For descriptions of all route map match clauses, see [“Route Maps” on page 4](#).

As in route maps, the match clauses in match policy lists contain permit and deny statements. When you reference a match policy list within a route map, the route map

evaluates and processes each match clause and permits or denies routes based on the match policy list configuration.

When you configure match policy lists, keep the following in mind:

- A route map evaluates and processes all match statements within any match policy list that it references.
- You can configure multiple match policy lists within a route map, and you can evaluate each match policy list by using a logical AND or a logical OR.
- You can reference match policy lists within a route map that also uses separate match and set statements (that is, the statements are not part of the match policy list).
- All match policy lists within a route map match on the incoming attribute only.

ip match-policy-list

- Use to create an IP match policy list and launch the match policy list configuration mode.
- Example

```
host1(config)#ip match-policy-list
host1(config-match-policy-list)#
```
- Use the **no** version to delete the match policy list.
- See `ip match-policy-list`.

Access Lists

An access list is a sequential collection of permit and deny conditions that you can use to filter inbound or outbound routes. You can use different kinds of access lists to filter routes based on either the prefix or the AS path.

Filtering Prefixes

To filter routes based on the prefix, you can do any of the following:

- Define an access list with the **access-list** or **ipv6 access-list** command, and apply the list to routes received from or passed to a neighbor with the **neighbor distribute-list** command.
- Define a prefix list with the **ip prefix-list** command, and apply the list to routes received from or passed to a neighbor with the **neighbor prefix-list** command.
- Define a prefix tree with the **ip prefix-tree** command, and apply the list to routes received from or passed to a neighbor with the **neighbor prefix-tree** command.

The router compares each route's prefix against the conditions in the list or tree, one-by-one. If the first match is for a permit condition, the route is accepted or passed. If the first match is for a deny condition, the route is rejected or blocked. The order of conditions is critical because testing stops with the first match. If no conditions match, the router rejects or blocks the address; that is, the last action of any list is an implicit

deny condition for all routes. The implicit rule is displayed by **show access-list** and **show config** commands.

You cannot selectively place conditions in or remove conditions from an access list, prefix list, or prefix tree. You can insert a new condition only at the end of a list or tree.

Configuration Example 1

The following example shows how the implicit deny condition appears:

```
host1(config)#access-list 1 permit 10.10.10.1 0.0.0.255
host1(config)#access-list 2 permit 10.25.25.1 0.0.0.255
host1(config)#access-list 3 permit any any
host1(config)#show access-list
IP Access List 1:
permit ip 10.10.10.1 0.0.0.255 any
deny ip any any
IP Access List 2:
permit ip 10.25.25.1 0.0.0.255 any
deny ip any any
IP Access List 3:
permit ip any any
```

The implicit deny rule does not appear in the display for access list 3, because any prefix matches access list 3.

Configuration Example 2

The following example demonstrates how to use a route map and an access list to redistribute static routes to IS-IS.

1. Configure three static routes.

```
host1(config)#ip route 20.20.20.0 255.255.255.0 192.168.1.0
host1(config)#ip route 20.20.21.0 255.255.255.0 192.168.2.0
host1(config)#ip route 20.21.0.0 255.255.255.0 192.168.30.0
```

2. Configure an access list, fltra, that filters routes 20.20.20.0/24 and 20.20.21.0/24.

```
host1(config)#access-list fltra permit 20.20.0.0 0.0.255.255
```

3. Configure route map 1 to match access list fltra, and apply an internal metric type.

```
host1(config)#route-map 1
host1(config-route-map)#match ip address fltra
host1(config-route-map)#set metric-type internal
```

4. Configure redistribution into IS-IS of the static routes with route map 1.

```
host1(config)#router isis testnet
host1(config-router)#redistribute static route-map 1
```

5. Verify the effect of the redistribution (the two static routes matching the route map are redistributed as level 2 internal routes).

```
host1#show isis database detail l2
IS-IS Level-2 Link State Database
LSPID LSP Seq Num LSP Checksum LSP Holdtime ATT/P/OL
0000.0000.6666.00-00 0x000002B7 0x3E1F 1198 0/0/0
```

```

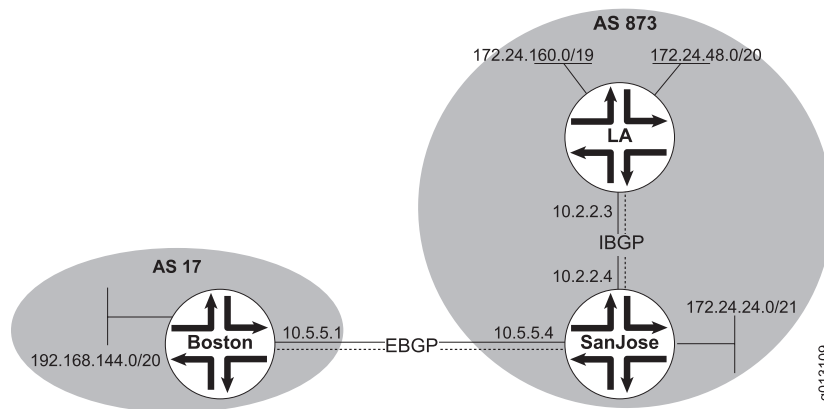
Area Address: 47.0005.80FF.F800.0000.0001.0001
NLPID:      0xcc
IP Address: 192.168.1.105
Metric: 10 IS 0000.0000.6666.01
Metric: 10 IS 0000.0000.3333.00
Metric: 10 IS 0000.0000.7777.00
Metric: 30 IP 20.20.20.0 255.255.255.0
Metric: 30 IP 20.20.21.0 255.255.255.0

```

Configuration Example 3

The following example demonstrates how to use an access list to filter routes advertised to a BGP device. Consider the network structure in [Figure 2 on page 22](#).

Figure 2: Filtering with Access Lists



The following commands configure router Boston to apply access list reject1 to routes inbound from router SanJose. Access list reject1 rejects routes matching 172.24.160.0/19.

```

host1(config)#router bgp 17
host1(config-router)#neighbor 10.5.5.4 remote-as 873
host1(config-router)#neighbor 10.5.5.4 distribute-list reject1 in
host1(config-router)#exit
host1(config)#access-list reject1 permit 172.24.48.0 0.0.255
host1(config)#access-list reject1 deny 172.24.160.0 0.0.0.255
host1(config)#access-list reject1 permit 172.24.24.0 0.0.0.255

```

Filtering AS Paths

You can use a filter list to filter incoming and outgoing routes based on the value of the AS-path attribute. Whenever a BGP route passes through an AS, BGP prepends its AS number to the AS-path attribute. The AS-path attribute is the list of ASs that a route has passed through to reach a destination.

To filter routes based on the AS path, define the access list with the **ip as-path access-list** command, and apply the list to routes received from or passed to a neighbor with the **neighbor filter-list** command. AS-path access lists use regular expressions to describe the AS path to be matched. A regular expression uses special characters—often referred to as metacharacters—to define a pattern that is compared with an input string. For a

full discussion of regular expressions, with examples of how to use them, see [“Using Regular Expressions” on page 42](#).

The router compares each route's AS path with each condition in the access list. If the first match is for a permit condition, the route is accepted or passed. If the first match is for a deny condition, the route is rejected or blocked. The order of conditions is critical because testing stops with the first match. If no conditions match, the router rejects or blocks the route; that is, the last action of any list is an implicit deny condition for all routes.

You cannot selectively place conditions in or remove conditions from an AS-path access list. You can insert a new condition only at the end of an AS-path access list.

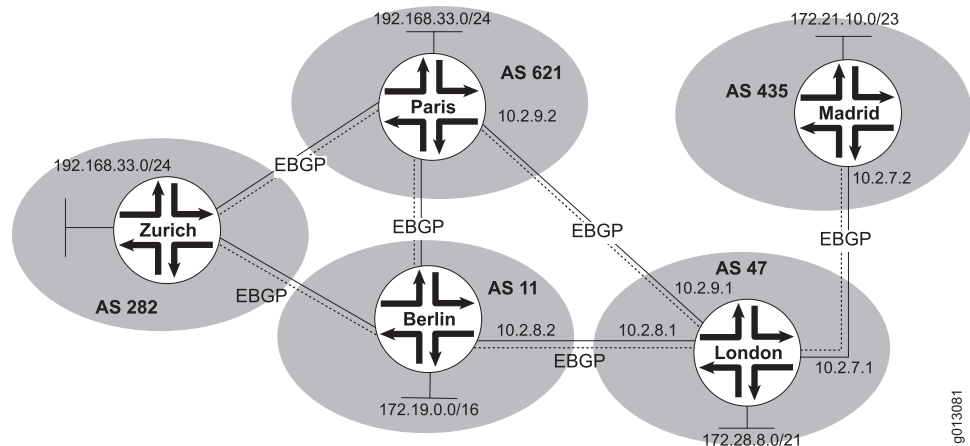
Configuration Example 1

Consider the network structure in [Figure 3 on page 23](#).

Suppose you want router London to behave in the following way:

- Accept routes originated in AS 621 only if they pass directly to router London.
- Accept routes originated in AS 11 only if they pass directly to router London.
- Forward routes from AS 282 to AS 435 only if they pass through either AS 621 or AS 11, but not both AS 621 and AS 11.

Figure 3: Filtering with AS-Path Access Lists



The following commands configure router London to apply filters based on AS path to routes received from router Berlin and router Paris and to routes forwarded to router Madrid.

```
host1(config)#router bgp 47
host1(config-router)#neighbor 10.2.9.2 remote-as 621
host1(config-router)#neighbor 10.2.9.2 filter-list 1 in
host1(config-router)#neighbor 10.2.8.2 remote-as 11
host1(config-router)#neighbor 10.2.8.2 filter-list 2 in
host1(config-router)#neighbor 10.2.7.2 remote-as 435
host1(config-router)#neighbor 10.2.7.2 filter-list 3 out
host1(config-router)#exit
```

```

host1(config)#ip as-path access-list 1 deny ^11
host1(config)#ip as-path access-list 1 permit .*
host1(config)#ip as-path access-list 2 deny ^621
host1(config)#ip as-path access-list 2 permit .*
host1(config)#ip as-path access-list 3 deny [621 11]
host1(config)#ip as-path access-list 3 permit .*

```

AS-path access list 1 is applied to routes that router London receives from router Paris. Router London rejects routes with the AS path 11 621 or 11 282 621.

AS-path access list 2 is applied to routes that router London receives from router Berlin. Router London rejects routes with the AS path 621 11 or 621 282 11.

Router London accepts routes with the AS path 282 11, 282 621, 282 621 11, or 282 11 621. However, it applies AS-path access list 3 to routes it forwards to router Madrid, and filters out routes with the AS path 282 621 11 or 282 11 621.

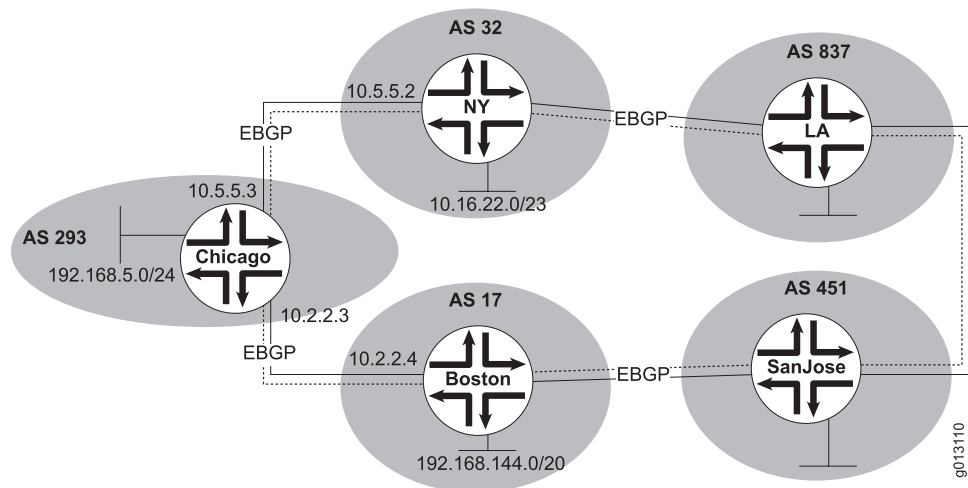
Using Access Lists in a Route Map

You can use a route map instead of the **neighbor filter-list** command to apply access lists for filtering routes.

Configuration Example 1

In [Figure 4 on page 24](#), a route map is used to determine the weight for routes learned by router Chicago.

Figure 4: Route Map Filtering



Access list 1 permits any route whose AS-path attribute includes 32 or 837. This condition permits routes that originate in (or pass through from elsewhere) AS 32 or AS 837. When these routes are advertised through AS 451 and AS 17 to router Chicago, instance 1 of route map 1 matches such routes and sets their weight to 25, overriding the neighbor weight set for updates received from 10.2.2.4.

Access list 2 permits any route whose AS-path attribute indicates that it originates in AS 74. When these routes are advertised through AS 837 and AS 32 to router Chicago,

instance 1 of route map 2 matches such routes and sets their weight to 175, overriding the neighbor weight set for updates received from 10.5.5.2.

The following example configures router Chicago:

```
host1(config)#router bgp 293
host1(config-router)#network 192.168.5.0 mask 255.255.255.0
host1(config-router)#neighbor 10.2.2.4 remote-as 17
host1(config-router)#neighbor 10.2.2.4 weight 150
host1(config-router)#neighbor 10.2.2.4 route-map 1 in
host1(config-router)#exit

host1(config-router)#neighbor 10.5.5.2 remote-as 32
host1(config-router)#neighbor 10.5.5.2 weight 50
host1(config-router)#neighbor 10.5.5.2 route-map 2 in

host1(config)#route-map 1 permit 1
host1(config-route-map)#match as-path 1
host1(config-route-map)#set weight 25
host1(config-route-map)#exit
host1(config)#ip as-path access-list 1 permit [ 32 837 ]

host1(config)#route-map 2 permit 1
host1(config-route-map)#match as-path 2
host1(config-route-map)#set weight 175
host1(config-route-map)#exit
host1(config)#ip as-path access-list 2 permit [ 74 ]
```

The result of this configuration is that router Chicago prefers routes learned through router Boston (weight 150) over routes learned through router NY (weight 50), except that:

- Router Chicago prefers routes learned via router NY that passed through AS 837 or AS 32 (weight 50) over the same routes learned via router Boston (weight 25 according to route map 1).
- Router Chicago prefers routes originating in AS 74 learned via router NY that passed through AS 837 and AS 32 (weight 175 according to route map 2) over the same routes learned via router Boston (weight 150).

access-list

- Use to define an IP access list to permit or deny routes based on the prefix.
- Each access list is a set of permit or deny conditions for routes based on matching a route's prefix.
- A zero in the wildcard mask means that the corresponding bit in the address must be exactly matched by the route. A one in the wildcard mask means that the corresponding bit in the address does not have to be matched by the route.
- Use the **neighbor distribute-list** command to apply the access list to routes received from or forwarded to a neighbor.

- Use the **log** keyword to log an Info event in the ipAccessList log whenever an access list rule is matched.
- Example

```
host1(config)#access-list bronze permit ip host any 228.0.0.0 0.0.0.255
```
- Use the **no** version to delete an IP access list (no other options specified), the specified entry in the access list, or the log for the specified access list or entry (by specifying the **log** keyword).
- See access-list.

default-information originate

- Use to enable RIP, OSPF, or BGP to advertise a default route (0.0.0.0/0) that exists in the IP routing table.
- If you specify the **always** option for OSPF, OSPF generates a default route, if it does not exist in the IP routing table and advertises it.
- Use to generate a default route to an IS-IS domain.
- Example

```
host1(config-router)#default-information originate
```
- Use the **no** version to disable advertisement of the default route.
- See default-information originate.

ip as-path access-list

- Use to define an AS-path access list to permit or deny routes based on the AS path.
- Each access list is a set of permit or deny conditions for routes based on matching a route's AS path to a regular expression. If the regular expression matches the representation of the AS path of the route as an ASCII string, the permit or deny condition applies. The AS path does not contain the local AS number.
- The AS path allows substring matching. For example, the regular expression *20* matches AS path = *20* and AS path = *100 200 300*, because *20* is a substring of each path. To disable substring matching and constrain matching to only the specified attribute string, place the underscore (*_*) metacharacter on both sides of the string; for example, *_20_*.
- Use the **neighbor filter-list** command to apply the AS-path access list. You can apply access-list filters to inbound and outbound BGP routes. You can assign weights to routes matching the AS-path access list.
- Example

```
host1(config)#ip as-path access-list 1 permit ^\()
```
- Use the **no** version to remove the AS-path access list; all entries that belong to this list are removed.
- See ip as-path access-list.

ipv6 access-list

- Use to define an IPv6 access list to permit or deny routes based on the prefix.
- Each access list is a set of permit or deny conditions for routes based on matching a route's prefix.
- Use the **neighbor distribute-list** command to apply the access list to routes received from or forwarded to a neighbor.
- Use the **log** keyword to log an Info event in the ipAccessList log whenever an access list rule is matched.
- Example


```
host1(config)#ipv6 access-list bronze deny 1::1/16 any
```
- Use the **no** version to delete an IPv6 access list (no other options specified), the specified entry in the access list, or the log for the specified access list or entry (by specifying the **log** keyword).
- See ipv6 access-list.

neighbor distribute-list

- Use to filter routes to selected prefixes as specified in an access list. Distribute lists are applied only to external peers.
- Use the **in** keyword to apply the list to inbound routes (inbound policy). Use the **out** keyword to apply the list to outbound routes (outbound policy).
- Besides using distribute lists to filter BGP advertisements, you can do the following:
 - Use AS-path filters with the **ip as-path access-list** and the **neighbor filter-list** commands
 - Use route map filters with the **route-map** and the **neighbor route-map** commands
- Example


```
host1:vr1(config-router)#neighbor group1 distribute-list list1 in
```
- Use the **no** version to disassociate the access list from a neighbor.
- See neighbor distribute-list.

neighbor filter-list

- Use to assign an AS-path access list to matching inbound or outbound routes.
- Use the **in** keyword to apply the list to inbound routes (inbound policy). Use the **out** keyword to apply the list to outbound routes (outbound policy).
- You can specify an optional weight value with the **weight** keyword to assign a relative importance to incoming routes that match the AS-path access list.
- Access list values can be in the range 0–65535.
- Example


```
host1:vr1(config-router)#neighbor group2 filter-list list2 out
```

- Use the **no** version to disassociate the access list from a neighbor.
- See neighbor filter-list.

neighbor prefix-list

- Use to assign an inbound or outbound prefix list.
- If you specify a BGP peer group by using the *peer-group-name* argument, all the members of the peer group inherit the characteristic configured with this command unless it is overridden for a specific peer.
- Use the **in** keyword to assign the prefix list to incoming routes (inbound policy)
- Use the **out** keyword to assign the prefix list to outgoing routes (outbound policy); you cannot configure a member of a peer group to override the inherited peer group characteristic for outbound policy
- Example

```
host1(config-router)#neighbor 192.168.1.158 prefix-list seoul19 in
```
- Use the **no** version to remove the prefix list.
- See neighbor prefix-list.

neighbor prefix-tree

- Use to assign an inbound or outbound prefix tree.
- If you specify a BGP peer group by using the *peer-group-name* argument, all the members of the peer group inherit the characteristic configured with this command unless it is overridden for a specific peer.
- Use the **in** keyword to assign the prefix tree to incoming routes (inbound policy)
- Use the **out** keyword to assign the prefix tree to outgoing routes (outbound policy); you cannot configure a member of a peer group to override the inherited peer group characteristic for outbound policy
- Example

```
host1(config-router)#neighbor 192.168.1.158 prefix-tree newyork out
```
- Use the **no** version to remove the prefix tree.
- See neighbor prefix-tree.

redistribute

- Use to redistribute routes from one routing domain to another routing domain.
- Example

```
host1(config)#router bgp 100
host1(config-router)#neighbor 192.56.10.2 remote-as 200
host1(config-router)#redistribute static
host1(config-router)#exit
host1(config)#ip route 155.30.0.0 0.0.255.255
```

- Use the **no** version to end redistribution of information.
- See redistribute.

Using Access Lists for PIM Join Filters

You can apply access lists to PIM sparse mode interfaces along with the **ip pim join-filter** or **ipv6 pim join-filter** command to use the access lists as PIM sparse mode join filters.

To configure PIM join filters:

1. Create the various access list services you want to use with the PIM join filter command.

```
host1(config)#! create bronze service
host1(config)#! - restrict SSM channels to 232.0.1/24 only
host1(config)#access-list bronze permit ip host any 228.0.0.0 0.0.0.255
host1(config)#access-list bronze permit ip host 1.1.1.1 232.0.1.0 0.0.0.255
host1(config)#access-list bronze permit ip host 2.2.2.2 232.0.1.0 0.0.0.255
host1(config)#
host1(config)#! create silver service
host1(config)#! - bronze service + new channels 232.0.2/24
host1(config)#access-list silver permit ip host any 228.0.0.0 0.0.0.255
host1(config)#access-list silver permit ip host 1.1.1.1 232.0.1.0 0.0.0.255
host1(config)#access-list silver permit ip host 2.2.2.2 232.0.1.0 0.0.0.255
host1(config)#access-list silver permit ip host 1.1.1.1 232.0.2.0 0.0.0.255
host1(config)#access-list silver permit ip host 2.2.2.2 232.0.2.0 0.0.0.255
host1(config)#
host1(config)#! create gold service
host1(config)#! - silver service + new channels 232.0.3/24
host1(config)#access-list gold permit ip host any 228.0.0.0 0.0.0.255
host1(config)#access-list gold permit ip host 1.1.1.1 232.0.1.0 0.0.0.255
host1(config)#access-list gold permit ip host 2.2.2.2 232.0.1.0 0.0.0.255
host1(config)#access-list gold permit ip host 1.1.1.1 232.0.2.0 0.0.0.255
host1(config)#access-list gold permit ip host 2.2.2.2 232.0.2.0 0.0.0.255
host1(config)#access-list gold permit ip host 1.1.1.1 232.0.3.0 0.0.0.255
host1(config)#access-list gold permit ip host 2.2.2.2 232.0.3.0 0.0.0.255
```

For additional information about how to create access lists, see [“Access Lists” on page 20](#).

2. Enable IP multicast routing.

```
host1(config)#ip multicast-routing
```

3. Enable PIM source-specific multicast router.

```
host1(config)#ip pim ssm
```

4. Identify the default PIM join filter.

```
host1(config)#ip pim join-filter bronze
```

5. Enable PIM sparse mode on a subinterface.

```
host1(config)#interface atm 3/0.101
host1(config-if)#ip address 101.0.0.1 255.255.255.255
host1(config-if)#ip pim sparse-mode
```

This interface (and any other PIM interface to which you do not specifically assign an access list filter) uses the default (bronze) join filter.

6. Enable PIM sparse mode on another subinterface and assign the silver join filter.

```
host1(config-if)#interface atm 3/0.102
host1(config-if)#ip address 102.0.0.1 255.255.255.255
host1(config-if)#ip pim sparse-mode
host1(config-if)#ip pim join-filter silver
```

7. Enable PIM sparse mode on another subinterface and assign the gold join filter.

```
host1(config-if)#interface atm 3/0.103
host1(config-if)#ip address 103.0.0.1 255.255.255.255
host1(config-if)#ip pim sparse-mode
host1(config-if)#ip pim join-filter gold
```

For information about the **ip pim join-filter** command, see *Configuring PIM for IPv4 Multicast* in *JunosE Multicast Routing Configuration Guide*. For information about the **ipv6 pim join-filter** command, see *Configuring PIM for IPv6 Multicast* in *JunosE Multicast Routing Configuration Guide*.

Clearing Access List Counters

Use the **clear access-list** or **clear ipv6 access-list** commands to clear access list counters.

clear access-list

clear ipv6 access-list

- Use to clear all access list counters or access list counters in the specified access list.
- Example 1

```
host1#clear access-list list1
```

- Example 2

```
host1#clear ipv6 access-list list2
```

- There is no **no** version.
- See `clear access-list`.
- See `clear ipv6 access-list`.

Creating Table Maps

For static routes and access routes, you can configure and apply a table map that filters routes before an access list adds them to the routing table. For static routes, you can use the **ip static-route table-map** or **ipv6 static-route table-map** command. For access routes, you can use the **ip access-route table-map** or **ipv6 access-route table-map** command.

Use these commands when triggering on the policy values listed in [Table 3 on page 31](#).

Table 3: Match and Set Policy Values

Match	Set
ip address	metric
metric	distance
distance	tag
tag	

For example, you can configure an access list and route map to filter, based on IP address, any routes that appear in the routing table:

```
host1(config)#ip access-route table-map just10net
host1(config)#access-list permit10 permit 10.0.0.0 0.255.255.255
host1(config)#access-list permit10 deny any
host1(config)#route-map just10net
host1(config-route-map)#match ip address permit10
```

Using the same name for both the table map and the route map creates an association specifying (in this case) that only IP addresses that match the access list criterion appear in the routing table.

ip access-route table-map

ipv6 access-route table-map

- Use to filter access routes before an access list adds them to the routing table.
- Example 1


```
host1(config)#ip access-route table-map just10net
```
- Example 2


```
host1(config)#ipv6 access-route table-map map2
```
- Use the **no** version to delete the table map.
- See *ip access-route table-map*.
- See *ipv6 access-route table-map*.

ip static-route table-map

ipv6 static-route table-map

- Use to filter static routes before adding them to the routing table.
- Example 1


```
host1(config)#ip static-route table-map map3
```
- Example 2


```
host1(config)#ipv6 static-route table-map map4
```
- Use the **no** version to delete the table map.

- See `ip static-route table-map`.
- See `ipv6 static-route table-map`.

Using the Null Interface

You can use access control lists to filter undesired traffic. Another way to handle undesired traffic is to send it to the null interface. The router automatically creates the null interface, which is always up, cannot be deleted, and acts as a data sink. In other words, the null interface cannot forward or receive traffic. However, the CLI does allow you to access the null interface.

The E Series router creates the null interface by default; you do not have to manually create it. You can direct traffic to the null interface by specifying the **null 0** keywords instead of a next-hop or destination address when you configure routes.

interface null

- Use to access the null interface.
- The null interface is a data sink; it does not accept or forward traffic.
- Although you can access the null interface, you cannot configure any values for it or delete it.
- Example

```
host1(config)#interface null 0
host1(config-if)#
```

- There is no **no** version.
- See `interface null`.

ip route

- Use to configure a static route and redirect traffic from it to the null interface.
 - Example
- ```
host1(config-if)#ip route 10.10.20.5 null 0
```
- Use the **no** version to remove the static route.
  - See `ip route`.

## Prefix Lists

---

A prefix list is a sequential collection of permit and deny conditions that apply to IP or IPv6 addresses. Like an access list, the router tests addresses one by one against the conditions in a prefix list. The first match determines whether the router accepts or rejects the address. Because the router stops testing conditions after the first match, the order of the conditions is critical. If no conditions match, the router rejects the address. An empty prefix list results in an automatic permit of the tested address.



Unlike access lists, the prefix list specifies a base IP or IPv6 address and a length (the number of bits applied to the base to determine the network prefix). The tested address is matched against the prefix.

Use the **ip prefix-list** command to define an IP prefix list, or the **ipv6 prefix-list** command to define an IPv6 prefix list. The **prefix-list** keyword with either the **match { ip | ipv6 } address** or **match { ip | ipv6 } next-hop** commands enables you to add a clause to a route map.

## Using a Prefix List

The following example creates a prefix list that permits routes with a prefix length up to 24 in the 151.0.0.0/8 network:

```
host1(config)#ip prefix-list abc permit 151.0.0.0/8 le 24
```

### *clear ip prefix-list*

- Use to clear all hit counts in the prefix lists or the specified entry from the specified prefix list. (The router increments the hit count by 1 each time an entry matches.)

- Example

```
host1#clear ip prefix-list abc
```

- There is no **no** version.
- See clear ip prefix-list.

### *clear ipv6 prefix-list*

- Use to clear all hit counts in the IPv6 prefix lists or the specified entry from the specified prefix list. (The router increments the hit count by 1 each time an entry matches.)

- Example

```
host1#clear ipv6 prefix-list abc
```

- There is no **no** version.
- See clear ipv6 prefix-list.

### *ip prefix-list*

#### *ipv6 prefix-list*

- Use to create a prefix list for route filtering and to specify a list entry—a deny or permit clause for a network address—to the prefix list. Use to add entries to prefix lists.
- The prefix list name can be up to 32 characters long.
- Specify the position of each entry in the list with the **seq** (sequence) keyword. If you do not specify a sequence number, the router uses the value of the last sequence number plus 5.
- Use the **ge** and **le** keywords to specify a range of network prefixes. These keywords have the following values:
  - prefix length < **ge** <= 32

- prefix length < **le** <= **ge**
- If you do not specify either the **ge** or **le** keyword, an exact match is expected.
- Example 1—IPv4; exact match required; the router permits only a route with a prefix length of 8 and a network address of 151.0.0.0.  

```
host1(config)#ip prefix-list abc permit 151.0.0.0/8
```
- Example 2—IPv6; exact match required; the router permits only a route with a prefix length of 8 and a network address of 1:0:0:0:0:0:5.  

```
host1(config)#ipv6 prefix-list abc permit 1::5/8
```
- Use the **no** version to remove the specified prefix list or the specified list entry.
- See ip prefix-list.
- See ipv6 prefix-list.

#### *match ip address*

- Use to specify an access list or use with the **prefix-list** or **prefix-tree** keyword to match routes that have a destination network number address that is permitted by any specified access list, prefix list, or prefix tree.
- Example  

```
host1(config-route-map)#match ip address prefix-list abc
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match ip address.

#### *match ipv6 address*

- Use to match any route that has a destination network number address that is permitted by the specified access list or prefix list.
- Example  

```
host1(config-route-map)#match ipv6 address prefix-list boston
```
- Use the **no** version to delete all address match clauses from a route map unless you specify an access list or prefix list, in which case only the list match is removed from the route map.
- See match ipv6 address.

#### *match ip next-hop*

- Use with the **prefix-list** keyword to match routes that have a next-hop router address passed by the specified access lists, prefix lists, or prefix trees.
- Example  

```
host1(config-route-map)#match ip next-hop prefix-list abc
```

- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match ip next-hop.

### *match ipv6 next-hop*

- Use to match any routes that have a next-hop router address passed by the specified access list or prefix list.
- Example
 

```
host1(config-route-map)#match ipv6 next-hop prefix-list next1
```
- Use the **no** version to delete all next-hop match clauses from a route map unless you specify an access list or prefix list, in which case the router removes only the list match from the route map.
- See match ipv6 next-hop.

## Prefix Trees

A prefix tree is a nonsequential collection of permit and deny conditions that apply to IP addresses. Like a prefix list, the prefix tree specifies a base IP address and a length, the number of bits applied to the base to determine the network prefix. The tested address is matched against the prefix. The prefix tree also enables route summarization.

However, the prefix tree does not match addresses one by one in sequence against the listed conditions. The router performs a binary search against the tree structure of the entries. If the tested address is less than a particular entry, it branches one way to another test pair; if it is greater than the entry, it branches the other way to another mutually exclusive test pair. The router stops testing conditions when it finds the best match. If no conditions match, the router rejects the address. An empty prefix tree results in an automatic permit of the tested address.

The prefix tree provides a faster search method and matches the test address more closely than either the access list or the prefix list.

Use the **ip prefix-tree** command to define an IP prefix tree. Use the **prefix-tree** keyword with the **match ip address** or **match ip next-hop** commands to add a clause to a route map. Use the **match-set summary prefix-tree** command to specify the prefix tree that summarizes routes for a particular route map.

### Using a Prefix Tree

The following example creates a prefix tree that permits routes with a prefix length of 24 or larger in the 10.10.2.0/24 network:

```
host1(config)#ip prefix-tree xyz permit 10.10.2.0/24
```

*clear ip prefix-tree*

- Use to clear all hit counts in the prefix trees or the specified entry from the specified prefix tree. (The router increments the hit count by 1 each time an entry matches.)
- Example

```
host1#clear ip prefix-tree xyz
```
- There is no **no** version.
- See clear ip prefix-tree.

### *ip prefix-tree*

- Use to create a prefix tree for best route filtering; specifies a tree entry—a deny or permit clause for a network address.
- The prefix tree name can be up to 32 characters long.
- Example

```
host1(config)#ip prefix-tree boston42 permit 10.10.2.0/24
```
- Use the **no** version to remove the specified prefix tree or the specified tree entry.
- See ip prefix-tree.

### *match ip address*

- Use with the **prefix-tree** keyword to match routes that have a destination network number address that is permitted by the prefix tree.
- Example

```
host1(config-route-map)#match ip address prefix-tree xyz
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match ip address.

### *match ip next-hop*

- Use with the **prefix-tree** keyword to match routes that have a next-hop router address passed by the specified prefix tree.
- Example

```
host1(config-route-map)#match ip next-hop prefix-tree xyz
```
- Use the **no** version to delete the match clause from a route map or a specified value from the match clause.
- See match ip next-hop.

### *match-set summary prefix-tree*

- Use to specify the prefix tree that summarizes routes for a particular route map.
- Use the **ip prefix-tree** command to set the conditions of the prefix tree, including which routes to summarize and how many bits of the network address to preserve.
- Example

```
host1(config-route-map)#match-set summary prefix-tree dog3
```

- Use the **no** version to disable use of the prefix tree by the route map.
- See match-set summary prefix-tree.

## Community Lists

A community is a logical group of prefixes that share some common attribute. Community members can reside on different networks and in different autonomous systems. BGP enables you to define the community to which a prefix belongs. A prefix can belong to more than one community. The community attribute lists the communities to which a prefix belongs.

You can use communities to simplify routing policies by configuring the routing information that a BGP device can accept, prefer, or distribute to other neighbors according to community membership. When a route is learned, advertised, or redistributed, a BGP device can set, append, or modify the community of a route. When routes are aggregated, the resulting BGP update contains a community attribute that contains all communities from all of the aggregated routes (if the aggregate is an AS-set aggregate).

Several well-known communities are predefined. [Table 4 on page 37](#) describes how a BGP device handles a route based on the setting of its community attribute.

**Table 4: Action Based on Well-Known Community Membership**

| Well-Known Community                         | BGP Device Action                                                                                               |
|----------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| no-export                                    | Does not advertise the route beyond the BGP confederation boundary                                              |
| no-advertise                                 | Does not advertise the route to any peers, IBGP, or EBGP                                                        |
| local-as (also known as no-export-subconfed) | Does not advertise the route to any external peers                                                              |
| internet                                     | Advertises this route to the Internet community; by default, all prefixes are members of the Internet community |

In addition to the well-known communities, you can define local-use communities, also known as private communities or general communities. These communities serve as a convenient way to categorize groups of routes to facilitate the use of routing policies. The community attribute consists of four octets, but it is common practice to designate communities in the *AA:NN* format. The autonomous system number (*AA*) comprises the higher two octets, and the community number (*NN*) comprises the lower two octets. Both are expressed as decimal numbers. For example, if a prefix in AS 23 belongs to community 411, the attribute could be expressed as 23:411. Use the **ip bgp-community new-format** command to specify that the **show** commands display communities in this format. You can also use a regular expression to specify the community attribute.

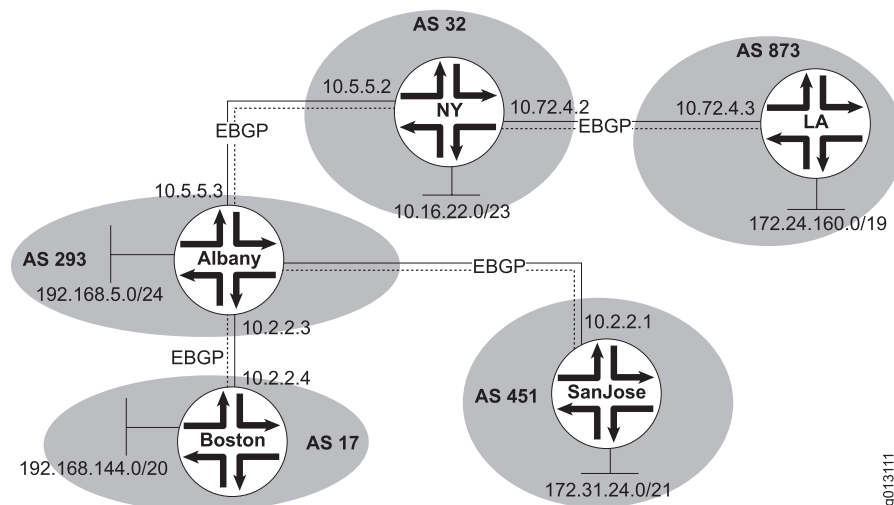
Use the **set community** command in route maps to configure the community attributes. You can add one or more communities to the attribute, or you can use the **list** keyword to add a list of communities to the attribute. By default, the community attribute is not sent to BGP peers. To send the community attribute to a neighbor, use the **neighbor send community** command.

A community list is a sequential collection of permit and deny conditions. Each condition describes the community number to be matched. If you issued the **ip bgp-community new-format** command, the community number is in *AA:NN* format; otherwise, it is in decimal format (the hexadecimal octets converted to decimal).

The router tests the community attribute of a route against each condition in a community list. The first match determines whether the router accepts (the route is permitted) or rejects (the route is denied) a route that has the specified community. Because the router stops testing conditions after the first match, the order of the conditions is critical. If no conditions match, the router rejects the route.

Consider the network structure shown in [Figure 5 on page 38](#).

**Figure 5: Community Lists**



Suppose you want router Albany to set metrics for routes that it forwards to router Boston based on the communities to which the routes belong. You can create community lists and filter the routes with a route map that matches on the community list. The following example configures router Albany:

```
host1(config)#router bgp 293
host1(config-router)#neighbor 10.5.5.2 remote-as 32
host1(config-router)#neighbor 10.2.2.1 remote-as 451
host1(config-router)#neighbor 10.2.2.4 remote-as 17
host1(config-router)#neighbor 10.2.2.4 route-map commtrc out
host1(config-router)#exit
host1(config)#route-map commtrc permit 1
host1(config-route-map)#match community 1
host1(config-route-map)#set metric 20
host1(config-route-map)#exit
```

```

host1(config)#route-map commtrc permit 2
host1(config-route-map)#match community 2
host1(config-route-map)#set metric 75
host1(config-route-map)#exit
host1(config)#route-map commtrc permit 3
host1(config-route-map)#match community 3
host1(config-route-map)#set metric 85
host1(config-route-map)#exit
host1(config)#ip community-list 1 permit 25
host1(config)#ip community-list 2 permit 62
host1(config)#ip community-list 3 permit internet

```

Community list 1 comprises routes with a community of 25; their metric is set to 20. Community list 2 comprises routes with a community of 62; their metric is set to 75. Community 3 catches all remaining routes by matching the Internet community; their metric is set to 85.

### *ip bgp-community new-format*

- Use to specify that communities must be displayed in *AA:NN* format, where *AA* is a number that identifies the autonomous system and *NN* is a number that identifies the community within the autonomous system.
- Example
 

```
host1(config)#ip bgp-community new-format
```
- Use the **no** version to restore the default display.
- See `ip bgp-community new-format`.

### *ip community-list*

- Use to create a community list for BGP and control access to it.
- The list name can be up to 32 characters long.
- A route can belong to any number of communities, so a community list can have many entries comprising many communities.
- You can specify one or more community values when you create a community list. A clause in a route map that includes a list that has more than one value matches only a route that has all of the values; that is, the multiple values are logical ANDed.
- You can specify community values with a number or a regular expression.
- Example

```

host1(config)#ip community-list 1 permit 100:2 100:3 100:4
host1(config)#route-map marengo permit 10
host1(config-route-map)#match community 1

```

A route matches this community list only if it belongs to at least all three communities in community list 1: communities 100:2, 100:3, and 100:4.

- Use the **no** version to remove the specified community list, including all list entries.
- See `ip community-list`.

### *neighbor send-community*

- Use to specify that a community attribute be sent to a BGP neighbor.
- If you specify a BGP peer group by using the *peer-group-name* argument, all the members of the peer group inherit the characteristic configured with this command.
- Example

```
host1:vr1(config-router)#neighbor 192.3.4.5 send-community standard
```
- Use the **no** version to specify that common attributes not be sent to a BGP neighbor.
- See neighbor send-community.

### ***set community***

- Use to set the community attribute in BGP updates.
- You can specify a community list number in the range 1–4294967295, or in the new community format of *AA:NN*, or you can specify one of the following well-known communities:
  - **local-as**—Prevents advertisement outside the local AS
  - **no-advertise**—Prevents advertisement to any peer
  - **no-export**—Prevents advertisement beyond the BGP confederation boundary
- Alternatively, you can use the **list** keyword to specify the name of a community list that you previously created with the **ip community-list** command.
- You can use this command with inbound, outbound, and redistribution route maps.
- Use the **none** keyword to remove the community attribute from a route.
- Example

```
host1(config)#route-map 1
host1(config-route-map)#set community no-advertise
```
- Use the **no** version to remove the set clause from a route map.
- See set community.

## Extended Community Lists

The router supports the BGP extended community attribute defined in Internet draft BGP Extended Communities Attribute— draft-ietf-idr-bgp-ext-communities-07.txt (February 2004 expiration). This attribute enables the definition of a type of IP extended community and extended community list unrelated to the community list that uses regular expressions.



**NOTE:** IETF drafts are valid for only six months from the date of issuance. They must be considered as works in progress. For the latest drafts, please see the IETF Web site at <http://www.ietf.org>.



BGP devices can use the extended community attribute to control routes much like they use the community attribute to determine routes that they accept, reject, or redistribute. A BGP device can append the extended community attribute to a route that does not have the attribute before it advertises the route. For routes that do have the attribute, BGP can modify the attribute.

### *ip extcommunity-list*

- Use to create an extended community list for BGP and control access to it.
- A route can belong to any number of communities, so an extended community list can have many entries comprising many communities.
- You can specify one or more community values when you create an extended community list. A clause in a route map that includes a list that has more than one value matches only a route that has all of the values; that is, the multiple values are logical ANDed.
- Use the **rt** keyword to specify a route target community, which consists of one or more routers that can receive a set of routes advertised by BGP that carry the extended community attribute.
- Use the **soo** keyword to specify a site-of-origin community, which consists of one or more routers that inject into BGP a set of routes that carry the extended community attribute.
- Example

```
host1(config)#ip extcommunity-list boston1 permit rt 100:2 rt 100:3 rt 100:4
host1(config)#route-map marengo permit 10
host1(config-route-map)#match extcommunity boston1
```

A route matches this community list only if it belongs to at least all three communities in extended community list boston1: communities 100:2, 100:3, and 100:4.

- Use the **no** version to remove a single extended community list entry if you specify the **permit** or **deny** keyword and a path expression. Otherwise, the router removes the entire community list.
- See ip extcommunity-list.

### *match extcommunity*

- Use to match an extended community list in a route map.
  - You can specify one or more extended community list names in a match clause. If you specify more than one extended community list, the lists are logically ORed.
  - Example
- ```
host1(config-route-map)#match extcommunity topeka10
```
- Use the **no** version to remove the match clause from a route map or a specified value from the match clause.
 - See match extcommunity.

set extcommunity

- Use to set the extended community attributes in a route map for BGP updates.
- Use the **rt** keyword to specify a route target community, which consists of one or more routers that can receive a set of routes advertised by BGP that carry the extended community attribute.
- Use the **soo** keyword to specify a site-of-origin community, which consists of one or more routers that inject into BGP a set of routes that carry the extended community attribute.
- You can specify both a route target community and a site-of-origin community at the same time in a set clause without them overwriting each other.
- Example

```
host1(config)#route-map 1
host1(config-route-map)#set extcommunity rt 10.10.10.2:325
```
- Use the **no** version to remove the set clause from the route map.
- See set extcommunity.

show ip extcommunity-list

- Use to display information about a specific extended community list or all extended community lists.
- Example

```
host1#show ip extcommunity-list
IP Extended Community List dresden1:
    permit soo 10.10.10.10:15
IP Extended Community List bonn:
    deny rt 12:12
```
- See show ip extcommunity-list.

Using Regular Expressions

You can use regular expressions when you define AS-path access lists and community lists to more easily filter routes. A regular expression uses special characters—often referred to as metacharacters—to define a pattern that is compared with an input string.

AS-path Lists

For an AS-path access list, the input string is the AS path of the routes to which the list is applied with the **route-map** or **neighbor filter-list** commands. If the AS path matches the regular expression in the access list, the route matches the access list.

Example The following commands apply access list 1 to routes inbound from BGP peer 10.5.5.2. Access list 1 uses a regular expression to deny routes that originate in autonomous system 32.

```
host1(config-router)#neighbor 10.5.5.2 remote-as 32
host1(config-router)#neighbor 10.5.5.2 filter-list 1 in
host1(config-router)#exit
host1(config)#ip as-path access-list 1 deny 32$
```

Community Lists

For a community list, the input string is the community attribute of the routes to which the list is applied using a **route-map** command. If the community attribute matches the regular expression in the community list, the route matches the community list.

Example The following commands apply route map 5 to routes forwarded to BGP peer 10.5.5.4. Route map 5 uses a regular expression to match community numbers ending with 305, setting the weight of matching routes to 150.

```
host1(config-router)#neighbor 10.5.5.4 remote-as 425
host1(config-router)#neighbor 10.5.5.4 route-map 5 out
host1(config-router)#exit
host1(config)#route-map 5 permit 10
host1(config-route-map)#match community 305$
host1(config-route-map)#set weight 150
```

Community Numbers

When you use a regular expression to match a community number, use the appropriate format for the community number in the community list. If you issue the **ip bgp-community new-format** command, the community number has the format *AA:NN* where *AA* is a number that identifies the autonomous system, and *NN* is a number that identifies the community within the autonomous system. Otherwise, the community number is an integer in the range 1–4294967295.

Metacharacters

Each regular expression consists of one or more metacharacters and zero or more complete or partial AS or community numbers. [Table 5 on page 43](#) describes the metacharacters supported for regular expression pattern-matching.

Table 5: Supported Regular Expression Metacharacters

Metacharacter	Description
^	Matches the beginning of the input string. Alternatively, when used as the first character within brackets— <code>[^]</code> —matches any number except the ones specified within the brackets.
\$	Matches the end of the input string.
.	Matches any single character, including white space.
*	Matches zero or more sequences of the immediately previous character or pattern.
+	Matches one or more sequences of the immediately previous character or pattern.
?	Matches zero or one sequence of the immediately previous character or pattern.

Table 5: Supported Regular Expression Metacharacters (*continued*)

Metacharacter	Description
()	Specifies patterns for multiple use when followed by one of the multiplier metacharacters: asterisk (*), plus sign (+), or question mark (?).
[]	Matches any enclosed character; specifies a range of single characters.
– (hyphen)	Used within brackets to specify a range of AS or community numbers.
_ (underscore)	Matches a ^, a \$, a comma, a space, a {, or a }. Placed on either side of a string to specify a literal and disallow substring matching. Numerals enclosed by underscores can be preceded or followed by any of the characters listed above.
	Matches characters on either side of the metacharacter; logical OR.

Using Metacharacters as Literal Tokens

You can remove the special meaning of a metacharacter by preceding it with a backslash (\). Such a construction denotes that the metacharacter is *not* treated as a metacharacter for that regular expression. It is simply a character or token with no special meaning, just as a numeral has no special meaning. The backslash applies only to the character immediately following it in the regular expression.

On an E Series router, you are likely to use the backslash only for the parentheses characters, (or). BGP indicates a segment of an AS path that is of type AS-confed-set or AS-confed-seq by enclosing that segment with parentheses.

Example The following AS-path access list uses a regular expression to match routes that have an AS-path attribute that *begins* with any AS-confed-set or AS-confed-seq:

```
host1(config)#ip as-path access-list 1 permit ^\(
```

The following AS-path access list uses a regular expression to match routes that have an AS-path attribute that *ends* with any AS-confed-set or AS-confed-seq:

```
host1(config)#ip as-path access-list 1 permit \$
```

The following AS-path access list uses a regular expression to match routes that have an AS-path attribute that *includes* the specific AS-confed-set or AS-confed-seq, (100 200):

```
host1(config)#ip as-path access-list 1 permit \((100 200\)
```

Regular Expression Examples

Table 6 on page 45 lists some representative regular expressions that you might use in an AS-path access list or community list, along with sample attribute values that match or do not match the regular expression.

Table 6: Sample Regular Expressions

Regular Expression	Matched AS-Path or Community Attribute	Example
<code>^12</code>	Begins with 12	12 23 4212 629 121245 19 but not 58 12 7
<code>[^12]</code>	Includes any numeral except 1 or 2	44 73 465 69 8 but not 1145 1912 2 49
<code>305\$</code>	Ends with 305	89 611 305305 42 30519 1305 6666:305
<code>.5</code>	Includes any one character followed by the numeral 5	89 611 3533 252 12 998 600:500
<code>1.9</code>	Includes a sequence of three characters, where the first character is numeral 1 and the third character is numeral 9	179 35 2433 252 129 48 2129 14600:2129 321:94
<code>.*</code>	Includes any character; matches all AS paths and community lists	
<code>42*</code>	Includes a number that has a numeral 4 followed by <i>zero or more</i> instances of the numeral 2	67 42 51314 33 252 422 483142 4 339 7831422
<code>1(37)*</code>	Includes a sequence that has a numeral 1 followed by <i>zero or more</i> instances of the pattern 37	137 42 211373737 29 4 1 but not 4 3737 78
<code>42+</code>	Includes a number that has a numeral 4 immediately followed by <i>one or more</i> instances of the numeral 2	67 42 2133 252 422 48 but not 4 329 78
<code>1(37)+</code>	Includes a sequence that has a numeral 1 immediately followed by <i>one or more</i> instances of the pattern 37	1373737 29 44 37137 78 137 42 21 but not 4 372 2121 37 5 1 456 881

Table 6: Sample Regular Expressions (*continued*)

Regular Expression	Matched AS-Path or Community Attribute	Example
42?	Includes a number that has a numeral 4 followed by <i>zero or only one</i> instance of the numeral 2	67 42 714 359 78 but not 33 252 422 48
1(37)?	Includes a sequence that has a numeral 1 followed by <i>zero or only one</i> instance of the pattern 37	137 42 2153 612 49 1 but not 4 13737 78
7..	Includes a sequence of three characters, where the first character is numeral 7	600 700 10025 7771 In the following examples, the three characters are 7, space, 8: 307 800 6127 888 999
^7..	Includes a number in the range 700 – 799	6127 723 999 700 100 600 but not 25 7771307 800
^7..\$	Consists only of a number in the range 700 – 799	723 700 but not 25 7771307 800 6127 723 999700 100 600
[621]	Includes any of the numerals 6, 2, or 1	60 4334 545 92 200710 86 53 The regular expression [162] has the same results.
[0-9]	Includes any number in the range 0–9	
^(22 431\)	(AS-path attribute only) Begins with the AS-confed-set or AS-confed-seq (22 431)	(22 431) 102(22 431) 55 76 but not 43 (22 431) 522 431 59
{41 19}	(AS-path attribute only) Includes the AS-set or AS-seq {41 19}	{41 19} 53 76 {41 19} 17 255 {41 19} but not 3 41 19 41 19 532
101 102 103 105	Includes either sequence 101 102 or sequence 103 105	43 101 102 5103 105 22 but not 19 102 101102 103

Table 6: Sample Regular Expressions (*continued*)

Regular Expression	Matched AS-Path or Community Attribute	Example
<code>_200_</code>	Includes the number 200 (as opposed to the pattern consisting of numeral 2, numeral 0, numeral 0) Our implementation of regular expressions is not literal. Substring matching is enabled by default. Specifying <code>200</code> (no underscores) results in a match on <code>200</code> and on <code>2005</code> . The underscore metacharacter disables substring matching.	33 200 422 48^200\$ ^200 500\$ but not 33 20 422 48 51 2005

For information about using AS-path access lists, see [“Access Lists” on page 20](#). For information about using community lists, see [“Community Lists” on page 37](#).

Managing the Routing Table

You can clear all routes from the IP routing table, and then enable the owning protocols—BGP, OSPF, RIP—to reinstall the routes.

clear ip routes

- Use to clear all routing entries or a specified entry from the IP routing table.
- Example
 `host1#clear ip routes`
- There is no **no** version.
- See `clear ip routes`.

ip refresh-route

- Use to reinstall routes removed from the IP routing table by the **clear ip route** command.
- Example
 `host1#ip refresh-route`
- There is no **no** version.
- See `ip refresh-route`.

Troubleshooting Routing Policy

You can turn on debugging for routing policy by issuing the **log severity debug ipRoutePolicy** command from Global Configuration mode. You can specify different levels of severity for `ipRoutePolicy`. For more information about using **log** commands for troubleshooting, see *Managing the System* in *JunosE System Basics Configuration Guide*.

Monitoring Routing Policy

You can monitor the following aspects of routing policy using **show** commands:

To Display	Commands
Access lists	show access-list show ip as-path access-list show ipv6 access-list
Community lists	show ip community-list
Policy lists	show ip match-policy-list
Prefix lists	show ip prefix-list
Prefix trees	show ip prefix-tree
Protocols	show ip protocols
Redistribution policies	show ip redistribute
Routes	show ip route
Route maps	show route-map
Interfaces and next hops	show ip route slot 5 192.168.5.4
Static routes	show ip static
Traffic	show ip traffic

You can use the output filtering feature of the **show** command to include or exclude lines of output based on a text string that you specify. For details, see *Command Line Interface* in *JunosE System Basics Configuration Guide*.

show access-list

show ipv6 access-list

- Use to display information about access lists.
- The displayed information includes the instances of each access list.
- Use the **detail** keyword to display the automatically assigned element ID for each access list entry. Only rules that you explicitly create have element IDs.
- Example 1

```
host1#show access-list
IP Access List 1:
  permit ip host 172.31.192.217 any
  permit ip 12.40.0.0 0.0.0.3 any
```



```

deny ip any any
IP Access List 2:
  permit ip 172.19.0.0 0.0.255.255 any
  deny ip 0.0.0.0 255.255.255.255 any
IP Access List 10:
  permit ip any any
IP Access List 11:
  deny ip any any

```

- Example 2

```

host1#show access-list detail
IP Access List 1:
  1: permit ip host 172.31.192.217 any
  2: permit ip 12.40.0.0 0.0.0.3 any
  deny ip any any

```

- See show access-list.
- See show ipv6 access-list.

show ip as-path-access-list

- Use to display information about AS-path access lists.
- Example

```

host1#show ip as-path-access-list
AS Path Access List 1:
  permit .*
AS Path Access List 2:
  deny .*
AS Path Access List 3:
  permit _109_
  deny .*
AS Path Access List 4:
  permit _109$
  deny .*
AS Path Access List 10:
  deny _109$
  permit ^108_
  deny .*

```

- See show ip as-path-access-list.

show ip community-list

- Use to display community list information.
- Display varies based on whether you issued the **ip bgp community new-format** command.
- Example 1—If you did not issue the **ip bgp community new-format** command, the display appears as follows:

```

host1#show ip community-list
Community List 1:
  permit 81200109
  permit 81200110
  permit 81200108
Community List 2:

```

```
deny 81200109
permit 81200110
permit 81200108
Community List 4:
  permit local-as
Community List 5:
  permit no-advertise
Community List 6:
  permit no-export
Community List 7:
  permit internet
```

- Example 2—If you did issue the **ip bgp community new-format** command, the display appears as follows:

```
host1#show ip community-list
Community List 1:
  permit 1239:1005
  permit 1239:1006
  permit 1239:1004
Community List 2:
  deny 1239:1005
  permit 1239:1006
  permit 1239:1004
Community List 4:
  permit local-as
Community List 5:
  permit no-advertise
Community List 6:
  permit no-export
Community List 7:
  permit internet
```

- See show ip community-list.

show ip match-policy-list

- Use to display configured policy lists.
- Example

```
host1#show ip match-policy-list
match-policy-list list1, permit
Match clauses:
  match access-list addrList1
  match distance 100
```

- See show ip match-policy-list.

show ip prefix-list

- Use to display information about the prefix lists currently configured on the router.
- Use the **summary** keyword to display abbreviated information about prefix lists.
- Example 1

```
host1#show ip prefix-list
Prefix-list with the last deletion/insertion: def
ip prefix-list name abc: 4 entries
  seq 5 permit 192.168.0.0/16 le 24
```

```

seq 10 permit 192.178.0.0/16 le 24
seq 15 deny 195.178.0.0/16 le 24
seq 20 deny 195.178.0.0/16 le 32
ip prefix-list name def: 1 entries
seq 5 deny 192.170.0.0/16

```

- Example 2

```

host1#show ip prefix-list summary
Total memory used for prefix-list: 310 bytes
Prefix-list with the last deletion/insertion: def
ip prefix-list name abc:
  count: 4, range entries: 4, sequences: 5-20
ip prefix-list name def:
  count: 1, range entries: 0, sequences: 5-5

```

- See show ip prefix-list.

show ip prefix-tree

- Use to display information about the prefix trees currently configured on the router.
- Use the **summary** keyword to display abbreviated information about prefix trees.
- Example 1

```

host1#show ip prefix-tree
Prefix-tree with the last deletion/insertion: t_abc5
ip prefix-tree name t_abc1: 1 entries
  permit 108.243.0.0/16
ip prefix-tree name t_abc2: 3 entries
  permit 101.10.254.0/24
  permit 102.10.248.0/21
  permit 103.10.192.0/18
  permit 108.109.0.0/16
  permit 108.109.241.0/24
ip prefix-tree name t_abc3: 1 entries
  deny 108.0.0.0/8

```

- Example 2

```

host1#show ip prefix-tree summary
Total memory used for prefix-tree: 860 bytes
Prefix-tree with the last deletion/insertion: t_abc5
ip prefix-tree name t_abc1:
  count: 1
ip prefix-tree name t_abc2:
  count: 5
ip prefix-tree name t_abc3:
  count: 1

```

- See show ip prefix-tree.

show ip protocols

- Use to display detailed information about the protocols currently configured on the router.
- Use the **summary** keyword to display only a list of the configured protocols.

- For field descriptions, see the **show** commands for the individual routing protocols in their respective *Configuration Guide* chapters.
- Example

```
host1#show ip protocols
Routing Protocol is " bgp 1"

  Default local preference is 100
  IGP synchronization is enabled
  Always compare MED is disabled
  Router flap damping is disabled
  Administrative Distance: external 20 internal 200 local 200
  Neighbor(s):
    No neighbors are configured
  Routing for Networks:

Routing Protocol is " ospf 255" with Router ID 100.100.100.1
  Distance is 110

  Address Summarization:
    None
  Routing for Networks:

Routing Protocol is " rip"
  Router Administrative State: enable
  System version RIP1: send = 1, receive = 1 or 2
  Update interval: 30 seconds
  Invalid after: 180 seconds
  hold down time: 120 seconds
  flushed interval: 300 seconds
  Filter applied to outgoing route update is not set
  Filter applied to incoming route update is not set
  No global route map
  Distance is 120
  Interface          Tx    Rx    Auth

  Routing for Networks:
    10.2.1.0/255.255.255.0
```

- See show ip protocols.

show ip redistribute

- Use to display configured route redistribution policy.
- Field descriptions
 - To—Protocol into which routes are distributed
 - From—Protocol from which routes are distributed
 - status—Redistribution status
 - route map number—Number of the route map
- Example

```

host1#show ip redistribute
To ospf, From static is enabled with route map 4
To ospf, From connected is enabled with route map 3

```

- See show ip redistribute.

show ip route

- Use to display the current state of the routing table, including routes that are not used for forwarding.
- You can display all routes, a specific route, all routes beginning with a specified address, routes for a particular protocol (BGP, IS-IS, OSPF, or RIP), locally connected routes, internal control routes, static routes, or summary counters for the routing table.
- Field descriptions
 - Prefix—IP address prefix
 - Length—Prefix length
 - Type—Protocol type
 - Next Hop—IP address of the next hop
 - Dist—Distance metric for the route
 - Met—Number of hops
 - Intf—Interface type and interface specifier
- Example 1

```

host1#show ip route
Protocol/Route type codes:
  I1- ISIS level 1, I2- ISIS level2,
  I- route type intra, IA- route type inter, E- route type external,
  i- metric type internal, e- metric type external,
  O- OSPF, E1- external type 1, E2- external type2,
  N1- NSSA external type1, N2- NSSA external type2

Prefix/Length  Type    Next Hop    Dist/Met  Intf
-----
172.16.2.0/24   Bgp      192.168.1.102  20/1      fastEthernet0/0
10.10.0.112/32  Static   192.168.1.1    1/1        fastEthernet0/0
10.1.1.0/24     Connect  10.1.1.1       0/1        atm3/0.100

```

- Example 2

```

host1#show ip route static
Protocol/Route type codes:
  I1- ISIS level 1, I2- ISIS level2,
  I- route type intra, IA- route type inter, E- route type external,
  i- metric type internal, e- metric type external,
  O- OSPF, E1- external type 1, E2- external type2,
  N1- NSSA external type1, N2- NSSA external type2

Prefix/Length  Type    Next Hop    Dist/Met  Intf
-----
10.10.0.112/32  Static   192.168.1.1    1/1        fastEthernet0/0

```

- Example 3

```
host1#show ip route summary
Unicast routes:
8 total routes, 576 bytes in route entries
0 isis routes
0 rip routes
3 static routes
2 connected routes
1 bgp routes
0 ospf routes
2 other internal routes
0 access routes
0 internally created access host routes

Last route added/deleted: 2::4/128 by BGP
At MON FEB 04 2008 14:18:25 UTC

Unicast routes used only for Multicast RPF check:
0 total routes, 0 bytes in route entries
0 isis routes
0 rip routes
0 static routes
0 connected routes
0 bgp routes
0 ospf routes
0 other internal routes
0 access routes
0 internally created access host routes
0 mbgp routes
0 dvmrp routes

Last route added/deleted: null by Invalid
At MON FEB 04 2008 14:18:04 UTC

MPLS tunnel routes (not used for forwarding):
3 total routes, 216 bytes in route entries
1 bgp tunnel routes
1 ldp tunnel routes
1 rsvp tunnel routes

Last route added/deleted: 2::4/128 by BGP Tunnel
At MON FEB 04 2008 14:18:26 UTC
```

- See show ip route.

show ip route slot

- Use to display the interface and next hop for an IP address in the routing table of a line module specified by the slot it occupies.
 - *slotNumber*—Number of the slot that contains the line module for which the information is displayed
 - *ipAddress*—IP address to look up in the routing table
- Field descriptions
 - IP address—Address that is reachable through the interface

- Interface—Interface type and specifier associated with the IP address; displays “Local Interface” if a special interface index is present in the routing table for special IP addresses, such as broadcast addresses
- Next Hop—Next hop to reach the IP address; displays “---” if no next hop is associated with the IP address
- Example 1

```
host1#show ip route slot 6 10.10.0.231
  IP address      Interface      Next Hop
-----
10.10.0.231      fastEthernet 6/0  10.10.0.231
```

- Example 2

```
host1#show ip route slot 9 90.248.1.2
  IP address      Interface      Next Hop
-----
90.248.1.2       serial9/23:2    ---
```

- Example 3

```
host1#show ip route slot 9 90.249.255.255
  IP address      Interface      Next Hop
-----
90.249.255.255   Local Interface ---
```

- See show ip route slot.

show ip static

- Use to display the status of static routes in the routing table.
- You can specify an optional IP mask that filters specific routes.
- Field descriptions
 - Prefix—IP address prefix
 - Length—Prefix length
 - Next Hop—IP address of the next hop
 - Met—Number of hops
 - Dist—Administrative distance or weight assigned to the route
 - Tag—Tag value assigned to the route
 - Intf—Interface type and interface specifier
- Example

```
host1#show ip static
Prefix/Length  Next Hop:  Met: Dist: Tag: Intf:
10.2.0.0/24    192.168.1.1  1   1    0   ethernet6/0
```

10.2.1.0/24	192.168.1.1	1	1	1	ethernet6/0
172.31.1.48/32	172.18.2.2	1	1	0	atm5/1.1

- See show ip static.

show ip traffic

- Use to display statistics about IP traffic.
- Field descriptions
 - IP Statistics Rcvd:
 - Router Id—Router ID number
 - total—Number of frames received
 - local destination—Frames with this router as their destination
 - hdr errors—Number of packets received that contain header errors
 - addr errors—Number of packets received that contain addressing errors
 - unkn proto—Number of packets received that contain unknown protocols
 - discards—Number of discarded packets
 - IP Statistics Frags:
 - reassembled—Number of reassembled packets
 - reasm timed out—Number of reassembled packets that timed out
 - reasm req—Number of requests for reassembly
 - reasm fails—Number of reassembly failures
 - frag ok—Number of fragmented packets reassembled successfully
 - frag fail—Number of fragmented packets reassembled unsuccessfully
 - frag creates—Number of packets created by fragmentation
 - IP Statistics Sent:
 - forwarded—Number of packets forwarded
 - generated—Number of packets generated
 - out disc—Number of outbound packets discarded
 - no routes—Number of packets that could not be routed
 - routing discards—Number of packets that could not be routed that were discarded
 - IP Statistics Route:

- routes in table—Number of routes in the routing table
- timestamp req—Number of requests for a timestamp
- timestamp rpy—Number of replies to timestamp requests
- addr mask req—Number of address mask requests
- addr mask rpy—Number of address mask replies
- ICMP Statistics Rcvd:
 - total—Total number of ICMP packets received
 - errors—Number of error packets received
 - dst unreachable—Number of packets received with destination unreachable
 - time exceed—Number of packets received with time-to-live exceeded
 - param probs—Number of packets received with parameter errors
 - src quench—Number of source quench packets received
 - redirects—Number of receive packet redirects received
 - echo req—Number of echo request (ping) packets received
 - echo rpy—Number of echo replies received
 - timestamp req—Number of requests for a timestamp received
 - timestamp rpy—Number of replies of timestamp requests received
 - addr mask req—Number of mask requests received
 - addr mask rpy—Number of mask replies received
- ICMP Statistics Sent:
 - total—Total number of ICMP packets sent
 - errors—Number of error packets sent
 - dest unreachable—Number of packets sent with destination unreachable
 - time excd—Number of packets sent with time-to-live exceeded
 - param prob—Number of packets sent with parameter errors
 - src quench—Number of source quench packets sent
 - redirects—Number of send packet redirects sent
 - echo req—Number of echo request (ping) packets sent
 - echo rpy—Number of echo replies sent

- timestamp req—Number of requests for a timestamp sent
- timestamp rpy—Number of replies to timestamp requests sent
- addr mask req—Number of address mask requests sent
- addr mask rpy—Number of address mask replies sent
- UDP Statistics Rcvd:
 - total—Total number of UDP packets received
 - checksum—Number of checksum error packets received
 - no port—Number of packets received for which no application listener was listening on the destination port
- UDP Statistics Sent:
 - total—Total number of UDP packets sent
 - errors—Number of error packets sent
- TCP Global Statistics Connections:
 - attempted—Number of outgoing TCP connections attempted
 - accepted—Number of incoming TCP connections accepted
 - established—Number of TCP connections established
 - dropped—Number of TCP connections dropped
 - closed—Number of TCP connections closed
- TCP Global Statistics Rcvd:
 - total pkts—Total number of TCP packets received
 - in-sequence pkts—Number of packets received in sequence
 - bytes—Number of bytes received
 - chksum err pkts—Number of checksum error packets received
 - authentication err pkts—Number of authentication error packets received
 - bad offset pkts—Number of packets received with bad offsets
 - short pkts—Number of short packets received
 - duplicate pkts—Number of duplicate packets received
 - out of order pkts—Number of packets received out of order
- TCP Global Statistics Sent:

- total pkts—Total number of TCP packets sent
- data pkts—Number of data packets sent
- bytes—Number of bytes sent
- retransmitted pkts—Number of packets retransmitted
- retransmitted bytes—Number of retransmitted bytes
- OSPF Statistics—Not supported for this version of the router
- IGMP Statistics—Not supported for this version of the router
- ARP Statistics—Not supported for this version of the router

- Example

```

host1#show ip traffic
IP statistics: Router Id: 172.31.192.217
  Rcvd: 97833 total, 171059 local destination
    0 hdr errors, 0 addr errors
      167 unkn proto, 0 discards
    Frags: 4 reassembled, 30 reasm timed out, 8 reasm req
      0 reasm fails, 145 frag ok, 0 frag fail
      290 frag creates
  Sent: 15 forwarded, 25144 generated, 0 out disc
    0 no routes, 0 routing discards
  Route: 57680 routes in table
    0 timestamp req, 0 timestamp rpy
    0 addr mask req, 0 addr mask rpy

ICMP statistics:
  Rcvd: 561 total, 0 errors, 15 dst unreachable
    0 time exceed, 0 param probs, 0 src quench
    0 redirects, 0 echo req, 0 echo rpy
    0 timestamp req, 0 timestamp rpy
    0 addr mask req, 0 addr mask rpy
  Sent: 0 total, 0 errors, 0 dest unreachable
    0 time excd, 0 param prob, 0 src quench
    0 redirects, 0 echo req, 0 echo rpy

UDP Statistics:
  Rcvd: 93326 total, 0 checksum errors, 90610 no port
  Sent: 0 total, 0 errors

TCP Global Statistics:
  Connections: 7358 attempted, 4 accepted, 7362 established
    0 dropped, 14718 closed
  Rcvd: 75889 total pkts, 53591 in-sequence pkts, 3120283 bytes
    0 chksum err pkts, 0 authentication err pkts, 0 bad offset
    0 short pkts, 0 duplicate pkts, 0 out of order pkts
  Sent: 82318 total pkts, 44381 data pkts, 656321 bytes
    34 retransmitted pkts, 487 retransmitted bytes

OSPF Statistics:

IGMP Statistics:

```

ARP Statistics:

- See show ip traffic.

show route-map

- Use to display the configured route maps.
- The displayed information includes the instances of each access list such as **match** and **set** commands.
- Example

```
host1(config)#route-map 1 permit 10
host1(config-route-map)#match community 44
host1(config-route-map)#set local-pref 400
host1(config-route-map)#exit
host1(config)#exit
host1#show route-map 1
route-map 1, permit, sequence 10
  Match clauses:
    match community 44
  Set clauses:
    set local-pref 400
```

- See show route-map.

