



**Concepts & Examples
ScreenOS Reference Guide**

**Volume 14:
Dual-Stack Architecture with IPv6**

Release 5.4.0, Rev. C

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

Copyright Notice

Copyright © 2009 Juniper Networks, Inc. All rights reserved.

Juniper Networks, the Juniper Networks logo, JUNOS, NetScreen, ScreenOS, and Steel-Belted Radius are registered trademarks of Juniper Networks, Inc. in the United States and other countries. JUNOSe is a trademark of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

All specifications are subject to change without notice. Juniper Networks assumes no responsibility for any inaccuracies in this document or for any obligation to update information in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

FCC Statement

The following information is for FCC compliance of Class A devices: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. The equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

The following information is for FCC compliance of Class B devices: The equipment described in this manual generates and may radiate radio-frequency energy. If it is not installed in accordance with Juniper Networks' installation instructions, it may cause interference with radio and television reception. This equipment has been tested and found to comply with the limits for a Class B digital device in accordance with the specifications in part 15 of the FCC rules. These specifications are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Consult the dealer or an experienced radio/TV technician for help.
- Connect the equipment to an outlet on a circuit different from that to which the receiver is connected.

Caution: Changes or modifications to this product could void the user's warranty and authority to operate this device.

Disclaimer

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

Table of Contents

	About This Volume	vii
	Document Conventions	viii
	Web User Interface Conventions	viii
	Command Line Interface Conventions	viii
	Naming Conventions and Character Types	ix
	Illustration Conventions	x
	Technical Documentation and Support	xi
Chapter 1	Internet Protocol Version 6 Introduction	1
	Overview	2
	IPv6 Addressing	2
	Notation	2
	Prefixes	3
	Address Types	3
	Unicast Addresses	3
	Anycast Addresses	4
	Multicast Addresses	4
	IPv6 Headers	4
	Basic Header	4
	Extension Headers	5
	IPv6 Packet Handling	6
	IPv6 Router and Host Modes	7
	IPv6 Tunneling Guidelines	8
Chapter 2	IPv6 Configuration	9
	Overview	11
	Address Autoconfiguration	11
	Extended Unique Identifier	11
	Router Advertisement Messages	12
	Router Solicitation Messages	12
	Prefix Lists	12
	Neighbor Discovery	13
	Neighbor Cache Table	13
	Neighbor Unreachability Detection	13
	Neighbor Entry Categories	14
	Neighbor Reachability States	14
	How Reachability State Transitions Occur	15
	Enabling an IPv6 Environment	18
	Enabling IPv6 at the Device Level	18
	Disabling IPv6 at the Device Level	19
	Configuring an IPv6 Host	19
	Binding the IPv6 Interface to a Zone	20
	Enabling IPv6 Host Mode	20

Setting an Interface Identifier	20
Configuring Address Autoconfiguration	21
Configuring Neighbor Discovery	21
Configuring an IPv6 Router	22
Binding the IPv6 Interface to a Zone	22
Enabling IPv6 Router Mode	22
Setting an Interface Identifier	23
Setting Address Autoconfiguration	23
Outgoing Router Advertisements Flag	23
Managed Configuration Flag	24
Other Parameters Configuration Flag	24
Disabling Address Autoconfiguration	24
Setting Advertising Time Intervals	25
Advertised Reachable Time Interval	25
Advertised Retransmit Time Interval	26
Maximum Advertisement Interval	26
Minimum Advertisement Interval	26
Advertised Default Router Lifetime	27
Advertising Packet Characteristics	27
Link MTU Value	27
Current Hop Limit	28
Advertising Router Characteristics	28
Link Layer Address Setting	28
Advertised Router Preference	28
Configuring Neighbor Discovery Parameters	29
Neighbor Unreachability Detection	29
MAC Session-Caching	29
Static Neighbor Cache Entries	30
Base Reachable Time	30
Probe Time	31
Retransmission Time	31
Duplicate Address Detection Retry Count	31
Viewing IPv6 Interface Parameters	32
Viewing Neighbor Discovery Configurations	32
Viewing the Current RA Configuration	32
Configuration Examples	33
IPv6 Router	33
IPv6 Host	33
Chapter 3	Connection and Network Services 35
Overview	36
Dynamic Host Configuration Protocol Version 6	36
Device-Unique Identification	36
Identity Association Prefix Delegation-Identification	37
Prefix Features	37
Server Preference	38
Configuring a DHCPv6 Server	38
Configuring a DHCPv6 Client	40
Viewing DHCPv6 Settings	41
Configuring Domain Name System Servers	42
Requesting DNS and DNS Search List Information	43
Setting Proxy DNS Address Splitting	44
Configuring PPPoE	46
Setting Fragmentation	47