CHAPTER 2

Configuring NAT

This chapter describes how to configure Network Address Translation (NAT) on your ERX router; it contains the following sections:

- [Overview on page 61](#)
- [Platform Considerations on page 62](#)
- [References on page 62](#)
- [NAT Configurations on page 63](#)
- [Network and Address Terms on page 64](#)
- [Understanding Address Translation on page 65](#)
- [Address Assignment Methods on page 66](#)
- [Order of Operations on page 66](#)
- [PPTP and GRE Tunneling Through NAT on page 67](#)
- [Packet Discard Rules on page 68](#)
- [Before You Begin on page 68](#)
- [Configuring a NAT License on page 68](#)
- [Limiting Translation Entries on page 69](#)
- [Specifying Inside and Outside Interfaces on page 69](#)
- [Defining Static Address Translations on page 69](#)
- [Defining Dynamic Translations on page 71](#)
- [Clearing Dynamic Translations on page 76](#)
- [NAT Configuration Examples on page 77](#)
- [Tunnel Configuration Through NAT Examples on page 83](#)
- [GRE Flows Through NAT on page 84](#)
- [Monitoring NAT on page 85](#)

Overview

The Internet faces the challenges of conserving IP address space while continuing to provide scalability in routing. Network Address Translation (NAT) helps address these challenges by allowing the conservation of registered IP addresses within private networks and simplifying IP addressing management tasks through a form of *transparent routing*.

NAT enables you to translate IP addresses between two address realms (for example, between an intranet network that uses private, not publicly routable addresses and the Internet, or between two overlapping, private networks). When incoming traffic is received, the IP addresses are translated back for delivery within the private network.

Using NAT at the edge of your intranet provides the following advantages:

- Allows unregistered *private* addresses to connect to the Internet by translating those addresses into globally registered IP addresses
- Increases network privacy by hiding internal IP addresses from external networks

Platform Considerations

For information about modules that support NAT on ERX14xx models, ERX7xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support NAT.



NOTE: The E120 and E320 Broadband Services Routers do not support configuration of NAT.

Module Requirements

To configure NAT on ERX7xx models, ERX14xx models, and the ERX310 router, you must install a Service Module (SM). For information about installing modules in E Series Broadband Services Routers, see the *ERX Hardware Guide*.

Unlike other line modules, SMs do not pair with corresponding I/O modules that contain ingress and egress ports. Instead, they receive data from and transmit data to other line modules with access to ingress and egress ports on their own associated I/O modules.

For a list of the modules that support NAT, see *ERX Module Guide, Appendix A, Module Protocol Support*.

References

For more information about NAT, consult the following resources:

- RFC 2663-IP Network Address Translator (NAT) Terminology and Considerations (August 1999)
- RFC 2694-DNS extensions to Network Address Translators (DNS_ALG) (September 1999)
- RFC 2993-Architecture Implications of NAT (November 2000)

- RFC 3022-Traditional IP Network Address Translator (Traditional NAT) (January 2001)
- RFC 3027-Protocol Complications with the IP Network Address Translator (January 2001)

NAT Configurations

You can configure NAT in several different ways. Each of the following configuration methods provides a solution for different configuration requirements:

- Traditional NAT
- Bidirectional NAT
- Twice NAT

Traditional NAT

Traditional NAT is the most common method of using address translation. Its primary use is translating private addresses to legal addresses for use in an external network. When configured for dynamic operation, hosts within a private network can initiate access to the external (public) network, but external nodes on the outside network cannot initiate access to the private network.

Addresses on the private network and public network must not overlap. Also, route destination advertisements on the public network (for example, the Internet) can appear within the inside network, but the NAT router does not propagate advertisements of local routes that reference private addresses out to the public network.

There are two types of traditional NAT—basic NAT and NAT.

Basic NAT

Basic NAT provides translation for IP addresses only (called a *simple* translation) and places the mapping into a NAT table. In other words, for packets outbound from the private network, the NAT router translates the source IP address and related fields (for example, IP, TCP, UDP, and ICMP header checksums). For inbound packets, the NAT router translates the destination IP address (and related checksums) for entries that it finds in its translation table.



CAUTION: Although NAT is the simplest translation method, it is the least secure. By not including port or external host information in the translation, basic NAT allows access to any port of the private host by any external host.

NAPT

Network Address Port Translation (NAPT) extends the level of translation beyond that of basic NAT; it modifies both the IP address and the transport identifier (for example, the TCP or UDP port number, or the ICMP query identifier) and places the mapping into the translation table (this entry is called an *extended* translation). This method can translate the addresses and transport identifiers of many private hosts into a few external

addresses and transport identifiers, to make efficient use of globally registered IP addresses.

Similar to basic NAT, for outbound packets NAPT translates the source IP address, source transport identifier, and related checksum fields. For inbound packets NAPT translates the destination IP address, destination transport identifier, and checksum fields.

Bidirectional NAT

Bidirectional (or two-way) NAT adds support to basic NAT for the Domain Name System (DNS) so public hosts can initiate sessions into the private network, usually to reach servers intended for public access.

When an outside host attempts to resolve the name of an inside host on a private network, the NAT router intercepts the DNS reply and installs an address translation to allow the outside host to reach the inside host by using a public address. When the outside host initiates a connection with the inside host on the private network, the NAT router translates that public destination address to the private address of the inside host and, on the return path, replaces the source address with the advertised public address.

You might need to perform some additional configuration to allow public access from the Internet to a DNS server that resides in the private domain. (See [“Bidirectional NAT Example” on page 78.](#))

The same address space requirements and routing restrictions apply to bidirectional NAT that were described for traditional NAT. The difference between these two methods is that the DNS exchange might create entries within the translation table.

Twice NAT

In twice NAT, both the source and destination addresses are subject to translation as packets traverse the NAT router in either direction. For example, you would use twice NAT if you are connecting two networks in which all or some addresses in one network overlap addresses in another network, whether the network is private or public.

Network and Address Terms

The NAT implementation defines an address realm as either *inside* or *outside*, with the router that is running NAT acting as the defining boundary between the two realms.

From a NAT perspective, an *inside* network is the local portion of a network that uses private, not publicly routable IP addresses that you want to translate. An *outside* network is the public portion of a network that uses legitimate, publicly routable IP addresses to which you want private hosts to connect.

The addresses that are translated by NAT between address realms are labeled as *inside* or *outside*, and as *local* or *global*. When reading the terms in the following sections, keep the following definitions in mind:

- The terms *inside* and *outside* refer to the host that the address is associated with.
- The terms *local* and *global* refer to the network on which the address appears.

Inside Local Addresses

The *inside local* address is a configured IP address that is assigned to a host on the inside network. Addresses may be globally unique (not requiring translation), allocated from the private address space defined in RFC 1918, or officially allocated to some other organization.

Inside Global Addresses

The *inside global* address is the *translated* IP address of an inside host as seen by an outside host and network. Addresses may be allocated from a globally unique address space (often provided by the ISP, if the inside address is connected to the global Internet).

Outside Local Addresses

The *outside local* address is the *translated* IP address of an outside host as it appears to the inside network. Addresses may be globally unique (not requiring translation), allocated from the private address space defined in RFC 1918, or officially allocated to some other organization.

Outside Global Addresses

The *outside global* address is the configured, publicly routable IP address assigned to a host on the outside network.

Understanding Address Translation

Address translation can occur one of two ways: inside or outside source translation.

Inside Source Translation

Inside source translation is the most commonly used NAT configuration. When an inside host sends a packet to the outside network, the NAT router translates the source information (either the source address or the source address/port pair) and, in the inbound direction, restores the original information (this time operating on the destination address or address/port pair).

For outbound traffic, the NAT router translates the inside local address (or address/port) into the inside global address (or address/port), either through a statically defined translation or dynamically created translation. For inbound traffic, a translation must be found to revert the inside global address (or address/port) into the inside local address (or address/port), or the packet is not routed into the inside network.



NOTE: Dynamic inside source translations are established by outbound traffic.

You use inside source translation in traditional and bidirectional NAT configurations.

Outside Source Translation

Outside source translation is used in NAT configurations only when addresses of external hosts might create a conflict on the private network. This complementary translation process is performed on the opposite addressing fields in the IP packet. When an outside host sends a packet to the inside network, the NAT router translates the source information (either the source address or the source address/port pair) and, in the outbound direction, restores the original information (this time operating on the destination address or address/port pair).

For inbound traffic, the NAT router translates the outside global address (or address/port) into the outside local address (or address/port), either through a statically defined translation or dynamically created translation. For outbound traffic, a translation must be found to revert the outside local address (or address/port) into the outside global address (or address/port), or the packet is not routed into the outside network.



NOTE: Dynamic outside source translations are established by inbound traffic.

You use outside source translation along with inside source translation to configure twice NAT.

Address Assignment Methods

NAT uses one of two methods to assign a translated IP address: static translation or dynamic translation.

Static Translations

You enter static translations as direct configuration settings that remain in the translation table until you remove them. You use static translations when you must initiate connections from both the inside and outside interfaces, or when the translation is not subject to change.

Dynamic Translations

Dynamic translations use access list rules, to determine whether to apply NAT to incoming traffic, and NAT address pools, from which a NAT translation can obtain IP addresses. You use dynamic translation when you want the NAT router to initiate and manage address translation and session flows between address realms on demand.

Order of Operations

This section describes the order of operations for both inside-to-outside and outside-to-inside translation.

Inside-to-Outside Translation

Inside-to-outside translation occurs in the following order:

1. Inside (privately addressed) traffic enters the router on an interface marked as *inside*.
2. A route lookup is performed.
3. If the next interface is marked as *outside*, the router sends the traffic to the server module.
4. The server module performs the appropriate translation.
5. The router forwards the packet to the appropriate egress line module.
6. The line module sends the packet as outbound traffic using a globally unique source address (inside source translation), destination address (outside source translation), and ports (NAPT).

Outside-to-Inside Translation

Outside-to-inside translation occurs in the following order:

1. Traffic from the outside, public domain enters the router.
2. All traffic from an interface that is marked *outside*, whether or not it requires NAT, is sent to the server module.
3. The server module searches for an associated NAT match.
4. If the server module:
 - Finds a NAT match, and the destination interface is marked as *inside*, the server module performs the appropriate translation and sends the packet to the appropriate destination.
 - Does not find a NAT match, and the destination interface is marked as *inside*, the server module drops the packet.
 - Does not find a NAT match, and the destination interface is not marked as *inside*, the server module processes the packet normally for its destination.

PPTP and GRE Tunneling Through NAT

You can configure NAT traversal support for GRE flows using simple translations (Basic NAT). Because PPTP uses an enhanced GRE encapsulation for the PPP payload, configuring for GRE flows also supports NAT traversal for PPTP tunnels.



NOTE: Network Address Port Translation (NAPT) for GRE packets is not supported for GRE flows.

When configured, the following types of translations are supported for GRE and PPTP tunnels:

- Inside source static simple translations (inbound and outbound)
- Outside source static simple translations (inbound and outbound)

- Inside source dynamic simple translations (inbound and outbound)
- Outside source dynamic simple translations (inbound and outbound)
- Combinations of the preceding translations (for example, twice NAT)

Packet Discard Rules

For all supported types of traffic (TCP, UDP, ICMP, and GRE), NAT discards packets in the following cases:

- When the translation table is full (that is, no more entries can be added).
- When the address pool is exhausted for outbound packets with inside source dynamic translation.
- When no match can be found for the destination addresses of inbound packets.
- When the address pool is exhausted for inbound packets with outside source dynamic translation.

In addition, NAT discards GRE packets when the GRE packets match an NAPT rule.

Before You Begin

You can configure certain IP interfaces to participate in Network Address Translation. This chapter discusses how to configure NAT to function for certain IP interfaces. For information about general IP interface configuration, see *Configuring IP in JunosE IP, IPv6, and IGP Configuration Guide*.

Configuring a NAT License

You must configure a NAT license before you can use any NAT commands on the ERX router.

license nat

- Use to specify a NAT license.
- Purchase a NAT license to allow NAT configuration on the ERX router.



.....
NOTE: Acquire the license from Juniper Networks Customer Services and Support or from your Juniper Networks sales representative.
.....

- Example

```
host1(config)#license nat license-value
```
- Use the **no** version to disable the license.
- See `license nat`.

Limiting Translation Entries

You can configure the maximum number of dynamic translation entries that the translation table contains in global configuration mode for a given virtual router.

ip nat translation max-entries

- Use to specify the maximum number of dynamic translation entries that the translation table can contain in global configuration mode for the given virtual router.
- Example

```
host:VR1 (config-if) #ip nat translation max-entries 1000
```
- Use the **no** version to remove the configured limit and return the maximum number of translation entries to the default, which is no enforced limit, as capacity allows.
- See `ip nat translation max-entries`.

Specifying Inside and Outside Interfaces

You must mark interfaces that participate in NAT translation as residing on the inside or the outside network.



CAUTION: Only packets routed between an inside and an outside interface are subject to translation.

You can unmark an interface by using the `no` version of this command.

ip nat

- Use to mark an IP interface as participating in NAT translation.
- Use the keyword (**inside** or **outside**) to specify the side of the network on which the interface resides.
- Example

```
host (config-if) # ip nat inside
```
- Use the **no** version to unmark the interface (the default) so that it does not participate in NAT translation.
- See `ip nat`.

Defining Static Address Translations

Static address translation establishes a one-to-one mapping between a local and global address or local and global address/port pair. When you specify a static address translation or address/port pair translation, you issue commands to indicate how the translation is applied, along with more specific variables that further define the type of translation.



CAUTION: You must mark interfaces that participate in NAT translation as on the inside or the outside network. See [“Specifying Inside and Outside Interfaces” on page 69](#) for details.

Creating Static Inside Source Translations

You use the **ip nat inside source static** command to create static translations from a local IP address to a global IP address, and to *untranslate* the destination address when a packet returns from the outside network to the inside network. When you configure traditional NAT (both basic NAT and NAPT), you only need to use this command alone. However, when you configure twice NAT, you must also use [“ip nat outside source static” on page 71](#).

The **ip nat inside source static** command creates a simple (IP address only) or extended (IP address, port, and protocol) entry in the translation table that maps the two addresses.

ip nat inside source static

- Use to create static translations for a source address (or address/port pair) when routing a packet from the inside network to the outside network, and to *untranslate* the destination address (or address/port pair) when a packet returns from the outside network to the inside network.
- A static translation created with the **ip nat inside source static** command enables any outside host to contact the inside host by using the inside global address of the inside host. A static translation can be used by traffic that is initiated in either direction
- Example 1—Simple address translation

```
host (config) # ip nat inside source static 10.1.2.3 171.69.68.10
```
- Example 2—Extended address/port translation

```
host (config) # ip nat inside source static tcp 10.1.2.3 15 171.69.68.10 30
```
- Use the **no** version to remove the static translation and purge the associated translations from the translation table.
- See *ip nat inside source static*.

Creating Static Outside Source Translations

Less commonly used, outside source translation enables you to set up translation between two non-unique or not publicly routable networks (for example, two separate networks that use overlapping IP address blocks).

ip nat outside source static

- Use to translate the source address when routing a packet from the outside network to the inside network, and to *untranslate* the destination address when a packet travels from the inside network to the outside network.
- Creates a simple (IP address only) or extended (IP address, protocol, and port) entry in the translation table that maps the two addresses.
- A static translation created with the **ip nat outside source static** command enables any inside host to contact the outside host by using the outside local address of the outside host. A static translation can be used by traffic that is initiated in either direction.
- Example 1—Simple address translation

```
host (config) # ip nat outside source static 171.69.68.10 10.1.2.3
```
- Example 2—Extended address/port translation

```
host (config) # ip nat outside source static tcp 171.69.68.10 56 10.1.2.3 24
```
- Use the **no** version to remove the static translation and purge the associated translations from the translation table.
- See `ip nat outside source static`.

Defining Dynamic Translations

Dynamic translations use access list rules, to determine whether or not to apply NAT to incoming traffic, and NAT address pools, from which a NAT translation can allocate IP addresses. You use dynamic translation when you want the NAT router to initiate and manage address translation and session flows between address realms on demand.

To configure dynamic translations:

- Define any access list rules that the NAT router uses to decide which packets need translation.
- Define an address pool from which the NAT router obtains addresses.
- Define inside and outside source translation rules for the NAT router to create NAT translations.
- Mark interfaces as *inside* or *outside*.
- (Optional) Modify any translation timeout values.

Creating Access List Rules

Before you create a dynamic translation, create the access list rules that you plan to apply to the translation. For information about configuring access lists, see [“Configuring Routing Policy” on page 3](#).

The router evaluates multiple commands for the same access list in the order they were created. An undefined access list implicitly contains a rule to *permit any*. A defined access list implicitly ends with a rule to *deny any*.



NOTE: The access lists do not filter any packets; they determine whether the packet requires translation.

You use the **access-list** command to create an access list.

access-list

- Use to define an IP access list to permit or deny translation based on the addresses in the packets.
- Each access list is a set of permit or deny conditions for routes that are candidates for translation (that is, moving from the inside network to the outside network).
- A zero in the wildcard mask means that the route must exactly match the corresponding bit in the address. A one in the wildcard mask means that the route does not have to match the corresponding bit in the address.
- Use the **log** keyword to log an Info event in the ipAccessList log whenever matching an access list rule.
- Example

```
host1(config)#access-list bronze permit ip host any 228.0.0.0 0.0.0.255
```
- Use the **no** version to delete the access list (by not specifying any other options), the specified entry in the access list, or the log for the specified access list or entry (by specifying the **log** keyword).
- See access-list.

Defining Address Pools

Before you can configure dynamic translation, create an address pool. An address pool is a group of IP addresses from which the NAT router obtains an address when dynamically creating a new translation. You can create address pools with either a single range or multiple, nonoverlapping ranges.

When you create a single range, you specify the starting and ending IP addresses for the range in the root **ip nat pool** command. However, when you create multiple, nonoverlapping ranges, you omit the optional starting and ending IP addresses in the root **ip nat pool** command; this launches the IP NAT Pool Configuration (config-ipnat-pool) mode.

The config-ipnat-pool mode uses an **address** command to specify a range of IP addresses. You can repeat this command to create multiple, nonoverlapping ranges.

When you create or edit address pools, keep the following in mind:

- Starting and ending IP addresses for the specified range are inclusive and must reside on the same subnet.
- Address ranges are verified against other ranges in the specified pool to exclude range overlaps. Additional verification occurs when the pool is associated with a translation rule and the router can determine whether the rule is inside or outside.
- You cannot change the network mask if configured ranges already exist.
- The network mask (or prefix length) is used to recognize host addresses that end in either all zeros or all ones. These addresses are reserved as broadcast addresses and are not allocated from an address pool, even if they are included in an address pool range.
- You cannot remove an address pool if the pool is part of a translation rule or if any of the ranges within the pool are still in use. You must issue the **clear ip nat translation** command to clear any ranges before you can remove the pool to which they apply.

address

- Use to specify a range of IP addresses in config-ipnat-pool mode; you can repeat the **address** command to create multiple ranges.
- Example


```
host (config-ipnat-pool)#address 171.69.40.110 171.69.40.115
```
- Use the **no** version to remove the range for the current address pool.
- See address.

ip nat pool

- Use to create address pools.
- Example 1—Creating a single, continuous range


```
host (config) #ip nat pool singlerange 171.69.40.1 171.69.40.100 prefix-length 30
```
- Example 2—Creating multiple, discontinuous ranges


```
host (config) #ip nat pool multiplerange prefix-length 30
host (config-ipnat-pool)#address 171.69.40.110 171.69.40.112
host (config-ipnat-pool)#address 171.69.40.118 171.69.40.120
host (config-ipnat-pool)#exit
```
- Use the **no** version to remove the address range.
- See ip nat pool.

Defining Dynamic Translation Rules

You can use the CLI to define dynamic translation rules for inside and outside sources.



CAUTION: You must mark interfaces that participate in NAT translation as on the inside or the outside network. See [“Specifying Inside and Outside Interfaces” on page 69](#) for details.

You can create a dynamic translation rule to configure inside source or outside source translation. If the NAT router cannot locate a matching entry in its translation database for a given packet, it evaluates the access list of all applicable dynamic translation rules (inside source translation rules for outbound packets and outside source translation rules for inbound packets) against the packet. If an access list permits translation, the NAT router tries to allocate an address from the associated address pool to install a new translation.

When you create dynamic translation rules, keep the following in mind:

- You can associate a list with one pool at any given time. Associating a list with a different pool replaces the previous association.
- The optional **overload** keyword for inside source translation specifies that the router employ NAT.
- You can configure dynamic NAT for inside source translation only; you cannot configure dynamic NAT for outside source translation.
- When no match occurs for any dynamic translation rule, the NAT router does not translate the packet.
- When an address pool is empty, the NAT router drops the packet.
- Access lists and pools do not have to exist when you are defining dynamic translation rules; you may create them after you define the dynamic translations.

Creating Dynamic Inside Source Translation Rules

Use the **ip nat inside source list** command to create a dynamic inside source translation rule. This command creates a translation rule that:

- Translates inside local source addresses to inside global addresses when packets from the inside network are routed to the outside network
- Translates outside local source addresses to outside global addresses when packets from the outside network are routed to the inside network.
- Use the **overload** keyword to specify that the translation create NAT entries (protocol, port, and address) in the NAT table.

The **no** version of this command removes the dynamic translation rule, but does not remove any previously created translations (resulting from the rule evaluation) from the translation table. To remove active translations from the translation table, see [“Clearing Dynamic Translations” on page 76](#).

ip nat inside source list

- Use to create dynamic translation rules that specify when to create a translation for a source address when routing a packet from the inside network to the outside network.
- Example

```
host (config) #ip nat inside source list translation1 pool pool1
```

- Use the **overload** keyword to specify that the translation create extended entries (protocol, port, and address) in the translation table for NAT.
- Use the **no** version to remove the dynamic translation rule; this command does not remove any dynamic translations from the translation table.
- See `ip nat inside source list`.

Creating Dynamic Outside Source Translation Rules

Use the **ip nat outside source list** command to create a dynamic outside source translation rule. This command dynamically translates outside global source addresses to outside local addresses when packets are routed from the outside network to the inside network (and *untranslates* the destination address when a packet returns before a translation table entry times out).

The **no** version of this command removes the dynamic translation rule, but does not remove any previously created translations from the translation table. To remove active translations from the translation table, see [“Clearing Dynamic Translations” on page 76](#).

ip nat outside source list

- Use to create dynamic translation rules that specify when to create a translation for a source address when routing a packet from the outside network to the inside network.
- Example

```
host (config) # ip nat outside source list translation1 pool pool1
```
- Use the **no** version to remove the dynamic translation rule; this command does not remove any dynamic translations from the translation table.
- See `ip nat outside source list`.

Defining Translation Timeouts

The router removes unused dynamic translations in the translation table. Use the **ip nat translation** command to change or disable NAT translation timeouts.

You can set the aging time (in seconds) never) for any of the specified timers:

- **timeout**—Dynamic simple translations (not for overloaded translations); default is 86400 seconds (24 hours).
- **dns-timeout**—DNS-created protocol translations; default is 120 seconds. These dynamic translations are installed by the DNS but not yet used; as soon as the translation is used, the router applies the timeout value mentioned above.
- **udp-timeout**—UDP protocol extended translations; default is 300 seconds (5 minutes).
- **tcp-timeout**—TCP protocol extended translations; default is 86400 seconds (24 hours).
- **finrst-timeout**—TCP connections terminated with reset (RST) or bidirectional finished (FIN) flags; default is 120 seconds. This timeout applies only to TCP extended

translations. The timer removes unused, closed TCP translations, which allows for retransmissions.

- **icmp-timeout**—ICMP protocol extended translations; default is 300 seconds (5 minutes).
- **gre-timeout**—Aging time for GRE protocol translations; default value is 300 seconds (5 minutes)

All timeouts for this command support a maximum value of 2147483 seconds (about 25 days).

The **no** version of this command resets the timer to its default value.

ip nat translation

- Use to change translation timeouts for existing and newly created translations in the translation table.
- All timeouts for this command support a maximum value of 2147483 seconds (about 25 days).
- Example

```
host1 (config) # ip nat translation timeout 23200
```

- Use the **no** version to reset the timer to its default value.
- See *ip nat translation*.

Clearing Dynamic Translations

Use the **clear ip nat translation** command to clear dynamic translations from the NAT translation table. You can remove all dynamic translations from the translation table or restrict the removal of translation entries based on the protocol, address, or port values.

clear ip nat translation

- Use to clear dynamic translations from the NAT translation table.
- Use an asterisk (*) in the **clear ip nat translation** version of this command to clear all dynamic translations from the translation table.
- Use an asterisk (*) in the **clear ip nat translation { gre | icmp | tcp | udp } inside insideGlobalIpAddress * insideLocalIpAddress *** version of this command to match any global or local port and remove inside source extended GRE, ICMP, TCP, or UDP translations for the specified global IP address and local IP address.

- Example 1—Clear all dynamic translations

```
host1 #clear ip nat translation*
```

- Example 2—Clear a specific port translation

```
host1 #clear ip nat translation tcp inside 171.69.68.10 10.1.2.3 55
```

- There is no **no** version.
- See clear ip nat translation.

NAT Configuration Examples

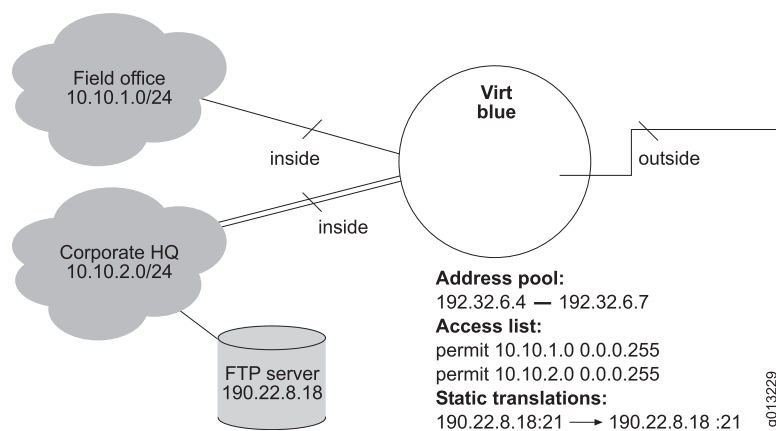
This section contains NAT configuration examples for a single virtual router configuration and NAT translation between two virtual routers.

NAPT Example

Figure 6 on page 77 illustrates a NAPT configuration for a private network with two inside subnetworks, a field office, and a corporate office.

Both offices use private addresses. The corporate office has a dual T-3 link and a public FTP server that has a global address (that is, it does not need translation).

Figure 6: NAPT Example



The address pool consists of three addresses (the number of addresses is small, because NAPT is used). Addresses matching the private address spaces of the corporate and field subnetworks are translated to global addresses from the pool through NAPT.

To configure this example:

1. Enter the correct virtual router context.

```
host1(config)#virtual-router blue
```

2. Mark the inside interfaces.

- a. Mark the field office:

```
host1:blue(config)#interface serial 2/1:1/1
host1:blue(config-interface)#ip nat inside
host1:blue(config-interface)#exit
```

- b. Mark the two corporate T-3 links:

```
host1:blue(config)#interface serial 1/1
host1:blue(config-interface)#ip nat inside
host1:blue(config-interface)#exit
```

```

host1:blue(config)#interface serial 1/2
host1:blue(config-interface)#ip nat inside
host1:blue(config-interface)#exit

```

3. Mark the outside interface.

```

host1:blue(config)#interface gigabitEthernet 3/0.1
host1:blue(config-interface)#ip nat outside
host1:blue(config-interface)#exit

```

4. Create a static NAT translation for the FTP server on the corporate network.

```

host1:blue(config)#ip nat inside source static tcp 190.22.8.18 21 190.22.8.18 21

```

5. Create the address pool for dynamic translations.

```

host1:blue(config)#ip nat pool corpxyz 192.32.6.4 192.32.6.7 prefix-length 24

```

6. Create the access list for addresses eligible for dynamic translation.

```

host1:blue(config)#access-list justcorp permit 10.10.1.0 0.0.0.255
host1:blue(config)#access-list justcorp permit 10.10.2.0 0.0.0.255

```

7. Create the NAT dynamic translation rule.

```

host1:blue(config)#ip nat inside source list justcorp pool corpxyz overload

```

8. Configure a default route to the outside interface.

```

host1:blue(config)#ip route 0.0.0.0 0.0.0.0 gigabitEthernet 3/0.1

```

9. Configure a null route for the inside global addresses to prevent routing loops when no matching translation exists.

```

host1:blue(config)#ip route 192.32.6.0 255.255.255.248 null 0

```



NOTE: Null route applies to 192.32.6.0–192.32.6.3, which do not exist in the address pool

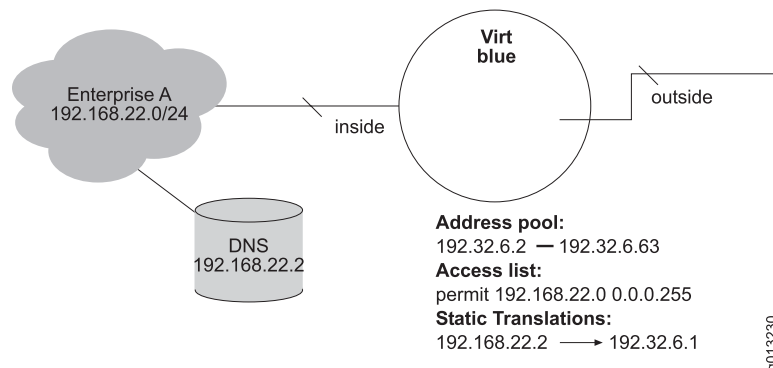
All hosts that use private addresses in both the field office and the corporate office must have their addresses translated to one of the three addresses in the pool. Because this example uses NAT, the interface can use only one pool address, depending on the number of inside hosts attempting to access the outside at any given time.

Bidirectional NAT Example

Figure 7 on page 79 illustrates how outside hosts can initiate conversations with inside hosts through the use of a DNS server that resides on the inside network.

The inside realm uses basic NAT. The inside network uses a mix of private subnetwork address space (192.168.22/24) and registered public addresses.

Figure 7: Bidirectional NAT Example



To configure this example:

1. Enter the correct virtual router context.

```
host1(config)#virtual-router blue
```

2. Mark the inside interface.

```
host1:blue(config)#interface serial 1/1:1/1
host1:blue(config-interface)#ip nat inside
host1:blue(config-interface)#exit
```

3. Mark the outside interface.

```
host1:blue(config)#interface gigabitEthernet 3/0.1
host1:blue(config-interface)#ip nat outside
host1:blue(config-interface)#exit
```

4. Create the translation for the DNS.

```
host1:blue(config)#ip nat inside source static 192.168.22.2 192.32.6.1
```

5. Create the address pool for dynamic translations.

```
host1:blue(config)#ip nat pool entA192 192.32.6.2 192.32.6.63 prefix-length 24
```

6. Create the access list for addresses eligible for dynamic translation (that is, private addresses).

```
host1:blue(config)#access-list entA permit 192.168.22.0 0.0.0.255
```

7. Create the dynamic translation rule.

```
host1:blue(config)#ip nat inside source list entA pool entA192
```

8. Configure a default route to the outside interface.

```
host1:blue(config)#ip route 0.0.0.0 0.0.0.0 gigabitEthernet 3/0.1
```

9. Configure a null route for the inside global addresses, to prevent routing loops when no matching translation exists.

```
host1:blue(config)#ip route 192.32.6.0 255.255.255.192 null 0
```



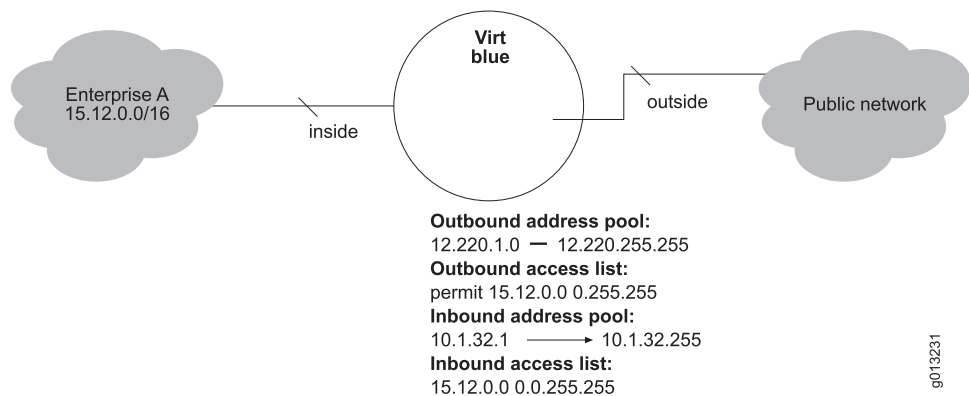
NOTE: Null route applies to 192.32.6.0 and 192.32.6.1, which do not exist in the address pool.

Twice NAT Example

Twice NAT is often useful when the inside network is using a nonprivate address space (unregistered usage of global address space) and you want it to connect to the public network. Inside local addresses need to be translated to legal global addresses. Legal addresses from the outside that overlap those used on the inside network need to be translated to unused and recognizable addresses in the inside network. Both inside source and outside source translations must be configured on the NAT router.

Figure 8 on page 80 illustrates how the inside network is using the unregistered global address space of 15.12.0.0/16. Outside hosts whose addresses overlap with this subnetwork that want to access the inside network need their global addresses translated.

Figure 8: Twice NAT Example



To configure this example:

1. Enter the correct virtual router context.

```
host1(config)#virtual-router blue
```

2. Mark the inside interface.

```
host1:blue(config)#interface fast-ethernet 6/1
host1:blue(config-interface)#ip nat inside
host1:blue(config-interface)#exit
```

3. Mark the outside Interface.

```
host1:blue(config)#interface atm 3/0.20
host1:blue(config-interface)#ip nat outside
host1:blue(config-interface)#exit
```

4. Create the address pool for inside source translations.

```
host1:blue(config)#ip nat pool entAoutpool 12.220.1.0 12.220.255.255 prefix-length
16
```



NOTE: This pool is purposely smaller than the size of the company network because not all private hosts are likely to access the public network at the same time.

5. Create the access list for addresses eligible for dynamic translation.

```
host1:blue(config)#access-list entAout permit 15.12.0.0 0.0.255.255
```

6. Create the dynamic translation rule for outbound traffic.

```
host1:blue(config)#ip nat inside source list entAout pool entAoutpool
```

7. Create the address pool for outside source translations.

Using an address range of 10.1.32.0/8 prevents any overlap with the private network (15.12.0.0/16).

```
host1:blue(config)#ip nat pool entAinpool 10.1.32.1 10.1.32.255 prefix-length 16
```



NOTE: This pool is purposely small, allowing for only a few connections.

8. Configure the access list for global addresses that overlap with inside addresses.

```
host1:blue(config)#access-list entAin permit 15.12.0.0 0.0.255.255
```

9. Create the dynamic translation rule for inbound traffic.

```
host1:blue(config)#ip nat outside source list entAin pool entAinpool
```

10. Create one of the following:

- A route to the outside interface for inside hosts to access outside hosts that have overlapping addresses.

```
host1:blue(config)#ip route 10.1.32.0 255.255.255.0 atm 3/0.1
```



NOTE: An inside host cannot directly access hosts on the outside network that use addresses that overlap with the inside subnetwork. However, by using outside source translation and DNS name resolution, the NAT router can install translations so inside hosts can access these outside hosts by using nonoverlapping addresses.

- A default route to the outside interface.

```
host1:blue(config)#ip route 0.0.0.0 0.0.0.0 atm 3/0.1
```

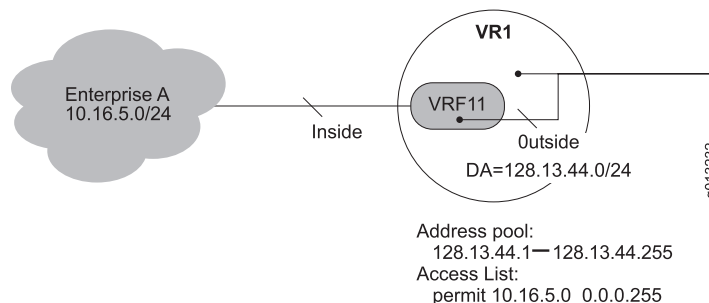
11. Configure a null route for the inside global addresses to prevent routing loops when no matching translation exists.

```
host1:blue(config)#ip route 12.220.1.0 255.255.0.0 null 0
```

Cross-VRF Example

In MPLS VPN configurations, you might want to offer public Internet access to VPN subscribers. MPLS VPNs are enabled through the use of VRFs. If a VPN is using a private or overlapping address space, you can use NAT to enable access to the public network because the NAT implementation is both VR and VRF aware. [Figure 9 on page 82](#) illustrates how the subscriber interface feature of the router is used in conjunction with NAT to connect the VPNs to the public network.

Figure 9: Cross-VRF Example



VRF11 is the local (this PE) representation of the MPLS VPN and connects enterpriseA to the VPN. Enterprise A communicates to VRFs in other PE devices (the rest of the VPN) through RFC2547bis (MPLS VPNs). VRF1, of which the VRF is administratively a member, represents the public network. The interface to EnterpriseA is marked as an inside interface. The normal steps for configuring inside source translation are applied. A subscriber interface is created off the uplink to the core network and anchored in the VRF. A DA-based demultiplexer matching the inside global address range is configured on the subscriber interface. The subscriber interface is marked as an outside interface.

To configure this example:

1. Enter the correct virtual routing and forwarding instance.


```
host1(config)#virtual-router vr1:vrf11
```
2. Mark the inside interfaces.


```
host1:vr1:vrf11(config)#interface fast-ethernet 6/1
host1:vr1:vrf11 (config-interface)#ip nat inside
host1:vr1:vrf11 (config-interface)#exit
```
3. Set the primary interface to DA-type demultiplexer (for subsequent shared interfaces).


```
host1:vr1(config)#interface atm 12/0.101
host1:vr1(config-interface)#ip demux-type da-prefix
host1:vr1(config-interface)#exit
```
4. Create the address pool for dynamic translations.


```
host1:vr1(config)#virtual-router vr1:vrf11
host1:vr1:vrf11(config)#ip nat pool entApool 128.13.44.0 128.13.44.255 prefix-length
24
```
5. Create the access list for addresses eligible for dynamic translation.


```
host1:vr1:vrf11(config)#access-list entA permit 10.16.5.0 0.0.0.255
```
6. Create the dynamic translation rule.


```
host1:vr1:vrf11(config)#ip nat inside source list entA pool entApool
```
7. Create the subscriber interface off the uplink.


```
host1:vr1:vrf11(config)#interface ip vrf11vr1
host1:vr1:vrf11(config-interface)#ip share-interface atm 12/0.101
host1:vr1:vrf11(config-interface)#ip unnumbered loopback 1
```

8. Configure a group of destination prefixes with which the device can communicate on the public network.

```
host1:vr1:vrfl1(config-interface)#ip destination-prefix 128.13.44.0 255.255.255.0
```

9. Mark the subscriber interface as outside.

```
host1:vr1:vrfl1(config-interface)#ip nat outside
host1:vr1:vrfl1(config-interface)#exit
```

10. Point the default route to the shared interface.

```
host1:vr1:vrfl1(config)#ip route 0.0.0.0 0.0.0.0 ip vrfl1vr1
```

11. Install a null route to avoid routing loops to the inside global address.

```
host1:vr1:vrfl1(config)#ip route 128.13.44.0 255.255.255.0 null 0
```

Tunnel Configuration Through NAT Examples

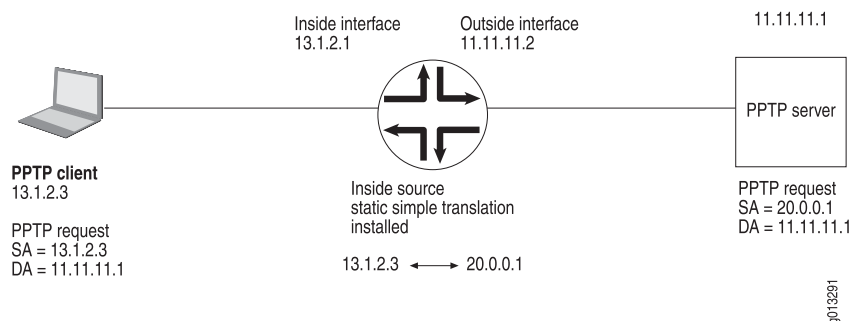
PPTP uses enhanced GRE encapsulation for PPP payloads. After the PPTP tunnel setup process, PPP packets are exchanged using GRE encapsulation. It is critical that a NAT device that resides between PPTP client and PPTP server allow GRE flows.

This section contains NAT configuration examples for both inside and outside PPTP tunnel setup through NAT.

Clients on an Inside Network

In this example, a subscriber on the inside network is initiating PPTP tunnels to a PPTP server located in the outside network. The PPTP connection to the server traverses an E Series router that has NAT enabled.

Figure 10: PPTP Tunnels on an Inside Network



The router has installed an inside source static simple translation in its translation table as follows:

Inside Local Address	Inside Global Address
13.1.2.3	20.0.0.1

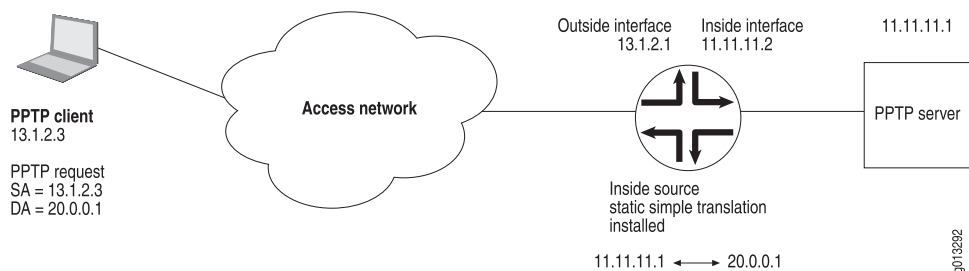
The PPTP client initiates its tunnels to the server at 11.11.11.1. The E Series router translates the SA from inside local 13.1.2.3 to inside global SA 20.0.0.1. Because GRE traffic can pass

through NAT, all matching PPTP control packets are translated and forwarded to the destination.

Clients on an Outside Network

In this example, an outside subscriber initiates PPTP tunnels to a PPTP server located in the service provider network. The PPTP connection to the server traverses an E Series router that has NAT enabled.

Figure 11: PPTP Tunnels on an Outside Network



The router has installed an inside source static simple translation in its translation table as follows:

Inside Local Address	Inside Global Address
11.11.11.1	20.0.0.1

The PPTP client initiates its tunnels to the inside global address 20.0.0.1. The E Series router translates packets destined for address 20.0.0.1 and forwards them to the inside local address of 11.11.11.1. Because GRE traffic can pass through NAT, all matching PPTP control packets are translated and forwarded to the destination.

GRE Flows Through NAT

Because PPTP requires the use of GRE flows, the examples in the previous section also work for any GRE traffic flows that traverse NAT.

GRE flows can terminate at an E Series router if NAT is or is not enabled. When the router receives locally terminating inbound GRE packets, the router transmits the packets to the tunnel server module for GRE processing. If the packets require translating, they are again sent through the tunnel server module.



NOTE: Only inner IP headers are translated for terminating GRE flows; outer IP headers are never translated.

For outbound GRE packets, the process works in reverse. If the packets require translation, the router transmits the packets to the tunnel server module for translation. If the packets are destined for a GRE tunnel, they are again sent through the tunnel server module where an outer header is prepended to the packet and the packet is then sent to the appropriate GRE tunnel.

Monitoring NAT

This section explains how to view NAT license information, NAT statistics, NAT translation entries, NAT address pool information, and NAT inside and outside rule settings.

Displaying the NAT License Key

The **show license nat** command displays the NAT license key.

show license nat

- Use to display the NAT license key configured on the router.
- Example

```
host1#show license nat
Nat license is nat_license
```

- See show license.

Displaying Translation Statistics

The **show ip nat statistics** command displays internal statistics that apply to NAT operation.

show ip nat statistics

- Use to display internal NAT statistics.
- Field descriptions
 - Last dynamic allocation failure—Completion level of any dynamic allocation failures; the number of times the router attempted dynamic allocation but reached the dynamic allocation entry limit
 - Current static translation entries
 - Inside Source Simple—Number of inside source simple static translations
 - Outside Source Simple—Number of outside source simple static translations
 - Inside Source Extended—Number of inside source extended static translations
 - Outside Source Extended—Number of outside source extended static translations
 - Dynamic Translation Type—Type of dynamic translation (inside source simple, outside source simple, inside source extended)
 - Current—Current number of dynamic translations of the associated translation type
 - Peak—Peak number of dynamic translations of the associated translation type
 - Accumulated—Accumulated number of dynamic translations of the associated type; this value reflects the accumulation of dynamic translations since the last router reboot operation

- Failed—Total number of installation attempts that failed for an associated translation type
- Forwarding statistics for packets received on inside or outside interfaces
 - forwarded directly—Number of packets forwarded directly (that is, without the need of translation)
 - forwarded through translator—Number of packets forwarded through the NAT translator
 - discarded—Number of packets discarded immediately upon receipt
 - discarded by translator—Number of packets discarded by the NAT translator when no matching translation could be located
- Example

```
host1#show ip nat statistics
```

```
NAT database statistics for virtual router vr1:
```

```
-----
Last dynamic allocation failure: normal, successful completion
Dynamic entry limit was reached 10318 times
```

```
Current static translation entries:
```

```
-----
Inside Source Simple:          10
Outside Source Simple:         3
Inside Source Extended:        8
Outside Source Extended:       12
```

Dynamic Translation Type	Current	Peak	Accumulated	Failed
Inside Source Simple	69999	69999	69999	12568
Outside Source Simple	4518	4518	4518	25
Inside Source Extended	70000	70000	70000	568
Fully Extended	26855	26855	26855	2565

```
Forwarding statistics for virtual router vr1:
```

```
-----
Packets received on inside interface and
  forwarded directly          8
  forwarded through translator 111763104
  discarded                   2
  discarded by translator     28524565
```

```
Bytes received on inside interface and
  forwarded directly          544
  forwarded through translator 5141098074
```

```
Packets received on outside interface and
  forwarded directly          7
  forwarded through translator 1031624
  discarded                   3
  discarded by translator     578961
```

```
Bytes received on outside interface and
  forwarded directly          476
```

forwarded through translator 47454704

- See show ip nat statistics.

Displaying Translation Entries

The **show ip nat translations** command displays current translations that reside in the translation table.

Simple translation entries appear with inside/outside and local/global address information. Extended entries appear with added protocol and port numbers (or query IDs).

Using verbose mode additionally provides the time since creation and time since last use for each translation entry.

show ip nat translations

- Use to display current translations that reside in the NAT translation table.
- Field descriptions
 - Prot—Protocol (TCP, UDP, ICMP, or GRE) for this translation entry; this field appears only for extended table entries
 - Inside local—Inside local IP address for this translation entry; this field also provides the port number, separated by a colon (:) for extended entries
 - Inside global—Inside global IP address for this translation entry; this field also provides the port number, separated by a colon (:) for extended entries
 - Outside global—Outside global IP address for this translation entry; this field also provides the port number, separated by a colon (:) for extended entries
 - Outside local—Outside local IP address for this translation entry; this field also provides the port number, separated by a colon (:) for extended entries
 - Time since creation—Amount of time elapsed since the translation entry appeared in the translation table
 - Time since last use—Amount of time elapsed since the translation entry was used
- Example 1

```
host1# show ip nat translations
Prot    Inside local  Inside global  Outside global  Outside local
----    -
GRE     13.1.2.1:*    20.0.0.1:*    ---            ---
ICMP    13.1.2.2:4    20.0.0.2:4    ---            ---
TCP     13.1.2.3:20   20.0.0.3:50   ---            ---
```



NOTE: Because they are not NAT translations, port numbers for GRE translations appear as asterisks (*).

- Example 2

```
host1# show ip nat translations verbose
```

Prot	Inside local	Inside global	Outside global	Outside local	Time since creation	Time since last use
	20.0.0.3	30.0.0.3	---	---	00:04:50	00:00:01
	21.0.0.3	30.208.0.3	---	---	00:02:12	00:00:01
	21.0.0.4	30.208.0.4	---	---	00:02:12	00:00:01
	---	---	50.0.0.3	70.0.0.3	00:03:24	Never
	---	---	51.0.0.3	70.208.0.3	00:01:44	00:00:01
	---	---	51.0.0.4	70.208.0.4	00:01:44	00:00:01
UDP	---	---	50.50.0.3:87	70.50.0.3:87	00:03:10	Never
UDP	22.0.0.4:63	30.224.0.3:4097	---	---	00:02:12	00:00:01
UDP	22.0.0.3:63	30.224.0.3:4096	---	---	00:02:12	00:00:01
TCP	---	---	50.50.0.3:80	70.50.0.3:80	00:03:10	Never
UDP	20.50.0.3:87	30.50.0.3:87	---	---	00:03:35	Never

- See show ip nat translations.

Displaying Address Pool Information

The **show ip nat pool** command displays NAT address pool information. The command output displays configuration (mask and address ranges) of all address pools, unless you supply a specific pool name.

show ip nat pool

- Use to display NAT address pool information.
- Field descriptions
 - pool—Name of the address pool
 - netmask—Network prefix associated with the NAT address pool

- prefix length—Prefix length associated with the NAT address pool
- range—Address ranges used by this NAT address pool
- Example 1

```
host1#show ip nat pool
```

```
pool: pool1 netmask: 255.255.255.0 prefix length: 24
      range: 3.3.3.1 to 3.3.3.255
      range: 4.4.4.1 to 4.4.4.32
```

```
pool: pool2 netmask: 255.255.255.0 prefix length: 24
      range: 1.1.1.1 to 1.1.1.24
      range: 2.2.2.1 to 2.2.2.55
```

- Example 2

```
host1#show ip nat pool pool1
```

```
pool: pool1 netmask: 255.255.255.0 prefix length: 24
      range: 3.3.3.1 to 3.3.3.255
      range: 4.4.4.1 to 4.4.4.32
```

- See show ip nat pool.

Displaying Inside and Outside Rule Settings

The **show ip nat inside rule** and **show ip nat outside rule** commands display access list and pool usage for all dynamic translation rules configured for the virtual router. If you do not specify an access list, the output displays address pool associations for each of the access lists for either inside or outside translation rules in the virtual router. Specifying an access list filters the output to display only the address pool associated with the specified list.

show ip nat inside rule

- Use to display NAT access list and pool usage information for inside source translation rules.
- Field descriptions
 - access list name—Name of the access list
 - pool name—Name of the address pool
 - rule type—Type of rule assigned
- Example

```
host1#show ip nat inside rule
```

```
access list name: list1 pool name: poolA rule type: inside source
access list name: list2 pool name: poolB rule type: inside source
access list name: list3 pool name: poolC rule type: inside source overload
```

- See show ip nat inside rule.

show ip nat outside rule

- Use to display NAT access list and pool usage information for outside source translation rules.
- Field descriptions
 - access list name—Name of the access list
 - pool name—Name of the address pool
 - rule type—Type of rule assigned
- Example

```
host1#show ip nat outside rule
access list name: list4 pool name: poolD rule type: outside source
```
- See show ip nat outside rule.

CHAPTER 3

Configuring J-Flow Statistics

This chapter describes how to configure J-Flow statistics on your ERX router; it contains the following sections:

- [Overview on page 91](#)
- [Platform Considerations on page 94](#)
- [Before You Configure J-Flow Statistics on page 94](#)
- [Configuring Flow-Based Statistics Collection on page 94](#)
- [Monitoring J-Flow Statistics on page 101](#)

Overview

The JunosE J-Flow feature provides a method by which you can collect IP traffic flow statistics on your routing devices. J-Flow does not require any special protocol for connection setup. It also does not require any external changes to networked traffic, packets, or any other devices in the network. In other words, J-Flow is transparent to the existing network, including end stations and application software and network devices such as LAN switches.

The JunosE implementation of J-Flow allows you to export data to the UDP port of a remote workstation for data collection and further processing. In addition, the ability to enable J-Flow on an individual virtual router, interface, or subinterface allows you to collect network statistics for specific locations within your network.

Interface Sampling

For any given IP interface, enabling J-Flow causes packets from the input stream to be sampled at a globally configured rate. For each packet sampled, the main flow cache is examined to see if there is an existing entry. If no entry exists, J-Flow creates a new entry and records attributes of the flow. If the packet matches an existing entry, J-Flow updates the existing flow.

In general, the system samples packets that it can forward. In other words, the system does not sample packets that it discards. As sampling occurs, the system records flow characteristics as they would appear for a packet that the virtual router transmits. This means, for example, that if a packet uses the address of an output interface or next-hop value altered by a policy setting, the system records the altered value in the flow record.

Aggregation Caches

Data from flow cache entries is summarized to build aggregated views or aggregation caches. Aggregation caches are created and maintained along with the main cache. Aggregation caches have their own history area where the aging aggregation cache records are collected. Aggregation caches have a set of configuration parameters: number of entries, active and inactive time out, and export destination.

Types of aggregation caches include:

- AS-Aggregates flow data based on source and destination AS, and ingress and egress interface values.
- Destination Prefix-Aggregates flow data based on the destination address, mask, destination AS, and egress interface.
- Prefix-Aggregates flow data based on source prefix, destination prefix, source mask, destination mask, source AS, destination AS, ingress interface, and egress interface.
- Protocol Port-Aggregates flow data based on protocol, source port, and destination port.
- Source Prefix-Aggregates flow data based on source address, source mask, source AS, and ingress interface.

Aggregation caches contain a subset of the fields collected in the raw flow data. For example, TCP flags, Next Hop Address, and ToS values are not maintained in any of the aggregation caches. Unlike the main cache, aggregation caches are not enabled by default.

Flow Collection

The JunosE J-Flow functionality allows statistics collection at the VR/VRF level. This means that each virtual router (VR)/VPN routing and forwarding (VRF) table has its own main cache for statistics gathering.

Although you can export flow statistics only at the VR level, VRF data is rolled up for each VR. The reason for supporting export flow at the VR level is that existing export formats cannot discriminate between VRs and VRFs. However, even though export formats do not allow for segregation, the JunosE CLI commands do. Segregating each collection by VR removes any ambiguity and aliasing that may occur with overlapping address spaces (as may occur in virtual private network [VPN] configurations).

Main Flow Cache Contents

The following 7-tuple distinguishes an entry in the flow cache for a VR:

- Source IP address (SA)
- Destination IP address (DA)
- Source port number (SP)
- Destination port number (DP)

- Layer 3 protocol type
- Type of service (ToS byte) or Differentiated Services code point (DSCP)
- Input interface

Cache Flow Export

Using UDP as the transport method, the ERX router can export the content of the flow cache as the system removes the entries. You can specify one export destination for each VR.

Each export packet contains a header and flow records. The version 5 header contains the following fields:

- Version-Format version
- Count-Number of records in this packet
- SysUpTime-System up time value when this packet was built
- Unix Timestamp-Number of seconds and nanoseconds since 0000 UTC 1970 (Coordinated Universal Time)
- Sequence Number-Number of total records sent on this export stream
- Engine type-Type of switching engine (line module or route processor)



NOTE: The J-Flow setting for Engine type is always RP=0.

- Engine ID-SRP slot number

If, for any reason, the virtual router is unable to export records to the collector, the unsent records are discarded. However, the virtual router continues to increase the sequence number by one as if it sent the records. Discrepancies between the sequence number and sent records can assist in recognizing discontinuities at the collector end.

Aging Flows

After the virtual router creates a flow in the cache, the flow is removed at the expiration of either the active or the inactive timer.

In sampled environments, methods for detecting the end of a flow can be unreliable. The active timer places a hard limit on how long a flow may last before the virtual router closes it and gathers the necessary statistics. If the flow is still active when the active timer expires, the virtual router creates a new flow entry to replace the closed flow.

The inactive timer removes flows if they do not contain any data traffic for a specified period of time.

Operation with NAT

When functioning with Network Address Translation (NAT), J-Flow sampling occurs before NAT applies any translation.

Operation with High Availability

When high availability is enabled, the following occurs in the event of a switchover:

- Any flows that are collected but not exported off of the router are lost.
- Flow history is lost.
- Counters are reset to zero.

After the standby SRP becomes active, and all other applications indicate that they have recovered, sampling and flow-collecting resume.

Platform Considerations

For information about modules that support J-Flow statistics on ERX14xx models, ERX7xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support NAT.

For information about modules that support J-Flow on the E120 and E320 Broadband Services Routers:

- See *E120 and E320 Module Guide, Table 1, Modules and IOAs* for detailed module specifications.
- See *E120 and E320 Module Guide, Appendix A, IOA Protocol Support* for information about the modules that support J-Flow.

Before You Configure J-Flow Statistics

Before you configure J-Flow statistics, be sure you have created IP interfaces from which J-Flow can extract traffic flow information. For information about configuring IP interfaces, see *Configuring IP* in *JunosE IP, IPv6, and IGP Configuration Guide*.

Configuring Flow-Based Statistics Collection

To configure J-Flow on a virtual router:

1. Enable J-Flow statistics.
2. Enable J-Flow statistics on the desired interfaces.
3. (Optional) Define the sampling interval at which you want to collect statistics.
4. (Optional) Customize the size of the main flow cache.
5. (Optional) Define flow cache aging timers.
6. (Optional) Specify to where you want to export J-Flow statistics.

Enabling Flow-Based Statistics

Use the **ip flow statistics** command to explicitly enable J-Flow.



NOTE: Issuing any configuration-level commands implicitly enables J-Flow.

ip flow statistics

- Use to enable J-Flow.
- Example

```
host1(config)#ip flow statistics
```
- Use the **no** version to disable J-Flow on the virtual router.
- See ip flow statistics.

Enabling Flow-Based Statistics on an Interface

Use the **ip route-cache flow sampled** command to enable J-Flow statistics on an interface. You can also use this command to configure an IP profile that is applied to dynamically created IP interfaces. This feature provides J-flow capability on all dynamically created IP interfaces, including those used for MPLS-to-IP forwarding scenarios.



NOTE: Issuing an interface-level flow command does not enable J-Flow on the virtual router. To enable J-Flow, issue the **ip flow statistics** command.

ip route-cache flow sampled

- Use to enable J-Flow on an interface, or in an IP profile for dynamically created IP interfaces.
- Examples

```
host1(config-if)#ip route-cache flow sampled
```

or

```
host1(config-profile)#ip route-cache flow sampled
```
- Use the **no** version to disable J-Flow statistics on the interface.
- See ip route-cache flow sampled.

Defining a Sampling Interval

Use the **ip flow-sampling-mode packet-interval** command to define the packet-sampling interval for the virtual router. The sampling interval specifies the rate at which the virtual router samples J-Flow information. This rate is used for all interfaces that have J-Flow enabled. After you enable J-Flow on an interface, the virtual router samples one packet

at the specified packet interval. You can specify an interval in the range 1–4,000,000,000 packets.

When you use the **ip flow-sampling-mode packet-interval** command to define the packet-sampling interval for Gigabit Ethernet interfaces configured on the ES2 10G LM (line module) with either the ES2-S1 GE-8 IOA or the ES2-S2 10GE PR IOA on E120 routers and E320 routers, the J-Flow application makes the following internal adjustments to achieve better performance on the ES2 10G LM, regardless of the packet-sampling interval that you configure:

- J-Flow adjusts the maximum sampling interval to 8,388,608, which is the decimal equivalent of 0x800000.
- J-Flow changes the packet-sampling value to the closest integer that is a power of two and that is less than or equal to the configured value.

For performance reasons, J-Flow applies these adjustments to the sampling interval only for the interfaces configured on the ES2 10G LM on the virtual router. The configured sampling interval does not change for interfaces not configured on the ES2 10G LM on the virtual router.

When the data rate increases on a given interface, J-Flow packet sampling might not be able to maintain the configured sampling rate and might drop the intended sampled packets. If this occurs, you can address the issue by reducing the sampling rate.



NOTE: For all modules except the ES2 10G LM on the E120 router and the E320 router, packet sampling occurs individually for each processor. Because the router distributes packets over multiple processors, sampling occurs when each processor reaches the specified packet interval.

Even though each flow is sampled, the flow sample is not necessarily cached because of system constraints.

ip flow-sampling-mode packet-interval

- Use to define the J-Flow packet-sampling interval.
- Specify a packet-sampling interval in the range 1–4000000000 packets; the default value is 4000000000.
- Specifying an interval less than 10 sets a very high sampling rate that can severely degrade performance. The lower the packet-sampling interval you configure, the faster the sampling rate.
- For information about the effects of using the **ip flow-sampling-mode packet-interval** command for the ES2 10G LM with either the ES2-S1 GE-8 IOA or the ES2-S2 10GE PR IOA on E120 routers and E320 routers, see [“Defining a Sampling Interval” on page 95](#).
- Example—Samples 1 out of 50 packets from the line module on which the interface resides

```
host1(config)#ip flow-sampling-mode packet-interval 50
```


- Use the **no** version to return the sampling interval to its default value, 4 billion.
- See `ip flow-sampling-mode packet-interval`.

Setting Cache Size

Use the **ip flow-cache entries** command to limit the number of main flow cache entries for the virtual router (as collected across all line modules that are running J-Flow). After the cache size exceeds the flow-cache entry limit, the least recently used flow is removed.

The possible flow-cache range is 1,024 – 524,288 entries. The default value is 65,536 entries.

ip flow-cache entries

- Use to limit J-Flow main flow cache entries.
- Example

```
host1(config)#ip flow-cache entries 80000
```
- Use the **no** version to return the cache size to its default value, 65535.
- See `ip flow-cache entries`.

Defining Aging Timers

After the virtual router creates a flow in the cache, the virtual router can remove the flow at the expiration of either the active or the inactive timer.

Specifying the Activity Timer

Use the **ip flow-cache timeout active** command to specify a value for the activity timer. The activity timer measures the amount of time that the virtual router has been recording a datagram for a given flow. When this timer expires, the virtual router exports the flow cache entry from the cache and removes the entry. This process prevents active flows from remaining in the flow cache, and allows collected data to appear in a timely manner. The possible range for the activity timer value is 1 – 60 minutes. The default value is 30 minutes.

ip flow-cache timeout active

- Use to define the activity timer, in minutes.
- Example

```
host1(config)#ip flow-cache timeout active 50
```
- Use the **no** version to return the activity timer to its default value (30 minutes).
- See `ip flow-cache timeout`.

Specifying the Inactivity Timer

Use the **ip flow-cache timeout inactive** command to specify a value for the inactivity timer. The inactivity timer measures the length of time expired since the virtual router

recorded the last datagram for a given flow. When this timer expires, the virtual router exports the flow cache entry from the cache and removes it. When, at a later time, another datagram begins that uses the same flow characteristics, the virtual router allocates a new flow cache entry, and the inactivity timer begins again. The possible range for the inactivity timer value is 10 – 600 seconds. The default value is 15 seconds.

ip flow-cache timeout inactive

- Use to define the inactivity timer, in seconds.
- Example

```
host1(config)#ip flow-cache timeout inactive 90
```
- Use the **no** version to return the inactivity timer to its default value (15 seconds).
- See ip flow-cache timeout.

Specifying Flow Export

Use the **ip flow-export** command to specify the location to which you want to export the J-Flow datagrams.

ip flow-export

- Use to specify the location to which you want to export J-Flow datagrams or specify an alternate source address for outbound export J-Flow datagrams.
- Example 1-Specifies the destination address for J-Flow datagrams

```
host1(config)#ip flow-export 192.168.2.73 2055 version 5 peer-as
```
- Example 2-Specifies the source address for outbound export J-Flow datagrams

```
host1(config)#ip flow-export source fastEthernet 5/0
```
- Use the **no** version to remove the export setting.
- See ip flow-export.

Configuring Aggregation Flow Caches

Aggregation caches are disabled by default. Exporting flow records from the router does not occur while it is in the disabled state. When the configuration for an aggregation cache is changed from enabled to disabled state, all flow records from that cache are removed and flow collection stops.

For Prefix, Destination Prefix, and Source Prefix aggregation caches, you can specify a minimum source and destination mask size to affect the granularity of the IP address space captured in the aggregation cache. The commands to configure the minimum mask size for the source and destination address are issued in Flow Cache Configuration mode and are specific to each aggregation cache:

```
host1(config-flow-cache)#mask source minimum value
host1(config-flow-cache)#mask destination minimum value
```

The value (a number in the range 1–32) specifies the size of the minimum mask. The no version restores the default minimum mask size, which is 0. A mask of size N has the N most significant bits set in the corresponding bit mask.

You cannot configure a minimum mask size for aggregation caches that do not retain an IP address in their aggregation scheme (like the AS aggregation cache). You can configure the Prefix aggregation cache for both source and destination minimum mask size. You can configure only the source minimum mask size for the Source Prefix aggregation cache. You can configure only the destination minimum mask size for the Destination Prefix aggregation cache.

The peer/origin information configured with the export command for the man V5 cache is used to display the AS number of the AS aggregation cache for both the source and destination AS. If no (default) configuration is present, zero appears in the AS numbers for both V5 export and V8 export and in the **show** commands for the V8 AS aggregation cache.

Establish an aggregation cache:

1. Enter Flow Cache Configuration mode for the AS aggregation cache.

```
host1(config)#ip flow-aggregation cache as
```

2. Configure the number of entries (1024–524288) in the aggregation cache; the no version sets the number of entries back to its default value of 4096 (flow-data may be lost if the previous setting is larger than the default).

```
host1(config-flow-cache)#cache entries entryNumber
```

3. Set the active (1-60) and inactive (10-600) aging timers.

```
host1(config-flow-cache)#cache timeout active active-tmo
host1(config-flow-cache)#cache timeout inactive inactive-tmo
```

4. Configure an export destination for the aggregation cache; the no version removes the destination.

```
host1(config-flow-cache)#export destination { hostname | ip address }
udp-port-number
```

5. Set the source IP address for datagrams containing information from this cache: the no version removes the explicit setting of the source address.

```
host1(config-flow-cache)#export source interfacetype interface
```

6. Enable the aggregation cache.

```
host1(config-flow-cache)#enabled
```

The aggregation cache starts accumulating information from the flow cache; the no version stops the accumulation of information from the flow cache, but does not suspend the operation of the flow cache.

cache entries

- Use to set the number of entries in the aggregation cache.
- Example

host1(config-flow-cache)#cache entries 524288

- Use the **no** version to reset the number of entries to the default value 4096.
- See cache entries.

cache timeout

- Use to set the active and inactive timers.
- Example

host1(config-flow-cache)#cache timeout active 50

- Use the **no** version to reset the default value.
- See cache timeout.

enabled

- Use to enable the aggregation cache to accumulate information from the flow cache.
- Example

host1(config-flow-cache)#enabled

- Use the **no** version to stop the information flow from the flow cache.
- See enabled.

export destination

- Use to configure an export destination for the aggregation cache.
- Example

host1(config-flow-cache)#export destination myhost udp-port

- Use the **no** version to remove the destination.
- See export destination.

export source

- Use to configure an export source for the aggregation cache.
- Example

host1(config-flow-cache)#export source interface inf1

- Use the **no** version to remove the destination.
- See export source.

ip flow-aggregation cache

- Use to create an aggregation cache.
- Example

host1(config)#ip flow-aggregation cache

- Use the **no** version to remove the aggregation cache and its configuration.
- See ip flow-aggregation cache.

mask destination

- Use to set the minimum mask size for the destination address for the prefix and destination prefix aggregation caches.
- Example

```
host1(config-flow-cache)#mask destination 128
```
- Use the **no** version to restore the default mask size, which is 0.
- See mask destination.

mask source

- Use to set the minimum mask size for the source address for the prefix and source prefix aggregation caches.
- Example

```
host1(config-flow-cache)#mask source 60
```
- Use the **no** version to restore the default mask size, which is 0.
- See mask source.

Monitoring J-Flow Statistics

This section shows how to clear J-Flow statistics and use the **show** commands to view J-Flow settings and statistical results.

Clearing J-Flow Statistics

Use the **clear ip flow stats** command to clear all entries from all flow caches on the virtual router.

clear ip flow stats

- Use to clear entries from all flow caches on the VR/VRF.
- Example

```
host1(config)#clear ip flow stats
```
- There is no **no** version.
- See clear ip flow stats.

J-Flow show Commands

You can monitor the following aspects of J-Flow statistics by using the following commands:

To Display	Command
Main cache flow operational statistics	show ip cache flow

To Display	Command
J-Flow sampling state	show ip flow sampling
J-Flow export state and export statistics	show ip flow export

You can use the output filtering feature of the **show** command to include or exclude lines of output based on a text string that you specify. See *Command Line Interface in JunosE System Basics Configuration Guide*, for details.

show ip cache flow

- Use to display IP flow cache operational statistics.
- Field descriptions
 - Main Cache
 - Max Entries-Maximum number of entries allowed in the main cache
 - Activity Timeout-Activity timer value
 - Inactivity Timeout-Inactivity timer value
 - Size-Distribution of IP packets by size
 - Percent-Percent distribution of different-sized IP packets
 - Protocol - Port-Protocol of the sample and port destination for that sample
 - Total Flows-Total number of flows
 - Flows/Sec-Number of flows per second
 - Packets/Flow-Number of packets per flow
 - Bytes/Package-Number of bytes per packet
 - Packets/Sec—Number of packets per second
 - Src. Addr—Source address of sampled packets
 - Src. Intf—Source interface of sampled packets
 - Dst. Addr—Destination address of sampled packets
 - Dst. Intf—Destination interface of sampled packets
 - Summary
 - Total Flows Processed—Total number of flows processed
 - Total Packets—Total number of packets sampled
 - Total Bytes—Total number of bytes received
- Example 1—Brief output

```

host1#
show ip cache flow active brief
29140 packets sampled.
Distribution of IP packets by size.
  Size          Percent
-----
  1 - 32        0.000
    64          0.000
    96          0.000
   128          0.000
   160          0.000
   192          0.000
   224          0.000
   256          0.000
   288          0.000
   320          0.000
   352          0.000
   384          0.000
   416          0.000
   448          0.000
   480          0.000
   512          0.000
   544          0.000
   576          0.000
  1024         96.791
  1536          3.209
  2048          0.000
  2560          0.000
  3072          0.000
  3584          0.000
  4096          0.000
  4608          0.000

Protocol-Port    Total    Flows    Packets    Bytes    Packets
-----
TCP-telnet       1        0.000    118.000    1014.000  0.000
UDP-whois++      1        0.008    935.000    1026.000  7.664

----- Summary -----
Total Flows Processed: 2
Total Packets 1053
Total Bytes 1078962
-----

```

- Example 2—Detailed output



NOTE: The output format for this command was modified slightly to fit within the confines of this document.

```

host1# show ip cache flow active detail
Main Cache
Max Entries: 65536
Activity Timeout: 60 mins.
Inactivity Timeout: 600 secs.
Cache Enabled
32012 packets sampled.

```

Distribution of IP packets by size.

Size	Percent
-----	-----
1 - 32	0.000
64	0.000
96	0.000
128	0.000
160	0.000
192	0.000
224	0.000
256	0.000
288	0.000
320	0.000
352	0.000
384	0.000
416	0.000
448	0.000
480	0.000
512	0.000
544	0.000
576	0.000
1024	96.789
1536	3.211
2048	0.000
2560	0.000
3072	0.000
3584	0.000
4096	0.000
4608	0.000

Src.Addr	Src.Intf	Dst.Addr	Dst.Intf	Protocol Port	Packets /Flow	Bytes /Packet	Packets /Sec
-----	-----	-----	-----	-----	-----	-----	-----
10.20.30.41	258 GigE4/0	12.0.0.2	GigE2/0	TCP-telnet	58.000	1014.000	0.000
10.20.30.41	63 GE4/0	50.60.70.88		UDP-whois++	1028.000	1026.000	7.672
----- Summary -----							
Total Flows Processed: 2							
Total Packets 1086							
Total Bytes 1113540							

- Example 3—History output

```
host1# show ip cache flow history
```

```
35604 packets sampled.
```

Distribution of IP packets by size.

Size	Percent
-----	-----
1 - 32	0.000
64	0.000
96	0.000
128	0.000
160	0.000
192	0.000
224	0.000
256	0.000
288	0.000
320	0.000
352	0.000
384	0.000
416	0.000

	448	0.000				
	480	0.000				
	512	0.000				
	544	0.000				
	576	0.000				
	1024	96.784				
	1536	3.216				
	2048	0.000				
	2560	0.000				
	3072	0.000				
	3584	0.000				
	4096	0.000				
	4608	0.000				
Protocol	Port	Total Flows	Flows /Sec	Packets /Flow	Bytes /Pkt	Packets /Sec
-----		-----	-----	-----	-----	-----
TCP-telnet		216	1.450	159.264	1014.000	230.879
-----		Summary -----				
Total Flows Processed: 216						
Total Packets 34401						
Total Bytes 34882614						

- See show ip cache flow.

show ip cache flow aggregation

- Use to display IP flow cache operational statistics for an aggregation cache.
- Field descriptions
 - Aggregation Cache
 - AS—AS aggregation cache
 - Destination-prefix—Destination-prefix aggregation cache
 - Prefix—Prefix aggregation cache
 - Protocol-port—Protocol-port aggregation cache
 - Source-prefix—Source-prefix aggregation cache
 - Total Flows—Total number of flows
 - Flows/Sec—Number of flows per second
 - Packets/Flow—Number of packets per flow
 - Bytes/Package—Number of bytes per packet
 - Packets/Sec—Number of packets per second
 - Src. Addr—Source address of sampled packets
 - Src. Intf—Source interface of sampled packets
 - Dst. Addr—Destination address of sampled packets

- Dst. Intf—Destination interface of sampled packets
- Summary
 - Total Flows Processed—Total number of flows processed
 - Total Packets—Total number of packets sampled
 - Total Bytes—Total number of bytes received
- See show ip cache flow aggregation.

Example—Aggregation cache flow output

```
host1#show ip cache flow aggregation as active brief
29140 packets sampled.
```

Src.AS	Dst.AS	Total Flows	Packets /Flow	Bytes /Pkt	Packets /Sec
400	100	0.000	118.000	1014.000	0.000
100	400	0.008	935.000	1026.000	7.664

----- Summary -----
 Total Flows Processed: 2
 Total Packets 1053
 Total Bytes 1078962

show ip flow export

- Use to display configuration values for IP flow cache export.
- Example


```
host1#show ip flow export
Flow export is enabled using version 5 format.
Exporting to 10.0.0.2 port 9898 using source ip interface
GigabitEthernet5/0/0.
```
- See show ip flow.

show ip flow sampling

- Use to display configuration values for IP flow cache sampling.
- Example


```
host1#show ip flow sampling
Flow sampling is enabled
'Packet Interval' sampling mode is configured.
1 out of every 1000 packets is being sampled.
```
- See show ip flow.

CHAPTER 4

Configuring BFD

This chapter describes how to configure bidirectional forwarding detection (BFD) on your E Series router; it contains the following sections:

- [Bidirectional Forwarding Detection Overview on page 107](#)
- [BFD Platform Considerations on page 110](#)
- [BFD References on page 110](#)
- [Configuring a BFD License on page 111](#)
- [BFD Version Support on page 111](#)
- [Configuring BFD on page 112](#)
- [Managing BFD Adaptive Timer Intervals on page 112](#)
- [Clearing BFD Sessions on page 113](#)
- [Monitoring BFD on page 114](#)

Bidirectional Forwarding Detection Overview

Fast failure detection is a high priority feature for any network element. Some media, like Ethernet, do not provide remote end failure. Networks must often rely on internal gateway protocol (IGP) hello messages to detect any failure and, in some cases (for example, static routes), even these hello messages are not used.

IGP hellos have their own limitations—it often takes one second or more to detect a remote end failure and processing IGP hello messages takes precious processing time. BFD overcomes IGP detection time and processing limitations in detecting any data path failures.

When configured for protocols like OSPF and IS-IS, BFD employs rapid, periodic, and inexpensive hello messages to detect path activity. You can also configure BFD to function with static routes, combining with the BFD poll bit to detect path activity.

When configured for various protocols like OSPF and IS-IS, BFD employs rapid, periodic and inexpensive hello messages to detect path activity. You can also configure BFD to function with static routes, combining with the BFD poll bit to detect path activity.

You can also configure a BFD session with a BGP neighbor or peer group to determine relatively quickly whether the neighbor or peer group is reachable. For information about

configuring BFD for EBGp routes, see *Configuring BGP Routing in JunosE BGP and MPLS Configuration Guide*.

How BFD Works

In a BFD-configured network, when a client launches a BFD session with a peer, BFD begins sending slow, periodic BFD control packets that contain the interval values that you specified when you configured the BFD peers. This is known as the initialization state and BFD does not generate any up or down notifications in this state.

When another BFD interface acknowledges the BFD control packets, the session moves into an up state and begins to more rapidly send periodic control packets.

If a data path failure occurs and BFD does not receive a control packet within the configured amount of time, the data path is declared down and BFD notifies the BFD client. The BFD client can then perform the necessary actions to reroute traffic. This process can be different for different BFD clients. All BFD-configured IGP clients (like IS-IS, OSPF, PIM, and RIP) launch BFD sessions when they detect neighbors through their own hello protocols. However, a static BFD client launches a BFD session when it detects that its next hop is resolved.

The BFD Admin Down state is used to bring down a BFD session administratively, to protect client applications from BFD configuration removal, license issues, and clearing of BFD sessions. When BFD enters the Admin Down state, BFD notifies the new state to its peer for a failure detection time and after the time expires, the client stops transmitting packets. For the Admin Down state to work, the peer, which receives the Admin Down state notification, must have the capability to distinguish between administratively down state and real link down. A BFD session moves to the Admin Down state under the following conditions:

- When a BFD configuration is removed for the last client tied to a BFD session, BFD moves to the Admin Down state and communicates the change to the peer to enable the client protocols to handle this in a seamless manner without going down.
- When a BFD license is removed on the client, it moves to the Admin Down state and communicates the change to the remote system to enable the client protocols to handle this in a seamless manner without going down.
- When the **clear bfd session** command is executed, the BFD sessions move to the Admin Down state before restarting the BFD sessions so that the client applications are not impacted.

Negotiation of the BFD Liveness Detection Interval

When you issue the appropriate **bfd-liveness-detection** command on an IS-IS, OSPF, RIP, or PIM interface, BFD liveness detection is established with all of its BFD-enabled peers. When an update is received from a peer—if BFD is enabled and if the session is not already present—the local peer attempts to create a BFD session to the remote peer.

Each pair of peers negotiates acceptable transmit and receive intervals for BFD packets. These values can be different on each peer.

The negotiated transmit interval for a peer is the interval between the BFD packets that it sends to its peers. The receive interval for a peer is the minimum time that it requires between packets sent from its peer; the receive interval is not negotiated between peers.

To determine the transmit interval, each peer compares its configured minimum transmit interval with its peer's minimum receive interval. The larger of the two numbers is accepted as the transmit interval for that peer.

Consider the following example. Router A and Router B are peers, with the following BFD liveness detection values configured.

Router	Configured Transmit Interval (ms)	Configured Receive Interval (ms)
A	400	500
B	450	450

- For Router A, the negotiated transmit interval is the greater of its transmit interval (400 ms) and the Router B receive interval (450 ms), or 450 ms.
- For Router B, the negotiated transmit interval is the greater of its transmit interval (450 ms) and the Router A receive interval (500 ms), or 500 ms.

The liveness detection interval is the period a peer waits for a BFD packet from its peer before declaring the BFD session to be down. The detection interval is determined independently by each peer and can be different for each. The detection interval for the local peer is calculated as the remote peer's negotiated transmit interval times the detection multiplier value configured on the remote peer.

Router	Negotiated Transmit Interval (ms)	Detection Multiplier	Liveness Detection Interval (ms)
A	450	2	1500
B	500	3	900

- For Router A, the detection interval is Router B's negotiated transmit interval times the Router B detection multiplier: $500 \text{ ms} \times 3 = 1500 \text{ ms}$.
- For Router B, the detection interval is Router A's negotiated transmit interval times the Router A detection multiplier: $450 \text{ ms} \times 2 = 900 \text{ ms}$.

If Router A fails to receive a BFD packet from Router B within 1500 milliseconds, Router A declares the BFD session to be down. Similarly, if Router B fails to receive a BFD packet from Router A within 900 milliseconds, Router B declares the BFD session to be down. In either case, all routes learned from the failed peer are purged immediately.



NOTE: Before the router can use any **bfd-liveness-detection** command, you must specify a BFD license key. To view an already configured license, use the **show license bfd** command.



NOTE: During a stateful SRP switchover, the BFD transmit interval is set to 1000 ms with a detection multiplier of 3. These values result in a liveness detection interval of 3000 ms. This longer interval helps prevent a BFD timeout during the switchover. BFD negotiates the interval with the remote peer before applying the temporary change. The BFD timers revert back to the configured values after 15 minutes (the maximum duration for graceful restart completion).

BFD Platform Considerations

For information about modules that support BFD on the ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support BFD.

For information about modules that support BFD on the E120 and E320 Broadband Services Routers:

- See *E120 and E320 Module Guide, Table 1, Modules and IOAs* for detailed module specifications.
- See *E120 and E320 Module Guide, Appendix A, IOA Protocol Support* for information about the modules that support BFD.

The ES2 4G LM supports faster transmission and failure detection. On this module, the configurable BFD timer range for each routing protocol is extended from a minimum of 100 ms to 10 ms.

BFD References

For information about BFD, see the following:

- BFD for IPv4 and IPv6 (Single Hop)—draft-ietf-bfd-v4v6-1hop-00.txt (January 2005 expiration)
- Bidirectional Forwarding Detection—draft-ietf-bfd-base-00.txt. (January 2005 expiration)

Configuring a BFD License

You must configure a BFD license before the router configuration can use any BFD commands.

license bfd

- Use to specify a BFD license.
- Purchase a BFD license to allow BFD configuration on the E Series router.



NOTE: Acquire the BFD license from Juniper Networks Customer Service or your Juniper Networks sales representative.

- Example

```
host1(config)#license bfd license-value
```
- Use the **no** version to disable the license.
- See `license bfd`.

BFD Version Support

The JunosE Software supports both BFD Version 0 and BFD Version 1. When establishing a BFD neighbor session, the E Series router attempts to establish version 0 or version 1 sessions based on the capability of the BFD neighbor. [Table 7 on page 111](#) indicates how the routers establish sessions based on BFD version support:

Table 7: Determining BFD Versions

		E Series Routers Running JunosE 7.2.x (and later)	E Series Routers Running Software Versions Earlier than JunosE 7.2.x
	BFD Version Support	Version 0 and Version 1	Version 0 Only
E Series Routers Running JunosE 7.2.x (and later) and Other Routers	Version 0 and Version 1	Result = Version 1	Result = Version 0
E Series Routers Running Software Versions Earlier than JunosE 7.2.x and Other Routers	Version 0 Only	Result = Version 0	Result = Version 0
Other Routers	Version 1 Only	Result = Version 1	No session (version mismatch)



NOTE: You cannot configure the JunosE Software to send BFD Version 0 or BFD Version 1 packets. The JunosE Software determines the BFD version through auto-negotiation.

Configuring BFD

You configure BFD on routing protocols that use BFD for fast failure detection. BFD does not require any stand-alone configuration; it works in conjunction with the application that it is supporting. Applications on which you configure BFD pass configuration information to BFD when they need fast failure detection.

BFD works with a wide variety of routing protocols. The JunosE Software currently supports only a few of these protocols. All BFD supported clients provide the ability to configure session parameters for each interface. Refer to the following table for added configuration information:

Configuration Topic	See
EBGP	<i>Configuring BGP Routing in JunosE BGP and MPLS Configuration Guide</i>
IPv4 static routes	<i>Configuring IP in JunosE IP, IPv6, and IGP Configuration Guide</i>
IS-IS	<i>Configuring IS-IS in JunosE IP, IPv6, and IGP Configuration Guide</i>
OSPF	<i>Configuring OSPF in JunosE IP, IPv6, and IGP Configuration Guide</i>
OSPFv3	<i>Configuring OSPF in JunosE IP, IPv6, and IGP Configuration Guide</i>
PIM	<i>Configuring PIM for IPv4 Multicast in JunosE IP, IPv6, and IGP Configuration Guide and Configuring PIM for IPv6 Multicast in JunosE Multicast Routing Configuration Guide</i>
RIP	<i>Configuring RIP in JunosE IP, IPv6, and IGP Configuration Guide</i>
RSVP-TE	<i>Configuring MPLS in JunosE BGP and MPLS Configuration Guide</i>

Managing BFD Adaptive Timer Intervals

The **bfd adapt** command enables timer intervals to adapt for all BFD sessions on all virtual routers on the router.

Enabling BFD adaptive timers avoids BFD session flaps that might occur because of misconfiguration or other errors. When enabled, BFD attempts to adapt timer intervals on the router by making them less restrictive and increasing the survival chances for the session.



NOTE: Enabling BFD adaptive timers targets only rapidly flapping events and not genuine BFD down events. If BFD down events occur in intervals longer than 5 seconds, the session does not attempt to adapt.

Disabling BFD adaptive timers does not affect current adaptive timer intervals for sessions. Disabling adaptive timers prohibits BFD from further adapting timer intervals for existing sessions or for new sessions.

To reset adapted intervals for all BFD sessions on the router, use the **clear bfd adapted-intervals** command.

bfd adapt

- Use to enable all BFD sessions to adapt timer intervals on all virtual routers on the router.
- Example

```
host1(config)#bfd adapt
```
- Use the **no** version to disable subsequent BFD sessions from adapting timer intervals without resetting any already adapted intervals.
- See `bfd adapt`.

clear bfd adapted-intervals

- Use to reset adapted timer intervals for all BFD sessions on the router.
- Does not disable the state of the BFD adaptive timer interval feature.
- Example

```
host1#clear bfd adapted-intervals
```
- There is no **no** version.
- See `clear bfd adapted-intervals`.

Clearing BFD Sessions

You can use the **clear bfd session** or **clear ipv6 bfd session** commands to clear one or more BFD sessions for IPv4 or IPv6 (respectively).

clear bfd session

- Use to restart all IPv4 BFD sessions or a specified IPv4 BFD session.
- Use the **address** keyword to indicate the IPv4 address of the destination to which the session has been established.
- Use the **discriminator** keyword to clear the BFD session associated with the unique system-wide identifier.
- Example 1

host1#clear bfd session

- Example 2

host1#clear bfd session address 10.10.5.24

- Example 3

host1#clear bfd session discriminator 4

- There is no **no** version.
- See clear bfd session.



NOTE: When the **clear bfd session** command is executed, the BFD sessions move to the Admin Down state before restarting the BFD sessions so that the client applications are not impacted.

clear ipv6 bfd session

- Use to restart all IPv6 BFD sessions or a specified IPv6 BFD session.
- Use the **address** keyword to indicate the IPv6 address of the destination to which the session has been established.

- Example 1

host1#clear ipv6 bfd session

- Example 2

host1#clear ipv6 bfd session address 1::4

- There is no **no** version.
- See clear ipv6 bfd session.

Monitoring BFD

This section lists the system event logs associated with the BFD protocol and describes the **show** commands you can use to view BFD-related information.

System Event Logs

To troubleshoot and monitor BFD, use the following system event logs:

- bfdGeneral
- bfdSession
- bfdEvents
- bgpConnections
- isisBfdEvents
- ospfEvents

- ospfv3General
- ripBfdLog

For more information about using event logs, see the *JunosE System Event Logging Reference Guide*.

Viewing BFD Information

You can monitor the following aspects of BFD by using the following **show** commands:

To Display	Command
BFD session information	show bfd session
BFD license key information	show license bfd

You can use the output filtering feature of the **show** command to include or exclude lines of output based on a text string that you specify. See *Command Line Interface in JunosE System Basics Configuration Guide*, for details.

show license bfd

- Use to display the bfd license key configured on the router.
- Example

```
host1#show license bfd
BFD license is bfd_license
```

- See show license.

show bfd session

- Use to display BFD protocol session information.
- Use the **address** keyword to specify an IPv4 or IPv6 session that you want to view.
- Use the **detail** keyword to view more detailed information about the BFD session.
- Field descriptions
 - Address—IP address of the remote interface with which the session is established. In unnumbered cases, the remote interface provides its reference IP address.
 - State—State of the BFD session, Up, Down, or AdminDown
 - Interface—Interface on which the BFD session has been established
 - Detect/Detection Time—Time (in seconds) taken to declare the remote interface down when no packets are received from that interface
 - Local discriminator—Value used to identify the session at the local end
 - Remote discriminator—Value used to identify the session at the remote end

- Session up time—Amount of time the session has been operational since the last session down event in *days:hours:minutes:seconds* format.
- Up/Down count—Number of times up/down transitions have occurred on the session
- Adaptivity—Number of times this session has adapted its intervals, or that additional adaptivity is disabled for this BFD session on the router
- Local
 - min tx interval—Minimum transmit interval (in seconds) configured on the session at the local end
 - min rx interval—Minimum receive interval (in seconds) configured on the session at the local end
 - multiplier—Multiplier configured on the session at the local end
- (Adapted)
 - min tx interval—Minimum transmit interval (in seconds) to which the session is adapted at the local end
 - min rx interval—Minimum receive interval (in seconds) to which the session is adapted at the local end
 - multiplier—Multiplier to which the session is adapted at the local end
- Remote
 - min tx interval—Minimum transmit interval (in seconds) configured on the session at the remote end
 - min rx interval—Minimum receive interval (in seconds) configured on the session at the remote end
 - multiplier—Multiplier configured on the session at the remote end
- Up/Down count—Number of up/down transitions that have occurred on the session
- Local diagnostic—Reason at the local end for the last session down event
- Remote diagnostic—Reason at the remote end for the last session down event
- Remote heard/Remote not heard—Whether the local end is receiving packets from the remote end
- hears us/doesn't hear us—Whether the remote end is receiving packets from the local end
- Min async interval—Minimum interval (in seconds) between packets sent when in asynchronous mode
- min slow interval—Minimum interval (in seconds) between packets when the remote end is first being detected

- Echo mode—State of echo mode (enabled or disabled; active or inactive)
- Client—Name of the client
 - desired tx—Minimum transmit interval (in seconds) requested by the client
 - required rx—Minimum required receive interval (in seconds) specified by the client
 - multiplier—Multiplier requested by the client
- TX FC-assisted?—Whether component in forwarding controller is acting to speed transmission times, yes or no; forwarding controller assist available only for ES2 4G LM
- Detection FC-assisted?—Whether component in forwarding controller is acting to speed fast failure detection times, yes or no; forwarding controller assist available only for ES2 4G LM
- Example 1

```
host1#show bfd session
Address          State      Interface      Detect   Interval   Mx
                  Time
-----
172.16.1.2       Up         FastEthernet1/4 0.900    0.300      3
172.16.1.1       Up         FastEthernet1/5 0.900    0.300      3
172.16.1.3       AdminDown FastEthernet1/2 0.900    0.300      3
```

- Example 2—IPv4 version

```
host1#show bfd session detail
Address: 172.16.1.2
  State UP on Interface FastEthernet1/4
  Detection Time0.900, version v0
  Local discriminator 3, Remote discriminator 1
  Session up time 00:00:01:04, Up/Down count 1, Adapted 3 times
  Local: min tx interval 0.3, min rx interval 0.3, multiplier 1
  (Adapted) min tx interval 0.6, min rx interval 0, multiplier 3
  Remote: min tx interval 0.3, min rx interval 0.3, multiplier 3
  Local diagnostic: None, Remote diagnostic: None
  Remote heard, hears us
  Min async interval 0.3, min slow interval 0.3
  Echo mode disabled/inactive
2 Clients:
  Client OSPFv2, desired tx: 0.3, required rx: 0.3, multiplier 3
  Client ISIS, desired tx: 0.3, required rx: 0.3, multiplier 3
  TX FC-assisted? Yes, Detection FC-assisted? Yes
```

- Example 3—IPv6 version

```
host1#show bfd session detail
Address fe80:1234::abcd
  State UP on Interface FastEthernet1/3
  Detection Time0.900, version v1
  Local discriminator 3, Remote discriminator 1
  Session up time 00:00:01:04, Up/Down count 1, Adaptivity disabled
  Local: min tx interval 0.3, min rx interval 0.3, multiplier 3
  (Adapted) min tx interval 0, min rx interval 0, multiplier 4
  Remote: min tx interval 0.3, min rx interval 0.3, multiplier 3
  Local diagnostic: None, Remote diagnostic: None
```

```
Remote heard, hears us
Min async interval 0.3, min slow interval 0.3
Echo mode disabled/inactive
1 Client:
  Client OSPFv3, desired tx: 0.3, required rx: 0.3, multiplier 3
```

- See show bfd session.

CHAPTER 5

Configuring IPSec

This chapter describes Internet Protocol Security (IPSec) capabilities of the ERX routers. It contains the following sections:

- [Overview on page 119](#)
- [Platform Considerations on page 121](#)
- [References on page 121](#)
- [IPSec Concepts on page 122](#)
- [IKE Overview on page 134](#)
- [Configuration Tasks on page 139](#)
- [Configuration Examples on page 153](#)
- [Monitoring IPSec on page 160](#)

Overview

The IP security functionality covered in this chapter includes the following major areas:

- Encapsulating protocols, including authentication (AH) and Encapsulating Security Payload (ESP), to provide security on specified packets
- The Internet Security Association and Key Management Protocol/Internet Key Exchange (ISAKMP/IKE) protocol suite to provide automatic negotiation of security associations, including session keys

IPSec Terms and Acronyms

[Table 8 on page 119](#) describes terms and abbreviations that are used in this discussion of IPSec.

Table 8: IPSec Terms and Abbreviations

Term or Abbreviation	Description
3DES	Triple DES encryption/decryption algorithm
AH	Authentication header. Provides authentication of the sender and of data integrity.

Table 8: IPSec Terms and Abbreviations (*continued*)

Term or Abbreviation	Description
CA	Certificate authority
DES	Data Encryption Standard encryption algorithm
DPD	Dead peer detection, which enables router to detect when communication to remote peer has been disconnected. Also known as IKE keepalive.
DSS	Digital Signature Standard authentication algorithm
ESP	Encapsulating Security Payload, which provides data integrity, data confidentiality and, optionally, sender's authentication
FQDN	Fully qualified domain name, which consists of the hostname and domain name for a specific system
HMAC	Hashed Message Authentication Code
IKE	Internet Key Exchange
IKE endpoint	IP address of the entity that is one of two endpoints in an IKE/ISAKMP SA.
Inbound traffic	In the context of a secure interface, already secured traffic arriving on that interface (identified based on its SPI). This traffic is cleared and checked against the security parameters set for that interface.
IPSec	Internet Protocol Security
IPSec endpoint	IP address of the entity that is one of two endpoints in an IPSec SA
ISAKMP	Internet Security Association and Key Management Protocol
ISAKMP SA	Security associations used to secure control channels between security gateways. These are negotiated via IKE phase 1.
MDx	Message Digest hash algorithm
Nonce	A random value used to detect and protect against replay attacks
Outbound traffic	In the context of a secure interface, the clear traffic forwarded to the interface (either by policy or by routing) that is typically secured according to security parameters set for that interface.
PFS	Perfect forward secrecy
RSA	Rivest-Shamir-Adleman encryption algorithm

Table 8: IPSec Terms and Abbreviations (*continued*)

Term or Abbreviation	Description
SA	Security association. The set of security parameters that dictate how IPSec processes a packet, including encapsulation protocol and session keys. A single secure tunnel uses multiple SAs.
Secure tunnel	A virtual connection between two security gateways used to exchange data packets in a secure way. A secure tunnel is made up of a local SA and a remote SA, where both are negotiated in the context of an ISAKMP SA.
SHA	Secure Hash Algorithm
SPI	Security parameter index
VPN	Virtual private network

Platform Considerations

For information about modules that support IPSec on ERX14xx models, ERX7xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support IPSec.



NOTE: E120 and E320 Broadband Services Routers do not support configuration of IPSec.

References

For information about IPSec, see the following RFCs:

- RFC 768—User Datagram Protocol (August 1980)
- RFC 2401—Security Architecture for the Internet Protocol (November 1998)
- RFC 2402—IP Authentication Header (November 1998)
- RFC 2403—The Use of HMAC-MD5-96 within ESP and AH (November 1998)
- RFC 2404—The Use of HMAC-SHA-1-96 within ESP and AH (November 1998)
- RFC 2405—The ESP DES-CBC Cipher Algorithm With Explicit IV (November 1998)
- RFC 2406—IP Encapsulating Security Payload (ESP) (November 1998)
- RFC 2407—The Internet IP Security Domain of Interpretation for ISAKMP (November 1998)

- RFC 2408—Internet Security Association and Key Management Protocol (ISAKMP) (November 1998)
- RFC 2409—The Internet Key Exchange (IKE) (November 1998)
- RFC 2410—The NULL Encryption Algorithm and Its Use With IPSec (November 1998)
- RFC 3706—A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers (February 2004)

For information about using digital certificates, see [“Configuring Digital Certificates” on page 205](#).

IPSec Concepts

This section provides an overview of IPSec concepts.

IPSec provides security to IP flows through the use of authentication and encryption.

- Authentication verifies that data is not altered during transmission and ensures that users are communicating with the individual or organization that they believe they are communicating with.
- Encryption makes data confidential by making it unreadable to everyone except the sender and intended recipient.

IPSec comprises two encapsulation protocols:

- Encapsulating Security Payload (ESP) provides confidentiality and authentication functions to every data packet.
- Authentication header (AH) provides authentication to every data packet.

Both protocols are defined with two modes of operation:

- Tunnel mode completely encapsulates the original packet within another IP header.
- Transport mode keeps the original header and does not add the extra IP header.

Secure IP Interfaces

Secure IP interfaces are virtual IP interfaces that you can configure to provide confidentiality and authentication services for the data flowing through such interfaces. The software provides these services using mechanisms created by the suite of IPSec standards established by the IETF.

Secure IP interfaces connect the router to any other endpoint through the routed network and allow much of the same functionality as other IP interfaces. Traffic can reach a secure IP interface via routing or policy routing.

- A secure tunnel is a layer 2 entity. It is a point-to-point connection that is mapped on top of other IP interfaces. Secure tunnels carry only IP traffic.
- A secure IP interface is a layer 3 entity; that is, an IP interface mapped on top of a secure tunnel that inherits all security associated with it.

Secure IP interfaces are a logical representation of a secure connection between two security endpoints, one of which is the local system. The remote endpoint can be another security gateway or a host.

RFC 2401 Compliance

RFC 2401 states that a security policy database (SPD) must exist for each physical interface in the router, and an administrator must configure these SPDs to determine which traffic must be IPSec-protected, not IPSec-protected, or denied. The ERX router does not support a systemwide SPD. Instead, the router takes advantage of routing policies that are applied to physical interfaces to describe which traffic to forward to a single IPSec tunnel, which traffic to discard, and so on. The router also applies IPSec selectors to traffic going into or coming out of a secure tunnel so that unwanted traffic is not allowed inside the tunnel. Supported selectors include IP addresses, subnets, and IP address ranges. An implementation that strictly follows RFC 2401 requires a separate IPSec tunnel for each SPD entry.

IPSec Protocol Stack

Figure 12 on page 123 shows the protocol stack on a client, an IPSec gateway, and a server. In the figure, HTTP and TCP are examples of higher-level protocols involved in the end-to-end communication; other end-to-end communication protocols are also supported. The layers where the data can be encrypted are shown in gray.

Figure 12: IPSec Tunneling Stack

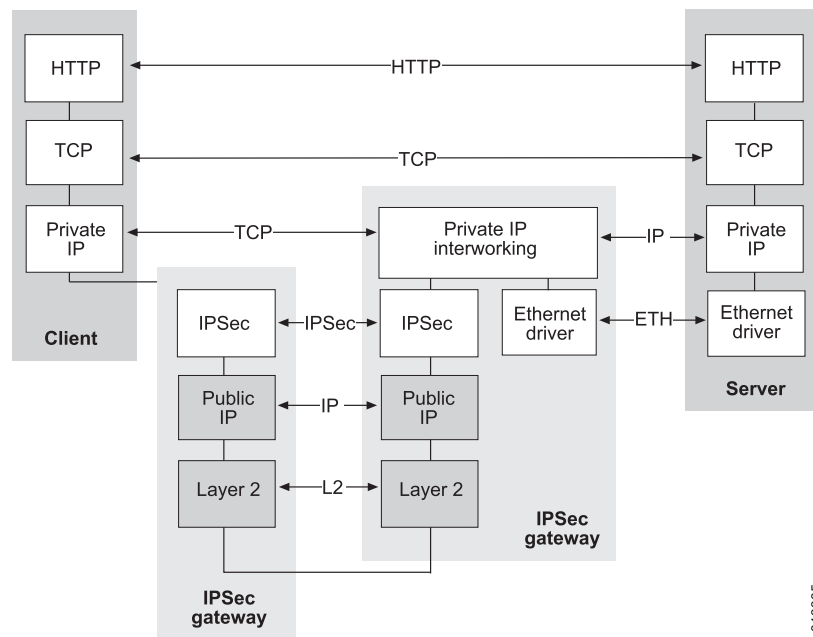
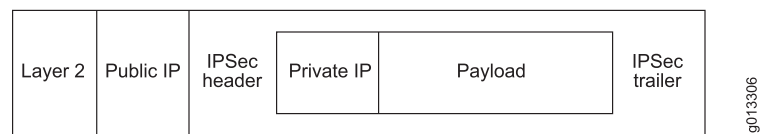


Figure 13 on page 124 shows the packet encapsulation for IPSec tunneling.

Figure 13: IPSec Tunneling Packet Encapsulation

Security Parameters

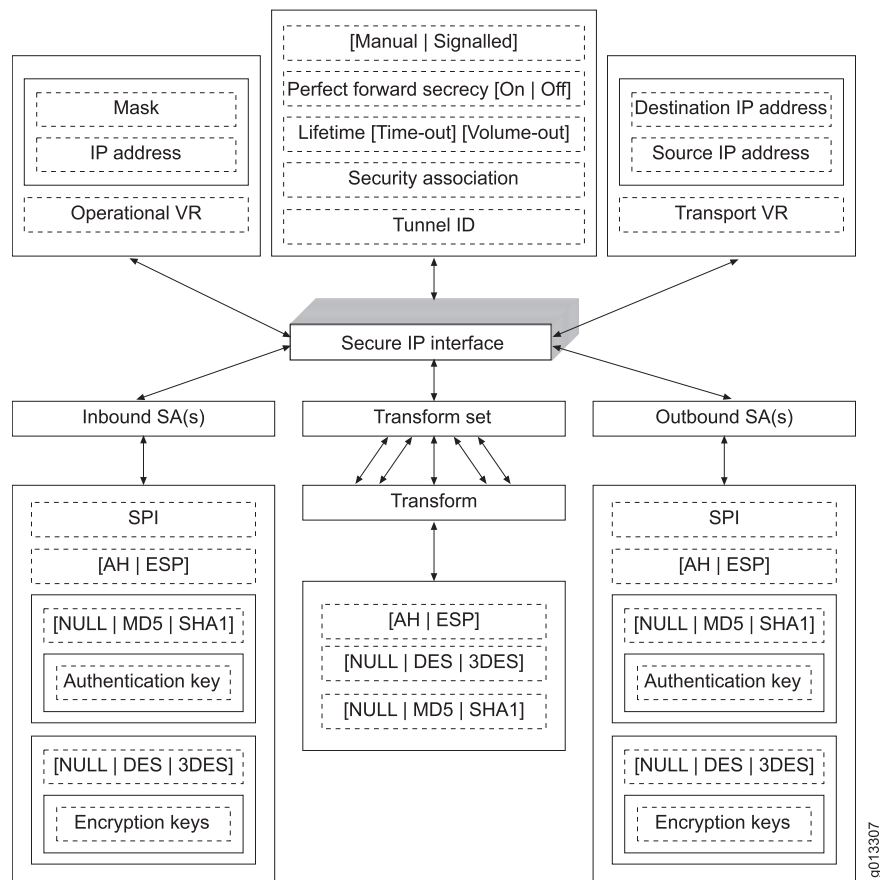
Secure IP interfaces allow tunneled traffic to be secured in many ways. For that, secure interfaces are associated with security parameters that are enforced for traffic that goes through these interfaces. [Table 9 on page 124](#) briefly describes all the parameters used for a secure IP interface.

Table 9: Security Parameters Used on Secure IP Interfaces

Security Parameter	Description
Manual or signaled	<p>A secure IP interface, which can be either manual or signaled.</p> <ul style="list-style-type: none"> You can configure manual interfaces manually on both local and remote security gateways. Signaled interfaces can dynamically set up connections between security gateways using ISAKMP/IKE.
Operational VR	Operational parameters for the secure IP interface, including the virtual router context to which this interface belongs and the network prefix reachable through the interface.
Transport VR	Transport network characteristics for the tunnel, including its virtual router context and source and destination IP addresses.
Perfect forward secrecy (PFS)	A key-generation approach that guarantees that every newly generated session key is not in any way related to the previous keys. PFS ensures that a compromised session key does not compromise previous and subsequent keys.
Lifetime	A limit on time and traffic volume allowed over the interface before an SA needs to be renegotiated.
Inbound and outbound SAs	<p>The actual session-related parameters used by both security gateways to secure the traffic between them. You can manually define the SA for manual secure IP tunnels or the SA can dynamically negotiate for signaled tunnels.</p> <p>Two sets of SA parameters exist; one for inbound traffic and another for outbound traffic.</p>
Transform set	The set of security parameters, including protocols and algorithms, that is considered adequate to provide a required security level to the traffic flowing through an interface.

[Figure 14 on page 125](#) shows the relationships of the various security parameters to the IPSec security interface. The following sections discuss each parameter in detail.

Figure 14: IPSec Security Parameters in Relation to the Secure IP Interface



Manual Versus Signaled Interfaces

The router supports both manual and signaled interfaces:

- Manual interfaces use a preconfigured set of SA parameters to secure traffic flowing through a secure IP interface. If SA parameters do not use a preconfigured, manual secure interface, the interface drops all traffic it receives. The router keeps statistics for dropped traffic. Both peer security gateways must contain a manually provisioned manual secure IP tunnel.
- Signaled interfaces negotiate an SA on demand with the remote security gateway. The remote security gateway must also support SA negotiation; otherwise the gateway drops traffic. Again, the router keeps statistics for dropped traffic.

The router supports SA negotiation within an IKE SA by means of the ISAKMP and IKE protocols. Only one IKE SA is maintained between a set of local and remote IKE endpoints. That means that if an IKE SA already exists between the two endpoints, it is reused.

Secure IP interface parameters can be required, optional, or not applicable, depending on whether the interface is manual or signaled. [Table 10 on page 126](#) presents how the other security parameters fit with manual and signaled interfaces.

Table 10: Security Parameters per IPSec Policy Type

Security Parameter	Manual	Signaled
Operational VR	Required	Required
Transport VR	Required	Required
Perfect forward secrecy	Optional	Optional
Lifetime	Optional	Optional
Inbound and outbound SAs	Required	Not applicable
Transform set	Required	Required

Operational Virtual Router

The operational VR for a secure IP tunnel is the VR in which a secure IP tunnel exists.

The IP address and mask associated with a secure IP interface exist only within the operational VR under which the interface is declared. The VR defines the network prefix, which is reachable through the logical IP interface.

A secure IP tunnel is always a member of one and only one operational VR. Therefore, the operational VR attributes are mandatory for any secure tunnel. These attributes include:

- IP address and mask
- Virtual router on which the secure IP interface exists

Transport Virtual Router

The transport VR for a secure IP tunnel is the VR in which both of the secure tunnel endpoints, the source and destination, are routable addresses. Normally, the transport VR is the default ISP routing infrastructure on top of which VPNs are provisioned.

The IPSec Service module (ISM) is a security gateway and, as such, is one of the endpoints for secure tunnels. The tunnel endpoints are the tunnel *source* and the tunnel *destination* IP addresses. For IKE signaled IPSec tunnels, you can use the fully qualified domain name (FQDN) instead of the IP address to identify the tunnel endpoints. You typically use this feature to identify the tunnel destination endpoint in DSL and broadband environments. See [“Transport VR Definitions with an FQDN” on page 127](#) in this section.

- The tunnel source IP address must be one of the local IP addresses configured on the router.
- The tunnel destination address must be a routable IP address within the transport VR routing tables.

The transport VR information is required, although its explicit configuration is not. If omitted, the transport VR is assumed to be the same as the operational VR. However, the tunnel source and destination are mandatory elements.

Transport VR Definition

The transport VR definition includes:

- Transport virtual router name—Name of the transport virtual router. If not explicitly configured, the operational VR is assumed.
- Tunnel source endpoint—IP address or FQDN used as the tunnel source endpoint on this end of the tunnel. In the case of signaled tunnels, the router monitors and transmits on port 500 of this address for IKE negotiations. The tunnel source endpoint must be a configured IP address or FQDN on the transport VR, or the router indicates an error. See [“Transport VR Definitions with an FQDN” on page 127](#) for information about using an FQDN rather than an IP address.
- Tunnel destination endpoint—IP address or FQDN associated with the termination or initiation point of the secure IP tunnel. This address must be routable within the context of the transport VR. Each secure IP tunnel can have a different remote IP address.

Transport VR Definitions with an FQDN

For signaled IPSec tunnels, you can use an FQDN instead of the IP address to specify tunnel endpoints. You typically use this feature to identify the tunnel destination in broadband and DSL environments in which the destination does not have a fixed IP address. The remote device uses the FQDN to establish and authenticate the IPSec connection, and then uses the actual IP address for rekeying and filtering operations.

The ERX router FQDN feature supports both preshared keys and digital certificates. If it uses preshared keys, the router must use IKE aggressive mode to support FQDNs.

An identity string can include an optional *user@* specification that precedes the FQDN. The entire string can be a maximum of 80 characters. For example, both of the following are supported:

```
branch245.customer77.isp.net
user4919@branch245.customer77.isp.net
```

With preshared key authentication, and when using the *user@fqdn* format, the router searches for the key based on the entire identity string. If the router cannot find that string, the router strips off the *user@* part and performs a second search based on the FQDN part of the string.

With digital certificates, the two sides of the tunnel must use the same identity format, with or without the *user@* specification; no stripping operation and no second search occurs.



NOTE: The E Series router does not support FQDN-to-IP address resolution by DNS.

Perfect Forward Secrecy

PFS is an optional feature that causes every newly refreshed key to be completely unrelated to the previous key. PFS provides added security, but requires extra processing for a new Diffie-Hellmann key exchange on every key refresh.

If PFS is enabled, the router mandates PFS during SA negotiation. The remote security gateway must accept PFS to successfully negotiate the SA. However, if PFS is disabled, PFS might still be negotiated if the remote security gateway requests PFS.

PFS supports three Diffie-Hellmann prime modulus groups:

- Group 1—A 768-bit Diffie-Hellmann prime modulus group
- Group 2—A 1024-bit Diffie-Hellmann prime modulus group
- Group 5—A 1536-bit Diffie-Hellmann prime modulus group

SA negotiation favors the highest request. For example, if group 2 is requested locally, the remote security gateway must support group 2 for the SA negotiation to be successful. If group 1 is requested locally, either groups 1 or 2 can be accepted, depending on requests from the remote security gateway.

Lifetime

You can set a lifetime for user SAs and IKE SAs. For information about setting the IKE SA lifetime, see [“Lifetime” on page 138](#).

For signaled IPSec interfaces, both the inbound and outbound SA must be assigned a lifetime. The lifetime parameter controls the duration for which the SA is valid. When a user SA is established, both a timer and a traffic volume counter are set. When either counter reaches the limit specified by the SA lifetime, a new SA is negotiated and the expired SA is deleted. The renegotiations refresh several SA parameters, including keys.

Note the following about how the lifetime parameters work:

- To avoid delays in the data flow, a new user SA is actually renegotiated before the expiration. If the SA expires in the middle of processing a packet, the router finishes processing that packet.
- The actual user SA lifetime may not equal the value configured in the router.
- There are both global and tunnel-specific lifetime parameters. If there is no tunnel-specific lifetime configured, the router uses the global lifetime. The global lifetime parameters have the following default settings:
 - 8 hours for the time-based lifetime
 - 100 MB for the traffic-based lifetime
- Lifetime parameters are valid only for user SAs established via IKE. Manually configured user SAs ignore this parameter.

You can set a lifetime for all SAs on a specific tunnel, and you can set a global lifetime.

- To set the tunnel lifetime, use the **tunnel lifetime** command.
- To set the global (default) lifetime, use the **ipsec lifetime** command.

Inbound and Outbound SAs

SA parameters are the actual session parameters used to secure a specific data flow associated with a specific secure IP interface. How SA parameters are set depends on how the IP interfaces are secured:

- For manual secure IP interfaces, the system administrator sets SA parameters. Manually setting SA parameters allows provisioning of IP security to destinations that do not support SA negotiation via IKE.
- For signaled secure IP interfaces, the two security gateway peers negotiate SA parameters; the system administrator is not allowed to set any of the parameters. In fact, for some of these parameters, such as session keys, the system administrator is not even granted read access.

Similarly to IPSec SAs, SA parameters are unidirectional. Therefore, for a two-way data flow, two SAs need to be established—one for inbound traffic and another for outbound traffic. For each direction, SA parameters must be set for each transform associated with a secure IP interface. Therefore, two sets of SA parameters exist for each secure IP interface, one being the inbound SA parameters and the other the outbound SA parameters.

The following parameters form each set of SA parameters:

- SPI—The SPI is a unique identifier that is applied to the SA when securing a flow. An SPI is unique for a given destination IP address and protocol tuple. The destination IP address is either the remote secure IP interface endpoint for the outbound direction or the local secure IP interface endpoint for the inbound direction.
- Encapsulation—The encapsulation options include both an encapsulating protocol and an encapsulating mode. The protocol can be either ESP or AH. The mode is tunnel mode.
- Transforms—The allowed transforms for given SA parameters depend on the encapsulation protocol. See [“Transform Sets” on page 130](#) for more information.
- Keys—The session key is used for the respective SA transform. The key length depends on the SA transform to which it applies, and is as follows:
 - DES—8 bytes
 - 3DES—24 bytes
 - MD5—16 bytes
 - SHA—20 bytes

Transform Sets

Transform sets are composed of security parameters that provide a required security level to a particular data flow. Transform sets are used during user SA negotiation to find common agreement between the local and the remote security gateway on how to protect that specific data flow.

A transform set includes encapsulation protocols and transforms; for example, encryption/decryption/authentication algorithms. These parameters are grouped to specify the acceptable protection for a given data flow. Many transform sets are supported, since different traffic requires distinct security levels.

A secure IP tunnel is associated with one transform set. Multiple secure IP tunnels can refer to the same transform set.

Changing existing transform sets affects only future user SA negotiations. User SAs that are already established remain valid and do not use the changed transform set until they are renegotiated.

For manually configured secure IP tunnels, the associated transform set must contain a single transform option.

Encapsulation Protocols

Both the AH and ESP protocols are supported. See supported transforms in [Table 11 on page 131](#).

- AH provides authentication.
- ESP provides data confidentiality and antireplay functions. ESP can also provide data authentication; although, in this implementation, ESP does not cover the outer IP header.

Encapsulation Modes

IPSec supports two encapsulation modes—tunnel mode and transport mode. Tunnel mode creates a second IP header in the packet and uses both the local and remote security gateway addresses as source and destination IP addresses. Also, tunnel mode allows an IP interface to be created and stacked right above it.

Transport mode does not add a second IP header and does not allow an IP interface to be created and stacked right above it. Instead, transport mode allows other tunneling applications, such as an L2TP tunnel, to be created and stacked on top of an IPSec transport mode connection. See [“Securing L2TP and IP Tunnels with IPSec” on page 277](#) for a description of L2TP transport mode.

Supported Transforms

[Table 11 on page 131](#) describes the supported transforms.

Table 11: Supported Transforms

Transform	Description
AH-MD5	IPSec performs AH protocol encapsulation using the MD5 hash function with HMAC message authentication.
AH-SHA	IPSec performs AH protocol encapsulation using the SHA-1 hash function with HMAC message authentication. SHA-1 is considered stronger than MD5.
ESP-MD5	IPSec performs ESP protocol encapsulation using the MD5 hash function with HMAC message authentication.
ESP-SHA	IPSec performs ESP protocol encapsulation using the SHA-1 hash function with HMAC message authentication. SHA-1 is considered stronger than MD5.
ESP-DES	IPSec performs ESP protocol encapsulation using the DES encryption algorithm. DES uses a 56-bit symmetric key and is considered a weak (breakable) encryption algorithm.
ESP-3DES	IPSec performs ESP protocol encapsulation using the 3DES encryption algorithm. 3DES uses a 168-bit symmetric encryption key and is widely accepted as a strong encryption algorithm. Export control issues apply to products that ship from the USA with 3DES.
ESP-DES-MD5	Combination of ESP-MD5 and ESP-DES transforms.
ESP-DES-SHA	Combination of ESP-SHA and ESP-DES transforms.
ESP-3DES-MD5	Combination of ESP-MD5 and ESP-3DES transforms.
ESP-3DES-SHA	Combination of ESP-SHA and ESP-3DES transforms.

Table 12 on page 131 lists the security functions achieved with the supported transforms, and provides a view of which combinations can be used, depending on security requirements.

Table 12: Supported Security Transform Combinations

Security Type	Supported Transform Combinations
Data authentication only	AH-HMAC-MD5
	AH-HMAC-SHA
	ESP-HMAC-MD5
	ESP-HMAC-SHA
Data confidentiality only	ESP-DES
	ESP-3DES

Table 12: Supported Security Transform Combinations (*continued*)

Security Type	Supported Transform Combinations
Data authentication and confidentiality	ESP-DES-MD5
	ESP-DES-SHA
	ESP-3DES-MD5
	ESP-3DES-SHA

The ISM does not support both the ESP and AH encapsulation modes concurrently on the same secure tunnel.

Negotiating Transforms

Inside a transform set, IPSec transforms are numbered in a priority sequence.

- During negotiation as an initiator of the user SA, the router uses transform number one first. If the remote system does not agree on the transform, the router then tries number two, and so on. If both end systems do not agree on a transform, the user SA fails and the secure IP tunnel is not established.
- During negotiation as a responder, the router compares the proposed transform from the remote end against each transform in the transform set. If there is no match, the router provides a negative answer to the remote end, which can either try another transform or give up. If no match is found, the secure IP tunnel is not established.

Other Security Features

The following sections briefly describe other supported security features for the ERX routers. These features include the following:

- [“IP Security Policies” on page 132](#)
- [“ESP Processing” on page 132](#)
- [“AH Processing” on page 133](#)

This section also provides a pointer to the IPSec system maximums.

IP Security Policies

The ERX router does not support a systemwide SPD. Instead, the router takes advantage of routing to forward traffic to and from a secure tunnel. The router still applies IPSec selectors to traffic going into or coming out of a secure tunnel so that unwanted traffic is not allowed inside the tunnel. Supported selectors include IP addresses, subnets, and IP address ranges.

ESP Processing

The router supports both the encryption and authentication functions of ESP encapsulation as defined in RFC 2406. Specifically, the router supports:

- DES and 3DES encryption algorithms
- The HMAC-SHA and HMAC-MD5 authentication algorithms
- ESP security options on a per-tunnel (per-SA) basis
- Tunnel mode

AH Processing

The router supports AH encapsulation as defined in RFC 2402. Specifically, the router supports:

- HMAC-SHA and HMAC-MD5 authentication algorithms
- AH authentication options on a per-tunnel (per-SA) basis
- Tunnel mode

IPSec Maximums Supported

See *JunosE Release Notes, Appendix A, System Maximums* corresponding to your software release for information about maximum values.

DPD and IPSec Tunnel Failover

Dead peer detection (DPD) is a keepalive mechanism that enables the E Series router to detect when the connection between the router and a remote IPSec peer has been lost. DPD enables the router to reclaim resources and to optionally redirect traffic to an alternate failover destination. If DPD is not enabled, the traffic continues to be sent to the unavailable destination.

When a disconnected state is detected between the E Series router and an IPSec peer, the router:

- Tears down the IPSec connection and displays the interface's state as down in output for the **show ipsec tunnel detail** command
- Clears all SAs that were established between the two endpoints
- Stops forwarding packets to the unavailable destination
- Generates SNMP traps
- Allows routing protocols running on the IP interfaces on top of the failed IPSec tunnel to switch to alternate paths
- (Optional) Redirects traffic to an alternate tunnel destination

Unlike other keepalive and heartbeat schemes, which require that peers frequently exchange Hello packets with each other at regular predetermined intervals, DPD uses two techniques to verify connectivity on an as-needed basis. In the first method, the router sends DPD inquiries to the remote peer when traffic has been sent to the peer in the last 30 seconds but no traffic has been received from the peer in the last 60 seconds. In the second method, DPD uses an idle timer. If there has been no traffic between the

router and the peer for 2.5 minutes, DPD sends an inquiry to the remote end to verify that the peer is still reachable.



NOTE: Not all IPSec connections need to verify connectivity between peers. For example, the ERX router does not use DPD to check secure remote access connections based on L2TP over IPSec, which have their own keepalive mechanism. However, the router does reply to a request from a remote peer in this type of connection.

Tunnel Failover

The ERX router provides a failover mechanism for IPSec tunnels that works in concert with both DPD and with IKE SA negotiation. The tunnel failover feature provides an alternate tunnel destination when DPD detects that the current destination is unreachable or when IKE SA set up is unsuccessful. During failover, the IPSec tunnel switches to the alternate destination and establishes IPSec SAs with the new peer. To configure tunnel failover, you specify the tunnel destination backup endpoint.

Tunnel failover is a two-way process. If the router detects that the remote peer is unreachable, it switches to sending traffic to the backup destination. Likewise, if the router is sending traffic to the backup destination when the connection is terminated, the router switches to sending the traffic to the original remote peer.



NOTE: Even without tunnel failover configured, DPD still provides many benefits, such as indicating that the destination interface is down, ensuring that the router stops sending packets to the unreachable destination, and generating SNMP traps.

IKE Overview

The IKE suite of protocols allows a pair of security gateways to:

- Dynamically establish a secure tunnel over which the security gateways can exchange tunnel and key information.
- Set up user-level tunnels or SAs, including tunnel attribute negotiations and key management. These tunnels can also be refreshed and terminated on top of the same secure channel.

IKE is based on the Oakley and Skeme key determination protocols and the ISAKMP framework for key exchange and security association establishment. IKE provides:

- Automatic key refreshing on configurable timeout
- Support for public key infrastructure (PKI) authentication systems
- Antireplay defense

IKE is layered on UDP and uses UDP port 500 to exchange IKE information between the security gateways. Therefore, UDP port 500 packets must be permitted on any IP interface involved in connecting a security gateway peer.

The following sections expand on the IKE functionality available for the router.

Main Mode and Aggressive Mode

IKE phase 1 negotiations are used to establish IKE SAs. These SAs protect the IKE phase 2 negotiations. IKE uses one of two modes for phase 1 negotiations: main mode or aggressive mode. The choice of main or aggressive mode is a matter of tradeoffs. Some of the characteristics of the two modes are:

- Main mode
 - Protects the identities of the peers during negotiations and is therefore more secure.
 - Enables greater proposal flexibility than aggressive mode.
 - Is more time consuming than aggressive mode because more messages are exchanged between peers. (Six messages are exchanged in main mode.)
- Aggressive mode
 - Exposes identities of the peers to eavesdropping, making it less secure than main mode.
 - Is faster than main mode because fewer messages are exchanged between peers. (Three messages are exchanged in aggressive mode.)
 - Enables support for fully qualified domain names (FQDNs) when the router uses preshared keys.

The next section describes aggressive mode in more detail.

Aggressive Mode Negotiations

During aggressive mode phase 1 negotiations, the E Series router behaves as follows:

- When the router is the initiator, the router searches all policy rules to find those that allow aggressive mode. The router then selects the rule with the highest priority and uses the rule to initiate phase 1 negotiations. If there are no policy rules with aggressive mode allowed, the router selects the highest-priority rule that allows main mode.
- When the router is the responder, the negotiation depends on what the initiator proposes, as well as what is configured in the policy rules.

[Table 13 on page 136](#) outlines the possible combinations of initiator proposals and policy rules. As indicated, allowing aggressive mode in a policy rule allows negotiation to take place no matter what the initiator requests.

Table 13: Initiator Proposals and Policy Rules

Aggressive Mode Setting	Initiator Requests (First Time)	Initiator Requests (Rekeyed)	Responder Policy Rule
Accepted	Main mode	Follows First Time	Aggressive or Main modes (follows initiator)
Requested	Aggressive mode	Follows First Time	Aggressive or Main modes (follows initiator)
Required	Aggressive mode	Aggressive Mode	Aggressive mode
None	Main mode	Main Mode	Main mode

The router responds to phase 1 negotiations with the highest-priority policy rule that matches the initiator. A match means that all parameters, including the exchange type, match.

IKE Policies

An IKE policy defines a combination of security parameters to be used during the IKE SA negotiation. IKE policies are configured on both security gateway peers, and there must be at least one policy on the local peer that matches a policy on the remote peer. Failing that, the two peers are not able to successfully negotiate the IKE SA, and no data flow is possible.

IKE policies are global to the router. Every ISM on a router uses the same set of policies when negotiating IKE SAs. The agreed-on IKE SA between the local system and a remote security gateway may vary, because it depends on the IKE policies used by each remote peer. However, the initial set of IKE policies the router uses is always the same and independent of which peer the router is negotiating with.

During negotiation, the router might skip IKE policies that require parameters that are not configured for the remote security gateway with which the IKE SA is being negotiated.

You can define up to ten IKE policies, with each policy having a different combination of security parameters. A default IKE policy that contains default values for every policy parameter is available. This policy is used only when IKE policies are not configured and IKE is required.

The following sections describe each of the parameters contained in an IKE policy.

Priority

Priority allows better (more secure) policies to be given preference during the negotiation process. However, every IKE policy is considered secure enough to secure the IKE SA flow.

During IKE negotiation, all policies are scanned, one at a time, starting from the highest-priority policy and ending with the lowest-priority policy. The first policy that the peer security gateway accepts is used for that IKE session. This procedure is repeated for every IKE session that needs to be established.

Encryption

A specific encryption transform can be applied to an IKE policy. The supported encryption algorithms are:

- DES
- 3DES

Hash Function

A specific hash function can be applied to an IKE policy. The supported ones are:

- MD5
- SHA-1

IKE also uses an authentication algorithm during IKE exchanges. This authentication algorithm is automatically set to the HMAC version of the specified hash algorithm. Therefore, you cannot have the hash function set to MD5 and the authentication algorithm set to HMAC-SHA.

Authentication Mode

As part of the IKE protocol, one security gateway needs to authenticate the other security gateway to make sure that the IKE SA is established with the intended party. The ERX router supports two authentication methods:

- Digital certificates (using RSA algorithms)

For digital certificate authentication, an initiator signs message interchange data using his private key, and a responder uses the initiator's public key to verify the signature. Typically, the public key is exchanged via messages containing an X.509v3 certificate. This certificate provides a level of assurance that a peer's identity (as represented in the certificate) is associated with a particular public key.

For more information, see [“Configuring Digital Certificates” on page 205](#).

- Preshared keys

With preshared key authentication mode, the same secret string (similar to a password) must be configured on both security gateways before the gateways can authenticate each other. It is not advisable to share a preshared key among multiple pairs of security gateways, because it reduces the key's security level.

The router allows preshared keys to be up to 256 ASCII alphanumeric characters.

Diffie-Hellman Group

An IKE policy must specify which Diffie-Hellmann group is used during the symmetrical key generation phase of IKE. The following Diffie-Hellmann groups are supported:

- Group 1 (768-bit)
- Group 2 (1024-bit)

- Group 5 (1536-bit)

Lifetime

Like a user SA, an IKE SA does not last indefinitely. Therefore, the router allows you to specify a lifetime parameter for an IKE policy. The timer for the lifetime parameter begins when the IKE SA is established using IKE.

IKE SA Negotiation

As the initiator of an IKE SA, the router sends its IKE policies to the remote peer. If the peer has an IKE policy that matches the encryption, hash, authentication method, and Diffie-Hellmann group settings, the peer returns the matching policy. The peers use the lesser lifetime setting as the IKE SA lifetime. If no match is found, the IKE SA fails, and a log alarm is generated.

As the responder of an IKE negotiation, the router receives all IKE policies from a remote security gateway. The router then scans its own list of IKE policies to determine whether a match exists, starting from the highest priority. If it finds a match, that policy is successfully negotiated. Again, the lifetime is negotiated to the lesser of the two lifetimes, and failures are logged.

Generating Private and Public Key Pairs

When any of the public key methods for authenticating remote security gateways is used, the system must have at least one valid pair of public or private keys. Therefore, the system provides a facility by which it can generate public and private key pairs for itself.

The private key is used only by the system itself. It is never exchanged with any other nodes. When generated, the private key is securely stored internally to the system in nonvolatile memory. Access to the private key is never given, not even to a system administrator or to a network management system.

The public key is used in either of the following scenarios:

- A network administration system or system administrator can retrieve it so that it can be entered into remote security gateways with which the system needs to establish an IKE SA.
- It can be given to CAs so that they can properly sign it. From there, the public key is distributed to remote security gateways that can handle a PKI.

The public/private key pair as provided by the system supports the RSA standard (512, 1024, or 2048 bits).

The public/private key pair is a global system attribute, regardless of how many ISMs exist in the system. Only one set of keys is available at any given time.

Configuration Tasks

This section explains the steps to configure an IPSec license and IPSec parameters, create an IPSec tunnel, and define an ISAKMP/IKE policy. The next section contains configuration examples.

Configuring an IPSec License

By default, and with no IPSec tunnel license, you can configure up to 10 IPSec tunnels on an ERX router. However, you can purchase licenses that support the following IPSec tunnel maximums:

- 1000
- 2000
- 4000
- 8000
- 16,000
- 32,000

The number of additional tunnels is independent of the number of ISMs installed in the router. However, the router chassis enforces the following tunnel limits:

- SRP 10G – 10,000
- SRP 40G – 20,000

license ipsec-tunnels

- Use to specify an IPSec tunnel license.



NOTE: Acquire the license from Juniper Networks Customer Services and Support or from your Juniper Networks sales representative.

- Example

```
host1(config)#license ipsec-tunnels license string
```

- Use the **no** version to disable the license.
- See `license ipsec-tunnels`.

Configuring IPSec Parameters

To configure IPSec:

1. For each endpoint, create a transform set that provides the desired encryption and authentication.

```
host1(config)#ipsec transform-set customerAprotection esp-3des-hmac-sha
```

```
host1(config)#ipsec transform-set customerBprotection ah-hmac-md5
```

2. Add a preshared key that the routers use to authenticate each other.

```
host1(config)#ipsec key manual pre-share 5.2.0.1
host1(config-manual-key)#key customerASecret
```

After you enter a preshared key, the router encrypts the key and displays it in masked form to increase the security of the key. If you need to reenter the key, you can enter it in its masked form using this command.