Chapter 4	Static and Dynamic Routing	49
	Overview	50
	Dual Routing Tables.....	50
	Static and Dynamic Routing	51
	Upstream and Downstream Prefix Delegation.....	51
	Static Routing.....	52
	RIPng Configuration.....	53
	Creating and Deleting a RIPng Instance.....	54
	Creating a RIPng Instance	54
	Deleting a RIPng Instance	54
	Enabling and Disabling RIPng on Interfaces	55
	Enabling RIPng on an Interface.....	55
	Disabling RIPng on an Interface.....	55
	Global RIPng Parameters	56
	Advertising the Default Route	56
	Rejecting Default Routes.....	57
	Configuring Trusted Neighbors	57
	Redistributing Routes	58
	Protecting Against Flooding by Setting an Update Threshold.....	59
	RIPng Interface Parameters	60
	Route, Interface, and Offset Metrics	60
	Access Lists and Route Maps.....	61
	Static Route Redistribution.....	61
	Configuring Split Horizon with Poison Reverse.....	64
	Viewing Routing and RIPng Information.....	64
	Viewing the Routing Table.....	65
	Viewing the RIPng Database.....	65
	Viewing RIPng Details by Virtual Router	66
	Viewing RIPng Details by Interface.....	67
	Viewing RIPng Neighbor Information	68
	Configuration Examples.....	69
	Enabling RIPng on Tunnel Interfaces.....	69
	Avoiding Traffic Loops to an ISP Router.....	71
	Configuring the Customer Premises Equipment.....	71
	Configuring the Gateway.....	75
	Configuring the ISP Router.....	78
	Setting a Null Interface Redistribution to OSPF.....	79
	Redistributing Discovered Routes to OSPF	80
	Setting Up OSPF-Summary Import	80
Chapter 5	Address Translation	81
	Overview	82
	Translating Source IP Addresses.....	83
	DIP from IPv6 to IPv4	83
	DIP from IPv4 to IPv6	83
	Translating Destination IP Addresses.....	84
	MIP from IPv6 to IPv4.....	84
	MIP from IPv4 to IPv6.....	85
	Configuration Examples.....	86
	IPv6 Hosts to Multiple IPv4 Hosts.....	86
	IPv6 Hosts to a Single IPv4 Host.....	88
	IPv4 Hosts to Multiple IPv6 Hosts.....	90
	IPv4 Hosts to a Single IPv6 Host.....	91