To see the masked form of the key:

```
host1#show config
ipsec key manual pre-share 10.10.1.1
masked-key "AAAAGAAAAAcAAAACfd+SAsaVQ6Qeopt2rJOP6LDg+0hX5cMO"
```

To enter the masked key:

```
host1(config-manual-key)#masked-key
AAAAGAAAAAcAAAACfd+SAsaVQ6Qeopt2rJOP6LDg+0hX5cMO
```

3. Define the local endpoint used for ISAKMP/IKE negotiations for all IPSec tunnels in the router.

```
host1(config)#ipsec local-endpoint 10.10.1.1 transport-virtual-router vr#8
```

4. (Optional) Set the global (default) lifetime for all SAs on the router.

```
host1(config)#ipsec lifetime kilobytes 42000000
```

ipsec key manual pre-share

- Use to specify that a peer use a preshared key for authentication during the tunnel establishment phase, and to display the prompt that lets you enter the preshared key. To enter a key, use the **key** command.
- Specify the peer by using its IP address or fully qualified domain name (FQDN).
 - FQDNs are supported only for signaled tunnels.
 - The router must be in aggressive mode to use FQDNs with preshared keys.
 - The identity string can include an optional *user@* specification preceding the FQDN.
- You must enter this command in the virtual router context where the IP address of the peer is defined.
- Example 1—using an IP Address

```
host1(config)#ipsec key manual pre-share ip address 10.10.1.1
host1(config-manual-key)#
```

- Example 2—using an FQDN

```
host1(config)#ipsec key manual pre-share identity branch245.customer77.isp.net
host1(config-manual-key)#
```

- Example 3—using an FQDN with *user@* specification

```
host1(config)#ipsec key manual pre-share identity
user4919@branch245.customer77.isp.net
host1(config-manual-key)#
```

- Use the **no** version to delete a manually configured key from the router.
- See ipsec key manual pre-share.

ipsec lifetime

- Use to set the global (default) lifetime in seconds or volume of traffic in kilobytes. The IPSec lifetime applies to tunnels that do not have a tunnel lifetime defined. When either limit is reached, the SA is renegotiated.
- To set a lifetime for all SAs on a tunnel, use the **tunnel lifetime** command.
- To set a lifetime for a specific SA, use [“lifetime” on page 151](#).
- Example 1

```
host1(config)#ipsec lifetime kilobytes 42000000
```
- Example 2

```
host1(config)#ipsec lifetime seconds 8600
```
- Use the **no** version to restore the default values of 4294967295 kilobytes and 28800 seconds (8 hours).
- See ipsec lifetime.

ipsec local-endpoint

- Use to define a default local endpoint for ISAKMP/IKE negotiations and all IPSec tunnels for a transport virtual router.
- You must specify the IP address used as the local endpoint and the transport virtual router on which the IP address is defined.
- Example

```
host1(config)#ipsec local-endpoint 10.10.1.1 transport-virtual-router VR#8
```
- Use the **no** version to delete a local endpoint. You cannot remove an endpoint if a tunnel is referencing the endpoint.
- See ipsec local-endpoint.

ipsec transform-set

- Use to create a transform set. Each transform in a set provides a different combination of data authentication and confidentiality.
- Transform sets used for manually configured tunnels can have one transform.
- Transform sets used for signaled tunnels can have up to six transforms. The actual transform used on the tunnel is negotiated with the peer. Transforms are numbered in a priority sequence in the order in which you enter them.
- To display the names of the transforms that you can use in a transform set, issue the **ipsec transform-set transformSetName ?** command.
- Example

```
host1(config)#ipsec transform-set espSet esp-3des-hmac-md5 esp-3des-null-auth
```

- Use the **no** version to delete a transform set. You cannot remove a transform set if a tunnel is referencing the transform set.
- See ipsec transform-set.

key

- Use to enter a manual preshared key.
- Preshared keys can have up to 256 ASCII alphanumeric characters. To include spaces in the key, enclose the key in quotation marks.

- Example 1

```
host1(config-manual-key)#key dj5fe23owi8er49fdsa
```

- Example 2

```
host1(config-manual-key)#key " my key with spaces"
```

- There is no **no** version. To delete a key, use the **no** version of the **ipsec key manual** command.
- See key.

masked-key

- Use to enter the preshared key in masked form.
- For security purposes, the router displays the key only in masked form. If you delete the key or reboot the router to factory defaults, you can use this command to reenter the key in its masked form so that the key is not visible while you enter it.
- To see the masked key, use the **show config** command.
- Example

```
host1#show config
ipsec key manual pre-share 10.10.1.1
masked-key " AAAAGAAAAAcAAAACfd+SAsaVQ6Qeopt2rJOP6LDg+0hX5cMO"
host1#configure terminal
host1(config)#ipsec key manual pre-share 10.10.1.1
host1(config-manual-key)#masked-key
AAAAGAAAAAcAAAACfd+SAsaVQ6Qeopt2rJOP6LDg+0hX5cMO
```

- There is no **no** version. To delete a key, use the **no** version of the **ipsec key manual** command.
- See masked-key.

Creating an IPSec Tunnel

To create an IPSec tunnel:

1. Enter virtual router mode. Specify the VR that contains the source and destination addresses assigned to the tunnel interface.

```
host1(config)#virtual-router vrA
host1:vrA(config)#
```

2. Create an IPSec tunnel, and specify the transport VR.

```
host1:vrA(config)#interface tunnel ipsec:Aottawa2boston transport-virtual-router
default
host1:vrA(config-if)#
```

3. Specify the IP address of this tunnel interface.

```
host1:vrA(config-if)#ip address 10.3.0.0 255.255.0.0
```

4. Specify the transform set that ISAKMP uses for SA negotiations.

```
host1:vrA(config-if)#tunnel transform-set customerAprotection
```

5. Configure the local endpoint of the tunnel.

```
host1:vrA(config-if)#tunnel local-identity subnet 10.1.0.0 255.255.0.0
```

6. Configure the peer endpoint of the tunnel.

```
host1:vrA(config-if)#tunnel peer-identity subnet 10.3.0.0 255.255.0.0
```

7. Specify an existing interface address that the tunnel uses as its source address.

```
host1:vrA(config-if)#tunnel source 5.1.0.1
```

8. Specify the address or identity of the tunnel destination endpoint.

```
host1:vrA(config-if)#tunnel destination identity branch245.customer77.isp.net
host1:vrA(config-if)#exit
```



NOTE: FQDNs are used when tunnel destination endpoints do not have a fixed address, as in cable and DSL environments.

9. For manual tunnels, specify the algorithm sets and the session key used for inbound SAs and for outbound SAs.

```
host1:vrA(config-if)#tunnel session-key-inbound esp-des-hmac-md5
a7bd567917bd5679 bd5678a7bd567917bd567917bd567678
host1:vrA(config-if)#tunnel session-key-outbound esp-3des-hmac-md5 421
567917bd567917bd567917bd545a17bd567917bd56784a7b
fda183bef567917bd567917bd567917b
```

10. (Optional) Configure PFS on this tunnel.

```
host1:vrA(config-if)#tunnel pfs group 5
```

11. (Optional) Set the tunnel type to signaled or manual. The default is signaled.

```
host1:vrA(config-if)#tunnel signaling isakmp
```

12. (Optional) Set the renegotiation time of the SAs in use by this tunnel.

```
host1(config-if)#tunnel lifetime seconds 48000 kilobytes 249000
```

13. (Optional) Set the MTU size for the tunnel.

```
host1(config-if)#tunnel mtu 2240
```

interface tunnel

- Use to create or configure an IPSec tunnel interface.
- Use the **transport-virtual-router** keyword to establish the tunnel on a virtual router other than the current virtual router context.
- Example

```
host1(config)#interface tunnel ipsec:jak transport-virtual-router tvr041
host1(config-if)#
```
- Use the **no** version to remove the tunnel.
- See interface tunnel.

tunnel destination

- Use to set the address or identity of the remote tunnel endpoint.
 - For signaled IPSec tunnels in cable or DSL environments, use the FQDN to identify the remote tunnel endpoint, which does not have a fixed IP address.
 - The identity string can include an optional *user@* specification preceding the FQDN.
- Example 1

```
host1(config-if)#tunnel destination 10.10.11.12
```
- Example 2

```
host1(config-if)#tunnel destination identity branch245.customer77.isp.net
```
- Example 3

```
host1(config-if)#tunnel destination identity user4919@branch245.customer77.isp.net
```
- Use the **no** version to remove the address.
- See tunnel destination.

tunnel lifetime

- Use to set the renegotiation time of the SAs in use by this tunnel.
- To configure the lifetime in number of seconds, use the **seconds** keyword to specify the lifetime in the range 1800–864000. The default value is 28800 seconds.
- To configure the lifetime in amount of traffic, use the **kilobytes** keyword to specify the lifetime in the range 102400–4294967295. The default is an unlimited volume.
- If you include the **seconds** keyword as the first keyword on the command line, you can also include the **kilobytes** keyword on the same line.
- Before either the volume of traffic or number of seconds limit is reached, the SA is renegotiated, which ensures that the tunnel does not go down during renegotiation.
- Example

```
host1(config-if)#tunnel lifetime seconds 48000 kilobytes 249000
```
- Use the **no** version to restore the default lifetime (28800 seconds) and an unlimited volume.
- See tunnel lifetime.

tunnel local-identity

- Use to configure the local identity (selector) of the tunnel. Specify the identity using one of the following keywords:
 - **address**—Specifies an IP address as the local identity
 - **subnet**—Specifies a subnet as the local identity
 - **range**—Specifies a range of IP addresses as the local identity
- Example 1
`host1(config-if)#tunnel local-identity range 10.10.1.1 10.10.2.1`
- Example 2
`host1(config-if)#tunnel local-identity subnet 10.10.1.1 255.255.255.0`
- Use the **no** version to restore the default identity, which is subnet 0.0.0.0/0.0.0.0
- See `tunnel local-identity`.

tunnel mtu

- Use to set the MTU size for the tunnel.
- Example
`host1(config-if)#tunnel mtu 2240`
- Use the **no** version to restore the default MTU (1440).
- See `tunnel mtu`.

tunnel peer-identity

- Use to configure the peer identity (selector) that ISAKMP uses. Specify the identity using one of the following keywords:
 - **address**—Specifies an IP address as the peer identity
 - **subnet**—Specifies a subnet as the peer identity
 - **range**—Specifies a range of IP addresses as the peer identity
- Example 1
`host1(config-if)#tunnel peer-identity range 10.10.1.1 10.10.2.2`
- Example 2
`host1(config-if)#tunnel peer-identity subnet 130.10.1.1 255.255.255.0`
- Use the **no** version to remove the peer identity.
- See `tunnel peer-identity`.

tunnel pfs group

- Use to configure perfect forward secrecy (PFS) on this tunnel.
- Assign a Diffie-Hellman prime modulus group using one of the following keywords:

- 1—768-bit group
- 2—1024-bit group
- 5—1536-bit group
- Example
`host1(config-if)#tunnel pfs group 5`
- Use the **no** version to remove PFS from this tunnel.
- See `tunnel pfs group`.

tunnel session-key-inbound

- Use to manually configure the authentication or encryption algorithm sets and session keys for inbound SAs on a tunnel. You can enter this command only on tunnels that have tunnel signaling set to manual.
- Use the online Help to see a list of available algorithm sets.
- Each key is an arbitrary hexadecimal string. If the algorithm set includes:
 - DES, create an 8-byte key using 16 hexadecimal characters
 - 3DES, create a 24-byte key using 48 hexadecimal characters
 - MD5, create a 16-byte key using 32 hexadecimal characters
 - SHA, create a 20-byte key using 40 hexadecimal characters
- Example
`host1(config-if)#tunnel session-key-inbound esp-des-hmac-md5 a7bd567917bd5679bd5678a7bd567917bd567917bd567678`
- Use the **no** version to remove inbound session keys from a tunnel.
- See `tunnel session-key-inbound`.

tunnel session-key-outbound

- Use to manually configure the authentication or encryption algorithm sets, SPI, and session keys for outbound SAs on a tunnel. You can enter this command only on tunnels that have tunnel signaling set to manual.
- Use the online Help to see a list of available algorithm sets.
- The SPI is a number in the range 256–4294967295 that identifies an SA.
- Each key is an arbitrary hexadecimal string. If the algorithm set includes:
 - DES, create an 8-byte key using 16 hexadecimal characters
 - 3DES, create a 24-byte key using 48 hexadecimal characters
 - MD5, create a 16-byte key using 32 hexadecimal characters
 - SHA, create a 20-byte key using 40 hexadecimal characters

- Example

```
host1(config-if)#tunnel session-key-outbound esp-3des-hmac-md5 421
567917bd567917bd567917bd545a17bd567917bd56784a7b
fda183bef567917bd567917bd567917b
```

- Use the **no** version to remove outbound session keys from a tunnel.
- See tunnel session-key-outbound.

tunnel signaling

- Use to set the tunnel type to signaled (ISAKMP) or manual. Specify a keyword:
 - **isakmp**—Specifies to use ISAKMP/IKE to negotiate SAs and to establish keys
 - **manual**—Specifies that security parameters and keys are configured manually

- Example

```
host1(config-if)#tunnel signaling manual
```

- Use the **no** version to restore the default value, **isakmp**.
- See tunnel signaling.

tunnel source

- Use to specify an existing interface address that serves as the tunnel's source address.
- For signaled IPSec tunnels in cable or DSL environments, you can optionally use an FQDN to identify the tunnel endpoint.
- Example

```
host1(config-if)#tunnel source 10.10.2.8
```

- Use the **no** version to remove the tunnel source.
- See tunnel source.

tunnel transform-set

- Use to specify the transform set that ISAKMP uses during SA negotiations on this tunnel. You create transform sets using [“ipsec transform-set” on page 141](#).

- Example

```
host1(config-if)#tunnel transform-set espSet
```

- Use the **no** version to remove the transform set from a tunnel.
- See tunnel transform-set.

Configuring DPD and IPSec Tunnel Failover

You can use the **ipsec option dpd** command to enable dead peer detection (DPD) on the router. DPD is also known as IKE keepalive. If an IPSec tunnel destination backup is configured, the router redirects traffic to the alternate destination when DPD detects a disconnection between the E Series router and the regular tunnel destination. See [“tunnel destination backup” on page 148](#).

To enable DPD and create an alternate IPSec tunnel destination for failover:

1. Enable DPD on the router.

```
host1(config)#ipsec option dpd
```

2. Enter virtual router mode. Specify the VR that contains the source and destination addresses assigned to the tunnel interface (that is, the transport virtual router context).

```
host1(config)#virtual-router vrA  
host1:vrA(config)#
```

3. Create an IPSec tunnel, and specify the transport VR.

```
host1:vrA(config)#interface tunnel ipsec:Aottawa2boston transport-virtual-router  
default  
host1:vrA(config-if)#
```

4. Specify the address or identity of the tunnel destination backup endpoint.

```
host1:vrA(config-if)#tunnel destination backup identity branch500.customer77.isp.net
```

ipsec option dpd

- Use to enable dead peer detection (DPD) on the router. DPD is also known as IKE keepalive.
- You configure DPD on a per-virtual router basis.
- Both peers must support DPD.
- Example

```
host1(config)#ipsec option dpd
```

- Use the **no** version to restore the default, which disables DPD.
- See `ipsec option dpd`.

tunnel destination backup

- Use to specify the address or identity of the remote IPSec tunnel endpoint that is a backup tunnel destination. When DPD detects a disconnection between the E Series router and the regular IPSec tunnel destination, the router redirects traffic to the tunnel destination backup, and vice versa.
- You can use either the IP address or fully qualified domain name (FQDN) to identify the backup IPSec tunnel, however you must use the same type of identity that is used to specify the regular tunnel destination.
 - For signaled IPSec tunnels in cable or DSL environments, use the FQDN to identify the tunnel destination backup, which does not have a fixed IP address.
 - The identity string can include an optional *user@* specification preceding the FQDN (this is also known as a user FQDN).



NOTE: If you use a FQDN to specify the IPSec tunnel destination backup, the tunnel is not initiated by the ERX router. However, the router does respond to negotiations for this backup tunnel.

- Examples

```
host1(config-if)#tunnel destination backup 10.10.11.15
host1(config-if)#tunnel destination backup identity branch245.customer88.isp.net
host1(config-if)#tunnel destination backup identity
user4925@branch245.customer88.isp.net
```

- Use the **no** version to restore the default in which the regular tunnel destination is also the backup tunnel destination.
- See tunnel destination backup.

Defining an IKE Policy

IKE policies define parameters that the router uses during IKE phase 1 negotiation.

To create an IKE policy:

```
host1(config)#ipsec ike-policy-rule 3
host1(config-ike-policy)#
```

You can then set the following parameters, or use the default settings:

- Allow aggressive mode negotiation.

```
host1(config-ike-policy)#aggressive-mode
```
- Specify the authentication method.

```
host1(config-ike-policy)#authentication pre-share
```
- Specify the encryption algorithm.

```
host1(config-ike-policy)#encryption 3des
```
- Assign a Diffie-Hellman group.

```
host1(config-ike-policy)#group 5
```
- Set the hash algorithm.

```
host1(config-ike-policy)#hash md5
```
- Specify the lifetime of IKE SAs created using this policy.

```
host1(config-ike-policy)#lifetime 360
```

aggressive-mode

- Use to enable aggressive mode negotiation for the tunnel.
- If you specify aggressive mode negotiation, the tunnel proposes aggressive mode to the peer in connections that the policy initiates.
- If the peer initiates a negotiation, the tunnel accepts the negotiation if the mode matches this policy.
- Use the **accepted** keyword to accept aggressive mode when proposed by peers
- Use the **requested** keyword to request aggressive mode when negotiating with peers

- Use the **required** keyword to only request and accept aggressive mode when negotiating with peers.
- Example

```
host1(config-ike-policy)#aggressive-mode accepted
```
- Use the **no** version to set the negotiation mode to main mode.
- See aggressive-mode.

authentication

- Use to specify the authentication method the router uses in the IKE policy: preshared keys or RSA signature.
- Example

```
host1(config-ike-policy)#authentication pre-share
```
- Use the **no** version to restore the default, preshared keys.
- See authentication.

encryption

- Use to specify one of the following encryption algorithms to use in the IKE policy:
 - **3des**—168-bit 3DES-CBC
 - **des**—56-bit DES-CBC
- Example

```
host1(config-ike-policy)#encryption 3des
```
- Use the **no** version to restore the default encryption algorithm, 3DES.
- See encryption.

group

- Use to assign a Diffie-Hellman group to the IKE policy. Specify:
 - **1**—768-bit group
 - **2**—1024-bit group
 - **5**—1536-bit group
- Example

```
host1(config-ike-policy)#group 5
```
- Use the **no** version to restore the default.
- See group.

hash

- Use to set the hash algorithm for the IKE policy:
 - **md5**—MD5 (HMAC variant)

- **sha**—SHA-1 (HMAC variant)
- Example

```
host1(config-ike-policy)#hash md5
```
- Use the **no** version to restore the default, **sha**.
- See hash.

ipsec ike-policy-rule

ipsec isakmp-policy-rule



NOTE: The command replaces the `ipsec isakmp-policy-rule` command, which may be removed completely in a future release.

- Use to define an IKE policy.
- When you enter the command, you include a number that identifies the policy and assigns a priority to the policy. You can number policies in the range 1–10000, with 1 having the highest priority.
- You can add up to 10 IKE policies per router.
- Example

```
host1(config)#ipsec ike-policy-rule 3
host1(config-ike-policy)#
```
- Use the **no** version to remove policies. If you do not include a priority number with the **no** version, all policies are removed.
- See `ipsec ike-policy-rule`.
- See `ipsec isakmp-policy-rule`.

lifetime

- Use to specify the lifetime of IKE SAs.
- The range is 60–86400 seconds.

```
host1(config-ike-policy)#lifetime 360
```
- Use the **no** version to reset the SA lifetime to the default, 28800 seconds.
- See lifetime.

Refreshing SAs

To refresh ISAKMP/IKE or IPSec SAs:

```
host1(config)#ipsec clear sa tunnel ipsec:Aottawa2boca phase 2
```

ipsec clear sa

- Use to refresh ISAKMP/IKE or IPsec SAs.
- To reinitialize all SAs, use the **all** keyword.
- To reinitialize SAs on a specific tunnel, use the **tunnel** keyword.
- To reinitialize SAs on tunnels that are in a specific state, use the **state** keyword.
- To specify the type of SA to be reinitialized, ISAKMP/IKE or IPSEC, use the **phase** keyword.
- Example

```
host1(config)#ipsec clear sa all phase 2
```
- There is no **no** version.
- See ipsec clear sa.

Enabling Notification of Invalid Cookies

The IKE protocol enables peers to exchange informational messages. The payload of these messages can be a notify type or a delete type. These messages are expected to be protected (encrypted) by the keys negotiated by the peers when they establish a security association as a result of the IKE phase 1 exchange.

If a responder peer does not recognize the initiator-responder cookie pair, it can send an invalid cookie notification message to the initiator. The responder might fail to recognize the cookie pair because it has lost the cookie, or because it deleted the cookie and then the peer lost the delete notification. Upon receipt of the invalid cookie notification, the initiator peer can delete the phase 1 state.

The ability to send the invalid cookie message is disabled by default. You can issue the **ipsec option tx-invalid-cookie** command to enable the feature on a per-transport-VR basis.

Even when you configure this feature, the E Series router does not respond when it receives an invalid cookie notification. These notifications are unprotected by a phase 1 key exchange and therefore are subject to denial-of-service (DOS) attacks. Instead, the E Series router can determine when a phase 1 relationship has gone stale by timeouts or use of dead peer detection (DPD). For this reason, this feature is useful only when the E Series router is a responding peer for non-E Series devices that cannot detect when the phase 1 relationship goes stale.

ipsec option tx-invalid-cookie

- Use to enable the router to send an invalid cookie notification to an IKE peer when the router does not recognize the initiator-responder cookie pair.
- Example

```
host1(config)#ipsec option tx-invalid-cookie
```
- Use the **no** version to restore the default, disabling the ability to send an invalid cookie notification.
- See ipsec option tx-invalid-cookie.

Configuration Examples

This section contains examples of two IPsec applications. The first example shows a customer who replaces a leased line network with an IPsec network that allows the company to connect its corporate locations over the Internet. The second example provides leased line replacement to two customers who use address schemes in the same range.

Configuration Notes

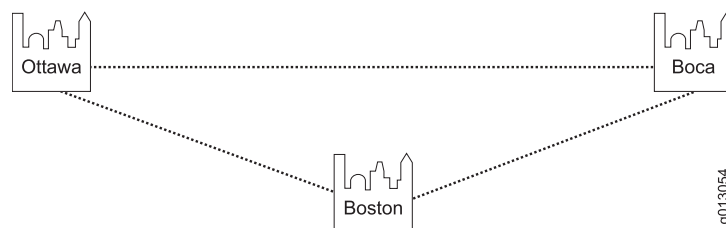
Both the local and remote identities shown in these examples serve two purposes:

- They identify multiple IPsec tunnels between the same endpoints.
- They filter traffic going into and coming out of the tunnels so that it is within the specified range. If the configuration requires that only one IPsec tunnel exists between two endpoints and no traffic filtering is required, you can omit the **tunnel local-identity** and **tunnel peer-identity** commands.

Example 1

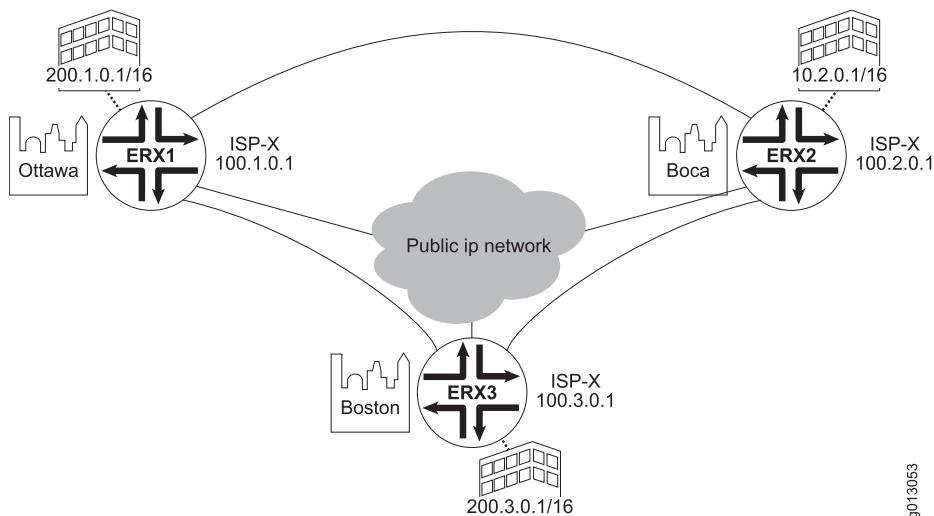
In [Figure 15 on page 153](#) customer A is using Frame Relay to connect its corporate offices in three cities: Boston, Ottawa, and Boca.

Figure 15: Customer A's Corporate Frame Relay Network



Customer A hires ISP-X to provide a leased line replacement over an IP infrastructure using IPsec. ISP-X can offer a replacement for long-haul Frame Relay links by creating IPsec tunnels to carry customer A's traffic securely between the sites over the public or ISP-provided IP network. This alternative costs only a fraction of the price of the Frame Relay links. [Figure 16 on page 154](#) shows the connectivity scheme.

Figure 16: ISP-X Uses ERX Routers to Connect Corporate Offices over the Internet



g013063

To configure the connections as shown in [Figure 16 on page 154](#):

1. On each ERX router, create a protection suite that provides 3DES encryption with SHA-1 authentication on every packet.

```

erx1(config)#ipsec transform-set customerAprotection esp-3des-hmac-sha
erx2(config)#ipsec transform-set customerAprotection esp-3des-hmac-sha
erx3(config)#ipsec transform-set customerAprotection esp-3des-hmac-sha

```

2. On each ERX router, create preshared keys for the three routers to use to authenticate each other:

```

erx1(config)#ipsec key manual pre-share 100.2.0.1
erx1(config-manual-key)#key customerASecret
erx1(config-manual-key)#exit
erx1(config)#ipsec key manual pre-share 100.3.0.1
erx1(config-manual-key)#key customerASecret
erx1(config-manual-key)#exit
erx2(config)#ipsec key manual pre-share 100.1.0.1
erx2(config-manual-key)#key customerASecret
erx2(config-manual-key)#exit
erx2(config)#ipsec key manual pre-share 100.3.0.1
erx2(config-manual-key)#key customerASecret
erx2(config-manual-key)#exit
erx3(config)#ipsec key manual pre-share 100.1.0.1
erx3(config-manual-key)#key customerASecret
erx3(config)#ipsec key manual pre-share 100.2.0.1
erx3(config-manual-key)#key customerASecret
erx3(config-manual-key)#exit

```

3. On erx1 create two IPSec tunnels, one to carry customer A's traffic between Ottawa and Boston and another to carry the traffic between Ottawa and Boca:

Tunnel 1:

```

erx1(config)#interface tunnel ipsec:Aottawa2boston
erx1(config-if)#tunnel transform-set customerAprotection
erx1(config-if)#tunnel local-identity subnet 200.1.0.0 255.255.0.0
erx1(config-if)#tunnel peer-identity subnet 200.3.0.0 255.255.0.0
erx1(config-if)#tunnel source 100.1.0.1
erx1(config-if)#tunnel destination 100.3.0.1
erx1(config-if)#ip address 200.3.0.0 255.255.0.0
erx1(config-if)#exit

```

Tunnel 2:

```

erx1(config)#interface tunnel ipsec:Aottawa2boca
erx1(config-if)#tunnel transform-set customerAprotection
erx1(config-if)#tunnel local-identity subnet 200.1.0.0 255.255.0.0
erx1(config-if)#tunnel peer-identity subnet 200.2.0.0 255.255.0.0
erx1(config-if)#tunnel source 100.1.0.1
erx1(config-if)#tunnel destination 100.2.0.1
erx1(config-if)#ip address 200.2.0.0 255.255.0.0
erx1(config-if)#exit

```

4. On erx2 create two IPSec tunnels, one to carry customer A's traffic between Boca and Ottawa and another to carry the traffic between Boca and Boston:

Tunnel 1:

```

erx2(config)#interface tunnel ipsec:Aboca2ottawa
erx2(config-if)#tunnel transform-set customerAprotection
erx2(config-if)#tunnel local-identity subnet 200.2.0.0 255.255.0.0
erx2(config-if)#tunnel peer-identity subnet 200.1.0.0 255.255.0.0
erx2(config-if)#tunnel source 100.2.0.1
erx2(config-if)#tunnel destination 100.1.0.1
erx2(config-if)#ip address 200.1.0.0 255.255.0.0
erx2(config-if)#exit

```

Tunnel 2:

```

erx2(config)#interface tunnel ipsec:Aboca2boston
erx2(config-if)#tunnel transform-set customerAprotection
erx2(config-if)#tunnel local-identity subnet 200.2.0.0 255.255.0.0
erx2(config-if)#tunnel peer-identity subnet 200.3.0.0 255.255.0.0
erx2(config-if)#tunnel source 100.2.0.1
erx2(config-if)#tunnel destination 100.3.0.1
erx2(config-if)#ip address 200.3.0.0 255.255.0.0
erx2(config-if)#exit

```

5. Finally, on erx3 create two IPSec tunnels, one to carry customer A's traffic between Boston and Ottawa and another to carry the traffic between Boston and Boca:

Tunnel 1:

```

erx3(config)#interface tunnel ipsec:Aboston2ottawa
erx3(config-if)#tunnel transform-set customerAprotection
erx3(config-if)#tunnel local-identity subnet 200.3.0.0 255.255.0.0
erx3(config-if)#tunnel peer-identity subnet 200.1.0.0 255.255.0.0
erx3(config-if)#tunnel source 100.3.0.1
erx3(config-if)#tunnel destination 100.1.0.1
erx3(config-if)#ip address 200.1.0.0 255.255.0.0
erx3(config-if)#exit

```

Tunnel 2:

```

erx3(config)#interface tunnel ipsec:Aboston2boca
erx3(config-if)#tunnel transform-set customerAprotection
erx3(config-if)#tunnel local-identity subnet 200.3.0.0 255.255.0.0
erx3(config-if)#tunnel peer-identity subnet 200.2.0.0 255.255.0.0
erx3(config-if)#tunnel source 100.3.0.1
erx3(config-if)#tunnel destination 100.2.0.1
erx3(config-if)#ip address 200.2.0.0 255.255.0.0
erx3(config-if)#exit

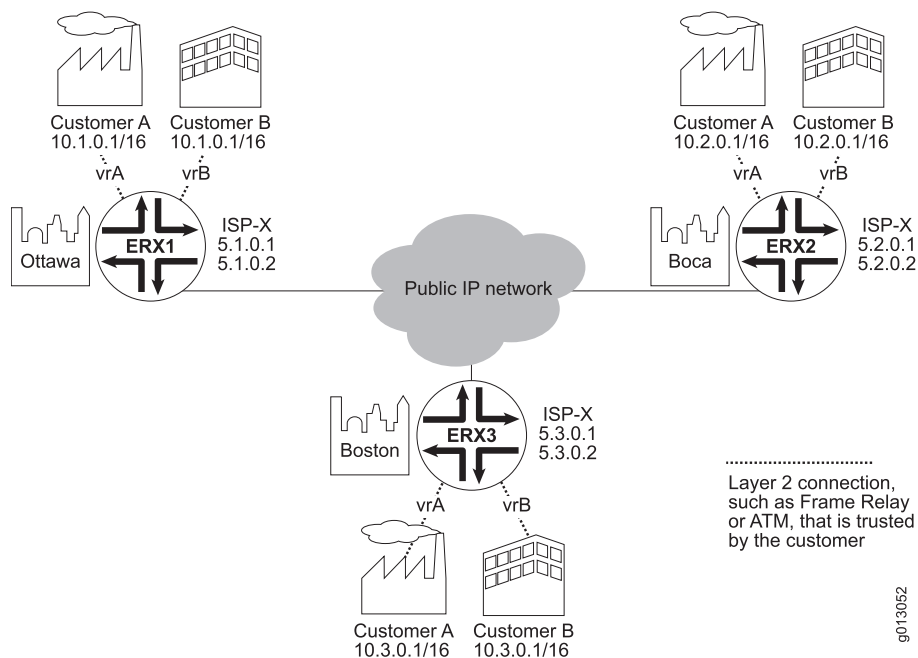
```

The configuration is complete. Now customer A traffic between different cities flows through the public, or untrusted, IP network inside a tunnel, where each packet is encrypted and authenticated. Of course, this example shows the basic secure encapsulation of customer traffic over the untrusted IP network. You can add features such as key refreshing.

Example 2 Example 2, shown in [Figure 17 on page 156](#), enhances the previous example by having the same ISP-X providing leased line replacement to two customers who use address schemes in the same range. There are two ways to solve scenarios in which different customers use similar IP address schemes:

- One solution is to have different transport virtual routers—a configuration similar to example 1, except that a different VR domain is possible.
- Another solution, as described in this example, simply duplicates the endpoints for the transport VR. This example assumes that the transport VR is the default VR.

Figure 17: Connecting Customers Who Use Similar Address Schemes



To configure the connections as shown in [Figure 17 on page 156](#):

1. On each ERX router, create a protection suite that provides customer A with 3DES encryption and SHA-1 authentication, and customer B with AH authentication using MD5.

```

erx1(config)#ipsec transform-set customerAprotection esp-3des-hmac-sha
erx1(config)#ipsec transform-set customerBprotection ah-hmac-md5
erx2(config)#ipsec transform-set customerAprotection esp-3des-hmac-sha
erx2(config)#ipsec transform-set customerBprotection ah-hmac-md5
erx3(config)#ipsec transform-set customerAprotection esp-3des-hmac-sha
erx3(config)#ipsec transform-set customerBprotection ah-hmac-md5

```

2. On each ERX router, create a protection suite for the three routers to use to authenticate each other:

```

erx1(config)#ipsec key manual pre-share 5.2.0.1
erx1(config-manual-key)#key customerASecret
erx1(config-manual-key)#exit
erx1(config)#ipsec key manual pre-share 5.3.0.1
erx1(config-manual-key)#key customerASecret
erx1(config-manual-key)#exit
erx1(config)#ipsec key manual pre-share 5.2.0.2
erx1(config-manual-key)#key customerBSecret
erx1(config-manual-key)#exit
erx1(config)#ipsec key manual pre-share 5.3.0.2
erx1(config-manual-key)#key customerBSecret
erx1(config-manual-key)#exit
erx2(config)#ipsec key manual pre-share 5.1.0.1
erx2(config-manual-key)#key customerASecret
erx2(config-manual-key)#exit
erx2(config)#ipsec key manual pre-share 5.3.0.1
erx2(config-manual-key)#key customerASecret
erx2(config-manual-key)#exit
erx2(config)#ipsec key manual pre-share 5.1.0.2
erx2(config-manual-key)#key customerBSecret
erx2(config-manual-key)#exit
erx2(config)#ipsec key manual pre-share 5.3.0.2
erx2(config-manual-key)#key customerBSecret
erx2(config-manual-key)#exit
erx3(config)#ipsec key manual pre-share 5.1.0.1
erx3(config-manual-key)#key customerASecret
erx3(config-manual-key)#exit
erx3(config)#ipsec key manual pre-share 5.2.0.1
erx3(config-manual-key)#key customerASecret
erx3(config-manual-key)#exit
erx3(config)#ipsec key manual pre-share 5.1.0.2
erx3(config-manual-key)#key customerBSecret
erx3(config-manual-key)#exit
erx3(config)#ipsec key manual pre-share 5.2.0.2
erx3(config-manual-key)#key customerBSecret
erx3(config-manual-key)#exit

```

3. On erx1, create two IPSec tunnels, one to carry customer A's traffic and another to carry customer B's traffic. You must create each pair of tunnels in the virtual routers where the IP interfaces reaching those customers are defined. Create the endpoints for the tunnels in the ISP default virtual router.

Virtual router A:

```
erx1(config)#virtual-router vrA
erx1:vrA(config)#
```

Tunnel from Ottawa to Boston on virtual router A:

```
erx1:vrA(config)#interface tunnel ipsec:Aottawa2boston transport-virtual-router
default
erx1:vrA(config-if)#tunnel transform-set customerAprotection
erx1:vrA(config-if)#tunnel local-identity subnet 10.1.0.0 255.255.0.0
erx1:vrA(config-if)#tunnel peer-identity subnet 10.3.0.0 255.255.0.0
erx1:vrA(config-if)#tunnel source 5.1.0.1
erx1:vrA(config-if)#tunnel destination 5.3.0.1
erx1:vrA(config-if)#ip address 10.3.0.0 255.255.0.0
erx1:vrA(config-if)#exit
```

Tunnel from Ottawa to Boca on virtual router A:

```
erx1:vrA(config)#interface tunnel ipsec:Aottawa2boca transport-virtual-router default
erx1:vrA(config-if)#tunnel transform-set customerAprotection
erx1:vrA(config-if)#tunnel local-identity subnet 10.1.0.0 255.255.0.0
erx1:vrA(config-if)#tunnel peer-identity subnet 10.2.0.0 255.255.0.0
erx1:vrA(config-if)#tunnel source 5.1.0.1
erx1:vrA(config-if)#tunnel destination 5.2.0.1
erx1:vrA(config-if)#ip address 10.2.0.0 255.255.0.0
erx1:vrA(config-if)#exit
```

Virtual router B:

```
erx1(config)#virtual-router vrB
erx1:vrB(config)#
```

Tunnel from Ottawa to Boston on virtual router B:

```
erx1:vrB(config)#interface tunnel ipsec:Bottawa2boston transport-virtual-router
default
erx1:vrB(config-if)#tunnel transform-set customerBprotection
erx1:vrB(config-if)#tunnel local-identity subnet 10.1.0.0 255.255.0.0
erx1:vrB(config-if)#tunnel peer-identity subnet 10.3.0.0 255.255.0.0
erx1:vrB(config-if)#tunnel source 5.1.0.2
erx1:vrB(config-if)#tunnel destination 5.3.0.2
erx1:vrB(config-if)#ip address 10.3.0.0 255.255.0.0
erx1:vrB(config-if)#exit
```

Tunnel from Ottawa to Boca on virtual router B:

```
erx1:vrB(config)#interface tunnel ipsec:Bottawa2boca transport-virtual-router default
erx1:vrB(config-if)#tunnel transform-set customerBprotection
erx1:vrB(config-if)#tunnel local-identity subnet 10.1.0.0 255.255.0.0
erx1:vrB(config-if)#tunnel peer-identity subnet 10.2.0.0 255.255.0.0
erx1:vrB(config-if)#tunnel source 5.1.0.2
erx1:vrB(config-if)#tunnel destination 5.2.0.2
erx1:vrB(config-if)#ip address 10.2.0.0 255.255.0.0
erx1:vrB(config-if)#exit
```

4. On erx2, create two IPSec tunnels, one to carry customer A's traffic and another to carry customer B's traffic. You must create each pair of tunnels in the virtual routers where the IP interfaces reaching those customers are defined. Create the endpoints for the tunnels in the ISP default virtual router.

Virtual router A:

```

erx2(config)#virtual-router vrA
erx2:vrA(config)#

```

Tunnel from Boca to Ottawa on virtual router A:

```

erx2:vrA(config)#interface tunnel ipsec:Aboca2ottawa transport-virtual-router default
erx2:vrA(config-if)#tunnel transform-set customerAprotection
erx2:vrA(config-if)#tunnel local-identity subnet 10.2.0.0 255.255.0.0
erx2:vrA(config-if)#tunnel peer-identity subnet 10.1.0.0 255.255.0.0
erx2:vrA(config-if)#tunnel source 5.2.0.1
erx2:vrA(config-if)#tunnel destination 5.1.0.1
erx2:vrA(config-if)#ip address 10.1.0.0 255.255.0.0
erx2:vrA(config-if)#exit

```

Tunnel from Boca to Boston on virtual router A:

```

erx2:vrA(config)#interface tunnel ipsec:Aboca2boston transport-virtual-router default
erx2:vrA(config-if)#tunnel transform-set customerAprotection
erx2:vrA(config-if)#tunnel local-identity subnet 10.2.0.0 255.255.0.0
erx2:vrA(config-if)#tunnel peer-identity subnet 10.3.0.0 255.255.0.0
erx2:vrA(config-if)#tunnel source 5.2.0.1
erx2:vrA(config-if)#tunnel destination 5.3.0.1
erx2:vrA(config-if)#ip address 10.3.0.0 255.255.0.0
erx2:vrA(config-if)#exit

```

Virtual router B:

```

erx2(config)#virtual-router vrB
erx2:vrB(config)#

```

Tunnel from Boca to Ottawa on virtual router B:

```

erx2:vrB(config)#interface tunnel ipsec:Bboca2ottawa transport-virtual-router default
erx2:vrB(config-if)#tunnel transform-set customerBprotection
erx2:vrB(config-if)#tunnel local-identity subnet 10.2.0.0 255.255.0.0
erx2:vrB(config-if)#tunnel peer-identity subnet 10.1.0.0 255.255.0.0
erx2:vrB(config-if)#tunnel source 5.2.0.2
erx2:vrB(config-if)#tunnel destination 5.1.0.2
erx2:vrB(config-if)#ip address 10.1.0.0 255.255.0.0
erx2:vrB(config-if)#exit

```

Tunnel from Boca to Boston on virtual router B:

```

erx2:vrB(config)#interface tunnel ipsec:Bboca2boston transport-virtual-router default
erx2:vrB(config-if)#tunnel transform-set customerBprotection
erx2:vrB(config-if)#tunnel local-identity subnet 10.2.0.0 255.255.0.0
erx2:vrB(config-if)#tunnel peer-identity subnet 10.3.0.0 255.255.0.0
erx2:vrB(config-if)#tunnel source 5.2.0.2
erx2:vrB(config-if)#tunnel destination 5.3.0.2
erx2:vrB(config-if)#ip address 10.3.0.0 255.255.0.0
erx2:vrB(config-if)#exit

```

5. Last, on erx3, create two IPSec tunnels, one to carry customer A's traffic and another to carry customer B's traffic.

Virtual router A:

```

erx3(config)#virtual-router vrA
erx3:vrA(config)#

```

Tunnel from Boston to Ottawa on virtual router A:

```
erx3:vrA(config)#interface tunnel ipsec:Aboston2ottawa transport-virtual-router
default
erx3:vrA(config-if)#tunnel transform-set customerAprotection
erx3:vrA(config-if)#tunnel local-identity subnet 10.3.0.0 255.255.0.0
erx3:vrA(config-if)#tunnel peer-identity subnet 10.1.0.0 255.255.0.0
erx3:vrA(config-if)#tunnel source 5.3.0.1
erx3:vrA(config-if)#tunnel destination 5.1.0.1
erx3:vrA(config-if)#ip address 10.1.0.0 255.255.0.0
erx3:vrA(config-if)#exit
```

Tunnel from Boston to Boca on virtual router A:

```
erx3:vrA(config)#interface tunnel ipsec:Aboston2boca transport-virtual-router default
erx3:vrA(config-if)#tunnel transform-set customerAprotection
erx3:vrA(config-if)#tunnel local-identity subnet 10.3.0.0 255.255.0.0
erx3:vrA(config-if)#tunnel peer-identity subnet 10.2.0.0 255.255.0.0
erx3:vrA(config-if)#tunnel source 5.3.0.1
erx3:vrA(config-if)#tunnel destination 5.2.0.1
erx3:vrA(config-if)#ip address 10.1.0.0 255.255.0.0
erx3:vrA(config-if)#exit
```

Virtual router B:

```
erx3(config)#virtual-router vrB
erx3:vrB(config)#
```

Tunnel from Boston to Ottawa on virtual router B:

```
erx3:vrB(config)#interface tunnel ipsec:Bboston2ottawa transport-virtual-router
default
erx3:vrB(config-if)#tunnel transform-set customerBprotection
erx3:vrB(config-if)#tunnel local-identity subnet 10.3.0.0 255.255.0.0
erx3:vrB(config-if)#tunnel peer-identity subnet 10.1.0.0 255.255.0.0
erx3:vrB(config-if)#tunnel source 5.3.0.1
erx3:vrB(config-if)#tunnel destination 5.1.0.1
erx3:vrB(config-if)#ip address 10.1.0.0 255.255.0.0
erx3:vrB(config-if)#exit
```

Tunnel from Boston to Boca on virtual router B:

```
erx3:vrB(config)#interface tunnel ipsec:Bboston2boca transport-virtual-router default
erx3:vrB(config-if)#tunnel transform-set customerBprotection
erx3:vrB(config-if)#tunnel local-identity subnet 10.3.0.0 255.255.0.0
erx3:vrB(config-if)#tunnel peer-identity subnet 10.2.0.0 255.255.0.0
erx3:vrB(config-if)#tunnel source 5.3.0.1
erx3:vrB(config-if)#tunnel destination 5.2.0.1
erx3:vrB(config-if)#ip address 10.2.0.0 255.255.0.0
erx3:vrB(config-if)#exit
```

The configuration is complete. Customer A's traffic and customer B's traffic can flow through the public, or untrusted, IP network inside a tunnel, where each packet is encrypted and authenticated.

Monitoring IPSec

This section contains information about troubleshooting and monitoring IPSec.

System Event Logs

To troubleshoot and monitor IPSec, use the following system event logs:

- auditIpsec—Lower layers of IKE SA negotiations
- ikepi—Upper layers of IKE SA negotiations
- stTunnel—Secure tunnel interface

For more information about using event logs, see the *JunosE System Event Logging Reference Guide*.

show Commands

To view your IPSec configuration and to monitor IPSec tunnels and statistics, use the following **show** commands.

show ipsec ike-policy-rule

show ike policy-rule



NOTE: The **show ipsec ike-policy-rule** command replaces the **show ipsec isakmp-policy-rule** command, which may be removed completely in a future release.

- Use to display the configuration of IKE phase 1 policy rules.
- Field descriptions
 - Protection suite priority—Priority number assigned to the policy rule
 - encryption algorithm—Encryption algorithm used in the IKE policy: des, 3des
 - hash algorithm—Hash algorithm used in the IKE policy: SHA, MD5
 - authentication method—Authentication method used in the IKE policy: RSA signature, preshared keys
 - Diffie-Hellman group—Size of the Diffie-Hellman group: 768-bit, 1024-bit, 1536-bit
 - lifetime—Lifetime of SAs created with this policy: 60 to 86400 seconds
 - aggressive mode—Allowed or not allowed

- Example

```
host1#show ipsec ike-policy-rule
```

```
IKE Policy Rules:
```

```
Protection suite priority: 5
```

```
  encryption algorithm :3DES Triple Data Encryption Standard(168 bit keys)
```

```
    hash algorithm      :SHA Secure Hash Standard
```

```

authentication method:RSA Signatures
Diffie-Hellman group :5 (1536 bit)
lifetime              :7200 seconds
aggressive mode       :Not Allowed

Protection suite priority: 6
  encryption algorithm :3DES Triple Data Encryption Standard(168 bit
keys)
  hash algorithm       :SHA Secure Hash Standard
  authentication method:Pre Shared Keys
  Diffie-Hellman group :2 (1024 bit)
  lifetime              :28800 seconds
  aggressive mode       :Not Allowed

```

- See `show ipsec ike-policy-rule`.
- See `show ike policy-rule`.

show ipsec ike-sa

show ike sa



NOTE: The `show ipsec ike-sa` command replaces the `show ike sa` command, which may be removed completely in a future release.

- Use to display IKE phase 1 SAs running on the router.
- Field descriptions
 - Local:Port—Local IP address and UDP port number of phase 1 negotiation
 - Remote:Port—Remote IP address and UDP port number of phase 1 negotiation
 - Time(Sec)—Time remaining in phase 1 lifetime, in seconds
 - State—Current state of the phase 1 negotiation. Corresponds to the messaging state in the main mode and aggressive mode negotiations. Possible states are:
 - AM_SA_I—Initiator has sent initial aggressive mode SA payload and key exchange to the responder
 - AM_SA_R—Responder has sent aggressive mode SA payload and key exchange to the initiator
 - AM_FINAL_I—Initiator has finished aggressive mode negotiation
 - AM_DONE_R—Responder has finished aggressive mode negotiation
 - MM_SA_I—Initiator has sent initial main mode SA payload to the responder
 - MM_SA_R—Responder has sent a response to the initial main mode SA
 - MM_KE_I—Initiator has sent initial main mode key exchange to the responder
 - MM_KE_R—Responder has sent a response to the key exchange

- MM_FINAL_I—Initiator has sent the final packet in the main mode negotiation
- MM_FINAL_R—Responder has finished main mode negotiation
- MM_DONE_I—Initiator has finished main mode negotiation
- DONE—Phase 1 SA negotiation is complete, as evidenced by receipt of some phase 2 messages
- Local Cookie—Unique identifier (SPI) for the local phase 1 IKE SA
- Remote Cookie—Unique identifier (SPI) for the remote phase 1 IKE SA
- Example

```
host1# show ipsec ike-sa
IKE Phase 1 SA's:
```

Local:Port	Remote:Port	Time(Sec)	State	Local Cookie	Remote Cookie
195.0.0.100:500	195.0.0.200:500	1551	DONE	0x90ee723e6cb0c016	0xf7d3651e93d56431
195.0.0.100:500	195.0.0.200:500	1552	DONE	0x821bccf81dcedbb0	0x35152bdb7a9c734e
195.0.1.100:500	195.0.1.200:500	1687	DONE	0x1b4fbcebe36d1b16	0xed742166a305a6a0
195.0.1.100:500	195.0.1.200:500	1687	DONE	0xacf3acd1b3555b6a	0x0af9edbc95622869
195.0.2.100:500	195.0.2.200:500	1688	DONE	0x3153379b32d8c936	0x17f5d77f9badc3cf
195.0.2.100:500	195.0.2.200:500	1688	DONE	0x6573dcbc9bf31fae	0x7af8b4d13078b463
195.0.3.100:500	195.0.3.200:500	1685	DONE	0xdc7df648fcac375a	0x0346752d2881d5c5
195.0.3.100:500	195.0.3.200:500	1685	DONE	0xe776e9ffb6678635	0x8de857af1c681874
195.0.4.100:500	195.0.4.200:500	1690	DONE	0x16410d890500e94e	0xbd47831b55e81c27

- See show ipsec ike-sa.
- See show ike sa.

show ipsec lifetime

- Use to display the configured IPSec default lifetime.
- Example

```
host1#show ipsec lifetime
Default lifetime in seconds is '7200'.
Default lifetime in kilobytes is '4294967295'.
```

- See show ipsec lifetime.

show ipsec local-endpoint

- Use to display the address and transport virtual router of local endpoints.
- To display the local endpoint of a specific transport virtual router, include the virtual router name.
- Example

```
host1#show ipsec local-endpoint transport-virtual-router default
Local endpoint for transport-virtual-router default is '0.0.0.0'.
```

- See show ipsec local-endpoint.

show ipsec option

- Use to display the status, enabled or disabled, of IPsec options configured on the current virtual router. Information is displayed for the following options:
 - Dead peer detection (DPD)
 - Network Address Translation Traversal (NAT-T). For information about configuring and monitoring NAT-T on L2TP/IPsec tunnels, see [“Securing L2TP and IP Tunnels with IPsec” on page 277](#).
 - Transmission of invalid cookie notification in ISAKMP messages to peers
- Example

```
host1:vrA#show ipsec option

IPsec options:
Dead Peer Detection: disabled
NAT Traversal      : enabled
TX Invalid Cookie  : disabled
```
- See show ipsec option.

show ipsec transform-set

- Use to display transform sets configured on the router.
- To display a specific transform set, include the transform set name.
- Field descriptions
 - Transform-set—Displays the transforms in the transform set
- Example 1

```
host1#show ipsec transform-set
Transform-set: Highest security = {esp-3des-hmac-sha }.
Transform-set: transform-esp-3des-hmac-sha = {esp-3des-hmac-sha }.
```
- Example 2

```
host1#show ipsec transform-set transform-esp-3des-hmac-sha
Transform-set: transform-esp-3des-hmac-sha = {esp-3des-hmac-sha}.
```
- See show ipsec transform-set.

show ipsec tunnel detail

- Use to display the running configuration and statistics for each tunnel.
- Field descriptions
 - IPSEC tunnel—Name and state of tunnel for which information is displayed
 - Tunnel operational configuration—Configuration running on the tunnel
 - Tunnel type—Manual, signaled
 - Tunnel mtu—MTU size of the tunnel

- Tunnel localEndpoint—IP address of local tunnel endpoint
- Tunnel remoteEndpoint—IP address of remote tunnel endpoint
- Tunnel source—IP address or FQDN of tunnel source
- Tunnel destination—IP address or FQDN of tunnel destination
- Tunnel backup destination—Alternate tunnel destination
- Tunnel transport virtual router—Name of transport virtual router over which tunnel runs
- Tunnel transform set—Tunnel transform set in use on this tunnel
- Tunnel local identity—IP address of local endpoint identity that ISAKMP uses
- Tunnel peer identity—IP address of peer endpoint identity that ISAKMP uses
- Tunnel outbound spi/SA—SPI and SA in use on traffic sent to the tunnel (manual tunnels only)
- Tunnel inbound spi/SA—SPI and SA in use on traffic received from the tunnel (manual tunnels only)
- Tunnel lifetime seconds—Configured time-based lifetime in seconds
- Tunnel lifetime kilobytes—Configured traffic-based lifetime in kilobytes
- Tunnel pfs—PFS group in use on the tunnel: 0 (PFS is not in use), 1 (768-bit group), 2 (1024-bit group), 5 (1536-bit group)
- Tunnel administrative state—Up, Down
- Tunnel Operational Attributes—Displays statistics related to the tunnel lifetime
 - inbound/outboundSpi/SA—SPI in use on traffic received from or sent to the tunnel
 - inbound/outboundSa—SA in use on traffic received from or sent to the tunnel
 - inbound/outbound lifetime allowed—Negotiated time-based lifetime in seconds
 - inbound/outbound lifetime remaining—Number of seconds remaining before time-based lifetime expires
 - inbound/outbound traffic allowed—Negotiated traffic-based lifetime in kilobytes
 - inbound/outbound traffic remaining—Number of additional kilobytes that tunnel can send or receive before traffic-based lifetime expires
- Tunnel Statistics—Displays statistics on traffic received on and sent from this tunnel
 - InUserPackets—Number of user packets received
 - InUserOctets—Number of octets received from user packets

- InAccPackets—Number of encapsulated packets received
 - InAccOctets—Number of octets received in encapsulated packets
 - InAuthErrors—Number of authentication errors received
 - InReplayErrors—Number of replay errors in received traffic
 - InPolicyErrors—Number of policy errors in received traffic
 - InOtherRxErrors—Number of packets received that have errors other than those listed above
 - InDecryptErrors—Number of decryption errors in received traffic
 - InPadErrors—Number of packets received that had invalid values after the packet was decrypted
 - OutUserPackets—Number of user packets sent
 - OutUserOctets—Number of octets sent in user packets
 - OutAccPackets—Number of encapsulated packets sent
 - OutAccOctets—Number of octets sent in encapsulated packets
 - OutPolicyErrors—Number of packets arriving at tunnel for encapsulation that do not meet specified tunnel identifier (selector)
 - OutOtherTxErrors—Number of outbound packets that have errors other than those listed above
- Example

```
host1#show ipsec tunnel detail
IPSEC tunnel r200000 is Up
Tunnel configuration:
  Tunnel type is signaled
  Tunnel mtu is 1440
  Tunnel local endpoint is 195.0.0.200
  Tunnel remote endpoint is 195.0.0.100
  Tunnel source is 195.0.0.200
  Tunnel destination is 195.0.0.100
  Tunnel backup destination is 0.0.0.0
  Tunnel transport virtual router is r
  Tunnel transform set is perf
  Tunnel local identity is ipAddress: 4.0.0.100
  Tunnel peer identity is ipAddress: 3.0.0.100
  Tunnel lifetime seconds is 7200
  Tunnel lifetime kilobytes is 1024000
  Tunnel pfs is group 5
  Tunnel administrative state is Up

Tunnel Operational Attributes:
  inboundSpi = 0x17270202, inboundSa = esp-3des-hmac-sha
  inbound lifetime: allowed 7200s, remaining 7100s
  inbound traffic: allowed 1024000KB, remaining 1023997KB
```

```

outboundSpi = 0x283b0201, outboundSa = esp-3des-hmac-sha
outbound lifetime: allowed 7200s, remaining 7100s
outbound traffic: allowed 1024000KB, remaining 1023997KB

```

```

Tunnel Statistics:
InUserPackets      15
InUserOctets       1920
InAccPackets       15
InAccOctets        2760
InAuthErrors       0
InReplayErrors     0
InPolicyErrors     0
InOtherRxErrors    0
InDecryptErrors    0
InPadErrors        0

OutUserPackets     15
OutUserOctets      1920
OutAccPackets      15
OutAccOctets       2760
OutPolicyErrors    0
OutOtherTxErrors   0

```

- See show ipsec tunnel.

show ipsec tunnel summary

- Use to display a summary of all tunnels configured on the router.
- Field descriptions
 - Total number of ipsec interface—Number of tunnels configured on the router
 - Administrative status—Number of tunnels with an administrative status of enabled and disabled
 - Operational status—Number of tunnels with an operational status of up, down, lower layer down, not present
- Example

```

host1#show ipsec tunnel summary
Total number of ipsec interface is 40
Administrative status   enabled   disabled
                        40           0
Operational status    up        down    lower-down  not-present
                      40          0          0          0

```

- See show ipsec tunnel.

show ipsec tunnel virtual-router

- Use to display the status of tunnels configured on a virtual router.
- To display only tunnels that are in a specific state, use the **state** keyword.
- To display tunnels that are using a particular IP address, use the **ip** keyword.
- Field descriptions

- For a description of fields, see the **show ipsec tunnel detail** command.
- Example

```
host1#show ipsec tunnel virtual-router default ip 10.255.1.13
IPSEC tunnel s011e3d0 is up
IPSEC tunnel s011e3d1 is up
IPSEC tunnel s012e3d0 is up
IPSEC tunnel s012e3d1 is up
IPSEC tunnel s013e3d0 is up
IPSEC tunnel s014e3d0 is up
IPSEC tunnel s014e3d1 is up
IPSEC tunnel s015e3d0 is up
```

- See show ipsec tunnel.

show license ipsec-tunnels

- Use to display the IPSec license key configured on the router and the number of tunnels allowed on the router.
 - Example
- ```
host1#show license ipsec-tunnels
ipsec-tunnels license is 'g1k23b23eb2j' which allows 5000 tunnels with 1
IPsec card and 7500 tunnels with 2 or more IPsec cards.
```
- See show license.



## CHAPTER 6

# Configuring Dynamic IPSec Subscribers

This chapter describes how to securely terminate IPSec remote access subscribers. These subscribers can reside on different VPNs and the router can support many VPNs simultaneously. It contains the following sections:

- [Overview on page 169](#)
- [Platform Considerations on page 172](#)
- [References on page 173](#)
- [Creating an IPSec Tunnel Profile on page 173](#)
- [Configuring IPSec Tunnel Profiles on page 174](#)
- [Defining IKE Policy Rules for IPSec Tunnels on page 180](#)
- [Monitoring IPSec Tunnel Profiles on page 182](#)

## Overview

---

You can use the E Series router to terminate users on multiple VPNs (that is, a private intranet where users can log in and access private servers). For the E Series router, VPNs appear as VRs or VRFs. Users that connect to the VPN terminate on the associated VR or VRF. The router contains a link between the VR or VRF and the private intranet containing the resources. This link can be a direct connection, or a tunnel (IPSec, IP-in-IP, GRE, or MPLS). Once establishing a connection, the router can pass traffic between the VPN and connected users.

The E Series router already supports termination of secure remote access subscribers using L2TP and IPSec. In this model, IPSec uses transport mode to “protect” PPP subscribers that use L2TP tunnels as described in RFC 3193. However, because they are handled by the PPP and L2TP application, IPSec has no direct information about the subscribers. By terminating dynamic IPSec subscribers, the IPSec protocol manages the subscribers completely.

## Dynamic Connection Setup

Dynamic secure remote access subscribers initiate connections to the E Series router by establishing an IPSec phase 1 security association (SA; also known as an IKE SA or P1) with the router.

After establishing a security association, the subscriber is instantiated in the IPSec software. Following this instantiation, the router initiates the extended authentication (Xauth) protocol exchange to invoke the user to enter a username and password. The router uses existing authentication, authorization, and accounting (AAA) functionality to authenticate the user data.

After granting access, the router instantiates an IP interface for the new subscriber as well as an access route for the IP address assigned to the subscriber on the terminating virtual router. The subscriber also obtains IP interface data (IP address, subnetwork mask, primary and secondary DNS address, primary and secondary WINS address, and so on) during a configuration exchange.

Once instantiated, an access router created, and the client successfully set with interface data parameters, the router can terminate the Xauth exchange and enable the IPSec layer and phase 2 SAs (IPSec SAs or P2s) can begin. Following these exchanges, the full data path is ready and subscribers can exchange packets with the VR on which they terminate.

## Dynamic Connection Teardown

The following events can trigger the teardown of a dynamic IPSec subscriber connection:

- All phase 1 and phase 2 SA deleted by a remote peer and no rekeying activity occurs for one minute
- Administrative logout
- IPSec card terminating the user becoming unavailable (for example, the card is reloading, disabled, or disconnected)
- Dead peer detection (DPD) reporting the phase 1 SA is unreachable
- Authentication, authorization, and accounting session or idle timeout values expire

## Dynamic IPSec Subscriber Recognition

The E Series router expects to receive the Xauth vendor ID from the remote peer for dynamic interface instantiation. The expected Xauth vendor ID is 0x09002689DFD6B712.



**NOTE:** The E Series router does not initiate connections to new subscribers. Acceptable vendor IDs are global to the router and not user-configurable.

Phase 2 SAs intended for static tunnels and those intended for dynamic subscribers do not share the same phase 1 SA. This means that dynamic phase 1 SAs are only used to negotiate dynamic phase 2 SAs. Conversely, phase 1 SAs that are not recognized as dynamic are used only to negotiate phase 2 SA static tunnels.

## Licensing Requirements

Each dynamic IPSec subscribers requires the use of two licenses:

- One B-RAS license

- One IPSec license

If either license is unavailable, the router denies access to the subscriber.

### Inherited Subscriber Functionality

Dynamic IPSec subscribers inherit much of the built-in AAA subscriber management functionality. This functionality includes the following:

- AAA subscriber management commands
- DNS (primary and secondary)
- WINS (primary and secondary)
- Session timeout
- Accounting features (interval, duplication, immediate update, broadcasting, Acct-stop)
- Duplicate address checking
- IP address pools
- Per virtual-router subscriber limit
- Policies
- Packet mirroring

For additional information on AAA functionality, see *JunosE Broadband Access Configuration Guide*.

### Using IPSec Tunnel Profiles

IPSec tunnel profiles serve the following purposes in the configuration of dynamic IPSec subscribers:

- Controlling which connecting user, based on the IKE identification, belongs to a given profile. Profile settings falling in this category include the following:
  - IKE identities from peers that can use this profile. These identities include IP addresses, domain names, and E-mail addresses. In addition, distinguished names that use X.509 certificates are permitted.
  - The router IKE identity.
- Terminating extraneous security and IP profile settings that exist after a subscriber is mapped to an IPSec tunnel. These settings include the following:
  - Maximum number of subscribers that this profile can terminate
  - AAA domain suffix intended for the username (helping to bridge users from a given IPSec tunnel profile to an AAA domain map)
  - Phase 2 SA selectors for use in phase 2 SA exchanges
  - IP profiles intended for users logging in using this profile (helping to bridge users from a given IPSec tunnel profile to an IP profile)

- Reachable networks on the VPN (allowing for split tunneling when supported by the client software)
- Security parameters intended to protect user traffic (including IPSec encapsulating protocol, encryption algorithms, authentication algorithms, lifetime parameters, perfect forward secrecy, and DH group for key derivation)
- Setting the IP address the router monitors for remote subscribers.

New subscribers are mapped only to IPSec tunnel profiles after the initial IKE SA is established. Like IPSec tunnels, IKE policy rules are required to control IKE SA acceptance and denial.

## Relocating Tunnel Interfaces

Unlike static IPSec tunnels interfaces, dynamic IPSec subscribers do not relocate if the IPSec server card becomes unavailable. If the IPSec server card becomes unavailable, all dynamic subscribers that are logged in and located on that server card are logged out and must log back in to connect.

## User Authentication

For IPSec subscribers, user authentication occurs in two phases. The first phase is an IPSec-level authentication (phase 1 or IKE authentication). Sometimes referred to as “machine” authentication, because the user PC is authenticated, the first authentication phase verifies private or preshared keys that reside on the PC. These keys are not easily moved from one PC to another and do not require user entry each time authentication is performed.

Depending on the IKE phase 1 exchange, restrictions on the authentication type or the access network setup might exist. To avoid any usage problems, keep the following in mind:

- If you are configuring a VPN where users perform preshared key IPSec authentication and use the IKE main mode exchange for phase 1, you must setup the access network such that the VPN has an exclusive local IP address.
- If you want to share a single server address on the access network for more than one VPN, you must either set the clients to use IKE aggressive mode or use a public and private key pair for authentication. This authentication type includes X.509v3 certificates).

After the IPSec-level authentication takes place, a user authentication occurs. Often considered a legacy form of authentication, the user authentication (like RADIUS) typically requires the user to enter information in the form of a username and password.

## Platform Considerations

---

For information about modules that support dynamic IPSec subscribers on the ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router:

- See IPSec Service support in *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See IPSec Service support in *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support IPSec service.

## References

---

For more information about dynamic IPSec subscribers, consult the following resources:

- The ISAKMP Configuration Method—draft-dukes-ike-mode-cfg-02.txt (March 2002 expiration)
- Extended Authentication within IKE (XAUTH)—draft-beaulieu-ike-xauth-02.txt (April 2002 expiration)
- Extended Authentication within ISAKMP/Oakley (XAUTH)—draft-ietf-ipsec-isakmp-xauth-06.txt (May 2000 expiration)



**NOTE:** IETF drafts are valid for only 6 months from the date of issuance. They must be considered as works in progress. Please refer to the IETF Web site at <http://www.ietf.org> for the latest drafts.

---

For additional configuration information, see:

- “Configuring IPSec” on page 119
- “Configuring Digital Certificates” on page 205
- “Configuring IP Tunnels” on page 237
- *JunosE Broadband Access Configuration Guide*

## Creating an IPSec Tunnel Profile

---

To create an IPSec tunnel profile, use the **ipsec tunnel profile** command. This command creates a tunnel profile of the name you specify and accesses the IPSec Tunnel Profile configuration mode (config-ipsec-tunnel-profile).

### *ipsec tunnel profile*

- Use to create or configure a tunnel profile for IPSec and accesses the IPSec Tunnel Profile configuration mode (config-ipsec-tunnel-profile). To create a new profile, you must specify a profile name.
- Use the optional **virtual-router** keyword to specify the name of the virtual router on which you want to create the profile (if you do not specify a virtual router name, the profile is created on the context virtual router)
- Example

```
host1(config)#ipsec tunnel profile tunnel1
host1(config-ipsec-tunnel-profile)#
```

- Use the **no** version to delete the tunnel profile.
- See ipsec tunnel profile.

## Configuring IPSec Tunnel Profiles

---

This sections explains how to configure the parameters that exist in the IPSec tunnel profile configuration mode.

### Limiting Interface Instantiations on Each Profile

To define the maximum number of interfaces that the IPSec tunnel profile can instantiate, use the **max-interfaces** command. Once the profile reaches the maximum number of interfaces, the profile rejects any new interface instantiations and generates a warning-level log. The default value (using the **no** version of the command) specifies unlimited interface instantiation on a given profile.

#### *max-interfaces*

- Use to define the maximum number of interfaces that the IPSec tunnel profile can instantiate.
- Example

```
host1(config-ipsec-tunnel-profile)#max-interfaces 500
```
- Use the **no** version to return the maximum value to unlimited, indicating no limit to the number of interfaces that can be instantiated on this profile.
- See max-interfaces.

## Specifying IKE Settings

This section describes how to define the IKE local identity and IKE peer identity values.

### Setting the IKE Local Identity

---

To set the IKE local identity (phase 1 identity) used for IKE security association negotiations, use the **ike local-identity** command.



**NOTE:** The authentication algorithm for an IKE SA is associated with its identity. You must ensure that the client and server are set accordingly to successfully establish IKE security associations.

---

#### *ike local-identity*

- Use to set the IKE local identity used for IKE security association (SA) negotiations.
- Example

```
host1(config-ipsec-tunnel-profile)#ike local-identity domain-name domain1
```

- Use the **no** version to remove the specified IKE local identity.
- See `ike local-identity`.

### Setting the IKE Peer Identity

To set the IKE peer identity values, use the **ike peer-identity** command. You can set the profile to accept logins from users that present one of the following:

- An `asn1DN` as an IKE identity type (an ASN.1-encoded distinguished name) and the user-provided IKE identity contains the substring configured for the profile.
- A `userFQDN` or `FQDN` as an IKE identity type and the domain name portion of the IKE identity matches the domain name setting for this profile. An empty string (default) means that IKE identity types of `userFQDN` and `FQDN` are not allowed for logins on this profile.

The IKE identity type of `userFQDN` also carries a domain name. Users presenting this identity must also pass any restrictions set for the peer domain name for this profile before they are able to log in.

- An IP address as an IKE identity type and the IP address resides within the specified network. The default of `0.0.0.0/0` allows any peer IP address to this profile.
- A `userFQDN` as an IKE identity type and the username portion of the IKE identity matches the username setting for this profile. An empty string (default) means that an IKE identity type of `userFQDN` is not allowed for logins on this profile.



**NOTE:** You can also use the wildcard (\*) for the username and domain name or as the first or last character in the username or domain name string.

*ike peer-identity distinguished-name*

*ike peer-identity domain-name*

*ike peer-identity ip address*

*ike peer-identity username*

- Use to set the IKE peer identity used for IKE security association (SA) negotiations.
- Example

```
host1(config-ipsec-tunnel-profile)#ike peer-identity domain-name domain2
```

- Use the **no** version to remove the specified IKE peer identity.
- See `ike peer-identity distinguished-name`.
- See `ike peer-identity domain-name`.
- See `ike peer-identity ip address`.
- See `ike peer-identity username`.

## Appending a Domain Suffix to a Username

The VPN to which a user is to be terminated is sometimes known from the IKE identities attached to the user. However, to assist in connecting users to the correct AAA domain for authentication, you can use the **domain-suffix** command to append a domain suffix to the username. Using the default, no domain suffix, passes usernames transparently to AAA.

### *domain-suffix*

- Use to specify a domain suffix that you want to append to any usernames received on this profile.
- Example

```
host1(config-ipsec-tunnel-profile)#domain-suffix domain2
```
- Use the **no** version to restore the default value, no domain suffix, and usernames are passed transparently to AAA.
- See domain-suffix.

## Overriding IPSec Local and Peer Identities for SA Negotiations

You can use the **local ip identity** and **peer ip identity** commands to override the local and peer identities used for SA negotiations (respectively).

### *local ip identity*

- Use to override the local identity (phase 2 identity) used for IPSec security association negotiations. For IPSec negotiations to succeed, the local and peer identities at one end of the tunnel must match the peer and local identities at the other end (respectively).
- Example

```
host1(config-ipsec-tunnel-profile)#local ip identity range 10.30.11.1 10.30.11.50
```
- Use the **no** version to restore the default value, the internal IP address allocated for the subscriber.
- See local ip identity.

### *peer ip identity*

- Use to override the peer identity (phase 2 identity) used for IPSec security association negotiations. For IPSec negotiations to succeed, the local and peer identities at one end of the tunnel must match the peer and local identities at the other end (respectively).
- Example

```
host1(config-ipsec-tunnel-profile)#peer ip identity address 10.227.1.2
```



- Use the **no** version to restore the default value, the internal IP address allocated for the subscriber.
- See peer ip identity.

## Specifying an IP Profile for IP Interface Instantiations

The **ip profile** command specifies the IP profile that is passed from the IPSec layer to the IP layer upon request for upper layer instantiation.

### *ip profile*

- Use to specify the IP profile that the IPSec layer passes on to the IP layer upon request for upper-layer instantiation.
- Example

```
host1(config-ipsec-tunnel-profile)#ip profile ipProfile1
```
- Use the **no** version to remove the association with this profile.
- See ip profile.

## Defining the Server IP Address

The **local ip address** command defines the specified local IP address as the server address. The router monitors UDP port 500 for incoming login requests (that is, IKE SA negotiations) from users.



**NOTE:** This address is typically made public to all users trying to connect to a VPN on this router.

This command enables you to optionally set a global preshared key for the specified server address. When using global preshared keys, keep the following in mind:

- Global preshared keys enable a group of users to share a single authentication key, simplifying the administrative job of setting up keys for multiple users.
- Specific keys for individual users have higher priority than global keys. If both individual and global keys are configured, the individual that also has a specific key must use that key or authentication fails.
- More than one profile can specify the same local endpoint and virtual router. Because the last value set overrides the other, we recommend that you avoid this type of configuration.

### *local ip address*

- Use to specify the given local IP address as a server address.
- Example

```
host1(config-ipsec-tunnel-profile)#local ip address 192.2.52.12
```

- Use the **no** version to stop the router from monitoring UDP port 500 for user requests and remove any preshared key associations with the local IP address.
- See local ip address.

## Specifying Local Networks

The **local ip network** command enables you to specify local, reachable networks through the IPSec tunnel. This type of “split tunneling” enables a remote station to separate VPN traffic from Internet traffic. For example a client connecting to a corporate Intranet could use split-tunneling to send all traffic destined to 10.0.0.0/8 through the secure tunnel and reach the VPN. Other traffic (for example, Web browsing) would travel directly to the Internet through the local service provider without passing through the tunnel.



**NOTE:** Split tunneling functions only when supported by the client software. It is up to the client to modify its routing table with the network information for split tunneling to occur

### *local ip network*

- Use to specify networks that are reachable through the IPSec tunnel. You can configure up to 16 networks for this method of “split-tunneling.”
- Example

```
host1(config-ipsec-tunnel-profile)#local ip network 10.0.0.0 255.255.255.252
```
- Use the **no** version to remove the specified network from the reachable list.
- See local ip network.

## Defining IPSec Security Association Lifetime Parameters

The **lifetime** command defines the IPSec SA lifetime parameters the tunnel profile can use for IPSec SA negotiations. These parameters include the phase 2 lifetime as a range in seconds or traffic volume.

### *lifetime*

- Use to specify the IPSec lifetime parameters used on IPSec SA lifetime negotiations.
- Example

```
host1(config-ipsec-tunnel-profile)#lifetime seconds 5000 25000
```
- Use the **no** version to return the lifetime to its default value, 28800 seconds (8 hours) and no traffic volume limit.
- See lifetime.

## Defining User Reauthentication Protocol Values

The **extended-authentication** command specifies the extended user authentication protocol for use during the extended user authentication protocol exchange.

The **re-authenticate** keyword enables the reauthentication option (a subsequent authentication procedure). When this option is enabled, rekeying of IKE SAs uses the initial authentication protocol to reauthenticate the user. When this option is disabled, authentication is only performed at the first IKE SA establishment. Subsequent IKE SAs rekey operations inherit the initial authentication and do not reauthenticate users.



**NOTE:** For maximum security, enable reauthentication.

The **skip-peer-config** keyword disables the router from configuring peer IP characteristics.

### *extended-authentication*

- Use to specify the extended user authentication protocol for use during the extended user authentication protocol exchange. This command can also enable or disable the reauthentication option (a subsequent authentication procedure).
- The **re-authenticate** keyword enables the reauthentication option (a subsequent authentication procedure).
- The **skip-peer-config** keyword disables the router from configuring peer IP characteristics.
- Example  

```
host1(config-ipsec-tunnel-profile)#extended-authentication chap
```
- Use the **no** version to reset the extended authentication to the default protocol, pap.
- See extended-authentication.

## Specifying IPSec Security Association Transforms

The **transform** command specifies the IPSec transforms that IPSec SA negotiations can use for this profile. The router accepts the first transform proposed by a client that matches one of the transforms specified by this command. During an IPSec SA exchange with a client, the router proposes all transforms specified by this command and one is accepted by the client.



**NOTE:** You can specify up to six transform algorithms for this profile.

For additional information about transforms and transform sets, see [“Configuring IPSec” on page 119](#).

### *transform*

- Use to specify the eligible transforms for this profile for IPSec security association negotiations.
- Example

```
host1(config-ipsec-tunnel-profile)#transform ah-hmac-md5
```
- Use the **no** version to reset the transform to the default, esp-3des-sha1.
- See transform.

## Specifying IPSec Security Association PFS and DH Group Parameters

The **pfs group** command specifies the IPSec SA perfect forward secrecy (PFS) option and Diffie-Hellman prime modulus group that IPSec SA negotiations can use for this profile.



**NOTE:** When the client initiates the IPSec negotiation, the router can accept Diffie-Hellman prime modulus groups that are higher than those configured.

For additional information about PFS, see [“Configuring IPSec” on page 119](#).

### *pfs group*

- Use to configure perfect forward secrecy for connections created with this IPSec tunnel configuration profile by assigning a Diffie-Hellman prime modulus group.
- Example

```
host1(config-ipsec-tunnel-profile)#pfs group 5
```
- Use the **no** version to remove PFS from the profile.
- See pfs group.

## Defining the Tunnel MTU

The **tunnel mtu** command configures the maximum transmission unit size for the tunnel.

### *tunnel mtu*

- Use to configure the maximum transmission unit size for the tunnel.
- Example

```
host1(config-ipsec-tunnel-profile)#tunnel mtu 3000
```
- Use the **no** version to restores the default value, an MTU size of 1400 bytes.
- See tunnel mtu.

---

## Defining IKE Policy Rules for IPSec Tunnels

This section describes enhancements to some IKE policy rule commands to support dynamic IPSec subscribers.

## Specifying a Virtual Router for an IKE Policy Rule

The **ip address virtual-router** command enables an IKE policy rule to limit its scope to a specific local IP address on a specific virtual router. When enabled, this limitation ensures that this policy rule is evaluated for IKE security association evaluations for only the specified IP address and virtual router.

When initiating and responding to an IKE SA exchange, the router evaluates the possible policy rules as follows:

- If an IP-address-specific IKE policy rule refers to the local IP address and virtual router for this exchange, the router evaluates this policy rule before any non-IP-address-specific IKE policy rules. If more than one IP-address-specific IKE policy rule exists, the router evaluates the policy rule with the lowest priority number first and then evaluates the policy rule with the next highest priority number and so on.
- If no IP-address-specific IKE policy rule refers to the local IP address and virtual router for this exchange, the router evaluates all non-IP-address-specific IKE policy rules in the normal IKE policy rule evaluation order.

You can define an IKE policy rule without specifying an IP address or virtual router (the default). When not specifically configured, the IKE policy rule remains valid for any local IP address on any virtual router residing on the router.

### *ip address virtual-router*

- Use to limit the scope of the IKE policy rule to the specified local IP address on the specified virtual router. This limitation ensures that this policy rule is evaluated for IKE security association evaluations for only the specified IP address and virtual router.
- Example

```
host1(config-ike-policy)#ip address virtual-router VR1
```
- Use the **no** version to remove the IP address and virtual router limitation.
- See `ip address virtual-router`.

## Defining Aggressive Mode for an IKE Policy Rule

The **aggressive-mode** command enables aggressive mode negotiation for the tunnel. For additional information about aggressive mode and how it works, see [“Main Mode and Aggressive Mode” on page 135](#).

### *aggressive-mode*

- Use to enable aggressive mode negotiation for the tunnel.
- If you specify aggressive mode negotiation, the tunnel proposes aggressive mode to the peer in connections that the policy initiates.
- If the peer initiates a negotiation, the tunnel accepts the negotiation if the mode matches this policy.
- Use the **accepted** keyword to accept aggressive mode when proposed by peers

- Use the **requested** keyword to request aggressive mode when negotiating with peers
- Use the **required** keyword to only request and accept aggressive mode when negotiating with peers.
- Example

```
host1(config-ike-policy)#aggressive-mode accepted
```
- Use the **no** version to set the negotiation mode to main mode.
- See aggressive-mode.

## Monitoring IPSec Tunnel Profiles

---

This section contains information about troubleshooting and monitoring dynamic IPSec subscribers.

### System Event Logs

To troubleshoot and monitor dynamic IPSec subscribers, use the following system event log:

- ipsecIdDb—IPsec ID database
- ipsecXcfgSM—IPsec Xauth/ModeCfg state machine
- ipsecP1Throttler—Ongoing Phase 1 negotiations

For more information about using event logs, see the *JunosE System Event Logging Reference Guide*.

### show Commands

To display user information for dynamic IPSec tunnel profiles or subscribers, use the following **show** commands.

#### *show ipsec tunnel profile*

- Use to display information about all existing IPSec tunnel profiles or a specified tunnel profile.
- Use the **detail** keyword to display detailed information about the tunnel profile.
- Example 1

```
host1#show ipsec tunnel profile
IPsec tunnel profile ipsec-spg is active with no subscriber
1 IPsec tunnel profile found
```

- Example 2

```
host1#show ipsec tunnel profile detail ipsec-spg
IPsec tunnel profile ipsec-spg is active with no subscriber
Extended-authentication: pap, no re-authentication
Peer IP characteristics configuration: enabled
Virtual router: default
Local IP address: 10.227.5.31
Local IKE identity: 10.227.5.31
```

```

Peer IKE identity: IP network: not allowed
 username: *
 domain-name: spg.juniper.net
 DN: not allowed
Maximum subscribers: no limit
Domain suffix: @spg
IP profile: ip-spg
Local IPsec identity: subnet 0.0.0.0 0.0.0.0, proto 0, port 0
Peer IPsec identity: invalid identity
Lifetime: between 1800 and 7200 seconds, and between 100000 and 500000
KB
Reachable networks: none
PFS not configured
Transforms: , tunnel-esp-3des-sha1
Subscribers rejected due to maximum subscribers limit: 0
Completed sessions: 43, totaling 4873 seconds, statistics:
ipsec stats:
 outbound:
 outboundUserPacketsReceived = 88
 outboundUserOctetsReceived = 74544
 outboundAccPacketsReceived = 88
 outboundAccOctetsReceived = 79168
 outboundOtherTxErrors = 0
 outboundPolicyErrors = 0
 inbound:
 inboundUserPacketsReceived = 88
 inboundUserOctetsReceived = 74880
 inboundAccPacketsReceived = 88
 inboundAccOctetsReceived = 79488
 inboundAuthenticationErrors= 0
 inboundReplayErrors = 0
 inboundPolicyErrors = 0
 inboundOtherRxErrors = 0
 inboundDecryptErrors = 0
 inboundPadErrors = 0

```

- See show ipsec tunnel profile.

### ***show subscribers***

- Use to display the active subscribers on the router.
- Field descriptions
  - User Name—Name of the subscriber
  - Type—Type of subscriber: atm, ip, ipsec, ppp, tnl (tunnel), tst (test)
  - Addr | Endpt—IP or IPv6 address and source of the address: l2tp, local, dhcp, radius, user. For local, dhcp, radius, and user endpoints, the address is that of the user. When the endpoint is l2tp, the address is that of the LNS.
  - Virtual Router—Name of the virtual router context
  - Interface—Interface specifier over which the subscriber is connected
  - Login Time—Date, in YY/MM/DD format, and time the subscriber logged in

- Circuit Id—User's circuit ID value specified by PPPoE
- Remote Id—User's remote ID value specified by PPPoE
- Example

```
host1#show subscribers
```

```

Subscriber List

User Name Type Addr|Endpt Virtual

xcfgUser1@vpn1 ipsec 10.227.5.106/loca Virtual
User Name Interface Router

xcfgUser1@vpn1 FastEthernet 5/2.4
User Name Login Time Circuit Id

xcfgUser1@vpn1 06/05/12 10:58:42 0.4.1.10.fe.25.3b.0
User Name Remote Id

xcfgUser1@vpn1 (800) 555-1212

```

- See show subscribers.



## CHAPTER 7

# Configuring ANCP

This chapter describes how to configure Access Node Control Protocol (ANCP), also known as Layer 2 Control (L2C), for IP multicast on an E Series router; it contains the following sections:

- [Overview on page 185](#)
- [Platform Considerations on page 187](#)
- [References on page 188](#)
- [Configuring ANCP on page 188](#)
- [Configuring ANCP Interfaces on page 189](#)
- [Configuring ANCP Neighbors on page 190](#)
- [Configuring Topology Discovery on page 192](#)
- [Configuring ANCP for QoS Adaptive Mode on page 192](#)
- [Triggering ANCP Line Configuration on page 193](#)
- [Adjusting the Data Rate Reported by ANCP for DSL Lines on page 194](#)
- [Configuring Transactional Multicast for IGMP on page 194](#)
- [Triggering ANCP OAM on page 197](#)
- [Monitoring ANCP on page 198](#)

## Overview

---

Access Node Control Protocol (ANCP), also known as Layer 2 Control (L2C), is based on a subset of the General Switch Management Protocol (GSMP), as defined in the GSMPv3 Base Specification (draft-ietf-gsmp-v3-base-spec-06.txt). GSMP is a general purpose protocol used to control a label switch.

GSMP enables a controller to establish and release connections across the switch, add and delete leaves on a multicast connection, manage switch ports, request configuration information, request and delete reservation of switch resources, and request statistics. It also enables the switch to inform the controller of asynchronous events such as a link going down.

Deploying value-added services across digital subscriber line (DSL) access networks requires special attention to quality of service (QoS) and service control. This control

depends on tighter coordination between network elements in the broadband access network while not causing added burden to the operations support system (OSS) layer.

ANCP is an extension to GSMPv3 that functions as a control plane between a service-oriented layer 3 edge device (the Broadband Remote Access Server) and a layer 2 access node. In this role, ANCP performs QoS-related, service-related, and subscriber-related operations. These operations include the following:

- Dynamic discovery of the access topology and enabling an authentication server to retrieve this information
- Subscriber and service data retrieval from the OSS by the Broadband Remote Access Server (B-RAS) and sending that information to the access node to simplify service management (referred to as line configuration)
- An optimized, layer 2 multicast (IGMP) replication
- On-demand access-line testing (ANCP Operation, Administration, and Maintenance)

JunosE Software supports the use of RADIUS attributes to monitor ANCP-related information, such as upstream and downstream data rates. For information about using RADIUS attributes see *JunosE Broadband Access Configuration Guide*.

## Access Topology Discovery

Many queuing or scheduling mechanisms must avoid congestion within the access network while contending with multiple flows and distinct QoS requirements. These mechanisms require that B-RAS devices obtain information about the access network topology, the links within that network, and their rates.

Operations support systems cannot enforce the consistency of this gathered information in a reliable and scalable way. ANCP discovery enables the automated discovery of the access network topology, resolving this problem.

## Line Configuration

Following access topology discovery, the B-RAS can query a subscriber management OSS component (for example, a RADIUS server) to retrieve subscriber authorization data. This type of query is typically managed by the B-RAS, but in some cases (for example, DSL-related enforcement) it can be useful to push service information to the access node for local enforcement of the corresponding subscriber line. Using the line configuration feature provides a more flexible way to achieve this on-demand service.



**NOTE:** JunosE Software supports only a CLI version of this feature.

---

## Transactional Multicast

IP multicasting in access networks often involves an access server replicating the same multicast stream to multiple subscribers. This type of replication wastes access bandwidth when multiple subscribers access network services using the same access node. The

amount of multicast replication is based on the number of subscribers, rather than the number of access nodes.

The access node sending a single copy of the multicast stream to a specific access node is a more efficient use of the bandwidth. Using this method, the access node performs the multicast replication for subscribers that reside beyond the access node.

ANCP transactional multicast enables the E Series router to set up a multicast replication state in the access node. In Asynchronous Transfer Mode (ATM) access networks, the B-RAS can use the ANCP to set up point-to-multipoint cross-connects in the access nodes.

## OAM

A simple solution based on ANCP provides B-RAS with an access-line test capability. When enabled through the CLI, the B-RAS uses an ANCP message to trigger the access-node to perform a loopback test on the local loop (between the access node and the CPE). The access node reports the results of the test by means of another ANCP message.

## Retrieval of DSL Line Rate Parameters

The router retrieves the DSL line rate parameters from ANCP and reports this information to the SRC software with the corresponding COPS messages. If the router cannot retrieve the DSL line rate parameters from ANCP, it retrieves the DSL information in the following ways:

- **From AAA layer**—For PPP interfaces, the router retrieves the DSL line rate parameters from the AAA layer and reports this information to the SRC software.
- **From DHCP options**—For DHCP external server and DHCP local server in equal-access mode, the router retrieves the DSL line rate parameters from DHCP options and reports this information to the SRC software. For more information about retrieval of DSL line rate parameters from DHCP options, see DHCP Overview Information.

## Learning the Partition ID from an Access Node

ANCP running on the network access server can learn the partition ID from an access node and use this partition ID to communicate with the access node. This learning option of the ANCP in the virtual router enables the network access server to learn the partition IDs from all the access nodes. This learned partition ID can be used to send messages to establish adjacency.

## Platform Considerations

---

Configuring ANCP is supported on all E Series Broadband Services Routers.

For information about the modules supported on E Series routers:

- See the *ERX Module Guide* for modules supported on ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router.

- See the *E120 and E320 Module Guide* for modules supported on the E120 and E320 Broadband Services Routers.

## References

---

For more information about ANCP, see the following resources:

- GSMP extensions for layer2 control (L2C) Topology Discovery and Line Configuration—draft-wadhwa-gsmp-l2control-configuration-00.txt (July 2006 expiration)
- IGMP-based Multicast Forwarding (“IGMP Proxying”)—draft-ietf-magma-igmp-proxy-00.txt (May 2002 expiration)
- GSMPv3 Base Specification—draft-ietf-gsmp-v3-base-spec-06.txt (March 2006 expiration)

## Configuring ANCP

---

ANCP uses several global-level configuration commands to enable ANCP to function on the router.

### Creating a Listening TCP Socket for ANCP

Use the **l2c ip listen** command to create a listening TCP socket for ANCP. ANCP monitors port 6068 for ANCP TCP connection requests.

#### *l2c ip listen*

- Use to create a listening TCP socket in the current virtual router context.
- Example

```
host1(config)#l2c ip listen
```
- Use the **no** version to remove the listening TCP socket and stop any new sessions from being established. The **no** version does not terminate any existing GSMP sessions.
- See l2c ip listen.

### Accessing L2C Configuration Mode for ANCP

Use the **l2c** command to launch the L2C Configuration (config-l2c) mode for ANCP. In this mode, you can define session timeout values and access the L2C Neighbor Configuration mode to configure an ANCP neighbor.

#### *l2c*

- Use to launch the L2C Configuration (config-l2c) mode for ANCP.
- Example

```
host1(config)#l2c
host1(config-l2c)#
```

- Use the **no** version to remove all ANCP configurations.
- See l2c.

## Defining the ANCP Session Timeout

In L2C Configuration (config-l2c) mode, you can use the **session-timeout** command to specify the ANCP session timeout value. The timer range is 1–25 seconds with a default value of 25 seconds.

### *session-timeout*

- Use to define the ANCP session timeout value (in seconds).
- Example  

```
host1(config-l2c)#session-timeout 10
```
- Use the **no** version to revert the session timeout to its default setting, 25 seconds.
- See session-timeout.

## Learning the Access Node Partition ID

In L2C Configuration (config-l2c) mode, you can use the **wait-for-gsmp-syn** command to enable the learning option. This learning option in the virtual router enables network access server to learn the partition ID from all the access nodes.

### **wait-for-gsmp-syn**

- Use to enable the learning option in ANCP.
- Example  

```
(config-l2c)# wait-for-gsmp-syn
```
- Use the **no** version to disable the learning option in ANCP.

If the access node does not send the GSMP\_SYN message after initiating the TCP session, the connection is lost because the session timer expires. To avoid this, you can use the **gsmp-syn-timeout** command to specify the timeout value in L2C Configuration (config-l2c) mode. The timer value cannot be more than the ANCP session timeout value with a default value of 60 seconds.

### **gsmp-syn-timeout**

- Use to define the TCP session timeout value (in seconds).
- Example  

```
(config-l2c)# gsmp-syn-timeout 10
```
- Use the **no** version to revert the session timeout to its default setting, 60 seconds.

## Configuring ANCP Interfaces ---

ANCP uses several interface-level configuration commands. These commands provide the ability to define GSMP input and output labels associated with the interface and specify the number of branches the ANCP end user can support.

### *l2c end-user-id*

- Use to create the GSMP output label associated with the interface. In addition to the label, this command also specifies the access node using the neighbor keyword.
- Example

```
host1(config-if)#l2c end-user-id out_subscriber_port_6 neighbor ACCESS_NODE_1
```
- Use the **no** version to remove the output label association.
- See l2c end-user-id.

### *l2c max-branches*

- Use to specify the maximum number of branches the ANCP end user can have.
- Example

```
host1(config-if)#l2c max-branches 5
```
- Use the **no** version to return the maximum number of branches to its default value, unlimited branches.
- See l2c max-branches.

### *l2c peer-attachment-id*

- Use to create the GSMP input label associated with the interface.
- Example

```
host1(config-if)#l2c peer-attachment-id in_multicast_port_5
```
- Use the **no** version to remove the input label association.
- See l2c peer-attachment-id.

---

## Configuring ANCP Neighbors

From the L2C Configuration mode (config-l2c), you can access the L2C Neighbor Configuration mode, from which you can create and manage ANCP neighbors.

### Accessing L2C Neighbor Configuration Mode for ANCP

Use the **neighbor** command to create an ANCP neighbor and access the L2C Neighbor Configuration (config-l2c-neighbor) mode.

#### *neighbor*

- Use to create an ANCP neighbor and access the L2C Neighbor Configuration (config-l2c-neighbor) mode.
- Example

```
host1(config-l2c)#neighbor ACCESS-NODE-1
host1(config-l2c-neighbor)#
```

- Use the **no** version to remove a specific ANCP neighbor configuration or, by omitting the neighbor name, all ANCP neighbor configurations.
- See neighbor.

## Defining an ANCP Neighbor

The L2C Neighbor Configuration mode enables you to define an ANCP neighbor by specifying a neighbor ID and the maximum number of branches that the neighbor can have.

### *id*

- Use to specify the ANCP neighbor ID in the L2C Neighbor Configuration (config-l2c-neighbor) mode.
- Example

```
host1(config-l2c)#neighbor
host1(config-l2c-neighbor)#id 1234.5678.9123
```
- Use the **no** version to remove the neighbor ID.
- See id.

### *max-branches*

- Use to specify the maximum number of branches the ANCP neighbor can have.
- Example

```
host1(config-l2c-neighbor)#max-branches 50
```
- Use the **no** version to return the maximum number of branches to its default value, unlimited branches.
- See max-branches.

## Limiting Discovery Table Entries

You use the **max-discovery-table-entries** command to specify the maximum number of discovery table entries that a neighbor can have.

Using this command to change the maximum number of entries when an already greater number of current entries exists in the discovery table does not remove any existing entries. Instead, future entries are prevented. For example, if the current table contains 5000 entries, and you specify a maximum of 4000 entries, the software does not remove any existing entries from the table. The software prevents any new entries until the number of table entries falls below the specified maximum.

### *max-discovery-table-entries*

- Use to specify the maximum number of discovery table entries a neighbor can have in the range 1–64000 entries.
- Example

```
host1(l2c-neighbor)#max-discovery-table-entries 4000
```

- Use the **no** version to return the maximum number of discovery table entries to its default value, 10,000 entries.
- See max-discovery-table-entries.

## Clearing ANCP Neighbors

You can clear an existing ANCP neighbor using the **clear l2c** command.

### *clear l2c neighbor*

- Use to reset a specific GSMP neighbor session.
- Example

```
host1#clear l2c neighbor ACCESS_NODE_1
```

- There is no **no** version.
- See clear l2c neighbor.

## Configuring Topology Discovery

---

You use the **discovery-mode** command to enable the dynamic discovery of access topology. When ANCP discovery is enabled, an authentication server (like RADIUS) retrieves upstream and downstream access loop information from the configured access node.

For information about configuring RADIUS, see *JunosE Broadband Access Configuration Guide*.

### *discovery-mode*

- Use to enable ANCP discovery for a neighbor.
- Example

```
host1(l2c-neighbor)#discovery-mode
```

- Use the **no** version to disable discovery mode.
- See discovery-mode.

## Configuring ANCP for QoS Adaptive Mode

---

The system can QoS adjust VLAN and ATM VC downstream rates received from ANCP when you enable QoS adaptive mode by issuing the **qos-adaptive-mode** command.

When QoS adaptive mode is enabled, ANCP dynamically creates QoS parameter instances associated with the QoS downstream rate application. ANCP also determines the value the system uses when recalculating the QoS shaping rate. The values of the parameter instances track the bandwidth of the local loop that is communicated by ANCP.



Issuing the **clear l2c neighbor** command removes all QoS parameter instances associated with the neighbor, including those associated with the QoS downstream rate and QoS cell mode applications.

Similarly, issuing the **clear l2c discovery-table** command without specifying an entry removes all QoS parameter instances associated with the neighbor. Specifying an entry in the table removes the QoS parameter instance associated with that entry.

After clearing ANCP entries or neighbor sessions, the system recreates the QoS parameter instances when QoS adaptive mode is enabled and ANCP learns the rates again.

For more information about the QoS downstream rate application and QoS parameters, see QoS Downstream Rate Application Overview in *JunosE Quality of Service Configuration Guide*.

#### ***clear l2c discovery-table***

- Use to clear all entries or a specified entry from the topology discovery table associated with the neighbor.
- Example  

```
host1#clear l2c discovery-table neighbor ACCESS_NODE_1
```
- There is no **no** version.
- See clear l2c discovery-table.

#### ***qos-adaptive-mode***

- Use to enable the QoS adaptive mode for ANCP.
- QoS adaptive mode enables the system to shape VLAN and ATM VC downstream rates received from ANCP by dynamically creating QoS parameter instances associated with the ANCP (L2C) downstream application.
- Example  

```
host1(config-l2c)#qos-adaptive-mode
```
- Use the **no** version to disable QoS adaptive mode for the system.
- See qos-adaptive-mode.

## Triggering ANCP Line Configuration

Use the **l2c line-configuration** command to trigger line configuration to the access node. Issuing this command sends a GSMP port management message to the access node. This message enables the B-RAS to configure a service profile name on an access loop.



**NOTE:** Before issuing the **l2c line-configuration** command, a profile must already be configured locally on the access node.

#### ***l2c line-configuration***

- Use to trigger a GSMP port management message to the access node. This message enables the B-RAS to configure a service profile name on an access loop.
- Example

```
host1#l2c line-configuration interface atm 2/0.11 profile1
```
- There is no **no** version.
- See l2c line-configuration.

---

## Adjusting the Data Rate Reported by ANCP for DSL Lines

When a DSLAM calculates the data rate, it ignores additional headers on the DSL line. When ANCP reports the upstream data rate (L2C Type 4 Sub-type 129) or the downstream data rate (L2C Type 4 Sub-type 130), it includes the headers in its calculation and therefore reports a slightly higher value. This discrepancy causes the QoS shaping rate to be slightly higher than the actual rate.

You can use the **adjustment-factor** command to set a percentage value ANCP applies to the data rate to generate a more accurate value that is reported to AAA whenever AAA requests the data rate from ANCP.

ANCP does not report the calculated data rate to RADIUS or to L2TP.

### *adjustment-factor*

- Use to configure a QoS adjustment factor that is applied to the upstream data rate and downstream data rate reported by ANCP for a DSL type. The adjustment factor is used to generate an accurate QoS shaping rate.
- The factor is applied for all subscribers that use the specified DSL line type
- Example

```
host1(config-l2c)#adjustment-factor adsl1 45
host1(config-l2c)#adjustment-factor adsl2 55
host1(config-l2c)#adjustment-factor adsl2+ 67
host1(config-l2c)#adjustment-factor vdsl 91
host1(config-l2c)#adjustment-factor vdsl2 55
host1(config-l2c)#adjustment-factor sds 12
```
- Use the **no** version to restore the default condition, wherein no adjustment factor is applied to the ANCP-reported value. This command has the same effect as applying an adjustment factor of 100 to a DSL line type.
- See adjustment-factor.

---

## Configuring Transactional Multicast for IGMP

By using ANCP, IGMP is no longer terminated or proxied at the access node. Instead, IGMP passes through the access node transparently. B-RAS terminates both the data PVC and IGMP. After any user permission verification, B-RAS may instruct the access node (using GSMP) to establish a multicast branch for the subscriber port.

ANCP works with a special IGMP session to collect OIF mapping events in a scalable manner. For additional information about configuring IGMP and about OIF mapping, see *Configuring IGMP in JunosE Multicast Routing Configuration Guide*.

## Creating an IGMP Session for ANCP

Use the **l2c ip oif** command to create an IGMP session for ANCP. This session enables ANCP to listen to OIF mappings and, in turn, convey cross connect events to the appropriate ANCP neighbor (access node).

### **l2c ip oif**

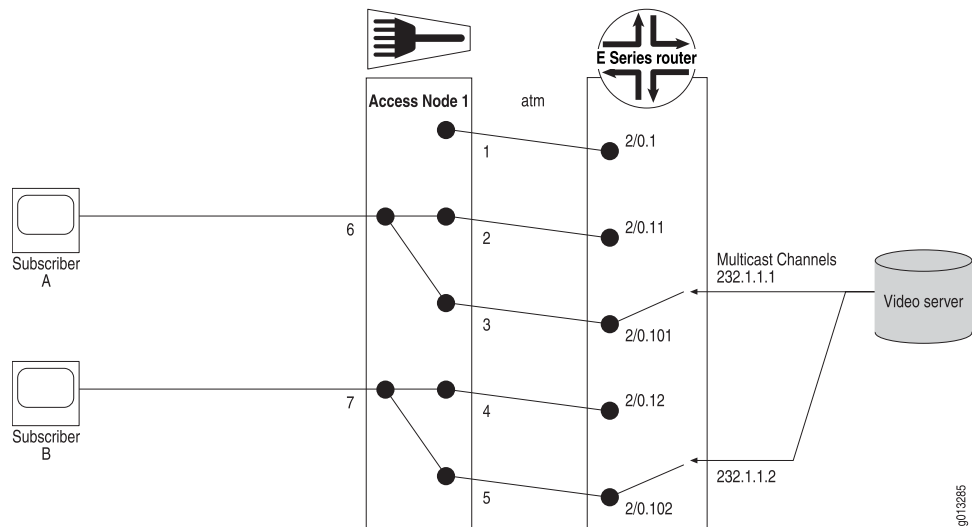
- Use to create an IGMP session at the virtual router within the context.
- Example
 

```
host1(config)#l2c ip oif
```
- Use the **no** version to remove the IGMP session.
- See **l2c ip oif**.

## ANCP IGMP Configuration Example

In the following example (Figure 18 on page 195), two subscribers access individual multicast channels through cross connections (branches) that occur on the access node.

Figure 18: Using ANCP with an Access Node



To configure the example, use the following general procedures:



**NOTE:** This example provides general information for configuring ANCP mapping. For detailed information about creating OIF maps, see [“Configuring Transactional Multicast for IGMP” on page 194](#).

1. Configure an OIF map for the access node that maps each multicast group to an outgoing interface.
2. Define ANCP parameters.
3. Enable ANCP to listen to OIF mapping events from IGMP in this virtual router.
4. Create a listening TCP socket in the virtual router (TCP port 6068).
5. Define ANCP neighbor parameters.



**NOTE:** The ID is a 48 bit quantity that identifies the ANCP neighbor.

6. Configure the ANCP multicast labels (input labels) on the corresponding outgoing interfaces.
7. Configure the ANCP output labels, the neighbor information on the subscriber interfaces, and apply the OIF map.

When Subscriber A requests to join 232.1.1.1, ANCP transmits an add branch message with the corresponding input and output labels that cross-connect port 3 and port 6 on Access Node 1.

## Complete Configuration Example

The following example contains the commands used to configure ANCP. You can customize and use this example in your own network.

You can copy the text into a text editor and modify it (removing all prompts and changing values) for immediate use or save the modified example as a script (.scr) file. Script files allow you to execute commands as though they were entered at the terminal. For information about creating and executing script files, see *Command Line Interface* in *JunosE System Basics Configuration Guide*.

```
!Configure an OIF map
host1(config)#ip igmp oif-map OIFMAP atm 2/0.101 232.1.1.1 10.1.1.1
host1(config)#ip igmp oif-map OIFMAP atm 2/0.102 232.1.1.2 10.1.1.2
!Define ANCP parameters
host1(config)#l2c
host1(config-l2c)#session-timeout 15
!Enable ANCP to listen to OIF mapping events from IGMP in this virtual router
host1(config)#l2c ip oif
!Create a listening TCP socket in the virtual router
host1(config)#l2c ip listen
!Define ANCP neighbor parameters.
host1(config-l2c)#neighbor ACCESS_NODE_1
host1(config-l2c-neighbor)#id 09af.15bc.3156
!Configure ANCP multicast labels on the corresponding outgoing interfaces
host1(config)#interface atm 2/0.101
host1(config-interface)#ip igmp version passive
host1(config-interface)#l2c peer-attachment-id "in_multicast_port_3"
host1(config)#interface atm 2/0.102
host1(config-interface)#ip igmp version passive
host1(config-interface)#l2c peer-attachment-id "in_multicast_port_5"
```

```
!Configure ANCP output labels, neighbor information, and apply OIF map
host1(config)#interface atm 2/0.11
host1(config-interface)#ip igmp apply-oif-map OIFMAP
host1(config-interface)#l2c end-user-id " out_subscriber_port_6" neighbor
ACCESS_NODE_1
host1(config)#interface atm 2/0.12
host1(config-interface)#ip igmp apply-oif-map OIFMAP
host1(config-interface)#l2c end-user-id " out_subscriber_port_7" neighbor
ACCESS_NODE_1
```

## Triggering ANCP OAM

The **l2c oam** command provides an access-line test and rudimentary fault isolation capability for the B-RAS. Issuing this command triggers the access node to perform a loopback test on the local-loop (between the access-node and the CPE). The B-RAS generates a GSMP port management message to the neighbor specifying the access line identifier on the access node and OAM test characteristics desired by the B-RAS (for example, the number of cells/message to generate and the timeout period). The access node responds with the result of the triggered loopback test by means of a GSMP port management message. For example, when using an ATM-based local-loop, the ANCP operation can trigger the access node to generate ATM (F4/F5) loopback cells on the local loop.

### *l2c oam*

- Use to trigger the access node to run a local loopback test on the specified interface.
- Field descriptions
  - Request status—Status of the OAM request (succeeded or failure)
  - Response code—Code that returned with the OAM response
  - DEFAULT RESPONSE—Response string (if any) that was included in the OAM response

- Example 1

```
host1#l2c oam interface atm 4/1.103
```

```
request succeeded
0x503 : DSL line status showtime.
DEFAULT RESPONSE
```

- Example 2

```
host1#l2c oam neighbor accessnode_1002 end-user-id enduser_1002
```

```
request succeeded
0x503 : DSL line status showtime.
DEFAULT RESPONSE
```

- There is no **no** version.
- See l2c oam.

## Monitoring ANCP

---

You can display ANCP information with the following commands.

### *show adjustment-factor*

- Use to display the configured values for the adjustment factor applied to the upstream data rate and downstream data rate reported by ANCP for each DSL type. The adjustment factor is used to generate an accurate QoS shaping rate.
- Field descriptions
  - L2C QoS Adjustment Rates—List of all DSL types and the corresponding adjustment factor for each
  - ADSL1—Adjustment factor for the ADSL1 DSL type
  - ADSL2—Adjustment factor for the ADSL2 DSL type
  - ADSL2+—Adjustment factor for the ADSL2+ DSL type
  - VDSL—Adjustment factor for the VDSL DSL type
  - VDSL2—Adjustment factor for the VDSL2 DSL type
  - SDS—Adjustment factor for the SDS DSL type
- Example 1—Displays the adjustment factor for each DSL type

```
host1#show adjustment-factor
L2C QoS Adjustment Rates:
ADSL1: 45
ADSL2: 55
ADSL2+: 100
VDSL: 100
VDSL2: 55
SDS: 100
```

- Example 2—Displays the adjustment factor for a specific DSL type verification criteria

```
host1#show adjustment-factor vdsl2
VDSL2:11
```

- See show adjustment-factor.

### *show l2c*

- Use to display information about the ANCP configuration on the router.
- Field descriptions
  - Current timeout—Configured session timeout (in seconds)
  - Qos adaptive mode—Whether QoS adaptive mode is enabled (true) or disabled (false)

- Wait-for-gsmp-syn—Whether learning is enabled or disabled
- gsmp-syn-timeout—Configured TCP session timeout (in seconds)
- Example

```
host1#show l2c
L2C:
 Current session timeout: 25 seconds
 Qos adaptive mode: false
 Wait-for-gsmp-syn: Enable
 gsmp-syn-timeout: 18 seconds
```

### *show l2c discovery-table*

- Use to display ANCP discovery table entries.
- Use the optional **delta** keyword to display baseline statistics.
- Field descriptions
  - Neighbor—Neighbor name
  - Access-Loop-Id—Access loop identifier
  - Down/Upstream (kbps)—Downstream and upstream rates, in Kbps
  - State—State of the access loop, UP or DOWN
  - Actual-Data-Rate-Upstream—Actual upstream data rate, in Kbps
  - Actual-Data-Rate-Downstream—Actual downstream data rate, in Kbps
  - Attainable-Data-Rate-Upstream—Attainable upstream data rate for this line, in Kbps
  - Attainable-Data-Rate-Downstream—Attainable downstream data rate for this line, in Kbps
  - Line-State—State of the access loop (SHOWTIME or IDLE)
  - Dsl-Type—Type of DSL
  - Total Line Attributes—Total number of line attributes reported
- Example 1

```
host1# show l2c discovery-table brief
Neighbor Access-Loop-Id Down/UpStream(kbps) State

ACCESSNODE_10 Accessnode_10 atm 2/2:0.0 8064/1184 UP
ACCESSNODE_10 Accessnode_10 atm 2/32:0.0 8064/1184 UP
ACCESSNODE_10 Accessnode_10 atm 2/33:0.0 8064/1184 DOWN
ACCESSNODE_10 Accessnode_10 atm 2/34:0.0 8064/1184 DOWN
```

- Example 2—Topology discovery table for a particular end-user-id

```
host1#
show l2c discovery-table end-user-id "Accessnode_10 atm 2/3:0.0"Access-Loop-Id:
```

```
Dslam_10 atm 2/3:0.0 UP
Neighbor: ACCESSNODE_10
Actual-Data-Rate-Upstream: 1152(kbps)
Actual-Data-Rate-Downstream: 8064(kbps)
Attainable-Data-Rate-Upstream: 1176(kbps)
Attainable-Data-Rate-Downstream: 9376(kbps)
Line-State: 1(SHOWTIME)
Dsl-Type: 0(Invalid transmission type)
Total Line Attributes: 6
```

- Example 3—Topology discovery table for a particular neighbor

```
host1# show l2c discovery-table neighbor "Accessnode_10"
Access-Loop-Id: Dslam_10 atm 10/5:0.0 DOWN
Neighbor: DSLAM_10
Line-State: 2(IDLE)
Dsl-Type: 0(Invalid transmission type)
Total Line Attributes: 2
Access-Loop-Id: Dslam_10 atm 10/6:0.0 DOWN
Neighbor: DSLAM_10
Line-State: 2(IDLE)
Dsl-Type: 0(Invalid transmission type)
Total Line Attributes: 2
Access-Loop-Id: Dslam_10 atm 10/7:0.0 DOWN
Neighbor: DSLAM_10
Line-State: 2(IDLE)
Dsl-Type: 0(Invalid transmission type)
Total Line Attributes: 2
Access-Loop-Id: Dslam_10 atm 2/0:0.0 UP
Neighbor: DSLAM_10
Actual-Data-Rate-Upstream: 1184(kbps)
Actual-Data-Rate-Downstream: 8064(kbps)
Attainable-Data-Rate-Upstream: 1184(kbps)
Attainable-Data-Rate-Downstream: 9408(kbps)
Line-State: 1(SHOWTIME)
Dsl-Type: 0(Invalid transmission type)
Total Line Attributes: 6
```

- See show l2c discovery-table.

### *show l2c label*

- Use to display information about known ANCP labels on the router.
- Use the **neighbor-input** keyword to display labels for input ports.
- Use the **neighbor-output** keyword to display labels for output ports.
- Use the **brief** keyword to show limited information.
- Field descriptions
  - Interface—Interface on which ANCP is configured
  - End-User-Id—Output label associated with the interface
  - Neighbor—Neighbor associated with the interface



- Max-Branches—Maximum number of branches to which the ANCP interface can subscribe
- Peer-Attach-Id—Input label associated with the interface
- Example 1

```

host1# show l2c label
Interface: ATM2/0.300
 End-User-Id: Accessnode_10 atm2/2:0.0
 Neighbor: accessnode _1002
 Max-Branches: 5
Interface: ATM2/0.301
 End-User-Id: Accessnode_10 atm2/3:0.0
 Neighbor: accessnode_1002
 Max-Branches: 5
Interface: ATM2/0.302
 End-User-Id: Accessnode_10 atm2/4:0.0
 Neighbor: accessnode _1002
 Max-Branches: 5
Interface: ATM2/0.303
 End-User-Id: Accessnode_10 atm2/5:0.0
 Neighbor: accessnode _1004
 Max-Branches: 5
Interface: ATM2/0.304
 End-User-Id: Accessnode_10 atm2/6:0.0
 Neighbor: accessnode _1004
 Max-Branches: 5

```

- Example 2

```

host1# show l2c label brief

```

| Interface  | End-User-Id               | Neighbor        |
|------------|---------------------------|-----------------|
| ATM4/0.300 | Accessnode_10 atm2/2:0.0  | accessnode_1002 |
| ATM4/0.301 | Accessnode_10 atm2/3:0.0  | accessnode_1002 |
| ATM4/0.302 | Accessnode_10 atm2/4:0.0  | accessnode_1002 |
| ATM4/0.303 | Accessnode_10 atm2/5:0.0  | accessnode_1004 |
| ATM4/0.304 | Accessnode_10 atm2/6:0.0  | accessnode_1004 |
| Interface  | Peer-Attach-Id            |                 |
| ATM4/0.11  | Accessnode_10 atm3/2:0.10 |                 |
| ATM4/0.12  | Accessnode_10 atm3/3:0.10 |                 |
| ATM4/0.13  | Accessnode_10 atm3/4:0.10 |                 |
| ATM4/0.14  | Accessnode_10 atm3/5:0.10 |                 |

- Example 3

```

host1# show l2c label neighbor-input brief

```

| Interface | Peer-Attach-Id            |
|-----------|---------------------------|
| ATM4/0.11 | Accessnode_10 atm3/2:0.10 |
| ATM4/0.12 | Accessnode_10 atm3/3:0.10 |
| ATM4/0.13 | Accessnode_10 atm3/4:0.10 |
| ATM4/0.14 | Accessnode_10 atm3/5:0.10 |

- See show l2c label.

*show l2c neighbor*

- Use to display information about all known ANCP neighbors or specified ANCP neighbors on the router along with their configurations.
- Use the **brief** keyword to display limited information.
- Use the **summary** keyword to display the number of active neighbors.
- Field descriptions
  - Neighbor Name—Name associated with the neighbor
  - Neighbor Id—ID associated with the neighbor
  - Maximum Branches—Maximum number of branches this neighbor can have
  - Topology Discovery—Whether topology discovery is enabled (true) or disabled (false)
  - Maximum Discovery Entries—Maximum number of discovery table entries allowed
  - Ip address—IP address of the neighbor
  - TCP port—TCP port associated with this neighbor
  - Connection Time—Date and time at which this neighbor was connected
  - Add Branches Sent—Number of add branch messages sent to this neighbor
  - Delete Branches Sent—Number of delete branch messages sent to this neighbor
  - Line-configurations—Number of line configurations sent to this neighbor
  - OAM Loopback Requests Sent—Number of OAM loopback requests sent to this neighbor
  - OAM Loopback Responses Received—Number of OAM loopback responses received from this neighbor
  - Protocol State—Protocol state of this neighbor
  - Number of configured neighbors—Number of configured ANCP neighbors
  - Number of Neighbors in GSMP\_ESTAB state—Number of ANCP neighbors that are in an established GSMP state
  - Number of neighbors in GSMP\_EMPTY state—Number of ANCP neighbors that are in an unestablished GSMP state
- Example 1

```
host1#show l2c neighbor name accessnode_1002
Neighbor Name: accessnode_1002
Neighbor Id: 0abc.0abc.1002
Maximum Branches: 10
Topology Discovery: true
Maximum Discovery Entries: 10000
Ip address: 1.1.1.2
TCP port: 1025
```

```

Connection Time: 03/02/2006 11:06:16
Add Branches Sent: 0
Delete Branches Sent: 0
Line-configurations: 0
OAM Loopback Requests Sent: 0
OAM Loopback Responses Received: 0
Protocol State: GSMP_ESTAB

```

- Example 2

```

host1#show l2c neighbor brief

```

| Name        | Mac Address    | Remote Address | Protocol State |
|-------------|----------------|----------------|----------------|
| -----       | -----          | -----          | -----          |
| accessnode1 | 0abc.0abc.0abc | null           | EMPTY          |

- Example 3

```

host1(config)#show l2c neighbor summary
L2C:
 Number of configured Neighbors: 1
 Number of Neighbors in GSMP_ESTAB state: 1
 Number of neighbors in GSMP_EMPTY state: 0

```

- See show l2c neighbor.

### *show l2c statistics*

- Use to display information about the ANCP statistics.
- Field descriptions
  - Current session timeout—Configured session timeout (in seconds)
  - Discovery—State of topology discovery (Enabled or Disabled)
  - Number of configured routers—Number of ANCP configured routers
  - Number of neighbors—Number of ANCP neighbors
  - Number of active neighbors—Number of active ANCP neighbors
  - Number of end-user-ids—Number of ANCP end user IDs (output labels)
  - Number of peer-attachment-ids—Number of ANCP peer attachment IDs (input labels)
  - Number of add-branches—Number of ANCP branches added
  - Number of delete-branches—Number of ANCP branches deleted
- Example

```

host1#show l2c statistics
L2C:
Current session timeout: 25 seconds
Discovery: Enabled
Number of configured routers: 1
Number of neighbors: 5
Number of active neighbors: 1
Number of end-user-ids: 25

```

Number of peer-attachment-ids: 39  
Number of add-branches: 0  
Number of delete-branches: 0

- See show l2c statistics.

## CHAPTER 8

# Configuring Digital Certificates

This chapter describes how to configure digital certificates; it contains the following sections:

- [Overview on page 205](#)
- [Platform Considerations on page 206](#)
- [References on page 206](#)
- [IKE Authentication with Digital Certificates on page 207](#)
- [IKE Authentication Using Public Keys Without Digital Certificates on page 212](#)
- [Configuring Digital Certificates Using the Offline Method on page 213](#)
- [Configuring Digital Certificates Using the Online Method on page 219](#)
- [Configuring Peer Public Keys Without Digital Certificates on page 224](#)
- [Monitoring Digital Certificates and Public Keys on page 228](#)

## Overview

---

You can use digital certificates in place of preshared keys for IKE negotiations. For more information about IKE, see [“IKE Overview” on page 134](#) in [“Configuring IPSec” on page 119](#).

## Digital Certificate Terms and Acronyms

[Table 14 on page 205](#) describes terms and abbreviations that are used in this discussion of digital certificates.

**Table 14: Digital Certificate Terms and Acronyms**

| Term or Abbreviation | Description                                                                                        |
|----------------------|----------------------------------------------------------------------------------------------------|
| 3DES                 | Triple DES encryption/decryption algorithm                                                         |
| Base64               | Method used to encode certificate requests and certificates before they are sent to or from the CA |
| CA                   | Certificate authority; an organization that creates digital certificates                           |
| Certificate          | Binds a person or entity to a public key using a digital signature                                 |

Table 14: Digital Certificate Terms and Acronyms (*continued*)

| Term or Abbreviation | Description                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CRL                  | Certificate revocation list; a list of certificates that a CA has revoked                                                                                                                    |
| ESP                  | Encapsulating Security Payload; provides data integrity, data confidentiality and, optionally, sender's authentication                                                                       |
| IKE                  | Internet Key Exchange                                                                                                                                                                        |
| PKCS                 | Public-Key Cryptography Standards; a series of standards established by RSA Laboratories                                                                                                     |
| PKCS10               | PKCS #10; describes a syntax for certification requests                                                                                                                                      |
| Root CA              | CA that signs the certificates of subordinate CAs                                                                                                                                            |
| Root certificate     | Self-signed public key certificate for a root CA; root certificates are used to verify other certificates                                                                                    |
| RSA                  | Rivest-Shamir-Adleman encryption algorithm                                                                                                                                                   |
| SA                   | Security association; the set of security parameters that dictate how IPSec processes a packet, including encapsulation protocol and session keys. A single secure tunnel uses multiple SAs. |
| SCEP                 | Simple certificate enrollment protocol; used to submit requests and to download certificates and CRLs                                                                                        |

## Platform Considerations

Digital certificates are supported on all ERX routers that support configuration of IPSec.

For information about modules that support IPSec on ERX14xx models, ERX7xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support IPSec.



**NOTE:** The E120 and E320 Broadband Services Routers do not support configuration of IPSec and digital certificates.

## References

For information about digital certificates, see the following references:

- RFC 2409—The Internet Key Exchange (IKE) (November 1998)
- RFC 2459—Internet X.509 Public Key Infrastructure Certificate and CRL Profile (January 1999)
- RFC 2986—PKCS #10: Certification Request Syntax Specification Version 1.7 (November 2000)
- RFC 3280—Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (April 2002)
- RFC 3447—Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1 (February 2003)

For more information about IPSec and IKE, see [“Configuring IPSec” on page 119](#).

## IKE Authentication with Digital Certificates

---

As part of the IKE protocol, one security gateway needs to authenticate another security gateway to make sure that IKE SAs are established with the intended party. The router supports two authentication methods:

- Digital certificates (using RSA algorithms)

For digital certificate authentication, an initiator signs message interchange data using his private key, and a responder uses the initiator's public key to verify the signature. Typically, the public key is exchanged via messages containing an X.509v3 certificate. This certificate provides a level of assurance that a peer's identity—as represented in the certificate—is associated with a particular public key. E Series Broadband Services Routers provide both an offline (manual) and an online (automatic) process when using digital certificates.

- Preshared keys

With preshared key authentication, the same secret must be configured on both security gateways before the gateways can authenticate each other.

The following sections provide information about digital certificates. For information about using preshared keys, see [“IKE Overview” on page 134](#).

You can also use public keys for RSA authentication without having to obtain a digital certificate. For details, see [“IKE Authentication Using Public Keys Without Digital Certificates” on page 212](#).

## Signature Authentication

The following are key steps for using public key cryptography to authenticate a peer. These steps are described in more detail in the following sections.

1. Generating a private/public key pair

Before the router can place a digital signature on messages, it requires a private key to sign, and requires a public key so that message receivers can verify the signature.

2. Obtaining a root CA certificate

The router requires at least one root CA certificate to send to IKE peers and also to verify that a peer's certificate is genuine.

### 3. Obtaining a public key certificate

The router requires at least one public key certificate, which binds the router identity to its public key. The CA verifies the identity represented on the certificate and then signs the certificate. The router sends the certificate to IKE peers during negotiations to advertise the router public key.

### 4. Authenticating the peer

As part of IKE negotiations, the router receives its peer's digital signature in a message exchange. The router must verify the digital signature by using the peer's public key. The public key is contained in the peer's certificate, which often is received during the IKE negotiation. To ensure that the peer certificate is valid, the router verifies its digital signature by using the CA public key contained in the root CA certificate. The router and its IKE peer require at least one common trusted root CA for authentication to work.

Generally, only Step 4 is required each time a phase 1 negotiation happens. The first three steps are required only if keys are compromised or router certificates require renewal.

## Generating Public/Private Key Pairs

The ERX router needs at least one valid pair of public/private keys whenever it uses any of the public key methods for authenticating an IKE peer. The ERX router can generate its own public/private key pairs. The public/private key pair supports the RSA standard (1024 or 2048 bits).

The private key is used only by the ERX router. It is never exchanged with any other nodes. It is used to place a digital signature on IKE authentication messages. When generated, it is securely stored internally to the ERX router in nonvolatile storage (NVS). Access to the private key is never allowed, not even to a system administrator or a network management system. Private key storage includes protection mechanisms to prevent improper private key usage, including encryption with 3DES using a unique internally generated key. The key is also tied to SRP-specific data to prevent swapping flash disks between routers.

The public key is used in the generation of the router certificate request, which is sent to a CA. Based on the certificate request, the CA generates a public key certificate for the E Series router.

The router public/private key pair is a global system attribute. It does not matter how many IPSec Service modules (ISMs) exist in the router; only one set of keys is available at any given moment. The private/public key pair applies across all virtual routers and is persistent across reloads and booting to factory defaults.

## Obtaining a Root CA Certificate

The ERX router enables the use of either a manual or automatic method to download the root CA's self-signed certificate. The standards supported for obtaining root CAs are X.509v3, base64, and basic-encoding-rules (BER)–encoded certificates.



In the manual method, an operator obtains the root CA certificate, typically through a Web browser, and copies the certificate file to the E Series router so that the router can use it as part of IKE negotiations.

In the automatic method, the router uses SCEP and HTTP to authenticate with the CA and retrieve the certificate. The requested root CA certificate is automatically downloaded to the router.



**NOTE:** You cannot view certificate files by their filenames if the files were created by online enrollment. However, the certificate information will appear in the output for show commands.

## Obtaining a Public Key Certificate

After the public key is generated, the router must obtain a public key certificate from a CA, a process called certificate enrollment. The procedure to obtain public keys depends on whether the offline or online digital certificate process is being used.

The standards supported for certificate enrollment are PKCS #10 certificate requests, PKCS #7 responses, and X.509v3 certificates. For manual enrollment, certificates are encoded in base64 (MIME) so that the files are easily transferred through cut-and-paste operations and e-mail.

### Offline Certificate Enrollment

---

Offline certificate enrollment works as follows:

1. An operator generates a certificate request by supplying identity information.
2. The ERX router creates a certificate request file and makes it available to the operator.
3. The operator supplies the certificate request file to a CA for approval, typically by copying and pasting the file to a Web page.
4. The CA approves the request and generates a certificate.
5. The operator copies the certificate file onto the ERX router so that it can be used for IKE negotiations.

### Online Certificate Enrollment

---

Online certificate enrollment works as follows:



**NOTE:** The ERX router must have a root CA certificate for the specified CA before online certificate enrollment.

- The router uses SCEP and HTTP to enroll with the specified CA and retrieve the certificate that the router uses in IKE negotiations.

## Authenticating the Peer

The ERX router validates X.509v3 certificates from the peer by confirming that the ID payload passed in IKE matches the identifiers in the peer certificate. The router also verifies that the signature is correct, based on the root CA public key.

The ERX router also validates the certificate based on its time window, so correct UTC time on the router is essential. In addition to the certificate checks, the router confirms that message data received from the peer has the correct signature based on the peer's public key as found in its certificate. After the IKE authentication is done, quick-mode negotiation of SAs can proceed.

## Verifying CRLs

You can control how the router handles CRLs during negotiation of IKE phase 1 signature authentication. Both the offline and online digital certificate processes enable you to verify CRLs.

To verify CRLs in the offline certificate process, you must copy CRL files that are published by CAs to the ERX router. Using the **ipsec crl** command, you can control how the router handles CRLs during negotiation of IKE phase 1 signature authentication.

In the online certificate method you use the **crl** command to control CRL verification. The router uses HTTP to support CRL verification when the CRL distribution point that appears in the certificate has an `http://name` Uniform Resource Indicator (URI) format.

The **ipsec crl** and **crl** commands have three possible settings:

- Ignored—Allows negotiations to succeed even if a CRL is invalid or the peer's certificate appears in the CRL; this is the most lenient setting.
- Optional—If the router finds a valid CRL, the router uses it.
- Required—Requires a valid CRL, and the certificates belonging to the E Series router or the peer must not appear in the CRL; this is the strictest setting.

Based on the CRL setting, you can expect the phase 1 IKE negotiations to succeed or fail depending on the following conditions:

- CRL OK—The certificate revocation list is present for the CA and valid (not expired).
- CRL expired—The CRL is present on the ERX router but is expired.
- Missing CRL—There is no CRL on the router for the CA.
- Peer Cert revoked—The CRL contains the peer certificate.
- ERX Cert revoked—The CRL contains the E Series router's certificate.

[Table 15 on page 211](#) presents how the CRL setting affects the outcome of IKE phase 1 negotiations. It lists common problem conditions such as ERX Cert revoked.

Table 15: Outcome of IKE Phase 1 Negotiations

|                   |         | CRL Setting |          |
|-------------------|---------|-------------|----------|
| Condition         | Ignored | Optional    | Required |
| CRL OK            | Succeed | Succeed     | Succeed  |
| CRL expired       | Succeed | Succeed     | Fail     |
| Missing CRL       | Succeed | Succeed     | Fail     |
| Peer Cert revoked | Succeed | Fail        | Fail     |
| ERX Cert revoked  | Succeed | Fail        | Fail     |

## File Extensions

Table 16 on page 211 describes the file extensions that the ERX routers use for digital certificates that are created by the offline process.

During the online digital certificate process, the certificate files are kept in NVS in hidden areas and are not visible to users (the files do not appear when you enter a **dir** shell command). Use the **show** commands to display information for the online certificate files. The router's private keys are similarly hidden from users.

Table 16: File Extensions (Offline Configuration)

| File Extension | Description                                                                                                                                                                                                                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| .crq           | Used for certificate request files that are generated on the ERX router and taken to CAs for obtaining a certificate.                                                                                                                                                                                                          |
| .cer           | Used for public certificate files. The public certificates for root CAs and the router public certificates are copied to the ERX router. They are automatically recognized as belonging to the ERX router or CA by certificate subject name and issuer name (in a CA they are the same). The ERX router supports multiple CAs. |
| .crl           | Used for certificate revocation lists that are obtained offline from CAs and copied to the ERX router. CRLs indicate which certificates from a particular CA are revoked.                                                                                                                                                      |

## Certificate Chains

In a basic CA model, there is a single CA from which the ERX router obtains the root CA certificates and the router's public key certificates. The E Series router also supports CA hierarchies, which consist of a top-level root CA and one or more sub-CAs (also called issuing CAs).

In a CA hierarchy, the router obtains its public key certificates and the CA certificate from a sub-CA. The sub-CA's certificate is signed by the root CA.

This process creates a certificate chain of trust in which the E Series router must verify all certificates in the chain until the router reaches a trusted CA, such as the root CA. For example, if the router receives traffic from a peer with a certificate signed by a sub-CA, the router first verifies the sub-CA's signature on the peer's certificate, then verifies the sub-CA's certificate, which is signed by the trusted root CA.

The ERX router supports CA hierarchies consisting of the root CA and one level of sub-CAs. When using a CA hierarchy, the router authenticates and enrolls for its public certificate with the sub-CA. When you use the **show ipsec ike-certificates** command, the root CA and sub-CA certificates are listed as CA certificates, and the router's public certificates are signed by the sub-CA.

## IKE Authentication Using Public Keys Without Digital Certificates

---

During IKE negotiations, peers exchange public keys to authenticate each other's identity and to ensure that IKE SAs are established with the intended party. Typically, public keys are exchanged in messages containing an X.509v3 digital certificate.

As an alternative to setting up digital certificates, you can configure and exchange public keys for IKE peers and use these keys for RSA signature authentication *without* having to obtain a digital certificate. This method offers the simplicity and convenience of using preshared key authentication without its inherent security risks.

With this method, you no longer need a digital certificate to do the following:

- Associate the router with its own public key
- Enable a remote peer to display the router's public key
- Learn the remote peer's public key

## Configuration Tasks

To set up public keys and peer public keys without obtaining a digital certificate, you use router commands to perform the following tasks:

- Display the router's public key by using the **show ipsec key mypubkey rsa** command. You can use the output from this command to provide information to the remote peer about the public key configured on the router. The remote peer can then enter the router's public key on its own system.
- Manually enter the public key for the remote peer with which you want to establish IKE SAs by using the **ipsec key pubkey-chain rsa** and **key-string** commands.
- Display the remote peer's public key by using the **show ipsec key pubkey-chain rsa** command.

For instructions on setting up peer public keys without a digital certificate, see [“Configuring Peer Public Keys Without Digital Certificates” on page 224](#).

## Public Key Format

RSA encryption and authentication require the use of a public key on both the ERX router and on the remote peer with which the router seeks to establish IKE SAs.

The length of the public key can be 1024 bits or 2048 bits, and the format conforms to the RSA standard defined in RFC 3447—Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1 (February 2003).

The public key consists of three components:

- Abstract Syntax Notation 1 (ASN.1) header information
- RSA public key modulus
- RSA public key exponent

In the following example of a 1024-bit public key, the first portion of the key (shown in **bold** typeface) represents the ASN.1 header information. The second portion of the key (shown in regular typeface) represents the RSA public key modulus. The third portion of the key (shown in **bold** typeface) represents the RSA public key exponent.

```
30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00A7E43C
3E2D399F 34EF6E16 F84464A9 8A145997 CC7F34C8 3DFF8216 57780FE9 D5CE2717
86239050 7A331044 EBA90120 EC13A78D C1B24285 333A9193 D94A59C8 492D8CB9
A46403A4 37461E00 768CF45C 580211AC 72793764 51E3AB3C F9A6665E 562E3681
F120405E 30235690 6FC093AA EB0FE956 51C38EE1 54D81E40 7687C387 07020301
0001
```

For more information about the format of an RSA public key and about ASN.1 syntax, see RFC 3447—Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1 (February 2003).

## Configuring Digital Certificates Using the Offline Method

To use the offline method to set up digital certificates on the router:

1. Generate RSA key pairs.

```
host1(config)#ipsec key generate rsa 2048
Please wait.....
.....
IPsec Generate Keys complete
```

2. In your IKE policy, set the authentication method to RSA signatures.

```
host1(config)#ipsec ike-policy-rule 1
host1(config-ike-policy)#authentication rsa-sig
host1(config-ike-policy)#exit
host1(config)#
```



**NOTE:** For more information about setting up IKE policies, see “Defining an IKE Policy” on page 149 in “Configuring IPSec” on page 119.

3. Enter IPsec Identity Configuration mode.

```
host1(config)#ipsec identity
host1(config-ipsec-identity)#
```

4. Specify the information that the router uses to generate a certificate request.

- a. Specify a country name.

```
host1(config-ipsec-identity)#country CA
```

- b. Specify a common name.

```
host1(config-ipsec-identity)#common-name Jim
```

- c. Specify a domain name.

```
host1(config-ipsec-identity)#domain-name myerx.kanata.junipernetworks.com
```

- d. Specify an organization.

```
host1(config-ipsec-identity)#organization juniperNetworks
host1(config-ipsec-identity)#exit
host1(config)#
```

5. Generate a certificate request using certificate parameters from the IPsec identity configuration.

```
host1(config)#ipsec certificate-request generate rsa myrequest.crq
```

6. After the certificate request is generated, you need to copy the file from the router and send it to the CA. Typically, you copy the file and paste it to a CA's Web page.

7. When you receive the certificate from the CA, copy the certificate to the router, and then inform the router that the new certificate exists.

```
host1(config)#ipsec certificate-database refresh
```

8. (Optional) Set the sensitivity of how the router handles CRLs.

```
host1(config)#ipsec crl ignored
```

9. (Optional) To delete RSA key pairs, use the **ipsec key zeroize** command.

```
host1(config)#ipsec key zeroize rsa
```

### ***authentication***

- Use to specify the authentication method that the router uses. For digital certificates, the method is set to RSA signature.

- Example

```
host1(config-ike-policy)#authentication rsa-sig
```

- Use the **no** version to restore the default, preshared keys.
- See authentication.

### ***common-name***

- Use to specify a common name used to generate certificate requests.
- Example

```
host1(config-ipsec-identity)#common-name Jim
```

- Use the **no** version to remove the common name.
- See common-name.

### *country*

- Use to specify a country name used to generate certificate requests.
- Example

```
host1(config-ipsec-identity)#country CA
```

- Use the **no** version to remove the country name.
- See country.