	Translating Addresses for Domain Name System Servers.....	93
Chapter 6	IPv6 in an IPv4 Environment	97
	Overview	98
	Configuring Manual Tunneling	99
	Configuring 6to4 Tunneling.....	102
	6to4 Routers.....	102
	6to4 Relay Routers	103
	Tunnels to Remote Native Hosts.....	104
	Tunnels to Remote 6to4 Hosts.....	107
Chapter 7	IPSec Tunneling	111
	Overview	112
	IPSec 6in6 Tunneling	112
	IPSec 4in6 Tunneling	115
	IPSec 6in4 Tunneling	120
	Manual Tunneling with Fragmentation Enabled	124
	IPv6 to IPv6 Route-Based VPN Tunnel	125
	IPv4 to IPv6 Route-Based VPN Tunnel	127
Chapter 8	IPv6 XAuth User Authentication	131
	Overview	132
	RADIUSv6.....	132
	Single Client, Single Server.....	132
	Multiple Clients, Single Server	132
	Single Client, Multiple Servers	133
	Multiple Hosts, Single Server	133
	IPSec Access Session Management.....	134
	IPSec Access Session.....	134
	Enabling and Disabling IAS Functionality	136
	Releasing an IAS Session.....	136
	Limiting IAS Settings	136
	Dead Peer Detection.....	137
	Configuration Examples.....	138
	XAuth with RADIUS.....	138
	RADIUS with XAuth Route-Based VPN.....	139
	RADIUS with XAuth and Domain Name Stripping	143
	IP Pool Range Assignment.....	147
	RADIUS Retries.....	153
	Calling-Station-Id	153
	IPSec Access Session	154
	Dead Peer Detection.....	163
Appendix A	Switching	A-I

About This Volume

Volume 14: Dual-Stack Architecture with IPv6 describes ScreenOS support for Internet Protocol version 6 (IPv6) and how to secure IPv6 and IPv4/IPv6 transitional networks with tunneling and IPSec.

Dual-stack architecture allows an interface to operate simultaneously in IPv4 and IPv6 modes and facilitates network management, while a network contains both IPv4 and IPv6 devices that pass traffic between IPv4/IPv6 boundaries.

This volume is for experienced network administrators who need to deploy and manage security solutions in an IPv6 or IPv4/IPv6 environment. It contains the following chapters and appendix:

- Chapter 1, “Internet Protocol Version 6 Introduction,” explains IPv6 headers, concepts, and tunneling guidelines.
- Chapter 2, “IPv6 Configuration,” explains how to configure an interface for operation as an IPv6 router or host.
- Chapter 3, “Connection and Network Services,” explains how to configure Dynamic Host Configuration protocol version 6 (DHCPv6), Domain Name Services (DNS), Point-to-Point Protocol over Ethernet (PPPoE), and fragmentation.
- Chapter 4, “Static and Dynamic Routing,” explains how to set up static and dynamic routing. This chapter explains ScreenOS support for Routing Information Protocol-Next Generation (RIPng).
- Chapter 5, “Address Translation,” explains how to use Network Address Translation (NAT) with dynamic IP (DIP) and mapped-IP (MIP) addresses to traverse IPv4/IPv6 boundaries.
- Chapter 6, “IPv6 in an IPv4 Environment,” explains manual and dynamic tunneling.
- Chapter 7, “IPSec Tunneling,” explains how to configure IPSec tunneling to connect dissimilar hosts.
- Chapter 8, “IPv6 XAuth User Authentication,” explains how to configure Remote Authentication Dial In User Service (RADIUS) and IPSec Access Session (IAS) management.
- Appendix A, “Switching,” lists options for using the security device as a switch to pass IPv6 traffic.

Document Conventions

This document uses the conventions described in the following sections:

- “Web User Interface Conventions” on page viii
- “Command Line Interface Conventions” on page viii
- “Naming Conventions and Character Types” on page ix
- “Illustration Conventions” on page x

Web User Interface Conventions

In the Web user interface (WebUI), the set of instructions for each task is divided into navigational path and configuration settings. To open a WebUI page where you can enter configuration settings, you navigate to it by clicking on a menu item in the navigation tree on the left side of the screen, then on subsequent items. As you proceed, your navigation path appears at the top of the screen, each page separated by angle brackets.

The following shows the WebUI path and parameters for defining an address:

Policy > Policy Elements > Addresses > List > New: Enter the following, then click **OK**:

```
Address Name: addr_1
IP Address/Domain Name:
  IP/Netmask: (select), 10.2.2.5/32
Zone: Untrust
```

To open Online Help for configuration settings, click the question mark (?) in the upper right of the screen.

The navigation tree also provides a Help > Config Guide configuration page to help you configure security policies and Internet Protocol Security (IPSec). Select an option from the dropdown menu and follow the instructions on the page. Click the ? character in the upper left for Online Help on the Config Guide.

Command Line Interface Conventions

The following conventions are used to present the syntax of command line interface (CLI) commands in examples and in text.

In examples:

- Anything inside square brackets [] is optional.
- Anything inside braces { } is required.
- If there is more than one choice, each choice is separated by a pipe (|). For example:

```
set interface { ethernet1 | ethernet2 | ethernet3 } manage
```

- Variables are in *italic* type:

```
set admin user name1 password xyz
```

In text, commands are in **boldface** type and variables are in *italic* type.

NOTE: When entering a keyword, you only have to type enough letters to identify the word uniquely. Typing **set adm u whee j12fmt54** will enter the command **set admin user wheezer j12fmt54**. However, all the commands documented here are presented in their entirety.

Naming Conventions and Character Types

ScreenOS employs the following conventions regarding the names of objects—such as addresses, admin users, auth servers, IKE gateways, virtual systems, VPN tunnels, and zones—defined in ScreenOS configurations:

- If a name string includes one or more spaces, the entire string must be enclosed within double quotes; for example:


```
set address trust "local LAN" 10.1.1.0/24
```
- Any leading spaces or trailing text within a set of double quotes are trimmed; for example, " local LAN " becomes "local LAN".
- Multiple consecutive spaces are treated as a single space.
- Name strings are case-sensitive, although many CLI keywords are case-insensitive. For example, "local LAN" is different from "local lan".

ScreenOS supports the following character types:

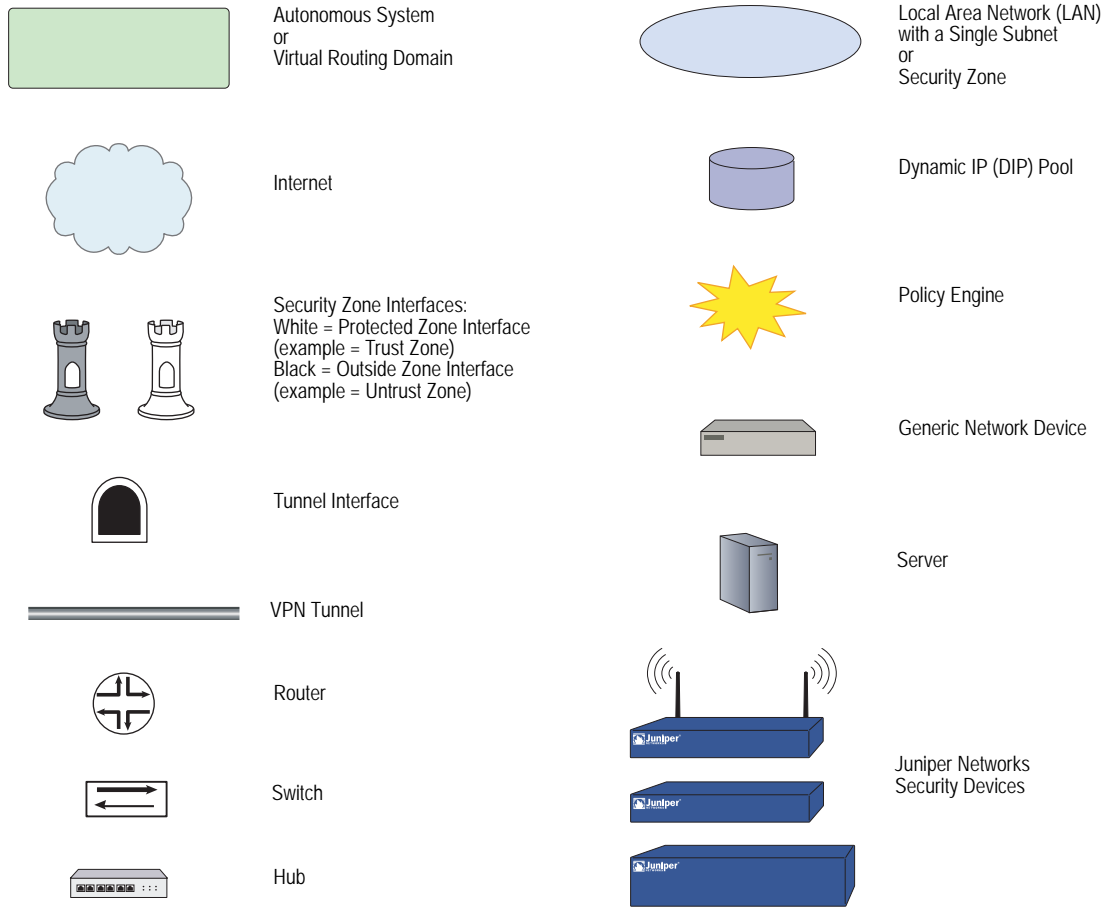
- Single-byte character sets (SBCS) and multiple-byte character sets (MBCS). Examples of SBCS are ASCII, European, and Hebrew. Examples of MBCS—also referred to as double-byte character sets (DBCS)—are Chinese, Korean, and Japanese.
- ASCII characters from 32 (0x20 in hexadecimal) to 255 (0xff), except double quotes ("), which have special significance as an indicator of the beginning or end of a name string that includes spaces.