### *domain-name*

- Use to specify the domain name that the router uses in IKE authentication messages and to generate certificate requests.
- The domain name is used in the SubjectAlternative DNS certificate extensions and as an FQDN (fully qualified domain name) ID payload for IKE negotiations.
- Example

```
host1(config-ipsec-identity)#domain-name myerx.kanata.junipernetworks.com
```

- Use the **no** version to remove the domain name.
- See domain-name.

### *ike crl*

- Use to control how the router handles CRLs during negotiation of IKE phase 1 signature authentication. Specify one of the following keywords:
    - **ignored**—Allows negotiations to succeed even if a CRL is invalid or the peer's certificate appears in the CRL; this is the most lenient setting
    - **optional**—If the router finds a valid CRL, it uses it; this is the default setting
    - **required**—Requires a valid CRL; either the certificates that belong to the E Series router or the peer must not appear in the CRL; this is the strictest setting
  - Example
- ```
host1(config)#ike crl ignored
```
- Use the **no** version to return the CRL setting to the default, optional.



NOTE: This command has been replaced by “[ipsec crl](#)” on page 216 and may be removed completely in a future release.

- See ike crl.

ipsec certificate-database refresh

- Use to inform the ERX router that a public key certificate has been copied to the router. The router then verifies public certificates found on its disk against its private key and prepares the certificates for use.



NOTE: On reload, the router scans all certificate files and determines which files are router public certificates and which are root CA certificates.

- Example
`host1(config)#ipsec certificate-database refresh`
- There is no **no** version.
- See `ipsec certificate-database refresh`.

ipsec certificate-request generate

- Use to cause the router to generate a certificate request using certificate parameters from the IPSec identity configuration.
- Include a name for the certificate request file. The file name must have a .crq extension.
- After the router generates the certificate, use offline methods to send the certificate request file to the CA.
- Example
`host1(config)#ipsec certificate-request generate rsa myrequest.crq`
- There is no **no** version.
- See `ipsec certificate-request generate`.

ipsec crl

- Use to control how the router handles CRLs during negotiation of IKE phase 1 signature authentication. Specify one of the following keywords:
 - **ignored**—Allows negotiations to succeed even if a CRL is invalid or the peer's certificate appears in the CRL; this is the most lenient setting
 - **optional**—If the router finds a valid CRL, it uses it; this is the default setting
 - **required**—Requires a valid CRL; either the certificates that belong to the E Series router or the peer must not appear in the CRL; this is the strictest setting
- Example
`host1(config)#ipsec crl ignored`
- Use the **no** version to return the CRL setting to the default, optional.



NOTE: This command replaces “ike crt” on page 215, which may be removed completely in a future release.

- See ipsec crt.

ipsec identity

- Use to enter IPSec Identity Configuration mode in which you can specify information that the router uses in certificate requests and during negotiations with its peers.

- Example

```
host1(config)#ipsec identity
host1(config-ipsec-identity)#
```

- Use the **no** version to remove the identity configuration.
- See ipsec identity.

ipsec ike-policy-rule

- Use to define an ISAKMP/IKE policy.
- When you enter the command, you include a number that identifies the policy and assigns a priority to the policy. You can number policies in the range 1–10000, with 1 having the highest priority.

- Example

```
host1(config)#ipsec ike-policy-rule 3
host1(config-ike-policy)#
```

- Use the **no** version to remove policies. If you do not include a priority number with the **no** version, all policies are removed.



NOTE: This command replaces “ipsec isakmp-policy-rule” on page 217, which may be removed completely in a future release.

- See ipsec ike-policy-rule.

ipsec isakmp-policy-rule

- Use to define an ISAKMP/IKE policy.
- When you enter the command, you include a number that identifies the policy and assigns a priority to the policy. You can number policies in the range 1–10000, with 1 having the highest priority.

- Example

```
host1(config)#ipsec isakmp-policy-rule 3
host1(config-ike-policy)#
```

- Use the **no** version to remove policies. If you do not include a priority number with the **no** version, all policies are removed.



NOTE: This command has been replaced by “[ipsec ike-policy-rule](#)” on [page 217](#) and may be removed completely in a future release.

- See `ipsec isakmp-policy-rule`.

ipsec key generate

- Use to generate RSA key pairs. Include a length of either 1024 or 2048 bits. The generated keys can be used only after the CA issues a certificate for them.
- Example

```
host1(config)#ipsec key generate rsa 2048
Please wait.....
.....
IPsec Generate Keys complete
```
- There is no **no** version. To remove a key pair, use the **ipsec key zeroize** command.
- See `ipsec key generate`.

ipsec key zeroize

- Use to delete RSA key pairs. Include one of the following keywords:
 - **rsa**—Removes the RSA key pair from the router
 - **pre-share**—Removes all preshared keys from the router
 - **all**—Removes all keys within the VR context from the router
- Example

```
host1(config)#ipsec key zeroize rsa
```
- There is no **no** version.
- See `ipsec key zeroize`.

organization

- Use to specify the organization used in the Subject Name field of certificates.
- Example

```
host1(config-ipsec-identity)#organization juniperNetworks
```
- Use the **no** version to remove the organization name.
- See `organization`.

Configuring Digital Certificates Using the Online Method

To use the online configuration method to set up digital certificates on the router:

1. Generate the RSA key pair.

```
host1(config)#ipsec key generate rsa 2048
Please wait.....
.....
IPsec Generate Keys complete
```

2. In your IKE policy, set the authentication method to RSA signatures.

```
host1(config)#ipsec ike-policy-rule 1
host1(config-ike-policy)#authentication rsa-sig
host1(config-ike-policy)#exit
```



NOTE: For more information about setting up IKE policies, see [“Defining an IKE Policy” on page 149](#) in [“Configuring IPSec” on page 119](#).

3. Enter IPSec CA Identity Configuration mode, and specify the name of the certificate authority.

```
host1(config)#ipsec ca identity trustedca1
host1(config-ca-identity)#
```

4. Specify the name of the CA issuer.

```
host1(config-ca-identity)#issuer-identifier BetaSecurityCorp
```

5. Specify the URL of the SCEP server from which the CA certificates and the router's public certificates is retrieved.

```
host1(config-ca-identity)#enrollment url http://192.168.99.105/scepurl
```

6. (Optional) Set the sensitivity of how the router handles CRLs.

```
host1(config-ca-identity)#crl ignored
```

7. (Optional) Specify the wait period between certificate request retries.

```
host1(config-ca-identity)#enrollment retry-period 5
```

8. (Optional) Specify the absolute time limit on enrollment.

```
host1(config-ca-identity)#enrollment retry-limit 60
```

9. (Optional) Specify the URL of your network's HTTP proxy server.

```
host1(config-ca-identity)#root proxy url http://192.168.5.45
host1(config-ca-identity)#exit
```

10. Retrieve the CA certificate.

```
host1(config)#ipsec ca authenticate trustedca1
```

11. Enroll with the CA and retrieve the router's certificate from the CA.

```
host1(config)#ipsec ca enroll trustedca1 My498pWd
```

12. (Optional) To delete RSA key pairs, use the **ipsec key zeroize** command.

authentication

- Use to specify the authentication method that the router uses. For digital certificates, the method is set to RSA signature.
- Example

```
host1(config-ike-policy)#authentication rsa-sig
```
- Use the **no** version to restore the default, preshared keys.
- See authentication.

crl

- Use to control how the router handles certificate revocation lists (CRLs) during negotiation of online IKE phase 1 signature authentication. Specify one of the following keywords:
 - **ignored**—Allows negotiations to succeed even if a CRL is invalid or the peer's certificate appears in the CRL; this is the most lenient setting
 - **optional**—If the router finds a valid CRL, it uses it; this is the default setting
 - **required**—Requires a valid CRL; either the certificates that belong to the E Series router or the peer must not appear in the CRL; this is the strictest setting
- Example

```
host1(config-ca-identity)#crl ignored
```
- Use the **no** version to return the CRL setting to the default, optional.
- See crl.

enrollment retry-limit

- Use to set the time period during which the router continues to send a certificate request to the CA. You can specify a time period in the range 0–480 minutes, with 0 specifying an infinite time period.
- Example

```
host1(config-ca-identity)#enrollment retry-limit 200
```
- Use the **no** version to restore the default of 60 minutes.
- See enrollment retry-limit.

enrollment retry-period

- Use to set the number of minutes that the router waits after receiving no response before resending a certificate request to the CA. You can specify a wait period in the range 0–60 minutes.
- Example

```
host1(config-ca-identity)#enrollment retry-period 40
```

- Use the **no** version to restore the default, 1 minute.
- See enrollment retry-period.

enrollment url

- Use to specify the URL of the SCEP server, in the format `http://server_ipaddress`. You can then use the **ipsec ca authentication** command to retrieve CA certificates from the SCEP server, and the **ipsec ca enroll** command to retrieve the router's public key certificates from the server.

- Example

```
host1(config-ca-identity)#enrollment url http://192.168.99.105/scepurl
```

- Use the **no** version to delete the enrollment URL specification.
- See enrollment url.

ipsec ca authenticate

- Use to retrieve the specified CA's certificate. If authentication is successful, the fingerprint is sent, and an `ikeEnrollment` message is logged at severity info.
- The CA must be previously declared by the **ipsec ca identity** command.

- Example

```
host1(config)#ipsec ca authenticate trustedca1
host1(config)#INFO 10/18/2003 03:45:16 ikeEnrollment (): Received CA certificate for
ca:trustedca1
INFO 10/18/2003 03:45:16 ikeEnrollment (): Received CA certificate for ca:trustedca1
fingerprint:28:19:ba:76:d8:e0:bb:22:60:cd:b9:2d:dc:b8:58:01
host1(config)#
```

- Use the **no ipsec ca identity** command for the specified CA, or boot the router using the factory defaults to remove the CA certificate that was generated during the online configuration.
- There is no **no** version.
- See `ipsec ca authenticate`.

ipsec ca enroll

- Use to enroll with the specified CA and to retrieve the router's public key certificate during online digital certificate configuration. If enrollment is successful, the CA sends the certificate to the router and logs an `ikeEnrollment` message is logged at severity info.
- Use the password option, if required by the CA, to access the CA and enable enrollment.
- The CA must be previously declared by the **ipsec ca identity** command.
- Example

```
host1(config)#ipsec ca enroll trustedca1 My498pWd
host1(config)#INFO 10/18/2003 03:49:33 ikeEnrollment (): Received erx certificate for
ca:trustedca1
```

host1(config)#

- Use the **no ipsec ca identity** command for the specified CA or boot the router using the factory defaults to remove the router's public certificate that was generated during the online configuration.
- There is no **no** version.
- See ipsec ca enroll.

ipsec ca identity

- Use to specify the CA that the ERX router uses for online certificate requests and to enter IPsec Identity Configuration mode.
- In IPsec Identity Configuration mode you specify information that the router uses in certificate requests and during negotiations with its peers.

- Example

```
host1(config)#ipsec ca identity trustedca1
host1(config-ipsec-identity)#
```

- Use the **no** version to remove the identity configuration.
- See ipsec ca identity.

ipsec ike-policy-rule

- Use to define an ISAKMP/IKE policy.
- When you enter the command, you include a number that identifies the policy and assigns a priority to the policy. You can number policies in the range 1–10000, with 1 having the highest priority.

- Example

```
host1(config)#ipsec ike-policy-rule 3
host1(config-ike-policy)#
```

- Use the **no** version to remove policies. If you do not include a priority number with the **no** version, all policies are removed.



NOTE: This command replaces “[ipsec isakmp-policy-rule](#)” on [page 217](#), which may be removed completely in a future release.

- See ipsec ike-policy-rule.

ipsec isakmp-policy-rule

- Use to define an ISAKMP/IKE policy.
- When you enter the command, you include a number that identifies the policy and assigns a priority to the policy. You can number policies in the range 1–10000, with 1 having the highest priority.
- Example

```
host1(config)#ipsec isakmp-policy-rule 3
host1(config-ike-policy)#
```

- Use the **no** version to remove policies. If you do not include a priority number with the **no** version, all policies are removed.



NOTE: This command has been replaced by “[ipsec ike-policy-rule](#)” on [page 217](#) and may be removed completely in a future release.

- See ipsec isakmp-policy-rule.

ipsec key generate

- Use to generate RSA key pairs. Include a length of either 1024 or 2048 bits. The generated keys can be used only after the CA issues a certificate for them.
- Example

```
host1(config)#ipsec key generate rsa 2048
Please wait.....
.....
IPsec Generate Keys complete
```

- There is no **no** version. To remove a key pair, use the **ipsec key zeroize** command.
- See ipsec key generate.

ipsec key zeroize

- Use to delete RSA key pairs. Include one of the following keywords:
 - **rsa**—Removes the RSA key pair from the router
 - **pre-share**—Removes all preshared keys from the router
 - **all**—Removes all keys within the VR context from the router
- Example


```
host1(config)#ipsec key zeroize rsa
```
- There is no **no** version.
- See ipsec key zeroize.

issuer-identifier

- Use to specify the name of the CA issuer for online digital certificate configuration. The identifier and the enrollment URL specified by the **enrollment url** command are used together to create the CA authentication requests.
- Example


```
host1(config-ca-identity)#issuer-identifier BetaSecurityCorp
```
- Use the **no** version to remove the name from the configuration.
- See issuer-identifier.

root proxy url

- Use to specify an HTTP proxy server that can submit HTTP requests on the E Series router's behalf to retrieve the root CA certificate. Use this command if your network has an HTTP proxy server installed between the E Series router and the Internet. Use the format `http://server_ipaddress` to specify the URL of the proxy server.
- Example

```
host1(config-ca-identity)#root proxy url http://192.168.5.45
```
- Use the **no** version to remove the root proxy URL from the configuration.
- See root proxy url.

Configuring Peer Public Keys Without Digital Certificates

During IKE negotiations, peers exchange public keys to authenticate each other's identity and to ensure that IKE SAs are established with the intended party. Typically, public keys are exchanged in messages containing an X.509v3 digital certificate. As an alternative, however, you can configure and exchange peer public keys and use them for RSA authentication *without* having to obtain a digital certificate.

To configure and exchange peer public keys without obtaining a digital certificate:

1. Generate the RSA key pair on the router.

```
host1(config)#ipsec key generate rsa 1024
Please wait...
IPsec Generate Keys complete
```

2. In your IKE policy, set the authentication method to RSA signature.

```
host1(config)#ipsec ike-policy-rule 1
host1(config-ike-policy)#authentication rsa-sig
host1(config-ike-policy)#exit
host1(config)#exit
```



NOTE: For more information about setting up IKE policies, see [“Defining an IKE Policy” on page 149](#) in [“Configuring IPSec” on page 119](#).

3. Display the router's public key.

```
host1#show ipsec key mypubkey rsa
30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 00daaa65
8082ac0a ec42e552 10e3489b 37463ed8 9bfa2541 f46a7b30 0e908749 5b652ae5
ae604e9a 81bc3268 270e7f68 69ffd2a8 be268afa 92849fd0 4e8c96be 3eddf1c2
12d9fe7a 68e8507c 99b59ff3 bb0c3942 b0a90c76 3ae3acbb 4a777037 31527ea0
23693bdc e5393c6f 2ef3e7e7 bb1a308e d42ce0ad a095273e d718384c dd020301
0001
```

For information about the format of an RSA public key, see [“Public Key Format” on page 213](#).

4. Use the output from the **show ipsec key mypubkey rsa** command to provide information to the remote peer about the public key configured on the E Series router. Providing this information enables the remote peer to enter the router's public key on its own system.

The **show ipsec key mypubkey rsa** command enables you to display the contents of the router's public key without having to obtain a digital certificate.

5. Obtain the public key from the remote peer.

For example, you might receive an e-mail message from the remote peer containing the public key information.

6. Configure the public key for the remote IKE peer.

- a. Access IPSec Peer Public Key Configuration mode.

You must identify the remote peer associated with the public key by specifying the remote peer's IP address, fully qualified domain name (FQDN), or FQDN preceded by an optional *user@* specification. For example, the following command enables you to enter the peer public key for the remote peer identified by IP address 192.168.15.5.

```
host1(config)#ipsec key pubkey-chain rsa address 192.168.15.5
host1(config-peer-public-key)#
```

- b. Enter the peer public key that you obtained in Step 5.

```
host1(config-peer-public-key)#key-string "
Enter remainder of text message. End with the character '""'.
```

```
30820122 300d0609 2a864886 f70d0101 01050003 82010f00 3082010a 02820101
00effc6f d91cbf23 5de66454 420db27a 0bacfc92 63a54e60 587c3e1c 951be4e8
09e7d130 da924040 0ceb797c ddc0df10 dabeb3fc a17145ff 6e7ff977 68ac0698
748d30f4 478252ed 29bf3e4e a6657cc8 cfaf1de4 e7dc2473 33231286 0ecfb15b
4aac505b 255f17ca faf884ca f0402022 5ad6f446 e0f3fb1e d48bbc00 5d4fe9b6
35f88b53 1bf4f07c b168e47b b7143181 5bad4586 0abb7b03 6dba9668 b45e3714
0b64ca82 3a53f69b 357a7d41 f512da37 71901b14 08212648 277f6d38 6bc34164
8c3ac8d4 d9c8baac dc006dac 8c09ce37 44a5d124 b69fec24 df0fc3a8 98e6efc8
5a1d65eb e4b832ba adc26c63 1996fe37 e797ecff 6e2acdd6 0981ef2c 3dd2f506
01020301 0001"
```

- c. (Optional) Verify the peer public key configuration.

```
host1#show ipsec key pubkey-chain rsa address 192.168.15.5

30820122 300d0609 2a864886 f70d0101 01050003 82010f00 3082010a 02820101
00effc6f d91cbf23 5de66454 420db27a 0bacfc92 63a54e60 587c3e1c 951be4e8
09e7d130 da924040 0ceb797c ddc0df10 dabeb3fc a17145ff 6e7ff977 68ac0698
748d30f4 478252ed 29bf3e4e a6657cc8 cfaf1de4 e7dc2473 33231286 0ecfb15b
4aac505b 255f17ca faf884ca f0402022 5ad6f446 e0f3fb1e d48bbc00 5d4fe9b6
35f88b53 1bf4f07c b168e47b b7143181 5bad4586 0abb7b03 6dba9668 b45e3714
0b64ca82 3a53f69b 357a7d41 f512da37 71901b14 08212648 277f6d38 6bc34164
8c3ac8d4 d9c8baac dc006dac 8c09ce37 44a5d124 b69fec24 df0fc3a8 98e6efc8
5a1d65eb e4b832ba adc26c63 1996fe37 e797ecff 6e2acdd6 0981ef2c 3dd2f506
01020301 0001
```

authentication

- Use to specify in the ISAKMP/IKE policy that the router uses the RSA signature authentication method for IKE negotiations.
- Example

```
host1(config-ike-policy)#authentication rsa-sig
```
- Use the **no** version to restore the default authentication method, preshared keys.
- See authentication.

ipsec ike-policy-rule

- Use to access IPsec IKE Policy Configuration mode to define an ISAKMP/IKE policy.
- For information about how to use this command, see [“ipsec ike-policy-rule” on page 217](#).
- Example

```
host1(config)#ipsec ike-policy-rule 2
host1(config-ike-policy)#
```
- Use the **no** version to remove policies. If you do not include a priority number with the **no** version, all policies are removed.
- See ipsec ike-policy-rule.

ipsec key generate

- Use to generate a 1024-bit or 2048-bit RSA key pair.
- Example

```
host1(config)#ipsec key generate rsa 2048
Please wait.....
.....
IPsec Generate Keys complete
```
- There is no **no** version. To remove a key pair, use the **ipsec key zeroize** command.
- See ipsec key generate.

ipsec key pubkey-chain rsa

- Use to access IPsec Peer Public Key Configuration mode to configure the public key for a remote peer with which you want to establish IKE SAs.
- The **ipsec key pubkey-chain rsa** command enables you to manually enter the public key data for the remote peer without having to obtain a digital certificate.
- To specify the IP address of the remote peer associated with the public key, use the **address** keyword followed by the IP address, in 32-bit dotted decimal format.
- To specify the identity of the remote peer associated with the public key, use the **name** keyword followed by either:
 - The fully qualified domain name (FQDN)

- The FQDN preceded by an optional *user@* specification; this is also referred to as user FQDN format
- The FQDN and user FQDN identifiers are case-sensitive.
- To ensure that the public key is associated with the correct remote peer, the router requires an exact match for the identifier string. For example, a public key for user FQDN `mjones@sales.company_abc.com` does not match a public key for FQDN `sales.company_abc.com`.
- From IPsec Peer Public Key Configuration mode, use the **key-string** command to enter the peer public key data. For information about how to use this command, see [“key-string” on page 227](#).
- Example 1—Enables you to configure the public key for a remote peer with IP address 192.168.50.10


```
host1(config)#ipsec key pubkey-chain rsa address 192.168.50.10
host1(config-peer-public-key)#
```
- Example 2—Enables you to configure the public key for a remote peer with the FQDN `sales.company_xyz.com`

```
host1(config)#ipsec key pubkey-chain rsa name sales.company_xyz.com
host1(config-peer-public-key)#
```
- Example 3—Enables you to configure the public key for a remote peer with the FQDN `tsmith@sales.company_xyz.com`

```
host1(config)#ipsec key pubkey-chain rsa name tsmith@sales.company_xyz.com
host1(config-peer-public-key)#
```
- Use the **no** version to remove the peer public key from the router.
- See `ipsec key pubkey-chain rsa`.

key-string

- Use to manually enter a 1024-bit or 2048-bit public key for a remote peer with which you want to establish IKE SAs.
- The key string represents the public key hexadecimal data that includes the ASN.1 object identifier and sequence tags for RSA encryption.
- Enter an alphanumeric key string with a maximum of 1999 characters.
- You must use the same character (for example, “ or x) at the beginning and end of the string to delimit the key string. The delimiter character is case-sensitive and must not occur anywhere else in the key string.
- For information about the format of an RSA public key, see [“Public Key Format” on page 213](#).
- Example 1—Configures the public key for a remote peer with IP address 192.168.50.10, using “ (double quotation marks) as the key string delimiter character

```
host1(config)#ipsec key pubkey-chain rsa address 192.168.50.10
host1(config-peer-public-key)#key-string "
Enter remainder of text message. End with the character '"'
```

```
30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 00d3a447
```

```
0b997844 213de4ae 13a2c09b f74051cd d404a187 c5e86867 d525cb6e 571a44f2
92bac7e8 bb282857 fb20357c d94ec241 b651596c 350dd770 6853526b c95e60c1
52ec06ce 094882a7 4a7275a6 af1b738f 29d1124d 21e49b2a 3b0b7f2f fe31f0cc
178ddbfe a587a7a9 83aa0601 e86e7de4 3ca78f60 89a758bf 4c1247ba cb020301
0001"
```

- Example 2—Configures the public key for a remote peer with the FQDN sales.company_xyz.com, using ' (single quotation mark) as the key string delimiter character

```
host1(config)#ipsec key pubkey-chain rsa name sales.company_xyz.com
host1(config-peer-public-key)#key-string '
Enter remainder of text message. End with the character '".
```

```
30820122 300d0609 2a864886 f70d0101 01050003 82010f00 3082010a 02820101
```

```
00c03cc6 0bad55ea b4f8a01f 5cf69de5 f03185e2 1338b5cb fa8418c3 6cbe1a77
bfeba5b 7a8f0ac2 6e2b223b 11e3c316 a30f7fb0 7bd2ab8a a614bb3d 2fce97bf
d6376467 0d5d1a16 d630c173 3ed93434 e690f355 00128ffb c36e72fa 46eae49a
5704eabe 0e34776c 7d243b8b fcb03c75 965c12f4 d68c6e63 33e0207c a985ffff
2422fb53 23d49dbb f7fd3140 a7f245ee bf629690 9356a29c b149451a 691a2531
9787ce37 2601bdf9 1434b174 4fd21cf2 48e10f58 9ac89df1 56e360b1 66fb0b3f
27ad6396 7a491d74 3b8379ea be502979 8f0270b2 6063a474 fadc5f18 f0ca6f7a
ddea66c7 cf637598 9cdb5087 0480af29 b9c174ab 1b1d033f 67641a8c 5918ddce
1f020301 0001'
```

- Example 3—Configures the public key for a remote peer with the user FQDN tsmith@sales.company_xyz.com, using lowercase x as the key string delimiter character

```
host1(config)#ipsec key pubkey-chain rsa name tsmith@sales.company_xyz.com
host1(config-peer-public-key)#key-string x
Enter remainder of text message. End with the character 'x'.
```

```
30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 00bcc106
8694a505 0b92433e 4c27441e 3ad8955d 5628e2ea 5ee34b0c 6f82c4fd 8d5b7b51
f1a3c94f c4373f9b 70395011 79b4c2fb 639a075b 3d66185f 9cc6cdd1 6df51f74
cb69c8bb dbb44433 a1faac45 10f52be8 d7f2c8cd ad5172a6 e7f14b1c bba4037b
29b475c6 ad7305ed 7c460779 351560c6 344ccd1a 35935ea3 da5de228 bd020301
0001x
```

- There is no **no** version. Use the **no** version of the **ipsec key pubkey-chain rsa** command to remove the peer public key from the router.
- See key-string.

Monitoring Digital Certificates and Public Keys

Use the following **show** commands to display information about IKE certificates, IKE configurations, CRLs, public keys, and peer public keys.

show ipsec ca identity

- Use to display information about IKE CA identities used by the router for online digital certificate configuration. You can display information for a specific CA or for all CAs configured on the router.
- Field descriptions
 - CA—Certificate authority that the router uses to generate certificate requests
 - enrollment url—URL of the SCEP server where the router sends certificate requests
 - issuer id—Name of the CA issuer providing the digital certificates
 - retry period—Number of minutes that the router waits after receiving no response from the CA before resending a certificate request
 - retry limit—Number of minutes during which the router continues to send a certificate request to the CA
 - crl setting—Setting that controls how the router checks the certificate revocation lists
 - proxy url—HTTP proxy server used to retrieve the root CA certificate, if any
- Example

```
host1#show ipsec ca identity mysecureca1

CA: mysecureca1 parameters:
enrollment url:http://192.168.10.124/scepurl
issuer id      :BetaSecurityCorp
retry period   :1
retry limit    :60
crl setting    :optional
proxy url      :
```

- See show ipsec ca identity.

show ipsec certificates

show ike certificates



NOTE: The `show ike certificates` command has been replaced by the `show ipsec certificates` command and may be removed completely in a future release.

- Use to display the IKE certificates and CRLs on the router. Specify the type of certificate you want to display:
 - **all**—All certificates configured on the router
 - **crl**—Certificate revocation lists
 - **peer**—Peer certificates

- **public-certs**—Public certificates
- **root-cas**—Root CA certificates
- Use the **hex-format** keyword to display certificate data, such as serial numbers, in hexadecimal format. Doing so allows easier comparison with CAs, such as Microsoft, that display certificates in hexadecimal format.
- Field descriptions
 - Ca identity—Certificate authority that the router uses to generate certificate requests
 - SubjectName—Distinguished name for the certificate
 - IssuerName—Organization that signed and issued the certificate
 - SerialNumber—Unique serial number assigned to the certificate by the CA
 - SignatureAlgorithm—Algorithm used for the digital signature
 - Validity—Beginning and ending period during which the certificate is valid
 - PublicKeyInfo—Information about the public key
 - Extensions—Fields that provide additional information for the certificate
 - Fingerprints—Unique hash of the certificate, which can be used to verify that the certificate is valid
- Example 1

```

host1#show ipsec certificates public-certs

----- Public Certificates: -----

Ca Identity:[trustedca1]Certificate =
  SubjectName = <C=us, O=junipernetworks, CN=jim>
  IssuerName = <C=CA, ST=ON, L=Kanata, O=BetaSecurityCorp, OU=VT Group,
CN=VT Root CA>
  SerialNumber= 84483276204047383658902
  SignatureAlgorithm = rsa-pkcs1-sha1
  Validity =
    NotBefore = 2003 Oct 21st, 16:14:42 GMT
    NotAfter  = 2004 Oct 21st, 16:24:42 GMT
  PublicKeyInfo =
    PublicKey =
      Algorithm name (SSH) : if-modn{sign{rsa-pkcs1-md5}}
      Modulus n (1024 bits) :

13409127965307061503054050053800642488356537668078160605242622661311625

19876607806686846822070359658649546374128540876213416858514288030584124

05896520823533525098960335493944208019747261524241389345208872551265097

58542773588125824612424422877870700028956172284401073039192457619002485
5366053321117704284702619
  Exponent e ( 17 bits) :
    65537
  Extensions =

```

```

    Available = authority key identifier, subject key identifier, key
usage,
    subject alternative name, authority information access, CRL
distribution
points
SubjectAlternativeNames =
    Following names detected =
        DNS (domain name server name)
    Viewing specific name types =
        DNS = host1.kanata.junipernetworks.com
KeyUsage = DigitalSignature
CRLDistributionPoints =
    % Entry 1
    FullName =
        Following names detected =
            URI (uniform resource indicator)
        Viewing specific name types =
            URI = http://vtsca1/CertEnroll/VTs%20Root%20CA.crl
    % Entry 2
    FullName =
        Following names detected =
            URI (uniform resource indicator)
        Viewing specific name types =
            No names of type IP, DNS, URI, EMAIL, RID, UPN or DN detected.
AuthorityKeyID =
KeyID =
    15:0a:17:4d:36:b6:49:96:fa:d5:be:df:51:3e:e4:90:51:a2:c0:95
AuthorityCertificateIssuer =
    Following names detected =
        DN (directory name)
    Viewing specific name types =
        No names of type IP, DNS, URI, EMAIL, RID, UPN or DN detected.
AuthorityCertificateSerialNumber =
79592882508437425959858112994892506178
SubjectKeyID =
KeyId =
    78:e0:3e:f7:24:65:2d:4b:01:d4:91:f9:66:c7:67:26:06:74:6c:5c
AuthorityInfoAccess =
AccessMethod = 1.3.6.1.5.5.7.48.2
AccessLocation =
    Following names detected =
        URI (uniform resource indicator)
    Viewing specific name types =
        No names of type IP, DNS, URI, EMAIL, RID, UPN or DN detected.
AccessMethod = 1.3.6.1.5.5.7.48.2
AccessLocation =
    Following names detected =
        URI (uniform resource indicator)
    Viewing specific name types =
        No names of type IP, DNS, URI, EMAIL, RID, UPN or DN detected.
Fingerprints =
MD5 = c4:c9:22:b6:19:07:4e:4f:ee:81:7a:9f:cb:f9:1f:7e
SHA-1 = 58:ba:fb:0d:68:61:42:2a:52:7e:19:82:77:a4:55:4c:25:8c:c5:60

```

- Example 2

```

host1# show ipsec certificates root-cas
----- Root CAs: -----
Ca Identity:[trustedca1]Certificate =
    SubjectName = <C=CA, ST=ON, L=Kanata, O=Juniper Networks, OU=VTS Group,
CN=VTS Root CA>

```

```

    IssuerName = <C=CA, ST=ON, L=Kanata, O=BetaSecurityCorp, OU=VT Group,
CN=VT Root CA>
    SerialNumber= 79592882508437425959858112994892506178
    SignatureAlgorithm = rsa-pkcs1-sha1
    Certificate seems to be self-signed.
    * Signature verification success.
    Validity =
        NotBefore = 2003 Mar 26th, 15:50:53 GMT
        NotAfter  = 2006 Mar 26th, 15:59:59 GMT
    PublicKeyInfo =
        PublicKey =
            Algorithm name (SSH) : if-modn{sign{rsa-pkcs1-md5}}
            Modulus n (1024 bits) :

14424807498766001201060433525671934401816213246866823722650117007030500

12414152472800629737773845549310833804653975288246486381759003010224672

53370575541853958272072875412915858260834056069053966369912244336288229

09443381900005615652631560044304863856421739848326865877661787314144447
    82765023232108941157077
    Exponent e ( 17 bits) :
    65537
    Extensions =
        Available = subject key identifier, key usage, basic
constraints(critical),
CRL distribution points, unknown
    KeyUsage = DigitalSignature NonRepudiation KeyCertSign CRLSign
    BasicConstraints =
        cA = TRUE
    [critical]
    CRLDistributionPoints =
    % Entry 1
    FullName =
    Following names detected =
    URI (uniform resource indicator)
    Viewing specific name types =
    URI = http://vtscal/CertEnroll/VTS%20Root%20CA.crl
    % Entry 2
    FullName =
    Following names detected =
    URI (uniform resource indicator)
    Viewing specific name types =
    No names of type IP, DNS, URI, EMAIL, RID, UPN or DN detected.
    SubjectKeyID =
    KeyId =
    15:0a:17:4d:36:b6:49:96:fa:d5:be:df:51:3e:e4:90:51:a2:c0:95
    Unknown 1.3.6.1.4.1.311.21.1 =
    02:01:00
    Fingerprints =
    MD5 = 8c:56:fb:a6:bd:ab:13:67:e6:13:09:c1:d0:de:1f:24
    SHA-1 = 22:3d:84:6d:d4:5f:18:87:ae:2c:15:7d:2a:94:20:ff:c6:12:fb:6f

```

- See show ike certificates.
- See show ipsec certificates.

show ipsec identity

show ike identity



NOTE: The `show ike identity` command has been replaced by the `show ipsec identity` command and may be removed completely in a future release.

- Use to display the IKE identity configuration.
- Field descriptions
 - Domain Name—Domain name the router uses in IKE authentication messages and to generate certificate requests
 - Common Name—Common name used to generate certificates
 - Organization—Name of the organization used in the Subject Name field of certificates
 - Country—Country used to generate certificates

- Example

```
host1#show ipsec identity
```

```
Ike identity:
```

```
Domain Name :myerx.kanata.junipernetworks.com
```

```
Common Name :jim
```

```
Organization:junipernetworks
```

```
Country      :ca
```

- See `show ipsec identity`.
- See `show ike identity`.

show ipsec ike-configuration

show ike configuration



NOTE: The `show ike configuration` command has been replaced by the `show ipsec ike-configuration` command and may be removed completely in a future release.

- Use to display a summary of the IKE configuration.
- Field descriptions
 - Ike identity—Information from your IKE identify configuration that the router uses to generate certificate requests
 - CRL Check—Setting of the CRL check: optional, required, ignored

- Example

```
host1#show ipsec ike-configuration
```

```
Ike configuration:
```

```
Ike identity:
```

```
Domain Name :treverxsys2.juniper.net
```

```
Common Name :Sys2 ERX
Organization:Juniper Networks
Country      :CA
CRL Check:optional
```

- See show ipsec ike-configuration.
- See show ike configuration.

show ipsec key mypubkey rsa

- Use to display the 1024-bit or 2048-bit RSA public key configured on the router.
- The public key is generated as part of a public/private key pair used to perform RSA authentication during ISAKMP/IKE SA negotiations.
- For information about the format of an RSA public key, see [“Public Key Format” on page 213](#).
- Example

```
host1#show ipsec key mypubkey rsa
30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 009cfbde
a16cf72c 49fbd3c1 10d5d9d4 8ba15ec0 9adcb19e 18d488f8 e0370c51 2d10e751
ddd81be4 dfc78aad 9deb797f b2c51172 18967cfb e18f6efa 69285fef 10337527
78ca6bbc 907abb9e 44b12713 ab70cb0e a86d9c6c 80c99bd1 e2bf6b70 91222295
616a88bb cc479e15 be04f3a5 a6160645 844598c3 314b66af 3a8b7602 ed020301
0001
```

- See show ipsec key mypubkey rsa.

show ipsec key pubkey-chain rsa

- Use to display a 1024-bit or 2048-bit ISAKMP/IKE public key that a remote peer uses for RSA authentication.
- To display a brief summary of the remote peers for which public keys are configured on the router, use the **summary** keyword.
- To display the public key for a remote peer with a specific IP address, use the **address** keyword followed by the IP address, in 32-bit dotted decimal format.
- To display the public key for a remote peer with a specific identity, use the **name** keyword followed by either:
 - The fully qualified domain name (FQDN)
 - The FQDN preceded by an optional *user@* specification; this is also referred to as user FQDN format
- The FQDN and user FQDN identifiers are case-sensitive and must exactly match the identifier specified in the **ipsec key pubkey-chain rsa** command. For example, a public key for user FQDN *mjones@sales.company_abc.com* does not match a public key for FQDN *sales.company_abc.com*.
- For information about the format of an RSA public key, see [“Public Key Format” on page 213](#).
- Field descriptions

- Remote Peer—IP address, FQDN, or user FQDN identifier of the remote peer for which the peer public key can be used
- Key Type—Type of remote peer identifier: ip address (if IP address is specified) or identity (if FQDN or user FQDN is specified)
- Example 1—Displays a summary of the remote peers for which peer public keys are configured

```
host1#show ipsec key pubkey-chain rsa summary
      Remote Peer          Key Type
-----
192.168.32.3              ip address
grp003.cust535.isp.net    identity
tsmith@grp003.cust535.isp.net identity
```

- Example 2—Displays the peer public key for a remote peer with the specified IP address

```
host1#show ipsec key pubkey-chain rsa address 192.168.32.3

30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 0082065f
841aa03a fadfd9f bf8be05c d2fe3596 abc3e265 0b86b99a df9b4907 29c7a737
8bf08491 5c96e72d 28471a12 f0735ff4 04d76ad1 3a80f10c 23dcadda b68ce8ec
5fdfbe58 a52008db 9a11f867 d38d0483 e4abd53c 89a4dc3c 985ea450 f17748c4
3f04def0 a3cf5d89 b62dfeae 5990641b 370bb113 73105ba7 585a41fc 3b020301
0001
```

- Example 3—Displays the peer public key for a remote peer with the specified FQDN identifier

```
host1#show ipsec key pubkey-chain rsa name grp003.cust535.isp.net

30820122 300d0609 2a864886 f70d0101 01050003 82010f00 3082010a 02820101
00c03cc6 0bad55ea b4f8a01f 5cf69de5 f03185e2 1338b5cb fa8418c3 6cbe1a77
bfefba5b 7a8f0ac2 6e2b223b 11e3c316 a30f7fb0 7bd2ab8a a614bb3d 2fce97bf
d6376467 0d5d1a16 d630c173 3ed93434 e690f355 00128ffb c36e72fa 46eae49a
5704eabe 0e34776c 7d243b8b fcb03c75 965c12f4 d68c6e63 33e0207c a985ffff
2422fb53 23d49dbb f7fd3140 a7f245ee bf629690 9356a29c b149451a 691a2531
9787ce37 2601bdf9 1434b174 4fd21cf2 48e10f58 9ac89df1 56e360b1 66fb0b3f
27ad6396 7a491d74 3b8379ea be502979 8f0270b2 6063a474 fadc5f18 f0ca6f7a
ddea66c7 cf637598 9cdb5087 0480af29 b9c174ab 1b1d033f 67641a8c 5918ddce
1f020301 0001
```

- Example 4—Displays the peer public key for a remote peer with the specified user FQDN identifier

```
host1#show ipsec key pubkey-chain rsa name tsmith@grp003.cust535.isp.net

30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 00bcc106
8694a505 0b92433e 4c27441e 3ad8955d 5628e2ea 5ee34b0c 6f82c4fd 8d5b7b51
f1a3c94f c4373f9b 70395011 79b4c2fb 639a075b 3d66185f 9cc6cdd1 6df51f74
cb69c8bb dbb44433 a1faac45 10f52be8 d7f2c8cd ad5172a6 e7f14b1c bba4037b
29b475c6 ad7305ed 7c460779 351560c6 344ccd1a 35935ea3 da5de228 bd020301
0001
```

- See show ipsec key pubkey-chain rsa.

CHAPTER 9

Configuring IP Tunnels

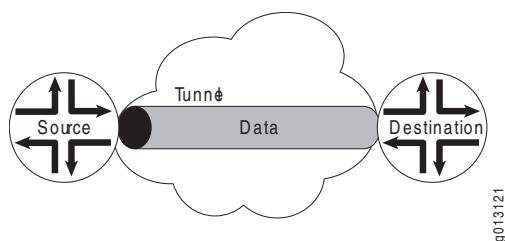
IP tunnels provide a way of transporting datagrams between routers separated by networks that do not support all the protocols that those routers support. This chapter describes how to configure IP tunnels on E Series routers; it contains the following sections:

- [Overview on page 237](#)
- [Platform Considerations on page 238](#)
- [References on page 239](#)
- [Configuration Tasks on page 240](#)
- [Monitoring IP Tunnels on page 244](#)

Overview

E Series routers support static IP tunnels. An IP tunnel is a virtual point-to-point connection between two routers. See [Figure 19 on page 237](#). To establish an IP tunnel, you specify a tunnel type and name, and then configure an interface on each router to act as an endpoint for the tunnel.

Figure 19: IP Tunneling



E Series routers support the following types of IP tunnels:

- Generic Routing Encapsulation (GRE) tunnels
- Distance Vector Multicast Routing Protocol (DVMRP) tunnels, also known as IP-in-IP tunnels

GRE Tunnels

GRE encapsulates IP packets to enable data transmission through an IP tunnel. The resulting encapsulated packet contains a GRE header and a delivery header. Consequently,

the packet requires more processing than an IP packet, and GRE can be slower than native routing protocols.

GRE tunnels can be secured with IPSec. See [“Securing L2TP and IP Tunnels with IPSec” on page 277](#).

DVMRP Tunnels

DVMRP tunnels allow the exchange of IP multicast traffic between routers separated by networks that do not support multicast routing. For information about DVMRP, see *Configuring IPv4 Multicast in JunosE Multicast Routing Configuration Guide*.

DVMRP tunnels can be secured with IPSec. See [“Securing L2TP and IP Tunnels with IPSec” on page 277](#).

Platform Considerations

For information about modules that support IP tunnels on the ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support IP tunnels.

For information about modules that support IP tunnels on the E120 and E320 Broadband Services Routers:

- See *E120 and E320 Module Guide, Table 1, Modules and IOAs* for detailed module specifications.
- See *E120 and E320 Module Guide, Appendix A, IOA Protocol Support* for information about the modules that support IP tunnels.

Module Requirements

The supported modules for creating IP tunnels depends on the type of E Series router that you have.

ERX7xx Models, ERX14xx Models, and the ERX310 Router

To create IP tunnels on an ERX7xx model, ERX14xx model, or an ERX310 router, you must install a Service line module (SM) or a module that supports the use of shared tunnel-server ports. For information about installing modules in the ERX routers, see the *ERX Hardware Guide*.

SMs provide dedicated tunnel-server ports that are always configured on the module. Unlike other line modules, SMs do not pair with corresponding I/O modules that contain ingress and egress ports. Instead, they receive data from and transmit data to other line modules with access to ingress and egress ports on their own associated I/O modules. However, you must assign interfaces on other line modules or loopback interfaces to act as source endpoints for the tunnel.

You can also create IP tunnels on router modules that support shared tunnel-server ports. You can configure (provision) a shared tunnel-server port to use a portion of the module's bandwidth to provide tunnel services. For a list of the modules that support shared tunnel-server ports, see the *ERX Module Guide*.

For information about configuring tunnel services on dedicated and shared tunnel-server ports, see *Managing Tunnel Service and IPSec Service Interfaces* in *JunosE Physical Layer Configuration Guide*.

All line modules forward traffic to IP tunnels. For information about which line modules accept traffic for IP tunnels, see the *ERX Module Guide*.

E120 Router and E320 Router

To create IP tunnels on an E120 router or an E320 router, you must install an ES2 4G line module (LM) or an ES2 10G ADV LM with an ES2-S1 Service I/O adapter (IOA), or an IOA that supports the use of shared tunnel-server ports. For information about installing modules in these routers, see the *E120 and E320 Hardware Guide*.

The combination of an ES2 4G LM or an ES2 10G ADV LM with an ES2-S1 Service IOA provides a dedicated tunnel-server port that is always configured on the IOA. Unlike SMs, the ES2 4G LM and the ES2 10G ADV LM require the ES2-S1 Service IOA to condition it to receive and transmit data to other line modules. The ES2-S1 Service IOA also does not have ingress or egress ports.

You can also create IP tunnels on IOAs that support shared tunnel-server ports. You can configure (provision) a shared tunnel-server port to use a portion of the bandwidth of the IOA to provide tunnel services. For a list of the IOAs that support shared tunnel-server ports, see the *E120 and E320 Module Guide*.

All line modules forward traffic to tunnels. For information about the IOAs that accept traffic for tunnels, see the *E120 and E320 Module Guide*.

Redundancy and Tunnel Distribution

For information about the redundancy and tunnel distribution mechanisms supported for SMs, the ES2-S1 Service IOA, and shared tunnel-server ports, see *Tunnel Service Interface Considerations* in *JunosE Physical Layer Configuration Guide*.

References

For more information about IP tunnels, see the following documents:

- RFC 791—Internet Protocol DARPA Internet Program Protocol Specification (September 1981)
- RFC 1700—Assigned Numbers (October 1994)
- RFC 1701—Generic Routing Encapsulation (October 1994)
- RFC 1702—Generic Routing Encapsulation over IPv4 Networks (October 1994)
- RFC 2003—IP Encapsulation within IP (October 1996)
- RFC 2784—Generic Routing Encapsulation (GRE) (March 2000)

Configuration Tasks

To configure an IP tunnel:

1. Create or select a physical or loopback interface.
This interface acts as an anchor for the source of the tunnel.
2. Assign an IP address to the physical or loopback interface.
3. Create a tunnel interface.
4. Set the source address for the tunnel.
5. Set the destination address for the tunnel.
6. (Optional) Enable error checking across a GRE tunnel.
7. Set the maximum transmission unit (MTU) size for the tunnel.



NOTE: On SM interfaces, issue only the commands listed below. Do not configure protocols such as Multilink PPP or Multilink Frame Relay on SM interfaces.

interface tunnel

- Use to create an IP tunnel interface.
- Specify the type and name of the tunnel you want to create.
- You can use the **transport-virtual-router** keyword to establish the tunnel on a virtual router other than the current virtual router.
- Example

```
host1(config)#interface tunnel dvmp:boston-tunnel-1 transport-virtual-router boston
```
- Use the **no** version to remove the tunnel.



NOTE: When you delete a virtual router that has been configured as a transport virtual router for a DVMRP tunnel, the **show configuration** output displays No Router for the transport virtual router. To remove the DVMRP tunnel interface, simply omit any reference to the transport virtual router. For example, to delete `interface tunnel dvmp:boston-tunnel-1 transport-virtual-router No Router` from the configuration, issue the command, `no interface tunnel dvmp:boston-tunnel-1`.

- See `interface tunnel`.

tunnel checksum

- Use to enable checksum computation across a GRE tunnel.
- Checksum computation is not supported for DVMRP tunnels.
- Selecting this feature causes the E Series router to drop corrupted packets it receives on the tunnel interface.
- Example

```
host1(config)#interface tunnel gre:tunnel2
host1(config-if)#tunnel checksum
```
- Use the **no** version to disable the checksum option.
- See tunnel checksum.

tunnel destination

- Use to configure the remote end of the tunnel.
- Specify either the IP address of an interface on the remote router or the hostname of the remote router.
 - The IP address is the address for the destination interface.
 - The hostname is the name of the destination interface.
- Example 1

```
host1(config)#interface tunnel dvmrp:tunnel2
host1(config-if)#tunnel destination 192.13.7.1
```
- Example 2

```
host1(config)#interface tunnel dvmrp:tunnel2
host1(config-if)#tunnel destination remoteHost
```
- Use the **no** version to remove the destination of a tunnel.
- See tunnel destination.

tunnel mtu

- Use to set the MTU for the tunnel.
- Specify a value in the range 1024–10240 bytes.
- Example

```
host1(config-if)#tunnel mtu 7500
```
- Use the **no** version to restore the default, 10240 bytes.
- See tunnel mtu.

tunnel source

- Use to configure the source of the tunnel.
- Specify either the primary IP address or the type and specifier of an interface.
- Do not specify an unnumbered interface.
- Example 1—Primary IP address

```
host1(config)#interface tunnel dvmrp:boston-tunnel-1
host1(config-if)#tunnel source 192.10.2.1
```

- Example 2—ATM interface on an ERX7xx model, ERX14xx model, or the ERX310 router that uses the *slot/port* format

```
host1(config)#interface tunnel dvmrp:boston-tunnel-1
host1(config-if)#tunnel source atm 5/0.12
```

- Example 3—ATM interface on an E320 router that uses the *slot/adaptor/port* format

```
host1(config)#interface tunnel dvmrp:boston-tunnel-1
host1(config-if)#tunnel source atm 5/1/0.12
```

- Use the **no** version to remove the source of a tunnel.
- See tunnel source.

Configuration Example

In this example, two GRE tunnel interfaces are configured on different virtual routers of an E Series router. The source of the first tunnel interface matches the destination of the second tunnel interface and vice versa.



NOTE: This example contains an ATM interface configuration for an ERX7xx model, ERX14xx model, or ERX310 router that uses the *slot/port* format.

1. Configure a virtual router called boston that supports one end of the tunnel.

```
host1#virtual-router boston
```

2. Configure a physical or loopback interface for the end of the tunnel on virtual router boston.

The IP address of this interface appears in the header of tunneled frames and is used for forwarding traffic.

```
host1:boston#interface atm 12/0.5
host1:boston(config-if)#ip address 10.5.5.5 255.255.255.0
```

3. Configure the tunnel interface on virtual router boston.

- a. Create the tunnel interface.

```
host1:boston(config)#interface tunnel gre:ChicagoTunnel
```

- b. Configure the source and destination points of the tunnel interface.

```
host1:boston(config-if)#tunnel source 10.5.5.5
host1:boston(config-if)#tunnel destination 10.6.6.6
```

- c. Set the MTU for the tunnel.

```
host1:boston(config-if)#tunnel mtu 8000
```

- d. Configure the IP address of the tunnel interface.

```
host1:boston(config-if)#ip address 10.7.7.7 255.255.255.0
```

4. Configure a virtual router called `chicago` that supports the other end of the tunnel.

```
host1(config)#virtual-router chicago
```

5. Configure a physical or loopback interface for the end of the tunnel on virtual router `chicago`.

```
host1:chicago(config)#interface atm 12/1.5
host1:chicago(config-if)#ip address 10.6.6.6 255.255.255.0
```

6. Configure the tunnel interface on virtual router `chicago`.

- a. Create the tunnel interface.

The name of the tunnel interface can differ from the tunnel interface configured in Step 3.

```
host1:chicago(config-if)#interface tunnel gre:BostonTunnel
```

- b. Configure the source and destination points of the tunnel interface.

The destination of this tunnel interface matches the source of the tunnel interface configured in Step 3 and vice versa.

```
host1:chicago(config-if)#tunnel source 10.6.6.6
host1:chicago(config-if)#tunnel destination 10.5.5.5
```

- c. Set the MTU for the tunnel.

The MTU must match the MTU configured in Step 3.

```
host1:chicago(config-if)#tunnel mtu 8000
```

- d. Configure the IP address of the tunnel interface.

```
host1:chicago(config-if)#ip address 10.9.9.9 255.255.255.0
```

Configuring IP Tunnels to Forward IP Frames

When a line module receives IP frames destined for a tunnel, the module forwards the frames to a tunnel-service module. Tunnel-service modules include SMs and modules that support the use of shared tunnel-server ports.

The tunnel-service module encapsulates the frames and forwards them to the tunnel through an interface determined by a route lookup of an IP frame. The source and destination addresses in the IP frame are the source and destination addresses of the tunnel.

Similarly, when a line module receives traffic from a tunnel, the module forwards the traffic to the tunnel-service module for deencapsulation. After deencapsulation, the tunnel-service module forwards the resulting IP frames to an interface determined by a route lookup.

When you have configured a tunnel interface, treat it in the same way as any IP interface on the router. For example, you can configure static IP routes or enable routing protocols on the tunnel interface. The IP configurations you apply to the tunnels control how traffic travels through the network.

Preventing Recursive Tunnels

If routing information about the tunnel network combines with routing information about the transport networks (the networks that the tunnel services), a *recursive* tunnel can occur. In this case, the routing table defines the tunnel itself as the best path to a tunnel destination. To prevent recursive tunnels, differentiate routing information for the tunnel network and the transport networks with one or both of the following techniques:

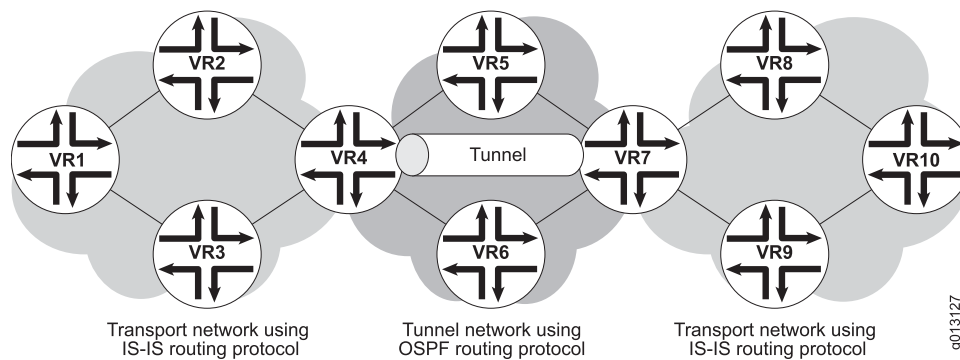
- Use different routing protocols for the tunnel network and the transport networks.
- Define a static route to the tunnel destination.



NOTE: If you define a static route to a tunnel destination, be careful not to create routing loops.

Figure 20 on page 244 illustrates how to prevent recursive tunnels by using different routing protocols for the tunnel network and the transport networks.

Figure 20: Transport and Tunnel Networks Using Different Routing Protocols



Creating Multicast VPNs Using GRE Tunnels

For information about configuring multicast VPNs using GRE tunnels, see *Configuring PIM for IPv4 Multicast* in *JunosE Multicast Routing Configuration Guide*.

Monitoring IP Tunnels

You can monitor DVMRP and GRE tunnels by using the following commands.

show dvmrp tunnel

- Use to display information about DVMRP tunnels.
- To view detailed information about tunnels, specify the **detail** keyword.
- To view the number of tunnels in a specific state, specify the **state** keyword and the state of the tunnel (disabled, down, enabled, lower-down, not-present, up).
- To view the state of a specific tunnel, specify a tunnel name.

- To view the number of tunnels associated with that IP address, specify an IP address.
- To view the number of tunnels associated with an IP address on the virtual router, specify an IP address with the **virtual-router** keyword and the name of the virtual router.
- Field descriptions
 - Tunnel status
 - up—Tunnel is operational
 - down—Tunnel is not operational
 - not-present—Tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
 - Tunnel name—Name of the tunnel
 - Tunnel mtu—Value of the maximum transmission unit for the tunnel
 - Tunnel source address—IP address of the source of the tunnel
 - Tunnel destination address—IP address of the destination of the tunnel
 - Tunnel transport virtual router—Name of the virtual router associated with the tunnel
 - Tunnel up/down trap—Indicates whether or not the E Series router sends traps to SNMP when the operational state of the tunnels changes, enabled or disabled
 - Tunnel server location—Location of the tunnel server in *slot/port* format (ERX7xx models, ERX14xx models, and the ERX310 router) or *slot/adaptor/port* format (E120 and E320 routers).
 - Tunnel secured by ipsec transport interface—IPSec interface that secures the tunnel.
 - Tunnel administrative state—Configured state of the tunnel: up or down
 - Statistics—Details of packets received or transmitted by the tunnel
 - Packets—Number of packets received or transmitted by the tunnel
 - Octets—Number of octets received or transmitted by the tunnel
 - Discards—Number of packets not accepted by the tunnel
 - Errors—Number of packets with errors received or transmitted by the tunnel
 - Data rx—Received data
 - Data tx—Transmitted data
 - Number of tunnels found—Total number of DVMRP tunnels found
 - Number of static tunnels—Number of tunnels created statically
- Example 1

```

host1#show dvmrp tunnel
DVMRP tunnel boston1 is up
1 DVMRP tunnel found
1 tunnel was created static

```

- Example 2

```

host1#show dvmrp tunnel detail
DVMRP tunnel boston1 is up (tunnel is static)
Tunnel operational configuration
  Tunnel name is 'boston1'
  Tunnel mtu is '10240'
  Tunnel source address is '0.0.0.0'
  Tunnel destination address is '0.0.0.0'
  Tunnel transport virtual router is vr1
  Tunnel up/down trap is enabled
  Tunnel server location is 4/0
  Tunnel secured by ipsec transport interface 1
  Tunnel administrative state is up
Statistics      packets      octets      discards      errors
Data rx        0              0            0              0
Data tx        0              0            0              0
1 DVMRP tunnel found
1 tunnel was created static

```

- Example 3

```

host1#show dvmrp tunnel state enabled
DVMRP tunnel boston1 is up
1 DVMRP tunnel found
1 tunnel was created static

```

- Example 4

```

host1#show dvmrp tunnel virtual-router vr1 ip 0.0.0.0
DVMRP tunnel boston1 is up
1 DVMRP tunnel found
1 tunnel was created static

```

- Example 5—Displays a DVMRP tunnel on an E320 router

```

host1#show dvmrp tunnel detail
DVMRP tunnel v1Tunnel1 is Up (tunnel is static)
Tunnel operational configuration
  Tunnel mtu is '10240'
  Tunnel source address is '50.1.1.1'
  Tunnel destination address is '50.1.1.2'
  Tunnel transport virtual router is v1
  Tunnel up/down trap is enabled
  Tunnel-server location is 13/0/0
  Tunnel administrative state is Up
Statistics      packets      octets      discards      errors
Data rx        5              740          0              0
Data tx        5              740          0              0

```

- See show dvmrp tunnel.

show dvmrp tunnel summary

- Use to display a summary of information about DVMRP tunnels.
- Field descriptions
 - Administrative status
 - enabled—Tunnel is available for use
 - disabled—Tunnel is not available for use
 - Operational status
 - up—Tunnel is operational
 - down—Tunnel is not operational
 - not-present—Tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible

- Example

```
host1#show dvmrp tunnel summary
Administrative status  enabled  disabled
                      1         0
Operational status    up      down    not-present
                      1         0         0
```

- See show dvmrp tunnel.

show gre tunnel

- Use to display information about a GRE tunnel or a list of GRE tunnels.
- To view detailed information about tunnels, specify the **detail** keyword.
- To view the number of tunnels in a specific state, specify the **state** keyword and the state of the tunnel (disabled, down, enabled, lower-down, not-present, up).
- To view the state of a specific tunnel, specify a tunnel name.
- To view the number of tunnels associated with an IP address, specify an IP address.
- To view the number of tunnels associated with an IP address on the virtual router, specify an IP address with the **virtual-router** keyword and the name of the virtual router.
- Field descriptions
 - Tunnel name—Name of the tunnel
 - Tunnel mtu—Value of the maximum transmission unit for the tunnel
 - Tunnel source address—IP address of the source of the tunnel
 - Tunnel destination address—IP address of the destination of the tunnel
 - Tunnel transport virtual router—Name of the virtual router associated with the tunnel
 - Tunnel mdt—State of the tunnel MDT
 - Tunnel checksum option—State of the checksum feature: enabled or disabled

- Tunnel up/down trap—Indicates whether or not the E Series router sends traps to SNMP when the operational state of the tunnels changes, enabled or disabled
- Tunnel server location—Location of the tunnel server in *slot/port* format (ERX7xx models, ERX14xx models, and the ERX310 router) or *slot/adaptor/port* format (E120 and E320 routers).
- Tunnel is secured by ipsec transport interface—IPSec interface that secures the tunnel.
- Tunnel administrative state—Configured state of the tunnel: up or down
- Statistics—Details of packets received or transmitted by the tunnel
- Packets—Number of packets received or transmitted by the tunnel
- Octets—Number of octets received or transmitted by the tunnel
- Discards—Number of packets not accepted by the tunnel
- Errors—Number of packets with errors received or transmitted by the tunnel
- Data rx—Received data
- Data tx—Transmitted data
- Number of tunnels found—Total number of GRE tunnels found
- Number of static tunnels—Number of tunnels created statically
- Example 1

```
host1#show gre tunnel
3 GRE tunnels found
3 tunnels were created static
```

- Example 2

```
host1#show gre tunnel detail
Tunnel operational configuration
Tunnel name is 'vr1'
Tunnel mtu is '10240'
Tunnel source address is '10.0.0.2'
Tunnel destination address is '10.0.0.1'
Tunnel transport virtual router is vr1
Tunnel mdt is disabled
Tunnel checksum option is disabled
Tunnel up/down trap is enabled
Tunnel server location is 4/0
Tunnel administrative state is up
```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

```
Tunnel operational configuration
Tunnel name is 'default'
Tunnel mtu is '10240'
Tunnel source address is '10.0.0.1'
```



```

Tunnel destination address is '10.0.0.2'
Tunnel transport virtual router is default
Tunnel checksum option is disabled
Tunnel up/down trap is enabled
Tunnel server location is 4/0
Tunnel secured by ipsec transport interface 1
Tunnel administrative state is up

```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

```

Tunnel operational configuration
Tunnel name is 'london2'
Tunnel mtu is '10240'
Tunnel source address is '0.0.0.0'
Tunnel destination address is '0.0.0.0'
Tunnel transport virtual router is vr1
Tunnel checksum option is disabled
Tunnel up/down trap is enabled
Tunnel server location is 4/0
Tunnel administrative state is up

```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

```

3 GRE tunnels found
3 tunnels were created static

```

- Example 3

```

host1#show gre tunnel state up
GRE tunnel VR1 is up
GRE tunnel default is up
GRE tunnel london2 is down
3 GRE tunnels found
3 tunnels were created static

```

- Example 4

```

host1#show gre tunnel virtual-router vr1 ip 10.0.0.1
GRE tunnel VR1 is up
1 GRE tunnel found
1 tunnel was created static

```

- Example 5—Displays a GRE tunnel on an E320 router

```

host1#show gre tunnel detail
GRE tunnel start is Up (tunnel is static)
Tunnel operational configuration
Tunnel mtu is '10240'
Tunnel source address is '15.0.0.1'
Tunnel destination address is '15.0.0.2'
Tunnel transport virtual router is default
Tunnel checksum option is disabled
Tunnel sequence number option is disabled
Tunnel up/down trap is enabled
Tunnel-server location is 1/0/0
Tunnel administrative state is Up

```

Statistics	packets	octets	discards	errors
------------	---------	--------	----------	--------

```

Data rx      0          0          0          0
Data tx      0          0          0          0
GRE tunnel end is Up
Tunnel operational configuration
Tunnel mtu is '10240'
Tunnel source address is '15.0.0.2'
Tunnel destination address is '15.0.0.1'
Tunnel transport virtual router is vpnA
Tunnel checksum option is disabled
Tunnel sequence number option is disabled
Tunnel up/down trap is enabled
Tunnel-server location is 1/0/0
Tunnel administrative state is Up
Statistics   packets          octets          discards    errors

Data rx      0          0          0          0
Data tx      0          0          0          0
2 GRE tunnels found
2 tunnels were created static

```

- See show gre tunnel.

show gre tunnel summary

- Use to display a summary of information about GRE tunnels.
- Field descriptions
 - Administrative status
 - enabled—Tunnel is available for use
 - disabled—Tunnel is not available for use
 - Operational status
 - up—Tunnel is operational
 - down—Tunnel is not operational
 - not-present—Tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
- Example

```

host1#show gre tunnel summary
Administrative status   enabled   disabled
                        3         0
Operational status     up        down     not-present
                        3         0         0

```

- See show gre tunnel.

Configuring Dynamic IP Tunnels

IP tunnels provide a way of transporting datagrams between routers separated by networks that do not support all the protocols that those routers support. This chapter describes how to configure dynamic IP tunnels on E Series routers; it contains the following sections:

- [Dynamic IP Tunnel Overview on page 251](#)
- [Platform Considerations on page 253](#)
- [References on page 255](#)
- [Configuring a Destination Profile for Dynamic IP Tunnels on page 255](#)
- [Monitoring Dynamic IP Tunnels on page 260](#)

Dynamic IP Tunnel Overview

E Series routers support the following types of dynamic IP tunnels:

- Generic Routing Encapsulation (GRE) tunnels
- Distance Vector Multicast Routing Protocol (DVMRP) tunnels, also known as IP-in-IP tunnels

To establish a dynamic IP tunnel for GRE or DVMRP interfaces, you must configure a destination profile for a specific transport virtual router that is used to store tunnel configuration options, including the source and destination addresses of the dynamic IP tunnel.

A client application triggers the creation of dynamic IP tunnels based on the information stored in the GRE or DVMRP destination profile. The application specifies a tunnel source, tunnel destination, transport virtual router, and tunnel mode (GRE or DVMRP). If these parameters match those configured in the destination profile, the system creates the dynamic IP tunnel.

The application can automatically create an upper layer IPv4 interface over the GRE or DVMRP interface by using the IP characteristics defined in a profile referenced in the GRE or DVMRP destination profile.

Data MDT for Multicast VPNs and Dynamic IP Tunnels

The data multicast distribution tree (MDT) application for multicast VPNs can create dynamic point-to-multipoint GRE tunnels.

The data MDT application enables you to solve the problem of IP routers flooding unnecessary multicast information to PE routers that have no interested receivers for a particular VPN multicast group. The multicast data MDT solution requires the creation of a new dynamic IP tunnel by the PE router if the source exceeds a configured rate threshold parameter.

The data MDT application supports a co-located tunnel interface. The base GRE interface and its co-located data MDT interface must be both static or both dynamic.

The data MDT application creates a dynamic IP tunnel using the attributes in a customized destination profile.

When creating the dynamic IP tunnel, the data MDT application assigns its name using the following format:

mvpn-dynamic-number

For the data MDT application, you should configure a customized destination profile. For information about configuring multicast VPNs using GRE tunnels, see *Configuring PIM for IPv4 Multicast* in *JunosE Multicast Routing Configuration Guide*.

Mobile IP and Dynamic IP Tunnels

The Mobile IP application can create dynamic point-to-point GRE and DVMRP tunnels.

The Mobile IP application is a tunneling-based solution that enhances the utility of E Series Broadband Services Routers at the edge of the network between fixed wire and wireless network domains. This tunneling-based solution enables a router on a user's home subnet to intercept and forward IP packets to users while they roam beyond traditional network boundaries.

To achieve mobility, the mobile node takes a secondary IP address that matches the new network and redirects the traffic bound to the primary or home address to the mobile node's new network. In the Mobile IP feature, the two agents that accomplish this task are the home agent and the foreign agent.

The Mobile IP application can create a dynamic IP tunnel using the attributes in a default destination profile or a customized destination profile.

When creating the dynamic IP tunnel, the Mobile IP application assigns its name using the following format:

mobileip-dynamic-number

When the Mobile IP application creates the dynamic IP tunnel, it sets a Don't Fragment bit in the packet and in the outer IP header.

The Mobile IP home agent uses the dynamic IP tunnel for routing loop detection. The home agent examines packets that are intercepted by the home agent and destined for the mobile node. If the packet is already encapsulated, and the inner destination address is the same as the outer destination address, then the system examines the outer source address. If the outer source address is the same as the tunnel destination address or the foreign agent care-of-address (CoA), the system silently discards the packet. In all other cases, the tunnel encapsulation is successful.

For more information about configuring Mobile IP using GRE or DVMRP tunnels, see [“Configuring the Mobile IP Home Agent” on page 305](#).

Combining Dynamic and Static IP Tunnels in the Same Chassis

You can configure both dynamic and static IP tunnels in the same chassis.

A tunnel pair consists of two endpoints; one side encapsulates and the other side decapsulates. You can create a tunnel pair with two statically configured endpoints, two dynamically created endpoints, or with one static and one dynamic endpoint.