NOTE: A console connection only supports SBCS. The WebUI supports both SBCS and MBCS, depending on the character sets that your browser supports.

Illustration Conventions

Figure 1 shows the basic set of images used in illustrations throughout this volume.

Figure 1: Images in Illustrations



Technical Documentation and Support

To obtain technical documentation for any Juniper Networks product, visit www.juniper.net/techpubs/.

For technical support, open a support case using the Case Management link at <http://www.juniper.net/customers/support/> or call 1-888-314-JTAC (from the United States, Canada, or Mexico) or 1-408-745-9500 (from elsewhere).

If you find any errors or omissions in this document, please contact Juniper Networks at techpubs-comments@juniper.net.

Chapter 1

Internet Protocol Version 6 Introduction

ScreenOS supports Internet Protocol version 6 (IPv6), developed by the Internet Engineering Task force (IETF).

NOTE: Some security devices support IPv6. Check the datasheet for your security platform to see which features it supports.

This chapter contains the following sections:

- “Overview” on page 2
- “IPv6 Addressing” on page 2
 - “Notation” on page 2
 - “Prefixes” on page 3
 - “Address Types” on page 3
- “IPv6 Headers” on page 4
 - “Basic Header” on page 4
 - “Extension Headers” on page 5
- “IPv6 Packet Handling” on page 6
- “IPv6 Router and Host Modes” on page 7
- “IPv6 Tunneling Guidelines” on page 8

Overview

By using addressing and schema that are different from IPv4, IPv6 allows a greater number of connected hosts than IPv4 can allow. In addition, IPv6 reduces packet processing overhead and increases network scalability. Together, these improvements allow a greater exchange of data traffic.

IPv6 provides for interoperability between IPv4 devices and IPv6 devices. It is usually possible to install IPv6 on security devices without losing IPv4 capability, so organizations can perform incremental upgrades and avoid service disruptions while migrating from IPv4 to IPv6.

NOTE: For more information about IPv6, refer to RFC 2460.

ScreenOS features dual-stack architecture, which allows an interface to operate simultaneously in IPv4 and IPv6 modes. Dual-stack architecture allows you to secure your network infrastructure while it contains both IPv4 and IPv6 devices and to secure traffic that passes across IPv4/IPv6 boundaries.