When configuring IP tunnels, you must consider that a tunnel is uniquely defined by its tunnel source, tunnel destination, transport virtual router, and mode (GRE or DVMRP). The system does not allow multiple tunnels with the same parameters. For example, when you configure a static tunnel with the same parameters as an existing dynamic IP tunnel, the system does not create the dynamic IP tunnel.

Changing and Removing Existing Dynamic IP Tunnels

You can modify the parameters in a destination profile referenced by existing dynamic IP tunnels. The changes only affect new dynamic IP tunnels that reference the destination profile.

You can relocate a dynamic IP tunnel for the Mobile IP application.

You cannot relocate a dynamic IP tunnel for the data MDT application because it is created using a profile. The system deletes dynamic IP tunnels that are relocated. Connections between a static tunnel endpoint and a dynamic tunnel endpoint can fail if the dynamic tunnel endpoint is deleted.

The client application removes dynamic IP tunnel interfaces when one of the following situations occur:

- The transport virtual router is removed.
- The tunnel interface relocates and the tunnel had an IP interfaced stacked on it.
- The tunnel interface indicates that setup is complete when the system is warm started, but it has no upper IP interface.

Platform Considerations

For information about modules that support IP tunnels on the ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support IP tunnels.

For information about modules that support IP tunnels on the E120 and E320 Broadband Services Routers:

- See *E120 and E320 Module Guide, Table 1, Modules and IOAs* for detailed module specifications.
- See *E120 and E320 Module Guide, Appendix A, IOA Protocol Support* for information about the modules that support IP tunnels.

Module Requirements

The supported modules for creating IP tunnels depends on the type of E Series router that you have.

ERX7xx Models, ERX14xx Models, and the ERX310 Router

To create dynamic IP tunnels on an ERX7xx model, ERX14xx model, or an ERX310 router, you must install a Service line module (SM) or a module that supports the use of shared tunnel-server ports. For information about installing modules in the ERX routers, see the *ERX Hardware Guide*.

SMs provide dedicated tunnel-server ports that are always configured on the module. Unlike other line modules, SMs do not pair with corresponding I/O modules that contain ingress and egress ports. Instead, they receive data from and transmit data to other line modules with access to ingress and egress ports on their own associated I/O modules. However, you must assign interfaces on other line modules or loopback interfaces to act as source endpoints for the tunnel.

You can also create IP tunnels on router modules that support shared tunnel-server ports. You can configure (provision) a shared tunnel-server port to use a portion of the module's bandwidth to provide tunnel services. For a list of the modules that support shared tunnel-server ports, see the *ERX Module Guide*.

To configure IPSec transport mode in the GRE or DVMRP destination profile, you must install an IPSec Service Module (ISM).

For information about configuring tunnel services on dedicated and shared tunnel-server ports, see *Managing Tunnel Service and IPSec Service Interfaces* in *JunosE Physical Layer Configuration Guide*.

All line modules forward traffic to IP tunnels. For information about which line modules accept traffic for IP tunnels, see the *ERX Module Guide*.

E120 Router and E320 Router

To create dynamic IP tunnels on an E120 router or an E320 router, you must install an ES2 4G line module or an ES2 10G ADV line module (LM) with an ES2-S1 Service I/O adapter (IOA), or an IOA that supports the use of shared tunnel-server ports. For

information about installing modules in these routers, see the *E120 and E320 Hardware Guide*.

The combination of an ES2 4G LM or an ES2 10G ADV LM with an ES2-S1 Service IOA provides a dedicated tunnel-server port that is always configured on the IOA. Unlike SMs, the ES2 4G LM and the ES2 10G ADV LM require the ES2-S1 Service IOA to condition it to receive and transmit data to other line modules. The ES2-S1 Service IOA also does not have ingress or egress ports.

You can also create IP tunnels on IOAs that support shared tunnel-server ports. You can configure (provision) a shared tunnel-server port to use a portion of the bandwidth of the IOA to provide tunnel services. For a list of the IOAs that support shared tunnel-server ports, see the *E120 and E320 Module Guide*.

All line modules forward traffic to tunnels. For information about the IOAs that accept traffic for tunnels, see the *E120 and E320 Module Guide*.

Redundancy and Tunnel Distribution

For information about the redundancy and tunnel distribution mechanisms supported for SMs, the ES2-S1 Service IOA, and shared tunnel-server ports, see [“Configuring Dynamic IP Tunnels” on page 251](#).

References

For more information about IP tunnels, see the following documents:

- RFC 1700—Assigned Numbers (October 1994)
- RFC 1701—Generic Routing Encapsulation (October 1994)
- RFC 1702—Generic Routing Encapsulation over IPv4 Networks (October 1994)
- RFC 2003—IP Encapsulation within IP (October 1996)
- RFC 2784—Generic Routing Encapsulation (GRE) (March 2000)

Configuring a Destination Profile for Dynamic IP Tunnels

The tasks in this section describe how to configure a destination profile for dynamic IP tunnels.

Modifying the Default Destination Profile

Default destination profiles for GRE and DVMRP are generated at system startup. The system supports only one default GRE destination profile and one default DVMRP destination profile.

The default destination profile enables the application to automatically create dynamic IP tunnels without user configuration for any virtual router, destination address, or source address.

By default, the data MDT application is disabled in the default destination profiles. The Mobile IP application can use the default destination profile. You can modify the configuration of the default destination profiles.

Modifying the Configuration of the Default Destination Profile

To modify the configuration in the default destination profile:

1. Specify the default destination profile for GRE or DVMRP.
host1(config)#gre destination profile global any-virtual-router

2. Modify the options for the default destination profile.

```
host1(config-dest-profile)#tunnel mtu 5000
host1(config-dest-profile)#tunnel checksum
```



NOTE: You cannot configure a tunnel source, tunnel destination, or virtual router in the default destination profile.

Configuring a Destination Profile for GRE Tunnels

To configure a destination profile for dynamic GRE tunnels:

1. Configure a destination profile for GRE.
host1(config-dest-profile)#gre destination profile kanata1 virtual-router vr1

2. Set the source address for the tunnel.

```
host1(config-dest-profile)#tunnel source 1.1.1.1
```

3. Set the destination address for the tunnel.

```
host1(config-dest-profile)#tunnel destination subnet 10.0.0.0 255.0.0.0
```

4. (Optional) Set the maximum transmission unit (MTU) size for the tunnel.

```
host1(config-dest-profile)#tunnel mtu 10240
```

5. (Optional) Configure an IP profile with parameters that are used to stack an upper IP interface over a dynamic GRE tunnel.

```
host1(config-dest-profile)#profile ip-kanata
```

6. (Optional) Enable error checking across a GRE tunnel.

```
host1(config-dest-profile)#tunnel checksum
```

7. (Optional) Enable sequence number generation for a GRE tunnel.

```
host1(config-dest-profile)#tunnel sequence-datagrams
```

8. (Optional) Enable IPSec transport mode.

```
host1(config-dest-profile)#enable ipsec-transport
```

9. (Optional) Create a multicast VPN tunnel.

```
host1(config-dest-profile)#tunnel mdt profile kanata-mdt
```


Creating a Destination Profile for DVMRP Tunnels

To configure a destination profile for dynamic DVMRP tunnels:

1. Configure a destination profile for DVMRP.
`host1(config-dest-profile)#dvmrp destination profile kanata1 virtual-router vr1`
2. Set the source address for the tunnel.
`host1(config-dest-profile)#tunnel source 1.1.1.1`
3. Set the destination address for the tunnel.
`host1(config-dest-profile)#tunnel destination subnet 10.0.0.0 255.0.0.0`
4. (Optional) Set the maximum transmission unit (MTU) size for the tunnel.
`host1(config-dest-profile)#tunnel mtu 10240`
5. (Optional) Configure an IP profile with parameters that are used to stack an upper IP interface over a dynamic DVMRP tunnel.
`host1(config-dest-profile)#profile ip-kanata`
6. (Optional) Enable IPsec transport mode.
`host1(config-dest-profile)#enable ipsec-transport`
7. (Optional) Create a multicast VPN tunnel.
`host1(config-dest-profile)#tunnel mdt profile kanata-mdt`

dvmrp destination profile

- Use to configure a destination profile for dynamic DVMRP tunnels.
- Use the **any-virtual-router** keyword to create a default destination profile for all virtual routers. There can only be one default destination profile defined in the system.
- Use the **virtual-router** keyword to specify a specific transport virtual router.
- Example
`host1(config)#dvmrp destination profile kanata1`
- Use the **no** version to delete the destination profile.
- See dvmrp destination profile.

enable ipsec-transport

- Use to specify that the router accepts only dynamic IP tunnels protected by an IPsec transport connection.
- This command is supported in the destination profile only when you have installed an ISM on ERX routers.
- Example
`host1(config-dest-profile)#enable ipsec-transport`

- Use the **no** version to disable IPSec transport mode.
- See enable ipsec-transport.

gre destination profile

- Use to configure a destination profile for dynamic GRE tunnels.
- Use the **any-virtual-router** keyword to create a default destination profile for all virtual routers. There can only be one default destination profile defined in the system.
- Use the **virtual-router** keyword to specify a specific transport virtual router.
- Example

```
host1(config)#gre destination profile kanata2
```
- Use the **no** version to delete the destination profile.
- See gre destination profile.

profile

- Use to assign an IP profile with parameters that are used to stack an upper IP interface over a dynamic GRE or DVMRP tunnel to the destination profile.
- Example

```
host1(config-dest-profile)#profile ip-kanata
```
- Use the **no** version to remove the profile assignment from the destination profile.
- See profile.

tunnel checksum

- Use to enable checksum computation across a GRE tunnel.
- Checksum computation is not supported for DVMRP tunnels.
- Selecting this feature causes the E Series router to drop corrupted packets it receives on the tunnel interface.
- Example

```
host1(config-dest-profile)#tunnel checksum
```
- Use the **no** version to disable the checksum option.
- See tunnel checksum.

tunnel destination

- Use to configure the remote end of the tunnel.
- Specify the IP address of an interface on the remote router or the range of destination addresses:
 - Use the **subnet** keyword to configure the IP address for the destination interface and the mask.
 - Use the **range** keyword to configure the first IP address and the last IP address of the destination interface range

- Example 1—Specifies an IP address and mask for the destination interface
`host1(config-dest-profile)#tunnel destination subnet 192.13.7.1 255.0.0.0`
- Example 2—Specifies a range of IP addresses for the destination interface
`host1(config-dest-profile)#tunnel destination range 192.13.7.1 192.13.7.20`
- Use the **no** version to remove the destination of a tunnel.
- See tunnel destination.

tunnel mdt profile

- Use to enable multicast distribution tree operation so the IP tunnel component can create an MDT interface.
- The command defines an IP profile with parameters that are used to stack an upper IP interface over a dynamic GRE or DVMRP tunnel.
- Example
`host1(config-dest-profile)#tunnel mdt profile kanata-mdt`
- Use the **no** version to disable MDT on the interface.
- See tunnel mdt profile.

tunnel sequence-datagrams

- Use to enable GRE sequence numbers.
- Specify GRE sequence numbers at both ends of the GRE tunnel.
- Example
`host1(config-dest-profile)#tunnel sequence-datagrams`
- Use the **no** version to disable sequence numbers.
- See tunnel sequence-datagrams.

tunnel source

- Use to configure the source of the tunnel.
- Specify either the primary IP address or the type and specifier of an interface. Do not specify an unnumbered interface.
- You can configure multiple sources in a GRE destination profile or a DVMRP destination profile.
- Example
`host1(config-dest-profile)#tunnel source 11.11.11.11`
- Use the **no** version to remove the source of a tunnel.
- See tunnel source.

Monitoring Dynamic IP Tunnels

You can monitor dynamic DVMRP and GRE tunnels by using the following commands.

show dvmrp destination profile

- Use to display the configuration of DVMRP destination profiles.
- Field descriptions
 - default dvmrp destination profile—Name of the modified default destination profile on the system
 - dvmrp destination profile—Name of the DVMRP destination profiles configured on the system
 - tunnel checksum—Status of tunnel checksum configuration; enabled or disabled
 - tunnel sequence-datagrams—Status of tunnel sequence datagrams configuration; enabled or disabled
 - tunnel mtu—Value of the tunnel MTU
 - ipsec transport mode—Status of IPSec transport mode configuration; enabled or disabled
 - tunnel mdt—Status of IPSec transport mode configuration; enabled or disabled
 - profile—Name of the profile assigned for upper IP interfaces
 - virtual router—Name of the transport virtual router assigned to the destination profile
 - tunnel destination subnet—Value of the configured destination address subnet
 - tunnel source—Value of the configured source address
- Example 1—Displays all destination profiles configured on the system

```
host1#show dvmrp destination profile
default dvmrp destination profile global
dvmrp destination profile kanata1
dvmrp destination profile kanata2

3 dvmrp destination profiles found
the default destination profile is present
```

- Example 2—Displays a specific destination profile

```
host1#show dvmrp destination profile kanata1
dvmrp destination profile kanata1
tunnel mtu 10240
ipsec transport mode disabled
tunnel mdt disabled
profile disabled
virtual router vr1
tunnel destination subnet 10.0.0.0 255.0.0.0
tunnel source 1.1.1.1
```

```
tunnel source 1.1.1.2
tunnel source 1.1.1.3
```

- See show dvmrp destination profile.

show dvmrp tunnel

- Use to display information about DVMRP tunnels.
- To view detailed information about tunnels, specify the **detail** keyword.
- To view the number of tunnels in a specific state, specify the **state** keyword and the state of the tunnel (disabled, down, enabled, lower-down, not-present, up).
- To view the state of a specific tunnel, specify a tunnel name.
- To view the number of tunnels associated with that IP address, specify an IP address.
- To view the number of tunnels associated with an IP address on the virtual router, specify an IP address with the **virtual-router** keyword and the name of the virtual router.
- Field descriptions
 - DVMRP tunnel—Name and state of the dynamic DVMRP tunnel:
 - Up—Tunnel is operational
 - Down—Tunnel is not operational
 - not-present—Tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
 - Application—Name of the application that created the tunnel
 - Tunnel mtu—Value of the maximum transmission unit for the tunnel
 - Tunnel source address—IP address of the source of the tunnel
 - Tunnel destination address—IP address of the destination of the tunnel
 - Tunnel transport virtual router—Name of the virtual router associated with the tunnel
 - Tunnel mdt—Tunnel MDT state
 - Tunnel checksum option—State of the checksum feature: enabled or disabled
 - Tunnel sequence number option—State of the sequence number feature; enabled or disabled
 - Tunnel up/down trap is enabled—Indicates whether or not the E Series router sends traps to SNMP when the operational state of the tunnels changes
 - Tunnel server location—Location of the tunnel server in *slot/port* format (ERX7xx models, ERX14xx models, and the ERX310 router) or *slot/adaptor/port* format (E120 and E320 routers).
 - Tunnel secured by ipsec transport interface—IPSec interface that secures the tunnel.

- Tunnel administrative state—Configured state of the tunnel: Up or Down
- Statistics—Details of packets received or transmitted by the tunnel
- packets—Number of packets received or transmitted by the tunnel
- octets—Number of octets received or transmitted by the tunnel
- discards—Number of packets not accepted by the tunnel
- Errors—Number of packets with errors received or transmitted by the tunnel
- Data rx—Received data
- Data tx—Transmitted data
- DVMRP tunnels found—Total number of DVMRP tunnels found
- Tunnels were created dynamic—Number of tunnels created dynamically

- Example 1—Displays three dynamic DVMRP tunnels

```
host1:vr11#show dvmrp tunnel
DVMRP tunnel mvpn-dynamic-1 is Up
DVMRP tunnel mvpn-dynamic-2 is Up
DVMRP tunnel mvpn-dynamic-3 is Down

3 DVMRP tunnels found
3 tunnels were created dynamic
```

- Example 2—Displays the detail of a dynamically created DVMRP tunnel for the data MDT application

```
host1:vr11#show dvmrp tunnel detail mvpn-dynamic-1
DVMRP tunnel mvpn-dynamic-1 is Up
tunnel is dynamic
Application is MVPN
Tunnel operational configuration
  Tunnel mtu is '5000'
  Tunnel source address is '1.1.1.1'
  Tunnel destination address is '2.2.2.2'
  Tunnel transport virtual router is vr1
  Tunnel mdt is disabled
  Tunnel up/down trap is enabled
  Tunnel-server location is 4/0
  Tunnel administrative state is Up
```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

```

1 DVMRP tunnel found
1 tunnel was created dynamically
```

- Example 3—Displays the detail of a dynamically created DVMRP tunnel for the Mobile IP application

```
host1:vr12#show dvmrp tunnel detail mobileip-dynamic-1
DVMRP tunnel mobileip-dynamic-1 is Up
tunnel is dynamic
```

```

Application is Mobile-IP
Tunnel operational configuration
Tunnel mtu is '5000'
Tunnel source address is '6.6.6.6'
Tunnel destination address is '3.3.3.3'
Tunnel transport virtual router is vr1
Tunnel mdt is disabled
Tunnel checksum option is disabled
Tunnel sequence number option is disabled
Tunnel key is disabled
Tunnel up/down trap is enabled
Tunnel-server location is 6/0
Tunnel administrative state is Up
Statistics      packets      octets      discards      errors
Data rx         0              0              0              0
Data tx         0              0              0              0

```

- See show dvmrp tunnel.

show dvmrp tunnel summary

- Use to display a summary of information about DVMRP tunnels.
- Field descriptions
 - Administrative status
 - enabled—Tunnel is available for use
 - disabled—Tunnel is not available for use
 - Operational status
 - up—Tunnel is operational
 - down—Tunnel is not operational
 - not-present—Tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
- Example

```

host1#show dvmrp tunnel summary
Administrative status  enabled  disabled
                      1           0
Operational status    up       down    not-present
                      1           0           0

```

- See show dvmrp tunnel.

show gre destination profile

- Use to display the configuration of GRE destination profiles.
- Field descriptions

- default gre destination profile—Name of the modified default destination profile on the system
- gre destination profile—Name of the GRE destination profiles configured on the system
- tunnel checksum—Status of tunnel checksum configuration; enabled or disabled
- tunnel sequence-datagrams—Status of tunnel sequence datagrams configuration; enabled or disabled
- tunnel mtu—Value of the tunnel MTU
- ipsec transport mode—Status of IPSec transport mode configuration; enabled or disabled
- tunnel mdt—Status of IPSec transport mode configuration; enabled or disabled
- profile—Name of the profile assigned for upper IP interfaces
- virtual router—Name of the transport virtual router assigned to the destination profile
- tunnel destination subnet—Value of the configured destination address subnet
- tunnel source—Value of the configured source address

- Example 1—Displays all GRE destination profiles configured on the system

```
host1#show gre destination profile
default gre destination profile global
gre destination profile boston1
gre destination profile boston2

3 gre destination profiles found
the default destination profile is present
```

- Example 2—Displays a specific GRE destination profile used for dynamic IP tunnel creation

```
host1#show gre destination profile boston1
gre destination profile boston1
  tunnel checksum disabled
  tunnel sequence-datagrams disabled
  tunnel mtu 10240
  ipsec transport mode disabled
  tunnel mdt disabled
  profile kanata
  virtual router vr1
  tunnel destination subnet 10.0.0.0 255.0.0.0
  tunnel source 1.1.1.1
  tunnel source 1.1.1.2
  tunnel source 1.1.1.3
```

- Example 3—Displays a specific GRE destination profile used in a MVPN

```
host1#show gre destination profile boston2
gre destination profile boston2
  tunnel checksum disabled
  tunnel sequence-datagrams disabled
```



```

tunnel mtu 10240
ipsec transport mode disabled
tunnel mdt profile kanata-mdt
profile kanata
virtual router vr2
tunnel destination subnet 224.0.0.0 255.0.0.0
tunnel source 1.1.1.1
tunnel source 1.1.1.2
tunnel source 1.1.1.3

```

- See show gre destination profile.

show gre tunnel

- Use to display information about a GRE tunnel or a list of GRE tunnels.
- To view detailed information about tunnels, specify the **detail** keyword.
- To view the number of tunnels in a specific state, specify the **state** keyword and the state of the tunnel (disabled, down, enabled, lower-down, not-present, up).
- To view the state of a specific tunnel, specify a tunnel name.
- To view the number of tunnels associated with an IP address, specify an IP address.
- To view the number of tunnels associated with an IP address on the virtual router, specify an IP address with the **virtual-router** keyword and the name of the virtual router.
- Field descriptions
 - GRE tunnel—Name and state of the dynamic GRE tunnel:
 - Up—Tunnel is operational
 - Down—Tunnel is not operational
 - not-present—Tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
 - Application—Name of the application that created the tunnel
 - Tunnel mtu—Value of the maximum transmission unit for the tunnel
 - Tunnel source address—IP address of the source of the tunnel
 - Tunnel destination address—IP address of the destination of the tunnel
 - Tunnel transport virtual router—Name of the virtual router associated with the tunnel
 - Tunnel mdt—State of the tunnel MDT
 - Tunnel checksum option—State of the checksum feature: enabled or disabled
 - Tunnel sequence number option—State of the sequence number feature; enabled or disabled
 - Tunnel up/down trap—Indicates whether or not the E Series router sends traps to SNMP when the operational state of the tunnels changes, enabled or disabled

- Tunnel server location—Location of the tunnel server in *slot/port* format (ERX7xx models, ERX14xx models, and the ERX310 router) or *slot/adaptor/port* format (E120 and E320 routers).
- Tunnel is secured by ipsec transport interface—IPSec interface that secures the tunnel.
- Tunnel administrative state—Configured state of the tunnel: up or down
- Statistics—Details of packets received or transmitted by the tunnel
- packets—Number of packets received or transmitted by the tunnel
- octets—Number of octets received or transmitted by the tunnel
- discards—Number of packets not accepted by the tunnel
- Errors—Number of packets with errors received or transmitted by the tunnel
- Data rx—Received data
- Data tx—Transmitted data
- Tunnels found—Total number of GRE tunnels found
- Tunnels were created dynamic—Number of tunnels created dynamically
- Example 1—Displays three dynamic GRE tunnels

```
host1:vr11#show gre tunnel
GRE tunnel mobileIp-dynamic-1 is Up
GRE tunnel mvpn-dynamic-2 is Up
GRE tunnel mvpn-dynamic-3 is Down

3 GRE tunnels found
3 tunnels were created dynamic
```

- Example 2—Displays the detail of a dynamically created GRE tunnel for the data MDT application

```
host1:vr11#show dvmrp tunnel detail mvpn-dynamic-1
GRE tunnel mvpn-dynamic-1 is Up
tunnel is dynamic
Application is MVPN
Tunnel operational configuration
Tunnel mtu is '5000'
Tunnel source address is '1.1.1.1'
Tunnel destination address is '2.2.2.2'
Tunnel transport virtual router is vr1
Tunnel mdt is disabled
Tunnel checksum option is disabled
Tunnel sequence number option is disabled
Tunnel up/down trap is enabled
Tunnel-server location is 4/0
Tunnel administrative state is Up
```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

```

1 GRE tunnel found
1 tunnel was created dynamically

```

- Example 3—Displays the detail of a dynamically created GRE tunnel for the Mobile IP application

```

host1:vr12#show gre tunnel detail mobileip-dynamic-1
GRE tunnel mobileip-dynamic-1 is Up
tunnel is dynamic
Application is Mobile-IP
Tunnel operational configuration
  Tunnel mtu is '5000'
  Tunnel source address is '6.6.6.6'
  Tunnel destination address is '3.3.3.3'
  Tunnel transport virtual router is vr1
  Tunnel mdt is disabled
  Tunnel checksum option is disabled
  Tunnel sequence number option is disabled
  Tunnel key is disabled
  Tunnel up/down trap is enabled
  Tunnel-server location is 6/0
  Tunnel administrative state is Up
Statistics      packets      octets      discards    errors
Data rx        0            0           0           0
Data tx        0            0           0           0

```

- See show gre tunnel.

show gre tunnel summary

- Use to display a summary of information about GRE tunnels.
- Field descriptions
 - Administrative status
 - enabled—Tunnel is available for use
 - disabled—Tunnel is not available for use
 - Operational status
 - up—Tunnel is operational
 - down—Tunnel is not operational
 - not-present—Tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
- Example

```

host1#show gre tunnel summary
Administrative status  enabled  disabled
                      3           0
Operational status    up       down    not-present
                      3           0           0

```

- See show gre tunnel.

CHAPTER 11

IP Reassembly for Tunnels

This chapter describes IP packet reassembly for tunneled protocols on E Series routers; it contains the following sections:

- [Overview on page 269](#)
- [Platform Considerations on page 270](#)
- [Configuring IP Reassembly on page 271](#)
- [Monitoring IP Reassembly on page 272](#)

Overview

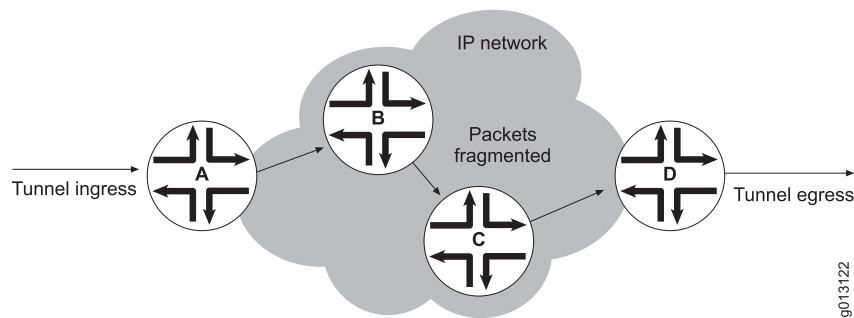
Tunneling protocols provide a method of forwarding packets of a particular protocol through a network of a different protocol type. For example, L2TP can transport a protocol such as PPP through a routed IP network. This capability requires a pair of devices that define the endpoints of the tunnel. Packets entering the tunnel are processed and encapsulated at the ingress endpoint, and packets exiting the tunnel are processed and de-encapsulated at the egress endpoint.

When packets are tunneled through an IP network, simple IP forwarding is performed. The IP forwarding process might fragment packets in the tunnel. Tunnel processing requires each packet to exit the tunnel in the same form in which it entered. Fragmented packets that are not reassembled before the tunnel egress processing are dropped.

For example, in [Figure 21 on page 270](#), traffic is tunneled through an IP network that has four hops. Because the MTU of the link between routers B and C is smaller than that of previous hops, some packets are fragmented. Router D must reassemble the packets before tunnel egress processing and de-encapsulation are performed.

For more information about configuring tunnel-service interfaces, see *Managing Tunnel Service and IPSec Service Interfaces* in *JunosE Physical Layer Configuration Guide*.

Figure 21: Tunneling Through an IP Network That Fragments Packets



Platform Considerations

For information about modules that support IP reassembly on the ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support IP reassembly.

For information about modules that support IP reassembly on the E120 or E320 Broadband Services Router:

- See *E120 and E320 Module Guide, Table 1, Modules and IOAs* for detailed module specifications.
- See *E120 and E320 Module Guide, Appendix A, IOA Protocol Support* for information about the modules that support IP reassembly.

Module Requirements

The types of modules that support IP reassembly for tunnel packets depend on the type of E Series router that you have.

ERX7xx Models, ERX14xx Models, and the ERX310 Router

To configure IP reassembly on ERX7xx models, ERX14xx models, and the ERX310 router, you must install one of a Service Module (SM), an IPSec Service line module (ISM), or a module that supports the use of shared tunnel-server ports. With these modules, an ERX router can perform reassembly of IP packets that it receives on DVMRP, GRE, IPSec, and L2TP tunnels.

Because IP reassembly is required only on tunnel egress packets, the router performs reassembly only on packets in which the IP destination address is local to the router and in which the underlying protocol is one of the supported tunneling protocols.

SMs provide dedicated tunnel-server ports that are always configured on the module. Unlike other line modules, SMs, and ISMs do not pair with corresponding I/O modules that contain ingress and egress ports. Instead, they receive data from and transmit data

to other line modules with access to ingress and egress ports on their own associated I/O modules.

You can also create tunnels on router modules that support shared tunnel-server ports. You can configure (provision) a shared tunnel-server port to use a portion of the module's bandwidth to provide tunnel services. For a list of the modules that support shared tunnel-server ports, see the *ERX Module Guide*.

For information about configuring tunnel services on dedicated and shared tunnel-server ports, see *Managing Tunnel Service and IPSec Service Interfaces* in *JunosE Physical Layer Configuration Guide*.

E120 Router and E320 Router

To configure IP reassembly on an E120 router or an E320 router, you must install an ES2 4G line module (LM) or an ES2 10G ADV LM with an ES2-S1 Service I/O adapter (IOA), or an IOA that supports the use of shared tunnel-server ports. For information about installing modules in these routers, see the *E120 and 320 Hardware Guide*.

The combination of an ES2 4G LM or an ES2 10G ADV LM with an ES2-S1 Service IOA provides a dedicated tunnel-server port that is always configured on the IOA. Unlike SMs, the ES2 4G LM and the ES2 10G ADV LM require the ES2-S1 Service IOA to condition it to receive and transmit data to other line modules. The ES2-S1 Service IOA also does not have ingress or egress ports.

You can also configure IP reassembly on IOAs that support shared tunnel-server ports. You can configure (provision) a shared tunnel-server port to use a portion of the IOA's bandwidth to provide tunnel services. For a list of the IOAs that support shared tunnel-server ports, see the *E120 and E320 Module Guide*.

For information about configuring tunnel services on dedicated and shared tunnel-server ports, see *Managing Tunnel Service and IPSec Service Interfaces* in *JunosE Physical Layer Configuration Guide*.

Configuring IP Reassembly

You can enable IP reassembly on a virtual router basis. Also, on a systemwide basis, you can control how the router handles verification of sequence numbers in data packets that it receives on L2TP tunnels.

ip tunnel reassembly

- Use to enable the reassembly of fragmented IP tunnel packets that are received on the current virtual router.
- You configure tunnel reassembly on VPN routing and forwarding routers independent of the tunnel reassembly configuration on the parent virtual router.
- Example—Enables reassembly for virtual router vr12 and disables reassembly for virtual router vr8

```
host1:vr12(config)#ip tunnel reassembly
host1:vr12(config)#virtual-router vr8
host1:vr8(config)#no ip tunnel reassembly
```

- Use the **no** version to return IP tunnel reassembly to the default, disabled.
- See ip tunnel reassembly.

l2tp ignore-receive-data-sequencing

- Use to prevent sequence number verification for data packets received on all L2TP tunnels in the router. This command does not affect the insertion of sequence numbers in packets sent from the router.
- If you are using IP reassembly, we recommend that you set up the router to ignore sequence numbers in received data packets. Because IP reassembly can reorder L2TP packets, out-of-order packets can be dropped if sequence numbers are being used on L2TP data packets.
- Example

```
host1(config)#l2tp ignore-receive-data-sequencing
```
- Use the **no** version to cause the router to verify sequence numbers on received L2TP data packets.
- See l2tp ignore-receive-data-sequencing.

Monitoring IP Reassembly

This section describes how to set a statistics baseline for tunnel reassembly statistics and how to display reassembly statistics.

Setting Statistics Baselines

You can use the **baseline ip tunnel-reassembly** command to set a statistics baseline for tunnel reassembly statistics on the current virtual router. The router implements the baseline by reading and storing the statistics at the time the baseline is set and then subtracting this baseline whenever you retrieve baseline-relative statistics.

baseline ip tunnel-reassembly

- Use to set a statistics baseline for tunnel reassembly statistics on the current virtual router.
- To display reassembly statistics relative to the baseline, use the **show ip tunnel reassembly statistics** command with the **delta** keyword. For information about displaying baselined statistics, see [“show ip tunnel reassembly statistics” on page 273](#).
- Example

```
host1:vr2#baseline ip tunnel-reassembly
```
- There is no **no** version.
- See baseline ip tunnel-reassembly.

Displaying Statistics

The router keeps several statistics that are useful for diagnostic purposes. These statistics are organized by virtual router, and some are broken out by protocol as well. You can display statistics for a single virtual router or for all virtual routers. You can also display statistics relative to a baseline.

show ip tunnel reassembly statistics

- Use to display tunnel reassembly statistics.
- To display statistics in brief form for the current virtual router, use the command with no keywords.
- To display statistics for all virtual routers, include the **all** keyword.
- To display detailed statistics, include the **detail** keyword.
- To display statistics relative to a baseline set with the **baseline ip tunnel-reassembly** command, include the **delta** keyword.
- Field descriptions
 - Tunnel IP Reassembly—Status of the IP reassembly feature: enabled, disabled
 - Total Fragments Received—Number of total fragments received for all tunneling protocols
 - Total Packets Reassembled—Number of packets reassembled; detailed display includes number of packets reassembled for each protocol; Control/Other increments for packets that are reassembled on a Tunnel Service module but are not forwarded, and instead sent to the SRP module
 - Reassembly Errors or Total Reassembly Errors—Number of errors in completing reassembly; detailed display includes types of reassembly errors
 - Reassembly Discards—Number of packets discarded because they were not reassembled
 - Reassembly Disabled Discards—Number of fragmented packets received when IP tunnel reassembly is disabled on the virtual router
- Example 1—Shows reassembly statistics for the default virtual router

```
host1#show ip tunnel reassembly statistics
```

```
Tunnel IP Reassembly Statistics for Virtual Router: default
```

```
Tunnel IP Reassembly enabled
Total Fragments Received:      15
Total Packets Reassembled:     5
Reassembly Errors:             0
Reassembly Discards:           0
```

- Example 2—Shows detailed reassembly statistics for the default virtual router

```
host1#show ip tunnel reassembly statistics detail
```

```
Tunnel IP Reassembly Statistics for Virtual Router: default
```

```
Tunnel IP Reassembly enabled
Total Fragments Received:      15
Total Packets Reassembled:     5
    L2TP:                      5
    GRE:                       0
    IPSec:                     0
    Control/Other:             0
Total Reassembly Errors:       0
    Fragmentation Errors:      0
    Too Many Fragments:       0
    Out of Resources:          0
    Packet Too Big:            0
    Reassembly Timeout:        0
Reassembly Disabled Discards:  0
```

- Example 3—Shows reassembly statistics for virtual router vr2 before and after setting a statistics baseline

The following command shows reassembly statistics for vr2 before setting the baseline.

```
host1:vr2#show ip tunnel reassembly statistics
```

```
Tunnel IP Reassembly Statistics for Virtual Router: vr2
```

```
Tunnel IP Reassembly enabled
Total Fragments Received:      45
Total Packets Reassembled:     15
Reassembly Errors:             0
Reassembly Discards:           0
```

The following command sets a baseline for reassembly statistics on vr2.

```
host1:vr2#baseline ip tunnel-reassembly
```

The following command shows reassembly statistics relative to the baseline before new packets arrive at the router for reassembly.

```
host1:vr2#show ip tunnel reassembly statistics delta
```

```
Tunnel IP Reassembly Statistics for Virtual Router: vr2
```

```
Tunnel IP Reassembly enabled
Total Fragments Received:      0
Total Packets Reassembled:     0
Reassembly Errors:             0
Reassembly Discards:           0
```

The following command shows reassembly statistics relative to the baseline as new packets start arriving at the router for reassembly.

```
host1:vr2#show ip tunnel reassembly statistics delta
```

```
Tunnel IP Reassembly Statistics for Virtual Router: vr2
```

```
Tunnel IP Reassembly enabled
Total Fragments Received:      15
Total Packets Reassembled:     5
```

Reassembly Errors:	0
Reassembly Discards:	0

- See show ip tunnel reassembly statistics.

CHAPTER 12

Securing L2TP and IP Tunnels with IPSec

This chapter describes how to secure generic routing encapsulation (GRE), Distance Vector Multicast Routing Protocol (DVMRP), and Layer 2 Tunneling Protocol (L2TP) tunnels with IP Security (IPSec) on your E Series router. It contains the following sections:

- [Overview on page 277](#)
- [Platform Considerations on page 278](#)
- [References on page 278](#)
- [L2TP/IPSec Tunnels on page 279](#)
- [GRE/IPSec and DVMRP/IPSec Tunnels on page 290](#)
- [Configuring IPSec Transport Profiles on page 292](#)
- [Monitoring DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec Tunnels on page 296](#)

Overview

You can provide additional security to L2TP and IP tunnels by protecting them with an IPSec transport connection. Secure IP interfaces are virtual IP interfaces that are configured to provide confidentiality and authentication services for the traffic flowing through the interface; that traffic can be L2TP, GRE, and DVMRP tunnel traffic. See [“Configuring IPSec” on page 119](#) for detailed information about IPSec.

GRE, DVMRP, and L2TP over IPSec provide security only between tunnel endpoints; they do not provide end-to-end security. For end-to-end security, you need additional security for the connection beyond the router.

Tunnel Creation

ERX routers can have both unsecured GRE, DVMRP, and L2TP tunnels and tunnels that are secured by IPSec. However, unsecured L2TP tunnels are not allowed on the ISM. You use the following commands to create a secure tunnel:

- L2TP tunnels—Use the **enable ipsec transport** command in the L2TP destination profile
- GRE and DVMRP tunnels—Use the **ipsec-transport** keyword in the **interface tunnel** command

IPSec Secured-Tunnel Maximums

See *JunosE Release Notes*, *Appendix A, System Maximums* corresponding to your software release for information about the maximum number of GRE/IPSec, DVMRP/IPSec, and L2TP/IPSec connections supported on E Series routers.

Platform Considerations

For information about modules that support L2TP and IP tunnels with IPSec on the ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router:

- See LNS and LAC support in *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See LNS and LAC support in *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support LNS and LAC.

For information about modules that support L2TP and IP tunnels with IPSec on the E120 and E320 Broadband Services Routers:

- See LNS and LAC support in *E120 and E320 Module Guide, Table 1, Modules and IOAs* for detailed module specifications.
- See LNS and LAC support in *E120 and E320 Module Guide, Appendix A, IOA Protocol Support* for information about the modules that support LNS and LAC.

Module Requirements

To create IPSec-secured tunnels, you must install an IPSec Service module (ISM) in the ERX router. The ISM is a security gateway and functions as one of the endpoints for secure tunnels. The tunnel endpoints are the tunnel *source* and the tunnel *destination* IP addresses. For an L2TP/IPSec tunnel, the source is the L2TP network server (LNS) and the destination is the L2TP access concentrator (LAC).

For information about installing ISMs in the ERX routers, see the *ERX Hardware Guide*.

References

For more information about the protocols for securing L2TP and IP tunnels with IPSec, consult the following resources:

- RFC 2401—Security Architecture for the Internet Protocol (November 1998)
- RFC 2661—Layer Two Tunneling Protocol “L2TP” (August 1999)
- RFC 3193—Securing L2TP using IPSec (November 2001)
- RFC 3715—IPsec-Network Address Translation (NAT) Compatibility Requirements (March 2004)

- Negotiation of NAT-Traversal in the IKE—draft-ietf-ipsec-nat-t-ike-08.txt (July 2004 expiration)
- UDP Encapsulation of IPsec ESP Packets—draft-ietf-ipsec-udp-encaps-09.txt (November 2004 expiration)



NOTE: IETF drafts are valid for only 6 months from the date of issuance. They must be considered as works in progress. Please refer to the IETF Web site at <http://www.ietf.org> for the latest drafts.

For additional configuration information, see:

- “Configuring IPSec” on page 119
- “Configuring Digital Certificates” on page 205
- “Configuring IP Tunnels” on page 237
- L2TP Overview

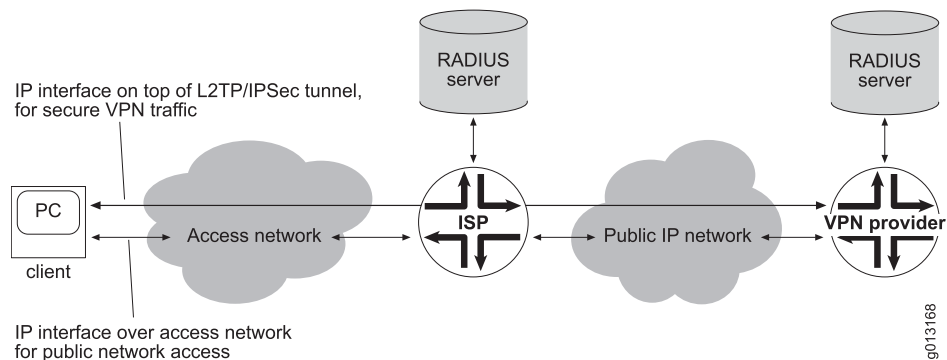
L2TP/IPSec Tunnels

L2TP/IPSec remote access allows clients to connect to a corporate VPN over the public Internet with a secure connection. The L2TP tunnel runs on top of an IPSec transport mode connection. The secure tunnel runs from the client PC to the E Series router that terminates the secure tunnel. For example, using L2TP with IPSec enables B-RAS clients to securely connect to a corporate or other VPN in addition to using another unsecured connection to the Internet, depending on the client software capabilities.

On the router side of the L2TP connection, the E Series router acts as the LNS. On the PC client side of the connection, the client acts as the LAC and runs the L2TP/IPSec client software on supported platforms. (For a list of the supported platforms, see “[Client Software Supported](#)” on page 281.) Both sides of the connection run IPSec in transport mode with Encapsulating Security Payload (ESP) encryption and authentication.

In the model shown in [Figure 22 on page 280](#), a client PC connects to its local provider, who gives the client a public IP address. Using the public IP address, the client PC initiates an IPSec connection toward the L2TP/IPSec gateway for the private network that it wants to connect to. After establishing the IPSec connection, the client establishes an L2TP tunnel to the same L2TP/IPSec gateway, which provides the client with another IP interface to access the private network it is connecting to. The L2TP tunnel is completely protected by the IPSec connection established earlier.

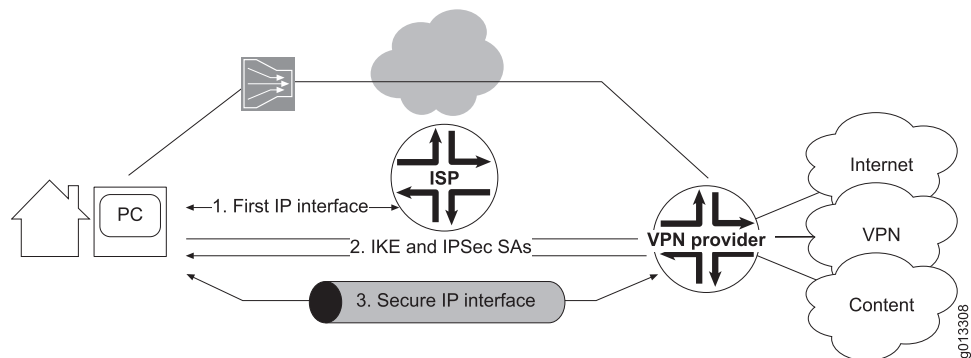
Figure 22: L2TP with IPsec Application



Setting Up the Secure L2TP Connection

Figure 23 on page 280 gives an overview of the process used to set up a secure connection between the client PC and an E Series router that is acting as a VPN provider.

Figure 23: L2TP/IPSec Connection



To set up the secure connection shown in Figure 23 on page 280:

1. Obtain an IP address from your ISP, using a normal B-RAS termination.
2. IKE signals a security association (SA) between the client PC and the E Series router that is acting as a VPN provider.
 - SAs are established to secure data traffic.
 - The IPsec connection secures L2TP traffic.
3. Set up an L2TP tunnel and session between the client PC (the LAC) and the E Series router (the LNS).

The tunnel runs over the SAs that IKE established.

L2TP with IPsec Control and Data Frames

L2TP and IPsec define control and data messages used for L2TP/IPsec. Figure 24 on page 281 shows an L2TP control frame encapsulated by IPsec. The shaded area shows the encrypted portion of the frame.

Figure 24: L2TP Control Frame Encapsulated by IPSec

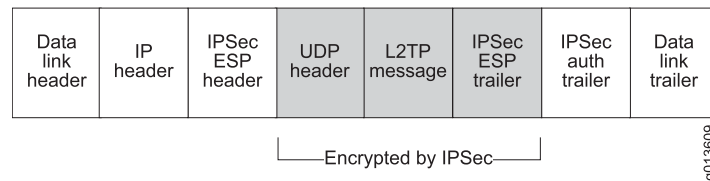
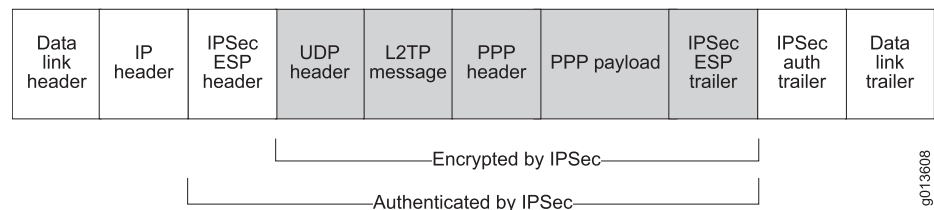


Figure 25 on page 281 is an L2TP data frame encapsulated by IPSec. The shaded area shows the encrypted portion of the frame.

Figure 25: L2TP Data Frame Encapsulated by IPSec



Compatibility and Requirements

This section covers various compatibility issues and requirements for the L2TP/IPSec traffic.

Client Software Supported

The L2TP/IPSec software supports the following client PC operating systems and L2TP and IPSec applications:

- Windows 2000 and Windows XP running built-in IPSec VPN software
- Microsoft L2TP/IPSec VPN client for Windows NT, Windows 98, and Windows Me
- SafeNet client software
- Mac OS X version 10.3 or higher

Interactions with NAT

There are two ways that you can configure E Series routers to interact with Network Address Translation (NAT) devices in the network:

- Configure the router to run in NAT passthrough mode by using the **application l2tp-nat-passthrough** command. For information, see [“NAT Passthrough Mode” on page 282](#).
- Configure the virtual router to enable NAT Traversal (NAT-T) by using the **ipsec option nat-t** command. For information, see [“NAT Traversal” on page 282](#).

Interaction Between IPSec and PPP

PPP defines the Compression Control Protocol (CCP) and the Encryption Control Protocol (ECP) modes. These modes are currently not supported in the E Series router. There is no interaction related to encryption directives between IPSec and PPP.

LNS Change of Port

In the L2TP world, the LNS is allowed to change its port number; this functionality is currently not supported in ERX routers. IPSec allows only port 1701 to be used for L2TP/IPSec tunnels. However, the LAC is allowed to use any source port it desires.

Group Preshared Key

Group preshared keys allow the provisioning of secure remote access by means of L2TP/IPSec to networks that do not use a certificate authority (CA) to issue certificates. A group preshared key is associated with a local IP address in the E Series router and is used to authenticate L2TP/IPSec clients that target this IP address as their VPN server address.



CAUTION: Group preshared keys are not fully secure, and we recommend that you use digital certificates in place of group preshared keys. Group preshared keys are open to man-in-the-middle attacks. To reduce this risk, the ERX routers accept only IPSec connections that specify L2TP traffic selectors for security associations (SAs) that are negotiated over IKE connections authenticated with group preshared keys.

NAT Passthrough Mode

NAT devices can change the IP address and port number of a traversing IP packet. Encrypted frames, in which an ESP header follows the IP header, may or may not get through the NAT device.

You can set up the router to run in NAT passthrough mode, which causes the router to not check UDP checksums. The reason is that a NAT device may change the IP address while the UDP header is encrypted. In this case, the UDP checksum cannot be recalculated. Not checking UDP checksums does not compromise security, because IPSec protects UDP with an authentication algorithm far stronger than UDP checksums. To set up the router to run in NAT passthrough mode, use the **application l2tp-nat-passthrough** command.

We recommend that you configure the router to use NAT passthrough mode when the NAT device provides a feature commonly known as IPSec passthrough.

For information about configuring NAT passthrough mode as part of an IPSec transport profile, see [“Configuring IPSec Transport Profiles” on page 292](#).

NAT Traversal

Using NAT passthrough mode is an adequate solution when a single remote user located behind a NAT device needs secure access to an E Series router. However, NAT passthrough mode does not support secure access to the router by multiple remote users at locations such as hotels or airports where a NAT device resides between the router and the remote users. In addition, NAT passthrough mode does not provide secure access for groups of remote users at corporate locations where a NAT device resides between the company's intranet and the public IP network.

To allow secure router access for multiple remote hosts located behind a NAT device, the router supports a set of IETF standards collectively known as NAT Traversal (NAT-T). For a list of the individual standards that NAT-T comprises, see [“References” on page 278](#).

How NAT-T Works

By default, NAT-T is enabled on every virtual router configured on the system. With NAT-T enabled, IPSec traffic flows transparently through a NAT device, thereby allowing one or more remote hosts located behind the NAT device to use secure L2TP/IPSec tunnel connections to access the router.

After NAT-T is enabled on a specific virtual router, either by default or by using the **ipsec option nat-t** command, the router performs the following actions, in this order:

1. The router monitors the exchange of private vendor ID (VID) payloads between the client PC and the E Series router during the IKE SA negotiation to determine whether both sides of the negotiation support NAT-T.
2. If both sides of the negotiation support NAT-T, the router detects whether a NAT device resides between the IPSec remote peers.
3. If a NAT device is detected between the remote peers, the router negotiates the appropriate type of UDP encapsulation as part of the IKE SA and uses this encapsulation method to process the IPSec traffic.

The **ipsec option nat-t** command affects only those IKE SAs negotiated on the virtual router *after* the command is issued. The command has no effect on IKE SAs that were previously negotiated.

UDP Encapsulation

As part of the IKE SA negotiation process, the router automatically negotiates UDP encapsulation for L2TP/IPSec control and data frames.

When NAT-T is enabled, L2TP/IPSec control frames and data frames are wrapped in an additional NAT-T UDP header that enables data to flow transparently through the NAT device. The NAT device can translate the IP address of the source port associated with the NAT-T UDP header, whereas the IPSec ESP header does not have a source port that the NAT device can translate.

[Figure 26 on page 283](#) shows an L2TP control frame encapsulated with a NAT-T UDP header. The shaded area shows the portion of the frame that is encrypted by IPSec.

Figure 26: L2TP Control Frame with NAT-T UDP Encapsulation

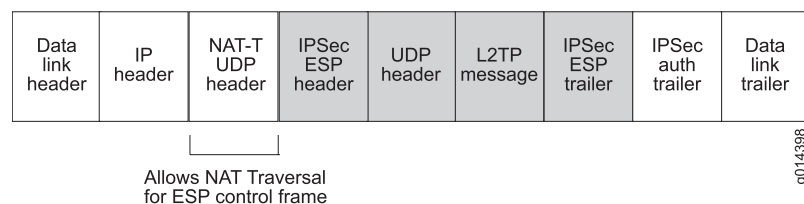
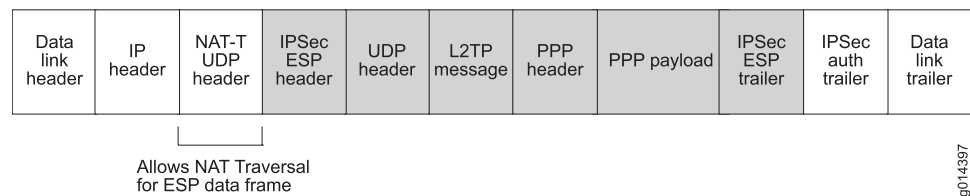


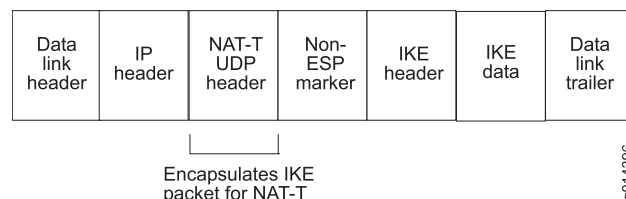
Figure 27 on page 284 shows an L2TP data frame encapsulated with a NAT-T UDP header. The shaded area shows the portion of the frame that is encrypted by IPSec.

Figure 27: L2TP Data Frame with NAT-T UDP Encapsulation



Additionally, IKE packets transmitted during the IKE SA negotiation process are encapsulated with a NAT-T UDP header, and include a non-ESP marker to distinguish them from standard ESP control and data frames. Figure 28 on page 284 shows an IKE packet encapsulated with a NAT-T UDP header.

Figure 28: IKE Packet with NAT-T UDP Encapsulation



Only frames that use the ESP encryption and authentication protocol can be UDP-encapsulated. Frames that use authentication header (AH) cannot be UDP-encapsulated; therefore, NAT-T is *not supported* for L2TP/IPSec connections that use AH.

For more detailed information about encapsulation and other IPSec security parameters, see “Configuring IPSec” on page 119.

UDP Statistics

When NAT-T is enabled, UDP-encapsulated IPSec packets arriving and leaving the router look like standard UDP packets. However, the router does not forward these packets to and from the SRP module, as it does for other UDP packets. As a result, the UDP statistics maintained by the SRP module do not reflect UDP-encapsulated IPSec packets.

NAT Keepalive Messages

The router does not generate NAT keepalive messages. The following reasons explain why this behavior does not generally pose problems for remote users.

- The primary application for using NAT-T is enabling secure L2TP/IPSec access to an E Series router for remote hosts located behind a NAT device. The L2TP protocol has its own keepalive mechanism that is sufficient for keeping NAT entries alive.
- In most NAT configurations, an ERX router does not operate behind the NAT device, thereby making the generation of keepalive messages unnecessary.

If the router receives NAT keepalive messages as part of the L2TP/IPSec traffic flow, it discards these messages at the ingress line module on which the messages were received.

Configuring and Monitoring NAT-T

For instructions on configuring and monitoring NAT-T, see the sections listed in [Table 17 on page 285](#).

Table 17: Configuration and Monitoring Tasks for NAT-T

Task	Command	See Section
Enabling and disabling NAT-T on a virtual router	ipsec option nat-t	"Configuring NAT-T" on page 288
Displaying information about the current NAT-T setting on a virtual router	show ipsec option	"Monitoring DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec Tunnels" on page 296
Displaying information about the IKE SA negotiation when NAT-T is enabled	show ipsec ike-sa	"Monitoring DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec Tunnels" on page 296

Single-Shot Tunnels

You can use the **single-shot-tunnel** command in L2TP Destination Profile Host Configuration mode to configure a single-shot L2TP tunnel. Although configuration of single-shot tunnels is more typically used with secure L2TP/IPSec tunnels, as described in this chapter, you can also configure single-shot tunnels for nonsecure L2TP tunnels that do not run over an IPSec connection.

A *single-shot tunnel* has the following characteristics:

- The L2TP tunnel can carry no more than a single L2TP session for the duration of its existence.
- The router ignores the idle timeout period for single-shot tunnels. This means that as soon a single-shot tunnel's session is removed, the single-shot tunnel proceeds to disconnect.
- The following characteristics apply only to secure L2TP/IPSec single-shot tunnels:
 - The underlying IPSec connection for a single-shot tunnel can carry no more than a single L2TP tunnel for the duration of its existence.
 - The router disconnects the underlying IPSec transport connection for a single-shot tunnel at the beginning of the destruct timeout period instead of waiting until the destruct timeout period expires.

For L2TP/IPSec single-shot tunnels, as soon as the tunnel or its single session fails negotiations or disconnects, the router prevents any further L2TP tunnels or L2TP sessions from connecting, and requires that a new IPSec connection be established for any subsequent connection attempts.

Table 18 on page 286 describes the differences between how the router handles the idle timeout period (configured with the **l2tp tunnel idle-timeout** command) and the destruct timeout period (configured with the **l2tp destruct-timeout** command) for standard L2TP/IPSec tunnels and for single-shot L2TP/IPSec tunnels when the last remaining tunnel session has been disconnected.

Table 18: Differences in Handling Timeout Periods for L2TP/IPSec Tunnels

Timeout Period	Standard L2TP/IPSec Tunnels (Not Single-Shot)	Single-Shot L2TP/IPSec Tunnels
Idle timeout period	<p>The tunnel persists until the idle timeout period expires. If a new L2TP session is created before the idle timeout period expires, the tunnel persists to carry the new session and any subsequent sessions that are established.</p> <p>When the idle timeout period expires, the router disconnects the tunnel.</p>	<p>The router ignores the idle timeout period.</p> <p>This behavior prevents a single-shot tunnel from passing traffic after its single L2TP session is disconnected.</p>
Destruct timeout period	<p>The router signals the underlying IPSec transport connection to disconnect when the destruct timeout period expires.</p>	<p>The router signals the underlying IPSec transport connection to disconnect at the beginning of the destruct timeout period.</p>

For information about configuring L2TP/IPSec single-shot tunnels on the router, see [“Configuring Single-Shot Tunnels” on page 289](#).

Configuration Tasks for Client PC

To set up client PCs, you need to:

1. Create an IPSec security policy to secure L2TP traffic to the E Series router.
2. Get a certificate for the client or set up preshared keys.
3. Create a VPN connection to the router.
4. Log the client in to the E Series router.

Configuration Tasks for E Series Routers

The main configuration tasks for setting up L2TP/IPSec are:

1. Set up IP connectivity to L2TP clients; for example, PPPoE, DHCP, or static IP.
2. Set up digital certificates on the router, or configure preshared keys for IKE authentication.
 - To set up digital certificates, see [“Configuring Digital Certificates” on page 205](#).
 - To set up preshared keys, see [“Configuring IPSec Parameters” on page 139](#) in [“Configuring IPSec” on page 119](#).

3. Create IPSec policies. See [“Defining an IKE Policy” on page 149](#) in [“Configuring IPSec” on page 119](#).
4. Configure RADIUS authentication and accounting. See *JunosE Broadband Access Configuration Guide*.
5. Configure L2TP destination profiles. See the next section, [“Enabling IPSec Support for L2TP” on page 287](#).
6. Configure NAT-T on the virtual router. See [“Configuring NAT-T” on page 288](#).
7. Configure single-shot L2TP/IPSec tunnels. See [“Configuring Single-Shot Tunnels” on page 289](#).
8. Configure IPSec transport profiles. See [“Configuring IPSec Transport Profiles” on page 292](#).

Enabling IPSec Support for L2TP

To configure an L2TP destination profile:

1. Create a destination profile that defines the location of the LAC, and access L2TP Destination Profile Configuration mode.


```
host1(config)#l2tp destination profile boston4 ip address 0.0.0.0
host1(config-l2tp-dest-profile)#
```
2. Define the L2TP host profile, and enter L2TP Destination Profile Host Configuration mode.


```
host1(config-l2tp-dest-profile)#remote host default
host1(config-l2tp-dest-profile-host)#
```
3. Specify that for L2TP tunnels associated with this destination profile, the router accept only tunnels protected by IPSec.


```
host1(config-l2tp-dest-profile-host)#enable ipsec-transport
```
4. (Optional) Assign a profile name for a remote host.


```
host1(config-l2tp-dest-profile-host)#profile georgeProfile1
```
5. Specify the local IP address to be used in any packets sent to the LAC.


```
host1(config-l2tp-dest-profile-host)#local ip address 10.0.0.1
```

For information about other L2TP destination profile commands, see LNS Configuration Prerequisites in *JunosE Broadband Access Configuration Guide*.

enable ipsec-transport

- Use to specify that the router accept only L2TP tunnels protected by an IPSec transport connection.
- Example


```
host1(config-l2tp-dest-profile-host)#enable ipsec-transport
```

- Use the **no** version to disable IPSec transport mode.
- See enable ipsec-transport.

l2tp destination profile

- Use to create the destination profile that defines the location of the LAC and to access L2TP Destination Profile Configuration mode.
- If no virtual router is specified, the current virtual router context is used.
- If the destination address is 0.0.0.0, then any LAC that can be reached via the specified virtual router is allowed to access the LNS. If the destination address is nonzero, then it must be a host-specific IP address.
- The router supports up to 4,000 L2TP destination profiles.
- Example

```
host1:boston(config)#l2tp destination profile boston ip address 10.10.76.12
host1:boston(config-l2tp-dest-profile)#
```

- Use the **no** version to remove the L2TP destination profile and all of its host profiles.



NOTE: If you remove a destination profile, all tunnels and sessions using that profile will be dropped.

- See l2tp destination profile.

Configuring NAT-T

To configure NAT-T on the current virtual router:

1. Select the name of the virtual router you want to configure.

```
host1(config)#virtual-router westford
host1:westford(config)#
```

2. Enable NAT-T for the current virtual router.

```
host1:westford(config)#ipsec option nat-t
```

ipsec option nat-t

- Use to enable NAT-T for the current virtual router.
- With NAT-T enabled, IPSec traffic flows transparently through a NAT device, thereby allowing one or more remote hosts located behind the NAT device to use secure L2TP/IPSec tunnel connections to access the router.
- The **ipsec option nat-t** command affects only those IKE SAs negotiated on this virtual router after the command is issued; it has no effect on previously negotiated IKE SAs.
- Example

```
host1:sunnyvale(config)#ipsec option nat-t
```


- Use the **no** version to disable NAT-T for the current virtual router.
- Use the **default** version to restore the default NAT-T setting on the virtual router, enabled.
- See `ipsec option nat-t`.

Configuring Single-Shot Tunnels

To configure a single-shot L2TP/IPSec tunnel:

1. Create an L2TP destination profile, which defines the location of the LAC. The **l2tp destination profile** command accesses L2TP Destination Profile Configuration mode.

```
host1(config)#l2tp destination profile boston4 ip address 0.0.0.0
host1(config-l2tp-dest-profile)#
```

2. Create an L2TP host profile, which defines the attributes that the router, acting as the LNS, uses when communicating with the LAC. The **remote host** command accesses L2TP Destination Profile Host Configuration mode.

```
host1(config-l2tp-dest-profile)#remote host default
host1(config-l2tp-dest-profile-host)#
```

3. Specify that, for L2TP tunnels associated with this host profile, the router accept only tunnels protected by IPSec.

```
host1(config-l2tp-dest-profile-host)#enable ipsec-transport
```

4. Specify that the L2TP tunnels associated with this host profile are single-shot tunnels.

```
host1(config-l2tp-dest-profile-host)#single-shot-tunnel
```

5. (Optional) Configure other attributes for the L2TP host profile.
6. (Optional) Use the **show l2tp destination profile** command to verify configuration of the single-shot tunnel for a particular L2TP host profile.

For information about how to use this command, see [“show l2tp destination profile” on page 302](#).

For information about the other commands you can use to configure L2TP destination profiles and L2TP host profiles, see *LNS Configuration Prerequisites in JunosE Broadband Access Configuration Guide*.

single-shot-tunnel

- Use to configure the L2TP/IPSec tunnels associated with a particular L2TP host profile as single-shot tunnels.
- A single-shot tunnel can carry no more than a single L2TP session for the duration of its existence.
- The router ignores the idle timeout period for single-shot tunnels.
- The following characteristics apply only to secure L2TP/IPSec single-shot tunnels:
 - The underlying IPSec connection for a single-shot tunnel can carry no more than a single L2TP tunnel for the duration of its existence.

- The router disconnects the underlying IPSec transport connection for a single-shot tunnel at the beginning of the destruct timeout period instead of waiting until the destruct timeout period expires.
- A single-shot tunnel does not persist beyond its last connected L2TP session. As a result, using single-shot L2TP/IPSec tunnels instead of the default (standard) tunnel behavior provides better protection against a brute force attack that makes multiple, simultaneous authentication attempts.
- Example


```
host1(config-l2tp-dest-profile-host)#single-shot-tunnel
```
- Use the **no** version to restore the default behavior for L2TP/IPSec tunnels, which disables the single-shot attribute.
- See single-shot-tunnel.

GRE/IPSec and DVMRP/IPSec Tunnels

In GRE/IPSec or DVMRP/IPSec connections, E Series routers can act as source and destination endpoints of the secure tunnel. Both sides of the connection run IPSec in transport mode with Encapsulating Security Payload (ESP) encryption and authentication.

In a GRE/IPSec or DVMRP/IPSec connection, the E Series router initiates an IPSec connection with a remote router. After establishing the IPSec connection, the E Series router establishes a GRE or DVMRP tunnel to the remote router. The tunnel is completely protected by the IPSec connection.

Setting Up the Secure GRE or DVMRP Connection

In [Figure 29 on page 290](#), a secure GRE/IPSec connection is set up between two E Series routers. To set up the secure connection:

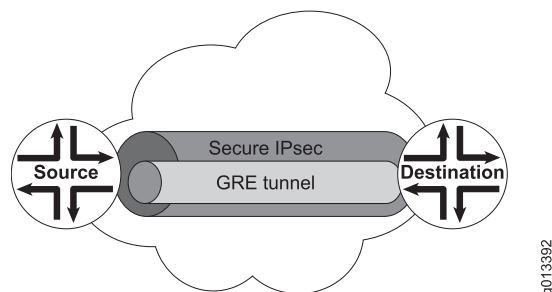
1. Set up the IPSec connection between the two routers. IKE signals a security association (SA) between the two IPSec tunnel endpoints.

Two unidirectional SAs are established to secure data traffic.

2. Set up a GRE tunnel between the two routers.

The GRE tunnel now runs over the SAs that IKE established.

Figure 29: GRE/IPSec Connection



Configuration Tasks

The main configuration tasks for setting up GRE or DVMRP over IPSec on E Series routers are:

- Set up the GRE or DVMRP tunnel, specifying the virtual router and destination address, and enabling IPSec support. See [“Configuring IP Tunnels” on page 237](#).
- Set up digital certificates on the router, or configure preshared keys for IKE authentication.
 - To set up digital certificates, see [“Configuring Digital Certificates” on page 205](#).
 - To set up preshared keys, see [“Configuring IPSec Parameters” on page 139](#) in [“Configuring IPSec” on page 119](#).
- Create IPSec policies. See [“Defining an IKE Policy” on page 149](#) in [“Configuring IPSec” on page 119](#).
- Configure IPSec transport profiles. See [“Configuring IPSec Transport Profiles” on page 292](#).

Enabling IPSec Support for GRE and DVMRP Tunnels

To create GRE/IPSec and DVMRP/IPSec tunnels, use the **ipsec-transport** keyword with the **interface tunnel** command.

interface tunnel dvmrp

interface tunnel gre

- Use with the **ipsec-transport** keyword to create a GRE or DVMRP tunnel that is protected with IPSec in transport mode.



NOTE: After you create a clear GRE or DVMRP tunnel, you cannot convert it to an IPSec-secured tunnel, or vice versa. You must delete the tunnel configuration, then reconfigure the tunnel as the new type.

- You can establish the tunnel on a virtual router other than the current virtual router.
- Example


```
host1(config)#interface tunnel gre:denver-tunnel-5 transport-virtual-router denver
ipsec-transport
host1(config-if)#
```
- Use the **no version** to remove the tunnel.
- See **interface tunnel**.

Configuring IPSec Transport Profiles

To configure an IPSec transport profile that will be used to secure DVMRP, GRE, or L2TP tunnels:

1. Create the profile.

```
host1(config)#ipsec transport profile secureGre virtual-router default ip address
5.5.5.5
host1(config-ipsec-transport-profile)#
```

2. Specify one or more types of application that the profile secures.

```
host1(config-ipsec-transport-profile)#application gre dvmrp l2tp
```

You can then set any of the following parameters for the profile:

- Set a lifetime range for the IPSec connection in volume of traffic or seconds.

```
host1(config-ipsec-transport-profile)#lifetime seconds 3600 28800 kilobytes 102400
4294967295
```

- Configure Perfect Forward Secrecy (PFS) for connections created with this IPSec transport profile.

```
host1(config-ipsec-transport-profile)#pfs group 5
```

- Specify one or more transform sets that an IPSec transport connection uses to negotiate a transform algorithm.

```
host1(config-ipsec-transport-profile)#transform-set esp-3des-hmac-sha
esp-3des-hmac-md5
```

To display the available transform sets, issue the **transform-set ?** command.