Each IPv6-enabled security device can operate as an IPv6 host or router.

IPv6 Addressing

IPv6 addresses differ from IPv4 addresses in several ways:

- Notation
- Prefixes
- Address Types

These differences give IPv6 addressing greater simplicity and scalability than IPv4 addressing.

Notation

IPv6 addresses are 128 bits long (expressed as 32 hexadecimal numbers) and consists of eight colon-delimited sections. Each section contains 2 bytes, and each byte is expressed as a hexadecimal number from 0 to FF.

An IPv6 address looks like this:

```
2080:0000:0000:0000:0008:0800:200c:417a
```

By omitting the leading zeroes from each section or substituting contiguous sections that contain zeroes with a double colon, you can write the example address as: 2080:0:0:0:8:800:200c:417a or 2080::8:800:200c:417a

For example, 0000:0000:0000:0000:0000:0000:93fc:9303 can be written as ::93fc:9303.

You can use the double-colon delimiter only once within a single IPv6 address. For example, you cannot express the IPv6 address 32af:0:0:0:ea34:0:71ff:fe01 as 32af::ea34::71ff:fe01.

Prefixes

Each IPv6 address contains bits that identify a network and a node or interface. An IPv6 prefix is the portion of an IPv6 address that identifies the network. The prefix length is a positive integer that denotes a number of consecutive bits, beginning with the most significant (left-most) bit. The prefix length follows a forward slash and, in most cases, identifies the portion of the address owned by an organization. All remaining bits (up to the right-most bit) represent individual nodes or interfaces.

For example, 32f1::250:af:34ff:fe26/64 has a prefix length of 64.

The first 64 bits of this address are the prefix (32f1:0000:0000:0000). The rest (250:af:34ff:fe26) identifies the interface.

Address Types

RFC 2373 describes three major categories of IPv6 addresses:

- Unicast
- Anycast
- Multicast

Unicast Addresses

A unicast address is an identifier for a single interface. When a network device sends a packet to a unicast address, the packet goes only to the specific interface identified by that address.

Devices use the following types of unicast addresses:

- A global unicast address is a unique IPv6 address assigned to a host interface. Global unicast addresses serve essentially the same purposes as IPv4 public addresses. Global unicast addresses are aggregatable for efficient and hierarchical addressing.
- A 6to4 address enables an IPv6 host interface for 6to4 tunneling. A 6to4 interface can serve as a border router between the host and IPv4 network space. In most cases, this method is not suitable for performing IPsec operations such as authentication and encryption.
- A link-local IPv6 address allows communication between neighboring hosts that reside on the same link. The device automatically generates a link-local address for each configured IPv6 interface.
- An IPv4-mapped address is a special IPv6 address that is the equivalent of an IPv4 address. A device uses IPv4-mapped addresses for address translation, when the device must send traffic from an IPv6 network to an IPv4 network.

Anycast Addresses

An anycast address is an identifier for a set of interfaces, which typically belongs to different nodes. When a network device sends a packet to an anycast address, the packet goes to one of the interfaces identified by that address. The routing protocol used in the network usually determines which interface is physically closest within the set of anycast addresses and routes the packet along the shortest path to its destination.

For more information about anycast addresses, see RFC 2526.

Multicast Addresses

A multicast address is an identifier for a set of interfaces, which typically belongs to different nodes. When a network device sends a packet to a multicast address, the device broadcasts the packet to all interfaces identified by that address.

Devices use the following types of multicast addresses:

- Solicited-node multicast addresses for Neighbor Solicitation (NS) messages.
- All-nodes multicast address for Router Advertisement (RA) messages.
- All-routers multicast address for Router Solicitation (RS) messages.

IPv6 Headers

The IETF designed IPv6 headers for low overhead and scalability. IPv6 headers allow optional extension headers, which contain extra information usable by network devices.