- Specify the local endpoint (for L2TP, the LNS address) of the IPSec transport connection, and enter Local IPSec Transport Profile mode.

```
host1(config-ipsec-transport-profile)#local ip address 10.10.1.1
host1(config-ipsec-transport-profile-local)#
```

- (Optional) Configure a key for IKE negotiations. For example:

Enter the unencrypted key. The router encrypts the key and stores it in encrypted form. You can no longer retrieve the unencrypted key.

```
host1(config-ipsec-transport-profile-local)#pre-share secretforGre
```

application

- Use to specify the types of application secured by connections created with this IPSec transport profile. You can specify multiple applications on the same command line:
 - **dvmrp**—Secures DVMRP tunnel traffic
 - **gre**—Secures GRE tunnel traffic

- **l2tp**—Secures L2TP traffic
- **l2tp-nat-passthrough**—Secures L2TP traffic and also allows clients to connect from behind NAT devices that support IPSec passthrough. To allow these clients to connect, the router:
 - Does not generate or verify UDP checksums. This does not compromise security, because IPSec protects UDP packets with an authentication algorithm far stronger than UDP checksums.
 - Provides IPSec filtering based on the received IP address (the NAT public IP address), rather than filtering based on the negotiated IKE identities.
- Example


```
host1(config-ipsec-transport-profile)#application gre dvmrp l2tp
```
- Use the **no** version to return to the default application type, L2TP.
- See application.

ipsec transport profile

- Use to create an IPSec transport profile and to enter IPSec Transport Profile Configuration mode. To create a new profile, you must include the following keywords:
 - **virtual-router**—Name of the virtual router on which you want to create the profile
 - **ip address**—Remote endpoint for the IPSec transport connection.

For L2TP/IPSec connections, you can enter a fixed IP address or the wildcard address, 0.0.0.0. If you use the wildcard address, the profile accepts any remote client connection, which is a typical scenario for secure remote access.

For GRE/IPSec and DVMRP/IPSec connections, you must enter a fixed address; the 0.0.0.0 wildcard address is not accepted and will return an error.
- Example


```
host1(config)#ipsec transport profile secureL2tp virtual-router default ip address
5.5.5.5
host1(config-ipsec-transport-profile)#
```
- Use the **no** version to delete the profile.
- See ipsec transport profile.

lifetime

- Use to set a lifetime range for the IPSec connection in volume of traffic or in seconds or both.
- If the PC client offers a lifetime within this range, the router accepts the offer. If the PC client offers a lifetime outside this range, the router rejects the connection.
- Example


```
host1(config-ipsec-transport-profile)#lifetime seconds 900 86400 kilobytes 100000
4294967295
```

- Use the **no** version to restore the default values, 100000–4294967295 KB and 900–86400 seconds (0.25–24 hours).
- See lifetime.

local ip address

- Use to specify the local endpoint (for L2TP, the LNS address) of the IPsec transport connection and to enter Local IPsec Transport Profile Configuration mode.
- You can enter this command multiple times in an IPsec transport profile.
- You can enter a fixed IP address or the wildcard address, 0.0.0.0. The wildcard address has a lower precedence than a fixed IP address.



CAUTION: We recommend that you do not use address 0.0.0.0, because it allows any address to accept IKE calls, and it creates a group preshared key, which is not fully secure.

- Example

```
host1(config-ipsec-transport-profile)#local ip address 192.168.1.2
host1(config-ipsec-transport-profile-local)#
```
- Use the **no** version to delete the IP address.
- See local ip address.

pfs group

- Use to configure perfect forward secrecy for connections created with this IPsec transport profile.
- Assign a Diffie-Hellman prime modulus group using one of the following keywords:
 - 1—768-bit group
 - 2—1024-bit group
 - 5—1536-bit group
- Example

```
host1(config-ipsec-transport-profile)#pfs group 5
```
- Use the **no** version to remove PFS from this profile, which is the default setting.
- See pfs group.

pre-share

- Use to configure an unencrypted (red) preshared key to authenticate IKE negotiations that arrive from any remote IP address specified for this transport profile and that are destined for the local IP address. If the remote endpoint address is a wildcard address, this preshared key is a group preshared key.



CAUTION: Group preshared keys are not fully secure, and we do not recommend using them. They are provided for trials and testing purposes where the missed security does not pose a risk to the provider.

- To have preshared key authentication take place, you must also specify the IKE policy rule as preshared by entering **authentication pre-share** in ISAKMP Policy Configuration mode.

- Example

```
host1(config-ipsec-transport-profile-local)#pre-share secretforL2tp
```

- Use the **no** version to remove the key.



NOTE: After you enter a preshared key, the original (unencrypted) key cannot be retrieved. If you need to reenter the original key (for example, the system goes to factory default and you have only the **show config** output) you can:

1. Use the **show config** command to see the encrypted (masked) form of the key.
2. Use the **pre-shared-masked** command to enter the masked key. The system will behave the same as when you entered the first **pre-share** key command.

- See pre-share.

pre-share-masked

- Use to specify an encrypted preshared key. To obtain this key, you enter an unencrypted key using the **pre-share** command. You then run the **show config** command, and the router displays the preshared key in encrypted form. You enter the encrypted key using the **pre-share-masked** command.
- The router uses the preshared key to authenticate IKE negotiations that arrive from any remote IP address specified for this transport profile and that are destined for any local IP address specified for this transport profile. If the remote endpoint address is a wildcard address, this preshared key is a group preshared key.



CAUTION: Group preshared keys are not fully secure, and we do not recommend using them. They are provided for trials and testing purposes, where the missed security does not pose a risk to the provider.

- To have preshared key authentication take place, you must also specify the IKE policy rule as preshared by entering **authentication pre-share** in ISAKMP Policy Configuration mode.
- Example

```
host1(config-ipsec-transport-profile-local)#pre-share-masked
AAAAGAAAAAcAAAACZquq4ABieTUBuNBELSY8b/L3CX/RcPX7
```
- There is no **no** version. To remove a key, use the **no pre-share** command.
- See pre-share-masked.

transform-set

- Use to specify the transform set(s) that an IPSec transport connection can use to negotiate a transform algorithm. Each transform in the set provides a different combination of data authentication and confidentiality.
- To display the available transform sets, issue the **transform-set ?** command.
- Example

```
host1(config-ipsec-transport-profile)#transform-set esp-3des-hmac-sha
```
- Use the **no** version to reset the transform to the default, esp-3des-hmac-sha.
- See transform-set.

Monitoring DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec Tunnels

This section contains information about troubleshooting and monitoring DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec tunnels.

System Event Logs

To troubleshoot and monitor DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec tunnels, use the following system event log:

- itm—IPSec transport mode

For more information about using event logs, see the *JunosE System Event Logging Reference Guide*.

show Commands

To display profile and connection information for DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec tunnels, use the following **show** commands.

show dvmrp tunnel

show gre tunnel

- Use to display information about DVMRP or GRE tunnels.
- If the tunnel is protected by IPSec, the **show dvmrp tunnel detail** and **show gre tunnel detail** commands include a line indicating the IPSec transport interface. The line is not

shown for unsecured tunnels. The following is a partial display. See [“Monitoring IP Tunnels” on page 244](#) in [“Configuring IP Tunnels” on page 237](#) for full descriptions of the commands.

- Example

```
host1#show gre tunnel detail
Tunnel operational configuration
Tunnel name is 'vr1'
Tunnel mtu is '10240'
Tunnel source address is '10.0.0.2'
Tunnel destination address is '10.0.0.1'
Tunnel transport virtual router is vr1
Tunnel checksum option is disabled
Tunnel up/down trap is enabled
Tunnel server location is 4/0
Tunnel secured by ipsec transport interface 1
Tunnel administrative state is up
. . .
```

- See `show dvmrp tunnel`.
- See `show gre tunnel`.

`show ipsec ike-sa`

`show ike sa`



NOTE: The `show ipsec ike-sa` command replaces the `show ike sa` command, which may be removed completely in a future release.

- Use to display IKE phase 1 SAs running on the router.
- When NAT-T is enabled on both the client PC and the E Series router, and the router has negotiated NAT-T as part of the IKE SA, the local UDP port number displayed in the Local:Port column is typically 4500. When NAT-T is disabled or not supported on one or both sides of the IKE SA negotiation, the local UDP port number is 500. (See the example under Field Descriptions for more information.)
- Field descriptions
 - Local:Port—Local IP address and UDP port number of phase 1 negotiation
 - Remote:Port—Remote IP address and UDP port number of phase 1 negotiation
 - Time(Sec)—Time remaining in phase 1 lifetime, in seconds
 - State—Current state of the phase 1 negotiation. Corresponds to the messaging state in the main mode and aggressive mode negotiations. Possible states are:
 - AM_SA_I—Initiator has sent initial aggressive mode SA payload and key exchange to the responder
 - AM_SA_R—Responder has sent aggressive mode SA payload and key exchange to the initiator

- AM_FINAL_I—Initiator has finished aggressive mode negotiation
- AM_DONE_R—Responder has finished aggressive mode negotiation
- MM_SA_I—Initiator has sent initial main mode SA payload to the responder
- MM_SA_R—Responder has sent a response to the initial main mode SA
- MM_KE_I—Initiator has sent initial main mode key exchange to the responder
- MM_KE_R—Responder has sent a response to the key exchange
- MM_FINAL_I—Initiator has sent the final packet in the main mode negotiation
- MM_FINAL_R—Responder has finished main mode negotiation
- MM_DONE_I—Initiator has finished main mode negotiation
- DONE—Phase 1 SA negotiation is complete, as evidenced by receipt of some phase 2 messages
- Local Cookie—Unique identifier (SPI) for the local phase 1 IKE SA
- Remote Cookie—Unique identifier (SPI) for the remote phase 1 IKE SA
- Example

The following example displays the IKE phase 1 SAs for three remote client PCs that are accessing an E Series router (IP address 21.227.9.8).

The first client PC listed (IP address 21.227.9.10) is *not* located behind a NAT device, and is therefore not using NAT-T to access the router. This PC appears in the Remote:Port column with its own IP address (21.227.9.10) and UDP port number 500.

The remaining two client PCs are located behind a NAT device that has IP address 21.227.9.11, and are using NAT-T to access the router. These PCs appear in the Remote:Port column with the same IP address (21.227.9.11) but with two different UDP port numbers, 4500 and 14500.

```
host1# show ipsec ike-sa
```

```
IKE Phase 1 SA's:
```

Local:Port	Remote:Port	Time(Sec)	State	Local Cookie	Remote Cookie
21.227.9.8:500	21.227.9.10:500	26133	DONE	0x87a943562124c711	0xaf2cf4a260399a4
21.227.9.8:4500	21.227.9.11:4500	28774	DONE	0x01f9efa234d45ad8	0xada4cb7cafee9243
21.227.9.8:4500	21.227.9.11:14500	28729	DONE	0x0c5ccb6b94b00051	0xe975c0ae3b9ca8bf

- See show ipsec ike-sa.
- See show ike sa.

show ipsec option

- Use to display whether NAT-T is enabled or disabled on the current virtual router.
- The **show ipsec option** command also displays the status of dead peer detection (DPD) on the virtual router. For information about configuring and monitoring DPD, see [“Configuring IPSec” on page 119](#).

- Example

```
host1:westford#show ipsec option
```

```
IPsec options:
Dead Peer Detection: disabled
NAT Traversal      : enabled
```

- See show ipsec option.

show ipsec transport interface

- Use to display information about transport connections.
- Field descriptions
 - IPSec transport interface—Number and status of the IPSec transport connection
 - Configuration
 - Virtual router—Virtual router on which this profile is configured
 - Application—Type of application the connection can protect
 - pfs group—PFS group being used for the connection
 - Mtu—Tunnel's MTU size
 - Local address—Local endpoint address
 - Remote address—Remote endpoint address
 - Local identity—Shows the subnet, protocol, and port
 - Remote identity—Shows the subnet, protocol, and port
 - Inbound spi—Inbound security parameter index
 - Inbound transform—Inbound algorithm
 - Inbound lifetime—Inbound configured lifetime in seconds and kilobytes
 - Outbound spi—Outbound security parameter index
 - Outbound transform—Outbound algorithm
 - Outbound lifetime—Outbound configured lifetime in seconds and kilobytes
 - Statistics
 - InUserPackets—Number of user packets received
 - InUserOctets—Number of octets received from user packets
 - InAccPackets—Number of encapsulated packets received
 - InAccOctets—Number of octets received in encapsulated packets
 - InAuthErrors—Number of authentication errors received

- InReplyErrors—Number of reply errors in received traffic
- InPolicyErrors—Number of policy errors in received traffic
- InOtherRxErros—Number of packets received that have errors other than those listed above
- InDecryptErrors—Number of decryption errors in received traffic
- InPadErrors—Number of packets received that had invalid values after the packet was decrypted
- OutUserPackets—Number of user packets sent
- OutUserOctets—Number of octets sent in user packets
- OutAccPackets—Number of encapsulated packets sent
- OutAccOctets—Number of octets sent in encapsulated packets
- OutPolicyErrors—Number of packets arriving at the transport connection for encapsulation that do not meet the specified identifier (selector)
- OutOtherTxErrors—Number of outbound packets that have errors other than those listed above

- Example 1

```
host1:vr11#show ipsec transport interface
IPSEC transport interface 5 is Up
IPSEC transport interface 6 is Up
2 Isec transport interfaces found
```

- Example 2

```
host1:vr11#show ipsec transport interface 5
IPSEC transport interface 5 is Up
```

- Example 3

```
host1:vr11#show ipsec transport interface detail 5
IPSEC transport interface 5 is Up
Configuration
  Virtual router vr00
  Application gre
  No pfs group
  Mtu is 1440
  Local address is 10.255.0.61
  Remote address is 10.255.0.62
  Local identity is subnet 10.255.0.61 255.255.255.255, proto 47, port
0
  Remote identity is subnet 10.255.0.62 255.255.255.255, proto 47, port
0
  Inbound spi 0x15c30204
  Inbound transform transport-esp-3des-sha1
  Inbound lifetime 900 seconds 102400 kilobytes
  Outbound spi is 0x16a10205
```

```

Outbound transform transport-esp-3des-sha1
Outbound lifetime 900 seconds 102400 kilobytes

Statistics
  InUserPackets          5
  InUserOctets           270
  InAccPackets           5
  InAccOctets            440
  InAuthErrors           0
  InReplayErrors         0
  InPolicyErrors         0
  InOtherRxErrors        0
  InDecryptErrors        0
  InPadErrors            0

  OutUserPackets         5
  OutUserOctets           270
  OutAccPackets           5
  OutAccOctets            440
  OutPolicyErrors        0
  OutOtherTxErrors       0

```

- See `show ipsec transport interface`.

show ipsec transport interface summary

- Use to display a summary of existing IPSec transport connections by application and state.
- Field descriptions
 - up—Number of IPSec transport interfaces that are currently up
 - down—Number of IPSec transport interfaces that are currently down
 - upper-bound—Number of IPSec transport interfaces that are currently bound to the upper layer
- Example

```

host1:vr11#show ipsec transport interface summary
Operational status      up      down      upper-bound
                        2        0         2

```

- See `show ipsec transport interface`.

show ipsec transport profile

- Use to display the configuration of an IPSec transport profile.
- Field descriptions
 - IPSec transport profile—Name of the profile
 - Virtual router—Virtual router on which this profile is configured
 - Peer address—Remote endpoint address
 - Application—Type(s) of application that this profile is protecting
 - Lifetime range in seconds—Lifetime range in seconds configured for the profile

- Lifetime range in kilobytes—Lifetime range in kilobytes configured for the profile
- TransformSet—Transform set(s) configured for the profile
- Pfs group—PFS group configured for the profile; 0 (zero) means that PFS is not configured for the profile
- Local ip address—Local endpoint address

- Example 1

```
host1:vr11#show ipsec transport profile
IPSEC transport profile goi1
IPSEC transport profile goi2
2 Isec transport profiles found
```

- Example 2

```
host1:vr11#show ipsec transport profile goi1
IPSEC transport profile goi1
Virtual router vr00
Peer address 10.255.0.62
Application gre,dvmrp
Lifetime range in seconds 900 900
Lifetime range in kilobytes 102400 4294967294
TransformSet transport-esp-3des-sha1
Pfs group 0
Local ip address : 10.255.0.61
```

- See show ipsec transport profile.

show l2tp destination profile

- Use to display configuration information for an L2TP destination profile and its associated L2TP host profiles.
- If single-shot tunnels are configured for a particular host profile, the command displays this information as an attribute of the profile for that remote host.
- Field descriptions
 - Destination profile attributes:
 - Transport—Method used to transfer traffic
 - Virtual router—Name of the virtual router
 - Peer address—IP address of the LAC
 - Destination profile maximum sessions—Maximum number of sessions allowed for the destination profile
 - Destination profile current session count—Number of current sessions for the destination profile
- Host profile attributes:

- Remote host is—Name of the remote host
 - Tunnel password is—Password for the tunnel
 - Interface profile is—Name of the host profile
 - Local host name is—Name of the local host
 - Isec transport is—Status of the IPSec transport connection: enabled or disabled
 - Disconnect-cause avp is—Status of the disconnect cause AVP generation: enabled or disabled
 - Tunnels are single-shot—Indicates that single-shot tunnels are configured for this host profile
 - Current session count is—Number of current sessions for the host profile
- Example


```

host1#show l2tp destination profile westford
L2TP destination profile westford
Configuration
  Destination address
    Transport ipUdp
    Virtual router default
    Peer address 172.31.1.99
Statistics
  Destination profile current session count is 1
Host profile attributes
  Remote host is lac-1
  Configuration
    Tunnel password is password
    Interface profile is tunneled-user
    Local host name is lns-1
    Isec transport is enabled
    Disconnect-cause avp is enabled
    Tunnels are single-shot
  Statistics
    Current session count is 1
1 L2TP host profile found
      
```
 - See show l2tp destination profile.

Configuring the Mobile IP Home Agent

This chapter describes how to configure the Mobile IP home agent on E Series routers.

- [Mobile IP Overview on page 305](#)
- [Mobile IP Platform Considerations on page 309](#)
- [Mobile IP References on page 309](#)
- [Before You Configure the Mobile IP Home Agent on page 309](#)
- [Configuring the Mobile IP Home Agent on page 310](#)
- [Monitoring the Mobile IP Home Agent on page 315](#)

Mobile IP Overview

Mobile IP is a tunneling-based solution that enhances the utility of E Series routers at the edge of the network between fixed wire and wireless network domains. This tunneling-based solution enables a router on a user's home subnet to intercept and forward IP packets to users who roam beyond traditional network boundaries. Mobile IP is useful in environments where mobility is desired and the traditional land line dial-in model does not provide an adequate solution, and in environments where a wireless technology is used.



NOTE: Currently, JunosE Software does not support configuration of the Mobile IP foreign agent.

Traditionally, IP addresses are associated with a fixed network location. To achieve mobility, the mobile node assumes a secondary IP address that matches the new network and redirects the traffic bound to the primary or home address to the mobile node's new network. In the Mobile IP architecture, the two agents that accomplish this task are the home agent and the foreign agent.

When a mobile node roams into a new network, it negotiates with the foreign agent to get a secondary IP address, which is referred to as the care-of address (CoA). The mobile node registers this CoA with the home agent. The home agent then establishes a tunnel to the CoA if the tunnel is not established earlier.



NOTE: You need to establish only one tunnel between the home agent and the CoA. Demultiplexing of the traffic is done through IP address inspection.

Packets sent to the home address of the mobile node are redirected by the home agent through the tunnel to the CoA at the foreign agent. The foreign agent routes the packets to the mobile node's home address. If the mobile node's home address is a private address or if the foreign agent implements ingress filtering, a reverse tunnel from the CoA to the home agent is required.

You can use the Mobile IP home agent feature to configure the home agent within a virtual router. The home agent handles the following tasks:

- Agent discovery
- Registration
- Routing and forwarding

Mobile IP Agent Discovery

Mobile nodes use the agent discovery process to identify whether they are on their home network or have roamed into a different network (referred to as a foreign network). Both the foreign agent and the home agent periodically multicast their agent advertisements. You can also request an agent advertisement from the mobile node through Internet Control Message Protocol (ICMP) router solicitations.

Mobile IP Registration

The home agent receives the registration requests on UDP port 434. The registration request contains the IP router ID as the home agent IP address. The home agent can support static home address allocation and dynamic home address allocation.

Home Address Assignment

The mobile node's home address can either be preconfigured, or dynamically allocated by the Mobile IP home agent. If a nonzero home address is preconfigured, the home agent processes the registration request using the home address. If the home address is dynamically allocated, the mobile node submits a nonzero home address and requests the home agent to assign an IP address. The mobile node then uses the address provided by the home agent for subsequent registration requests, until the mobile node is rebooted or the registration expires.

Home address allocation is done by one of the existing AAA back-end address mechanisms, such as:

- By RADIUS
- From an address pool returned by RADIUS
- From a local pool
- By the DHCP server

Authentication

The home agent authenticates the requests based on RFC 3344—IP Mobility Support for IPv4 (August 2002). The mobile home authentication is verified and the authentication algorithm and key are retrieved by checking the security association indexed by the security parameter index (SPI) value. This verification results in a 128-bit key and the authentication algorithm with which to compute an MD-5 message digest over the registration request. The Mobile IP home agent supports both HMAC-MD5 and keyed-MD5 authentication algorithms. When the result of this computation matches the 128-bit authenticator, the mobile-home extension is authenticated.

If a security association is configured for the foreign agent, the foreign-home authentication extension is verified; otherwise, authentication success is based only on the mobile-home authenticator.

The home agent checks the identification (ID) field used for matching registration requests with response and protection against replay attacks. The home agent uses timestamp-based replay protection and the ID field represents a 64-bit Network Time Protocol (NTP)-formatted time value. By default, the timestamp must be within 7 seconds of the home agent configured time value.

AAA

You can store the security associations and configuration information remotely on a RADIUS server. You can use the **ip mobile secure host** command and the **ip mobile secure foreign-agent** command to configure the security association (MD-5 key) for a specified user, or for a group of users (also known as a domain) for the home agent. The home agent can configure the security association (MD-5 key) for a specified user or a group of users (domain).

Authentication is accomplished either by generating an authentication, authorization, and accounting (AAA) access-request or querying the locally configured security parameters, depending on whether or not you use the **aaa** keyword when you issue the **ip mobile host** command to configure the mobile node. For AAA authentication, you must include the **aaa** keyword; for local authentication, do not include the **aaa** keyword. If AAA authentication is enabled, AAA queries the security information from the RADIUS server.

When both the network access identifier (NAI) and IP address of the mobile node are present in the registration request, then the authentication request from Mobile IP to AAA has the NAI as the user name and the IP address as the hint IP address. If only the NAI is present in the registration request, then the NAI address is used as the user name with no hint IP address in the authentication request. If only the IP address (home address) is present in the registration request, then it is used as both the user name and the hint IP address in the authentication request. If both the NAI address and the IP address are missing from the registration request, then the registration request is rejected.

If the optional **aaa** keyword is present in the **ip mobile host** command, then the authentication parameters are obtained by querying AAA. The authentication algorithm and security key are retrieved by AAA based on its configuration, depending on the SPI provided in the registration request. If the **aaa** keyword is absent, then the home agent

uses authentication parameters configured locally on the router to authenticate the registration request. In both cases, if security parameters are not retrieved, then the request for mobility service is rejected, a security violation error is logged, and no registration reply is generated.

When you configure the mobile host to use RADIUS authentication for home agent users by including the **aaa** keyword in the **ip mobile host** command, the Mobile IP home agent application generates a RADIUS access-request message. The RADIUS server then uses Juniper Networks vendor-specific attributes (VSAs) to provide the appropriate authentication algorithm and secure key for the authentication request.

For information about the specific Juniper Networks VSAs used for Mobile IP RADIUS-based authentication, see *JunosE Broadband Access Configuration Guide* and RADIUS IETF Attributes

Subscriber Management

The Mobile IP home agent interoperates with the subscriber management application on E Series routers. The subscriber management application enables customers to dynamically provision new IP subscribers and quickly create new value-added services.

You can set up your subscriber management environment to create dynamic IP subscriber interfaces to provision subscribers and provide differentiated service delivery. In this configuration, the service parameters for an IP subscriber are bound to a dynamic IP subscriber interface.

During the registration process when the Mobile IP home agent has authenticated the subscriber with AAA, the home agent locates or creates the appropriate IP tunnel to carry the data traffic to the foreign agent. When Mobile IP obtains all of the parameters required for interface creation, including the tunnel ID and the authentication context, it directs the subscriber management application to create the dynamic IP subscriber interface.

During the re-registration process when there is a handoff from an initial Mobile IP foreign agent to a new Mobile IP foreign agent, the home agent reauthenticates the subscriber with AAA and locates or creates the appropriate IP tunnel to carry the data traffic to the new foreign agent. When Mobile IP obtains all of the parameters required for interface creation, it directs the subscriber management application to move the dynamic IP subscriber interface from the initial tunnel for the previous foreign agent to the new tunnel that points to the new foreign agent. If this was the last subscriber on the tunnel for the previous foreign agent, then the home agent directs the IP tunneling application to tear down the initial tunnel.

For more information about subscriber management and dynamic IP subscriber interfaces, see *JunosE Broadband Access Configuration Guide*. For more information about dynamic IP subscriber interfaces, see *JunosE Broadband Access Configuration Guide*.

Mobile IP Routing and Forwarding

The home agent supports both generic routing encapsulation (GRE) and Distance Vector Multicast Routing Protocol (DVMRP, also known as IP-in-IP) tunnel encapsulation for forward and reverse tunneling. When packets destined for the mobile node reach a home agent, the home agent encapsulates the packets and tunnels them to the CoA. Packets

that exceed the maximum transmission unit (MTU) value of the tunnel are dropped and an ICMP error message is sent to the source IP address. Packets without an access route are returned to the source with an ICMP destination unreachable error message. For reverse tunnels, packets are de-tunneled and forwarded towards the next hop to the destination address.

For more information about configuring GRE and DVMRP dynamic IP tunnels, see [“Configuring Dynamic IP Tunnels” on page 251](#).

Mobile IP Platform Considerations

For information about modules that support the Mobile IP home agent on ERX7xx models, ERX14xx models, and the ERX310 Broadband Services Router:

- See *ERX Module Guide, Table 1, Module Combinations* for detailed module specifications.
- See *ERX Module Guide, Appendix A, Module Protocol Support* for information about the modules that support the Mobile IP home agent.

For information about modules that support the Mobile IP home agent on E120 and E320 Broadband Services Routers:

- See *E120 and E320 Module Guide, Table 1, Modules and IOAs* for detailed module specifications.
- See *E120 and E320 Module Guide, Appendix A, IOA Protocol Support* for information about the modules that support the Mobile IP home agent.

Mobile IP References

For more information about Mobile IP, consult the following resources:

- RFC 2006—The Definitions of Managed Objects for IP Mobility Support using SMIv2 (October 1996)
- RFC 2486—The Network Access Identifier (January 1999)
- RFC 2794—Mobile IP Network Access Identifier Extension for IPv4 (March 2000)
- RFC 2865—Remote Authentication Dial In User Service (RADIUS) (June 2000)
- RFC 3024—Reverse Tunneling for Mobile IP, revised (January 2001)
- RFC 3344—IP Mobility Support for IPv4 (August 2002)

Before You Configure the Mobile IP Home Agent

Before you can configure the Mobile IP home agent on a virtual router, perform the following tasks:

1. Create a virtual router to enable the Mobile IP license.
2. (Optional) Configure the access list for filtering foreign agents.

3. Configure an IP interface, which is used as the care-of address.
4. Configure the router ID of the virtual router, which becomes the home agent IP address.
5. (Optional) Configure the B-RAS license.
6. (Optional) Configure a RADIUS authentication server on the router.
7. (Optional) Configure a RADIUS accounting server on the router.
8. Configure a loopback interface to be used as the primary interface for a tunnel.
9. Configure an interface profile for mobile host associations.
10. Configure a destination profile for dynamic GRE or DVMRP tunnels, as described in [“Configuring Dynamic IP Tunnels” on page 251](#).

The following example illustrates this procedure:

```
! Create a virtual router.
host1(config)#virtual-router test
! Configure an access list.
host1:test(config)#access-list test deny ip host 100.1.1.3 any log
! Configure an IP interface.
host1:test(config)#interface loopback 0
host1:test(config-if)#ip address 10.10.10.1 255.255.255.255
! Configure the IP router ID.
host1:test(config)#ip router-id 10.10.10.1
! Configure the B-RAS license.
host1:test(config)#license b-ras demo
! Configure an authentication server.
host1:test(config)#radius authentication server 10.209.13.234
host1:test(config-radius)#key secret
host1:test(config-radius)#udp-port 1812
host1:test(config-radius)#radius update-source-addr 10.209.12.2
! Configure an accounting server.
host1:test(config-radius)#radius accounting server 10.209.13.234
host1:test(config-radius)#key secret
host1:test(config-radius)#udp-port 1813
! Create the primary interface for the tunnel.
host1:test(config)#interface loopback 1
! Configure a profile for mobile host associations.
host1:test(config)#profile virDefault
```

For information about configuring virtual routers and access lists, see the *JunosE System Basics Configuration Guide*. For information about configuring IP interfaces, see the *JunosE IP, IPv6, and IGP Configuration Guide*.

For information about configuring B-RAS licenses, RADIUS authentication servers, and RADIUS accounting servers, see the *JunosE Broadband Access Configuration Guide*.

Configuring the Mobile IP Home Agent

To configure the Mobile IP home agent on a virtual router:

1. Configure a license for the Mobile IP home agent.
2. Configure the Mobile IP home agent settings.

3. Configure one or more mobile hosts.
4. Configure the Mobile IP security associations for mobile hosts.
5. Configure the Mobile IP security associations for foreign agents.
6. Assign an interface profile to be referenced by the Mobile IP home agent.
7. (Optional) Verify the Mobile IP configuration. See [“Monitoring the Mobile IP Home Agent” on page 315](#).

The following example illustrates how you can configure a Mobile IP home agent on a virtual router named test:

```
! Configure the Mobile IP home agent license.
host1:test(config)#license mobile-ip home-agent demo
! Configure the Mobile IP home agent settings.
host1:test(config)#ip mobile home-agent care-of-access acl lifetime 2000 replay 255
reverse-tunnel-off
! Configure mobile hosts and their security associations.
host1:test(config)#ip mobile host 200.1.1.1 lifetime 200
host1:test(config)#ip mobile secure host 200.1.1.1 spi 0x398 key ascii w4ex algorithm
keyed-md5 replay timestamp within 225
! Configure foreign agents and their security associations.
host1:test(config)#ip mobile secure foreign-agent 100.1.1.3 spi 256 key ascii secret replay
timestamp within 255 algorithm hmac-md5
! Assign an interface profile for the Mobile IP home agent.
host1:test(config)#ip mobile profile testProfile
```

ip mobile home-agent

- Use to configure the Mobile IP home agent on a virtual router.
- To specify the access control list (ACL) applied to the care-of address (CoA) that restricts access for foreign agents or networks, include the **care-of-access** keyword followed by the ACL name.
- To specify the interval within which the registration requests are established, include the **lifetime** keyword followed by the number of seconds, in the range 5–65535; the default value is 36,000 seconds.
- To specify the interval within which a registration can exceed the home agent configured value, include the **replay** keyword followed by the number of seconds, in the range 1–255; the default value is 7 seconds.
- To disable reverse tunneling support by the home agent for denying T bit registration requests, include the **reverse-tunnel-off** keyword; reverse tunneling is enabled by default.
- Example


```
host1(config)#ip mobile home-agent care-of-access acl lifetime 2000 replay 255
reverse-tunnel-off
```
- Use the **no** version to disable the home agent service on the virtual router.



NOTE: The values for lifetime, replay, and care-of-access configured per mobile host by using the **ip mobile host** command override the values configured by using the **ip mobile home-agent** command.

- See ip mobile home-agent.

ip mobile host

- Use to configure a mobile node on a virtual router with an optional host network access identifier (NAI) address or the home address (IP address of the home agent).
- To specify the mobile node, include the required **nai** keyword or the required **address** keyword, as follows:
 - To specify the NAI for the mobile node, include the **nai** keyword. You must choose one of the following formats, where *user* represents the user name and *realm* represents the domain name: *user@realm*, *@realm*, or *@*.
 - To specify a nonzero home address of the mobile node, include the **address** keyword followed by the IP address of the mobile node.
- To specify that the AAA server should validate registration requests and obtain configuration and security associations, include the **aaa** keyword.
- To specify the access control list applied to the care-of address that restricts access for foreign agents or networks, include the **care-of-access** keyword followed by the ACL name.
- To specify the interval within which the registration requests are established, include the **lifetime** keyword followed by the number of seconds, in the range 5–65535; the default value is 36,000 seconds.
- Example 1—This example illustrates local authentication of a mobile node; do not specify the **aaa** keyword for local authentication.

```
host1(config)#ip mobile host 200.1.1.1 lifetime 200
```

or

```
host1(config)#ip mobile host nai @amazon.net
```

- Example 2—This example illustrates AAA authentication of a mobile node; you must specify the **aaa** keyword for AAA authentication.

```
host1(config)#ip mobile host nai @yahoo.com aaa care-of-access acl2
```

or

```
host1(config)#ip mobile host nai bob@msn.net aaa lifetime 400
```

- Use the **no** version to delete the configuration of the mobile node on the virtual router.
- See ip mobile host.

ip mobile profile

- Use to configure or associate a preconfigured interface profile with the home agent in a virtual router.
- For information about configuring a virtual router, see the *JunosE System Basics Configuration Guide*.
- Example


```
host1(config)#ip mobile profile virDefault
```
- Use the **no** version to remove the profile configuration from the virtual router.
- See ip mobile profile.

ip mobile secure foreign-agent

- Use to configure the security associations for a foreign agent.
- To specify a nonzero address for the foreign agent, include the IP address of the foreign agent.
- To specify the security parameter index (SPI) value to authenticate inbound requests and permit authentication for outbound registration requests, include the required **spi** keyword followed by a 4-octet hexadecimal number, in the range 0x100–0xFFFFFFFF.
- To specify the authentication key for this security association, include the required **key** keyword followed by either the **hex** keyword or the **ascii** keyword, as follows:
 - To specify a hexadecimal key, use the **hex** keyword followed by a 32-character (128-bit) hexadecimal value in the range 0x0–0xFFFFFFFFE.
 - To specify an ASCII key, use the **ascii** keyword followed by an alphanumeric value up to a maximum of 16 characters (128 bits).
- To specify the number of seconds by which a registration request can exceed the time value configured on the home agent, include the optional **replay timestamp within** keywords followed by the number of seconds, in the range 1–255; the default value is 7 seconds.
- To specify the type of authentication algorithm for Mobile IP messages, include the optional **algorithm** keyword followed by either the **hmac-md5** keyword or the **keyed-md5** keyword.
- Example


```
host1(config)#ip mobile secure foreign-agent 100.1.1.3 spi 256 key ascii secret replay
timestamp within 255 algorithm hmac-md5
```
- Use the **no** version to delete the security associations for the specified foreign agent on the virtual router.
- See ip mobile secure foreign-agent.

ip mobile secure host

- Use to configure the security associations for a mobile node.
- You must configure security associations only for mobile nodes on which local authentication is configured.



NOTE: If you delete a mobile node host by using the **no ip mobile host** command, all security associations that you configured for this host are deleted.

- To specify the mobile node, include the required **nai** keyword or the required **address** keyword, as follows:
 - To specify the network access identifier (NAI) for the mobile node, include the **nai** keyword. You must choose one of the following formats, where *user* represents the user name and *realm* represents the domain name: *user@realm*, *@realm*, or *@*.
 - To specify a nonzero home address of the mobile node, include the **address** keyword followed by the IP address of the mobile node.
- To specify the security parameter index (SPI) value to authenticate inbound requests and permit authentication for outbound registration requests, include the required **spi** keyword followed by a 4-octet hexadecimal number, in the range 0x100–0xFFFFFFFF.
- To specify the authentication key for this security association, include the required **key** keyword followed by either the **hex** keyword or the **ascii** keyword, as follows:
 - To specify a hexadecimal key, use the **hex** keyword followed by a 32-character (128-bit) hexadecimal value in the range 0x0–0xFFFFFFFF.
 - To specify an ASCII key, use the **ascii** keyword followed by an alphanumeric value up to a maximum of 16 characters (128 bits).
- To specify the number of seconds by which a registration request can exceed the time value configured on the home agent, include the optional **replay timestamp within** keywords followed by the number of seconds, in the range 1–255; the default value is 7 seconds.
- To specify the type of authentication algorithm for Mobile IP messages, include the optional **algorithm** keyword followed by either the **hmac-md5** keyword or the **keyed-md5** keyword.
- Examples


```
host1(config)#ip mobile secure host 200.1.1.1 spi 0x398 key ascii w4ex algorithm
keyed-md5 replay timestamp within 225
```

or

```
host1(config)#ip mobile secure host nai @amazon.net spi 0x100 key ascii pD4En
algorithm keyed-md5 replay timestamp within 100
```
- Use the **no** version to delete the security associations for the specified host on the virtual router.
- See `ip mobile secure host`.

license mobile-ip home-agent

- Use to configure the license key to enable a home agent.
- Specify a name for the license key; up to a maximum of 16 alphanumeric characters.
- Example

```
host1(config)#license mobile-ip home-agent demo
```
- Use the **no** version to delete the license key configuration.
- See license mobile-ip home-agent.

Monitoring the Mobile IP Home Agent

Use the commands described in this section to set a statistics baseline, remove the binding table, and verify the configuration of the Mobile IP home agent on a virtual router.

baseline ip mobile home-agent

- Use to set a statistics baseline for a specified Mobile IP home agent.
- Example

```
host1#baseline ip mobile home-agent
```
- There is no **no** version.
- See baseline ip mobile home-agent.

clear ip mobile binding

- Use to remove the binding table in the specified virtual router or a specified binding by the mobile node home address or NAI.
- Example

```
host1#clear ip mobile binding nai john@yahoo.com
```
- There is no **no** version.
- See clear ip mobile binding.

show ip mobile binding

- Use to display the binding table information of the home agent in the virtual router.
- Field descriptions
 - MN-NAI—Network access identifier of the mobile node in *user@realm*, *@realm*, or *@* format
 - AAA-NAI—Network access identifier returned from the AAA server in *user@realm*, *@realm*, or *@* format
 - Home IP address—IP address of the mobile node
 - Home agent address—IP address of the home agent
 - Care-of-address—IP address of the foreign agent care-of address or co-located care-of address

- Lifetime granted—Interval, in *hh:mm:sec* format, granted during registration before which the registration request exceeds the home agent configured time
- Lifetime remaining—Remaining interval, in *hh:mm:sec* format, at which the registration request exceeds the home agent configured time
- Tunnel—Configuration information provided while setting up the tunnel between the foreign agent and the home agent
- Reverse tunnel—Whether reverse tunneling is enabled or disabled
- Example

```
host1#show ip mobile binding
MN-NAI:  jr@zoom.com
AAA-NAI:  user@zoom.com
Home IP address:  55.0.0.5
Home agent address:  66.0.0.5
Care-of-address:  72.1.1.15
Lifetime granted :  10:00:00 (36000 seconds)
Lifetime remaining :  01:46:32
Tunnel:  Source 66.0.0.5, Destination 72.1.1.15, Encapsulation
GRE
Reverse tunnel: enabled
```
- See `show ip mobile binding`.

show ip mobile home-agent

- Use to display the configuration information of the home agent in the virtual router.
- Field descriptions
 - Access list name—Name of the access control list applied to the care-of address that restricts access for foreign agents or networks
 - Registration lifetime (in seconds)—Number of seconds before which the registration requests are established
 - Replay protection time (in seconds)—Number of seconds before which a registration request can exceed the home agent configured time value
 - Reverse tunnel—Whether reverse tunneling is enabled or disabled
- Example

```
host1#show ip mobile home-agent
Home Agent Parameters
Access list name          ---
Registration lifetime      (in seconds)  36000
Replay protection time     (in seconds)   7
Reverse tunnel             enabled
```
- See `show ip mobile home-agent`.

show ip mobile host

- Use to display configuration of all or specified mobile nodes or domain users.
- Field descriptions
 - MN-NAI—Network access identifier of the mobile node in *user@realm*, *@realm*, or *@* format
 - Home IP address—IP address of the mobile node
 - Lifetime—Number of seconds the registration request is active for a mobile node
 - Care-Of-Access—Name of the ACL applied to the care-of address to restrict network roaming
 - Aaa-Configured—Whether AAA server is configured or not
- Example 1

```
host1#show ip mobile host
Home
  MN-NAI      IP address  Lifetime  Care-Of-Access  Aaa-Configured
  -----
  @warner.com   ---         36000    ---             no
  @yahoo.com    ---         ---      ---             yes
  pj@juniper.net ---         100     ---             no
  pm@juniper.net ---         500     ---             no
```

- Example 2

```
host1#show ip mobile host nai @warner.com
  MN-NAI      Home IP address  Lifetime  Care-Of-Access  Aaa-Configured
  -----
  @warner.com   ---         36000    ---             no
```

- See `show ip mobile host`.

show ip mobile profile

- Use to display the interface profile name associated with the home agent.
- Field descriptions
 - Mobile IP profile is—Name of the interface profile name associated with the home agent
- Example

```
host1#show ip mobile profile
Mobile IP profile is: mobileIpProfile
```

- See `show ip mobile profile`.

show ip mobile secure foreign-agent

- Use to display the security associations configured for all foreign agents on the virtual router.
- Field descriptions

- IP address—IP address of foreign agent
- SPI—Security parameter index (SPI) key for authenticating registration requests
- Algorithm—Algorithm (hmac-md5 or keyed-md5) for authenticating Mobile IP messages
- Replay—Interval, in seconds, before which the registration request exceeds the home agent configured time
- Key—128-bit hexadecimal number for authenticating the security association.
- Example

```

host1#show ip mobile secure foreign-agent
IP address      SPI           Algorithm    Replay    Key
-----
10.10.10.1      628 ( 0x274 )  hmac-md5    ---       secret
20.20.20.1      628 ( 0x274 )  hmac-md5    255       secret
30.30.30.1      628 ( 0x274 )  hmac-md5    255       secret

```

- See `show ip mobile secure foreign-agent`.

show ip mobile secure host

- Use to display the security associations configured on all mobile node hosts in the virtual router.
- Field descriptions
 - MN-NAI—Network access identifier of the mobile node in *user@realm*, *@realm*, or *@* format
 - Home IP address—IP address of the mobile node host
 - SPI—Security parameter index (SPI) key for authenticating registration requests
 - Algorithm—Algorithm (hmac-md5 or keyed-md5) for authenticating Mobile IP messages
 - Replay—Interval, in seconds, before which the registration request exceeds the home agent configured time
 - Key—128-bit hexadecimal number for authenticating the security association
- Example

```

host1#show ip mobile secure host
MN-NAI          Home IP address      SPI           Algorithm    Replay    Key
-----
@warner.com     ---                 288 ( 0x120 )  hmac-md5    255       time

```

- See `show ip mobile secure host`.

show ip mobile traffic

- Use to display protocol statistics for the Mobile IP home agent traffic, including advertisements, solicitations, registrations, registration errors, and security violations.
- To display baseline-relative statistics for the Mobile IP home agent traffic, use the optional **delta** keyword.
- Field descriptions
 - Registration requests—Total number of registration requests, de-registration requests, and accepted registration requests for mobile nodes
 - Register—Number of registration requests received by the home agent
 - Deregister—Number of de-registration requests received by the home agent
 - Accept—Number of registration requests accepted by the home agent
 - Registration rejects—Total number of and reasons for unsuccessful responses to registration requests
 - Denied—Number of registration requests denied by the home agent
 - Unspecified—Number of registration requests rejected for an unspecified reason, such as an internal communication failure
 - Unknown HA—Number of registration requests rejected because of an unknown home agent address, or because the specified home agent address is not serviced by this home agent
 - Administratively prohibited—Number of registration requests prohibited for administrative reasons, such as the broadcast or B bit being set without the corresponding D bit, or a denial by the registration filters
 - No Resources—Number of registration requests rejected due to insufficient resources, such as a full binding table, an inability to create a tunnel, or an inability of the IP subscriber management application to create a dynamic subscriber interface
 - Authentication failed MN—Number of registration requests rejected because the mobile node failed authentication
 - FA—Number of registration requests rejected because the foreign agent failed authentication
 - Bad identification—Number of registration requests rejected because the registration ID field is out of range
 - Bad request form—Number of registration requests rejected because of a malformed request
 - Unavailable encapsulation—Number of registration requests rejected because of unsupported encapsulation
 - No reverse tunnel—Number of registration requests rejected because reverse tunneling is disabled

- Example

```
host1#show ip mobile traffic
Home Agent Registrations:
  Registration requests:
    Register: 0
    Deregister: 0
    Accept: 0

  Registration rejects:
    Denied: 0
    Unspecified: 0
    Unknown HA: 0
    Administratively prohibited: 0
    No Resources: 0
    Authentication failed MN: 0
    FA: 0
    Bad identification: 0
    Bad request form: 0
    Unavailable encapsulation: 0
    No reverse tunnel: 0
```

- See show ip mobile traffic.

show license mobile-ip home-agent

- Use to display the license key for the home agent.
- Field descriptions
 - Mobile IP license is—Mobile IP license key associated with the home agent and the maximum number of users allowed by this license
- Example

```
host1#show license mobile-ip home-agent
Mobile IP license is PcZJ93Mt17 which allows 48000 users
```

- See show license mobile-ip home-agent.

PART 2

Index

- [Index on page 323](#)

Index

A

AAA	
and Mobile IP home agent.....	305
access lists, BGP.....	20
access lists, IP	
monitoring.....	48
redirecting traffic with null interface	
instead.....	32
redistributing access routes.....	6
redistributing access-internal routes.....	6
redistributing static routes.....	20
access-list command.....	20, 25, 72
adjustment-factor command.....	194
aggregation flow caches.....	92
configuring.....	95
ANCP (Access Node Control Protocol)	
adjusting downstream rates.....	192
monitoring.....	198
overview.....	185
ANCP commands	
clear l2c neighbor.....	192
id.....	190
l2c.....	188
l2c end-user-id.....	189
l2c ip listen.....	188
l2c ip oif.....	195
l2c line-configuration.....	193
l2c max-branches.....	189
l2c peer-attachment-id.....	189
max-branches.....	190
neighbor.....	190
qos-adaptive-mode.....	192
session-timeout.....	188
AS-path attribute.....	22
authentication	
Mobile IP home agent.....	305
authentication commands	
authentication.....	219, 225

B

baseline commands	
baseline ip.....	85
baseline ip mobile home-agent.....	315
baseline ip tunnel-reassembly.....	272
baseline, setting	
Mobile IP home agent.....	315
tunnel reassembly.....	272
BFD (Bidirectional Forwarding Detection)	
BGP peer reachability detection.....	107
license.....	111
liveness detection.....	108
liveness detection interval, negotiating	
the.....	108
transmit interval, negotiating the.....	108
BFD commands	
clear bfd session.....	113
clear ipv6 bfd session.....	113
show license bfd.....	114
BGP (Border Gateway Protocol)	
BFD.....	107
clearing IP routing table.....	47
reinstalling routes in IP routing table.....	47
well-known communities.....	37
Bidirectional Forwarding Detection. See BFD	

C

cache flow, IP	
monitoring.....	102
certificate revocation list. See CRL	
checksum computation.....	240, 258
clear commands	
clear ip mobile binding.....	315
clear l2c neighbor.....	192
clear ip commands	
clear ip prefix-list.....	32
clear ip prefix-tree.....	35
clear ip routes.....	47
clearing L2C neighbors.....	192
communities, BGP.....	37
community lists, BGP.....	37
conventions	
notice icons.....	xxi
text and syntax.....	xxii
CRL (certificate revocation list).....	206
checking.....	208
viewing.....	228
customer support.....	xxiii
contacting JTAC.....	xxiii

D

dead peer detection. <i>See</i> DPD	
default-information originate command.....	26
destination profiles	
configuring.....	255
monitoring.....	260
destruct timeout period for single-shot tunnels.....	286
digital certificates	
authenticating the peer.....	208
base64.....	205
CA hierarchy.....	208
certificate chains.....	208
checking CRLs.....	208
configuring.....	213
file extensions.....	208
generating private/public key pairs.....	208
monitoring.....	228
obtaining a public key certificate.....	208
obtaining a root CA certificate.....	208
obtaining public keys without.....	212, 224
offline configuration.....	213
offline enrollment.....	208
online configuration.....	219
online enrollment.....	208
overview.....	205
signature authentication.....	207
standards.....	208
viewing.....	208, 209, 228
X.509v3.....	207
documentation set	
comments on.....	xxiii
DPD (dead peer detection).....	133
DVMRP (Distance Vector Multicast Routing Protocol)	
reassembly of tunnel packets.....	270
tunnels.....	238
dvmrp destination profile command.....	257
DVMRP with IPSec	
how it works.....	290
setting up secure connection.....	290
dynamic IP tunnels	
configuring.....	255
monitoring.....	260
overview.....	251
dynamic tunnels.....	251

E

enable commands	
enable ipsec-transport.....	287
enable ipsec-transport command.....	257
endpoints, tunnel.....	237

F

filter lists, BGP.....	22
filtering	
AS paths.....	22
network prefixes.....	20
undesirable traffic.....	32
flow statistics commands	
cache entries.....	95
cache timeout.....	95
enabled.....	95
export destination	95
export source.....	95
ip flow-aggregation cache	95
mask destination	95
FQDN (fully qualified domain name).....	127, 144, 148
aggressive mode.....	135
user@fqdn format.....	127
with digital certificates.....	127
with preshared keys.....	127
fully qualified domain name. <i>See</i> FQDN	

G

GRE (Generic Routing Encapsulation)	
reassembly of tunnel packets.....	270
tunnels.....	237
gre destination profile command.....	258
GRE with IPSec	
how it works.....	290
setting up secure connection.....	290

H

home agent, Mobile IP. <i>See</i> Mobile IP home agent	
--	--

I

idle timeout period for single-shot tunnels.....	286
IKE (Internet Key Exchange)	
aggressive mode characteristics.....	135
aggressive mode negotiations.....	135
authentication without digital certificates.....	212, 224
initiator proposals and policy rules.....	136
main mode characteristics.....	135
overview.....	134

-
- SA negotiation.....138
 - using digital certificates.....207
 - IKE commands.....174
 - ike local-identity.....174
 - ike peer-identity.....174
 - IKE message notification type.....151
 - IKE policies.....136
 - authentication mode.....136
 - Diffie-Hellman group.....136
 - encryption algorithms
 - 3DES.....136
 - DES.....136
 - hash function
 - MD5.....136
 - SHA-1.....136
 - IPSec tunnels.....180
 - lifetime.....138
 - priority.....136
 - instance, route map.....4
 - interface commands
 - interface null.....32
 - interface tunnel.....141, 240
 - ipsec-transport keyword.....291
 - interfaces
 - NAT, marking.....69
 - internet community, BGP.....37
 - Internet Key Exchange. *See* IKE
 - invalid cookies, IPSec.....151
 - IP
 - managing the routing table.....47
 - IP addresses
 - prefix lists.....32
 - prefix trees.....35
 - ip commands
 - ip as-path access-list.....22
 - ip bgp-community new-format.....38
 - ip community-list.....39
 - ip prefix-list.....20, 32
 - ip prefix-tree.....20, 35
 - ip refresh-route.....47
 - ip route.....32
 - ip tunnel reassembly.....271
 - IP flow
 - export.....102
 - sampling.....106
 - IP fragmentation
 - reassembling for tunnel packets.....269
 - ip mobile commands.....320
 - ip mobile home-agent.....310
 - ip mobile host.....310
 - ip mobile profile.....310
 - ip mobile secure foreign-agent.....310
 - ip mobile secure host.....310
 - See also* show ip mobile commands
 - ip nat commands.....72
 - address.....72
 - ip nat.....69
 - ip nat inside source list.....73
 - ip nat inside source static.....69
 - ip nat outside source list.....73
 - ip nat outside source static.....71
 - ip nat pool.....72
 - ip nat translation.....75
 - ip nat translation max-entries.....69
 - See also* show ip nat commands
 - IP reassembly of tunnel packets.....269
 - configuring.....271
 - monitoring.....272
 - IP security policies.....132
 - IP tunnels.....237, 251
 - configuring.....240
 - monitoring.....244, 260
 - IP-in-IP tunnels.....237, 251
 - IPSec (IP Security).....169, 277
 - AH.....122
 - AH processing.....133
 - concepts.....122
 - configuration
 - examples.....153
 - tasks.....139
 - configuring
 - IKE policy.....149
 - IPSec parameters.....139
 - tunnels.....141
 - digital certificates.....205
 - encapsulation modes.....130
 - encapsulation protocols.....130
 - ESP.....122
 - ESP processing.....132
 - invalid cookies.....151
 - L2TP with IPSec.....169, 277
 - license.....139
 - monitoring.....160
 - overview.....119
 - packet encapsulation.....123
 - protocol stack.....122

reassembly of tunnel packets.....	270	IPSec IKE policy commands	
remote access.....	169, 277	aggressive-mode.....	149, 180
secure IP interfaces.....	122	authentication.....	149, 213, 219, 225
security parameters.....	124	encryption.....	149
security parameters per policy type.....	126	group.....	149
tunnel destination endpoint.....	126	hash.....	149
tunnel failover.....	133	ip address virtual-router.....	180
tunnel source endpoint.....	126	ipsec ike-policy-rule.....	217, 222, 225
See also L2TP with IPSec		ipsec isakmp-policy-rule.....	217, 222
IPSec CA identity commands		lifetime.....	151
crl.....	219	IPSec security parameters	
enrollment retry-limit.....	219	in relation to IPSec interface.....	125
enrollment retry-period.....	219	inbound SAs.....	124, 127
enrollment url.....	219	lifetime.....	124
ipsec ca identity.....	222	lifetime for user SAs.....	127
issuer-identifier.....	222	manual versus signaled.....	125
root proxy url.....	222	negotiating transforms.....	130
ipsec certificate commands		operational VR.....	124
ipsec certificate-database refresh.....	215	outbound SAs.....	124, 127
ipsec certificate-request generate.....	215	per IPSec policy type.....	126
ipsec commands.....	219	perfect forward secrecy (PFS).....	124, 127
ipsec ca authenticate.....	219	transform combinations supported.....	131
ipsec ca enroll.....	219	transform sets.....	124, 130
ipsec ca identity.....	222	transforms supported.....	131
ipsec clear.....	151	transport VR.....	124, 126
ipsec crl.....	215, 216	IPSec transport local profile commands	
ipsec identity.....	216	pre-share.....	292
ipsec ike-policy-rule.....	151	pre-share-masked.....	292
ipsec isakmp-policy-rule.....	151	IPSec transport profile commands.....	292
ipsec key generate.....	217, 222, 225	application.....	292
ipsec key manual pre-share.....	140	ipsec transport profile.....	292
ipsec key pubkey-chain rsa.....	225	lifetime.....	292
ipsec key zeroize.....	217, 222	local ip address.....	292
ipsec lifetime.....	140	pfs group.....	292
ipsec local-endpoint.....	141	transform-set.....	292
ipsec option dpd.....	144	See also show ipsec transport commands	
ipsec option nat-t.....	288	IPSec tunnel profile commands	
ipsec option tx-invalid-cookie.....	151	domain-suffix.....	174
ipsec transform-set.....	141	extended-authentication.....	174
key.....	141	ike local-identity.....	174
masked-key.....	141	ike peer-identity.....	174
See also show ipsec commands		ip profile.....	174
IPSec identity commands		ipsec tunnel profile.....	173
common-name.....	213	lifetime.....	174
country.....	213	local ip address.....	174
domain-name.....	213	local ip identity.....	174
ipsec identity.....	216	local ip network.....	174
organization.....	217	max-interfaces.....	174
		peer ip identity.....	174

pfs group.....	174
transform.....	174
tunnel mtu.....	174
IPSec tunnel profiles.....	173
IPv6	
license.....	68
monitoring.....	85, 114

J

J-Flow commands	
clear ip flow stats.....	101
ip flow statistics.....	94
ip flow-cache entries.....	95
ip flow-cache timeout active.....	95
ip flow-cache timeout inactive.....	95
ip flow-export.....	95
ip flow-sampling-mode packet-interval.....	95
ip route-cache flow sampled.....	94
J-Flow statistics, clearing.....	101

K

keepalive messages, NAT-T.....	284
key-string command.....	227
keys, public	
displaying on router.....	228
format of.....	213
obtaining without digital certificates.....	212, 224

L

L2C (Layer 2 Control) See ANCP (Access Node Control Protocol)	
L2F, reassembly of tunnel packets.....	270
L2TP (Layer 2 Tunneling Protocol)	
reassembly of tunnel packets.....	270
l2tp commands	
l2tp destination profile.....	287
l2tp ignore-receive-data-sequencing.....	271
L2TP with IPSec.....	169, 277
client software supported.....	281
compatibility.....	281
configuring	
client PC.....	286
E Series router.....	286, 291
IPSec transport profiles.....	292
L2TP destination profiles.....	287, 291
single-shot tunnels.....	289
control and data frames.....	280
group preshared key.....	281
how it works.....	279

LNS change of port.....	281
monitoring.....	296
NAT interactions.....	281
overview.....	169, 277
references.....	278
requirements.....	281
setting up secure connection.....	280
troubleshooting.....	296
tunnel creation.....	277
with PPP.....	281
license commands	
license ipsec-tunnels	139
license mobile-ip home-agent.....	310
license nat.....	68
lifetime, IPSec.....	127, 138
limiting translation entries.....	69
local-as community, BGP.....	37
loopback interfaces.....	238, 254

M

manual IPSec interfaces.....	125
manuals	
comments on.....	xxiii
map tag, route map.....	4
match commands.....	10
and route maps.....	4
match as-paths.....	10
match community.....	10
match distance.....	10
match extcommunity.....	10, 41
match ip address.....	11, 32, 35, 36
match ip next-hop.....	32, 35, 36
match level.....	12
match metric.....	12
match metric-type.....	12
match policy-list.....	12
match route-type.....	13
match tag.....	13
match-set summary prefix-tree.....	35, 36
max-interfaces command.....	174
Mobile IP home agent.....	320
AAA.....	305
agent discovery.....	305
authentication.....	305
configuration prerequisites.....	309
configuring.....	310
home address assignment.....	305
licensing.....	310, 320
monitoring.....	315

overview.....	305	NAT-T (Network Address Translation Traversal)	
platform considerations.....	309	configuring.....	288
references.....	309	ipsec option nat-t command.....	288
registration.....	305	keepalive messages.....	284
routing and forwarding.....	308	overview.....	282
security associations		show ike sa command.....	297
for foreign agents.....	310	show ipsec ike-sa command.....	297
for mobile nodes.....	310	show ipsec option command.....	298
subscriber management.....	308	tasks.....	285
<i>See also</i> ip mobile commands		UDP encapsulation.....	282
MTU (maximum transmission unit)		UDP statistics.....	284
IP tunnels.....	240	neighbor commands	
N		neighbor distribute-list.....	20
NAT (Network Address Translation)		neighbor filter-list.....	22, 27
access list rules, creating.....	71	neighbor prefix-list.....	20
address pools, defining.....	72	neighbor prefix-tree.....	20
address translation		neighbor send-community.....	40
dynamic.....	66	Network Address Translation Traversal. <i>See</i> NAT-T	
inside source.....	65	Network Address Translation. <i>See</i> NAT	
outside source.....	66	network prefixes, filtering.....	20
static.....	66	next-hop routers	
bidirectional.....	64	setting or redistributing routes for.....	32
configuration examples.....	77	setting/redistributing routes for.....	12, 16
configuration types.....	63	no-advertise community, BGP.....	37
configuring.....	61	no-export community, BGP.....	37
dynamic address translation, defining.....	71	no-export-subconfed community, BGP.....	37
dynamic inside source translation,		notice icons.....	xxi
creating.....	73	null interface.....	32
dynamic outside source translation,		O	
creating.....	73	OSPF (Open Shortest Path First)	
interfaces, specifying inside and outside.....	69	clearing IP routing table.....	47
license.....	68	reinstalling routes in IP routing table.....	47
monitoring.....	85	P	
NAT-T	282	peer public keys	
overview.....	61	displaying on router.....	228
passthrough mode.....	282	obtaining without digital certificates.....	212, 224
references.....	62	perfect forward secrecy.....	127
static address translation, defining.....	69	policy list	
terms.....	64	monitoring.....	49
inside global address.....	64	prefix lists.....	32
inside local address.....	64	prefix trees.....	35
outside global address.....	64	prefixes	
outside local address.....	64	filtering network.....	20
timeouts, defining.....	75	preventing recursive tunnels.....	242
translation entries, limiting.....	69	profile commands	
translation rules, defining.....	73	profile.....	258
translations, clearing.....	76		

public keys
 displaying on router.....228
 format of.....213
 obtaining without digital certificates.....212, 224

Q

qos-adaptive-mode command.....192

R

recursive tunnels, preventing242
 redistribute command.....28
 redistribution policy (IP), monitoring.....49
 redundancy.....239
 tunnel server.....239, 255
 regular expressions and routing policy.....42
 AS-path lists.....42
 community lists.....42
 community number format.....42
 metacharacters
 defined.....43
 specifying as literals.....43
 RIP (Routing Information Protocol)
 clearing IP routing table.....47
 reinstalling routes in IP routing table.....47
 route maps
 and routing policy.....4
 deny keyword.....4
 filtering incoming/outgoing routes with access
 lists.....24
 instance.....4
 map tag.....4
 match clause.....4
 monitoring.....49
 permit keyword.....4
 sequence number.....4
 set clause.....4
 route-map command.....13
 routing policy
 community.....37
 community list.....37
 configuring.....3
 managing the routing table.....47
 monitoring.....48
 overview.....3
 prefix lists.....32
 prefix trees.....35
 route maps.....4
 troubleshooting.....47

routing policy, BGP
 access lists.....20, 24
 monitoring.....48, 49
 route maps.....4
 routing table
 managing the IP.....47
 routing, IP.....91
 monitoring.....49
 See also IP

S

secure IP interfaces.....122
 security parameters.....124
 sequence number, route map.....4
 Service Modules. *See* SMs
 set commands.....14
 and route maps.....4
 set as-path prepend.....14
 set automatic-tag.....14
 set comm-list delete.....14
 set community.....15, 40
 set dampening.....15
 set distance.....16
 set extcommunity.....16, 42
 set ip next-hop.....16
 set level.....17
 set local-preference.....17
 set metric.....17
 set metric-type.....17
 set origin.....18
 set route-class.....18
 set route-type.....19
 set tag.....19
 set weight.....19
 shared tunnel-server ports.....238, 254, 270
 show access-list command.....48
 show adjustment-factor command.....198
 show bfd session command.....115
 show dvmrp commands
 show dvmrp destination profile.....260
 show dvmrp tunnel.....244, 260, 296
 show dvmrp tunnel summary.....244, 260
 show gre commands
 show gre destination profile.....260
 show gre tunnel.....244, 260, 296
 show gre tunnel summary.....244, 260
 show ike commands
 show ike policy-rule.....160
 show ike sa.....160, 297

show ip commands	
show ip as-path-access-list.....	48
show ip cache flow.....	102
show ip cache flow aggregation.....	102
show ip community-list.....	49
show ip extcommunity-list.....	42
show ip prefix-list.....	49
show ip prefix-list detail.....	49
show ip prefix-list summary.....	49
show ip prefix-tree.....	49
show ip prefix-tree detail.....	49
show ip prefix-tree summary.....	49
show ip protocols.....	49
show ip redistribute.....	49
show ip route.....	49
show ip route slot.....	49
show ip static.....	49
show ip traffic.....	49
show ip tunnel reassembly statistics.....	273
show ip flow sampling command.....	102, 106
show ip match-policy-list command.....	49
show ip mobile commands.....	320
show ip mobile binding.....	315
show ip mobile home-agent.....	316
show ip mobile host.....	317
show ip mobile profile.....	317
show ip mobile secure foreign-agent.....	317
show ip mobile secure host.....	318
show ip mobile traffic.....	319
See also ip mobile commands	
show ip nat commands	
show ip nat inside rule.....	85
show ip nat outside rule.....	85
show ip nat statistics.....	85
show ip nat translations.....	85
show ipsec commands	
show ike certificates.....	228
show ike configuration.....	228
show ike identity.....	228
show ipsec ca identity.....	228
show ipsec certificates.....	228
show ipsec identity.....	228
show ipsec ike-configuration.....	228
show ipsec ike-policy-rule.....	160
show ipsec ike-sa.....	160, 297
show ipsec key mypubkey rsa.....	228
show ipsec key pubkey-chain rsa.....	228
show ipsec lifetime.....	160
show ipsec local-endpoint	160
show ipsec option.....	160, 298
show ipsec transform-set.....	160
show ipsec tunnel detail.....	160
show ipsec tunnel summary.....	160
show ipsec tunnel virtual-router.....	160
show license ipsec-tunnels.....	160
show ipsec transport commands	
show ipsec transport interface.....	298
show ipsec transport interface summary.....	298
show ipsec transport profile.....	298
show ipv6 commands	
show license nat.....	85
show l2c commands	
show l2c.....	198
show l2c label.....	198
show l2c neighbor.....	198
show l2c statistics.....	198
show l2tp commands	
show l2tp destination profile command.....	302
show license commands	
show license mobile-ip home-agent.....	320
show route-map command.....	49
single-shot tunnels	
configuring.....	289
handling timeout periods.....	286
monitoring.....	302
overview.....	285
single-shot-tunnel command.....	289
SMs (Service modules)	
installing.....	62, 238, 254
monitoring parameters.....	244
redundancy.....	239, 255
source, tunnel.....	237
static routes.....	49, 244
static tunnels.....	237
statistics, tunnel reassembly	
displaying.....	272
setting baseline for.....	272
subscriber management	
Mobile IP home agent.....	308
support, technical See technical support	
T	
table-map command	
IP.....	31
technical support	
contacting JTAC.....	xxiii
text and syntax conventions.....	xxii
timeout periods for single-shot tunnels.....	286

traffic, IP.....	49
transform sets, IPSec.....	130
transport network.....	244
troubleshooting	
DVMRP/IPSec, GRE/IPSec, and L2TP/IPSec	
tunnels.....	296
routing policy.....	47
tunnel commands	
tunnel mdt profile.....	258
tunnel commands, IP	
tunnel checksum.....	240, 258
tunnel destination.....	240, 258
tunnel mtu.....	240
tunnel sequence-datagrams.....	258
tunnel source.....	240, 258
tunnel commands, IPSec	
tunnel destination.....	144
tunnel destination backup.....	148
tunnel lifetime.....	144
tunnel local-identity.....	144
tunnel mtu.....	144
tunnel peer-identity.....	144
tunnel pfs group.....	144
tunnel session-key-inbound.....	144
tunnel session-key-outbound.....	144
tunnel signaling.....	144
tunnel source.....	144
tunnel transform set.....	144
tunnel-server ports	
shared.....	238, 254, 270
tunnels, IP	
DVMRP.....	251
DVMRP (IP in IP).....	238
dynamic.....	251
endpoints.....	237
GRE.....	251
reassembling tunnel packets.....	269
shared tunnel-server ports.....	238, 254, 270
static.....	237
tunnels, IPSec monitoring	
DVMRP/IPSec.....	296
GRE/IPSec.....	296
L2TP/IPSec.....	296
tunnels, single-shot	
configuring.....	289
handling timeout periods.....	289
monitoring.....	302
overview.....	289

U

UDP (User Datagram Protocol)	
encapsulation for NAT-T.....	282
statistics for NAT-T.....	284
updates, BGP	
AS-path filters.....	23

W

well-known communities, BGP.....	37
----------------------------------	----

X

X.509v3 certificates.....	207
---------------------------	-----