Basic Header

Every IPv6 packet has a basic IPv6 header. IPv6 headers occupy 40 bytes (320 bits). Figure 2 shows each field, arranged in order.

Figure 2: Header Structure

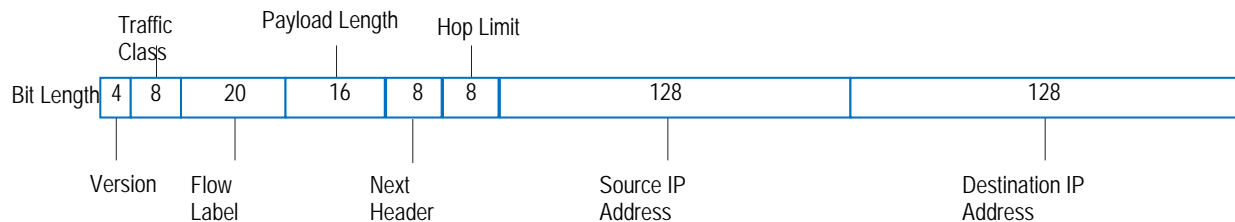


Table 1 lists the fields with bit lengths and their purposes.

Table 1: IPv6 Header Fields, Length, and Purpose

Field Name	Bit Length	Purpose
Version	4	Specifies the Internet Protocol used by the header and packet. This value tells destination internet devices which IP stack (IPv4 or IPv6) to use when processing the packet header and payload. IPv6 Version fields contain a value of 6. (IPv4 Version fields contain a value of 4.)
Traffic Class	8	Allows source nodes or routers to identify different classes (or priorities) of IPv6 packets. (This field replaces the IPv4 Type of Service field, which identified categories of packet transfer services.)
Flow Label	20	Identifies the flow to which the packet belongs. Packets in a flow share a common purpose, or belong to a common category, as interpreted by external devices such as routers or destination hosts. Typically, the source host inserts Flow Label values into outgoing packets to request special handling by the external devices. The external devices can uniquely identify each flow by evaluating the source address in combination with the Flow Label value. Traffic transmitted by a source host can contain packets in a single flow, multiple flows, no flow, or any combination. (Packets that do not belong to a flow carry a Flow Label of zero.)
Payload Length	16	Specifies the length of the IPv6 packet payload, expressed in octets.
Next Header	8	Identifies the type of IP protocol for the header that immediately follows the IPv6 header. This protocol can be one of two types: <ul style="list-style-type: none"> ■ An IPv6 extension header. For example, if the device performs IPSec security on exchanged packets, the Next Header value is probably 50 (ESP extension header) or 51 (AH extension header). Extension headers are optional. ■ An upper-layer Protocol Data Unit (PDU). For example, the Next Header value could be 6 (for TCP), 17 (for UDP), or 58 (for ICMPv6). The Next Header field replaces the IPv4 Protocol field. It is an optional field.
Hop Limit	8	Specifies the maximum number of hops the packet can make after transmission from the host device. When the Hop Limit value is zero, the device drops the packet and generates an error message. (This field is similar the to Time to Live IPv4 field.)
Source IP Address	128	Identifies the host device that generated the IPv6 packet.
Destination IP Address	128	Identifies the intended recipient of the IPv6 packet.

Extension Headers

Extension headers contain supplementary information used by network devices (such as routers, switches, and endpoint hosts) to decide how to direct or process an IPv6 packet. The length of each extension header is an integer multiple of eight octets. This allows subsequent extension headers to use 8-octet structures.

Any header followed by an extension header contains a Next Header value that identifies the extension header type.

Extension headers always follow the basic IPv6 header in order as follows:

1. The Hop-by-Hop Options header specifies delivery parameters at each hop on the path to the destination host. When a packet uses this header, the Next Header value of the previous header (the basic IPv6 header) must be 0.

2. The Destination Options header specifies packet delivery parameters for either intermediate destination devices or the final destination host. When a packet uses this header, the Next Header value of the previous header must be 60.
3. The Routing header defines strict source routing and loose source routing for the packet. (With strict source routing, each intermediate destination device must be a single hop away. With loose source routing, intermediate destination devices can be one or more hops away.) When an packet uses this header, the Next Header value of the previous header must be 43.
4. The Fragment header specifies how to perform IPv6 fragmentation and reassembly services. When a packet uses this header, the Next Header value of the previous header must be 44.
5. The Authentication header provides authentication, data integrity, and anti-replay protection. When a packet uses this header, the Next Header value of the previous header must be 51.
6. The Encapsulating Security Payload header provides data confidentiality, data authentication, and anti-replay protection for encapsulated security payload (ESP) packets. When a packet uses this header, the Next Header value of the previous header must be 50.

IPv6 Packet Handling

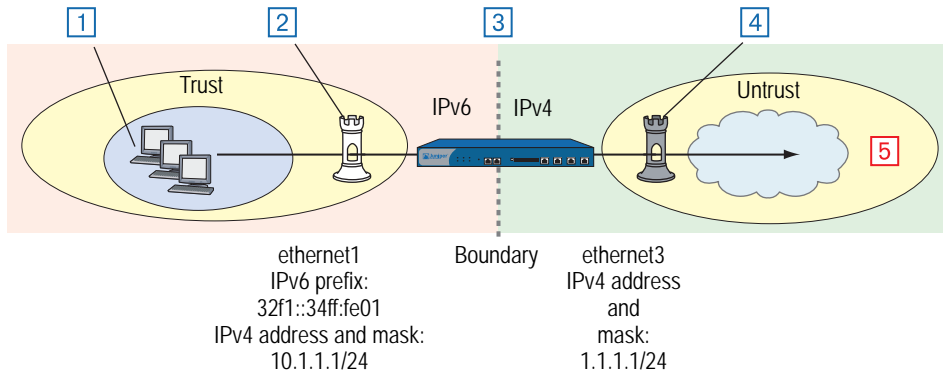
An interface configured for dual-stack operation provides both IPv4 and IPv6 capability. Such an interface can have an IPv4 address, at least one IPv6 address, or both.

If the interface resides at the boundary between an IPv4 network and an IPv6 network, the device can pass IP traffic over the boundary in one of two ways:

- Encapsulate (effectively hiding) any packet that passes across the boundary.
- Perform address translation on the packet source and destination addresses.

Figure 1 shows a packet-handling flow that might occur when an IPv6 host passes an outgoing service request packet across the IPv6/IPv4 boundary into an IPv4 network space.

Figure 3: Packet Flow Across IPv6/IPv4 Boundary



1. An IPv6 host transmits a service request packet. The source and destination addresses use IPv6 format.
2. IPv6 interface ethernet1 receives the packet.
3. The security device applies a policy to the packet. It either encapsulates the entire packet inside an IPv4 packet or translates the addresses to IPv4 format.
4. IPv4 interface ethernet3 transmits the packet (now using IPv4 address format).
5. A remote gateway node receives the packet.
 - If encapsulation occurred, this device might decapsulate the packet or continue to treat it as an IPv4 packet.
 - If IPv6/IPv4 address translation occurred, the device might translate the addresses back to IPv6 format or continue to treat it as an IPv4 packet.

IPv6 Router and Host Modes

You can configure each interface in a security device to function as an IPv6 host or router.

- In Host mode, the interface functions as an IPv6 host and autoconfigures itself by requesting and accepting Router Advertisement (RA) messages from other devices.
- In Router mode, the interface functions as an IPv6 router. An IPv6 router replies to Router Solicitation (RS) messages from IPv6 hosts by sending RAs. In addition, the interface can broadcast RAs periodically or in response to configuration changes to keep the on-link hosts updated.

IPv6 Tunneling Guidelines

Before deciding which kind of tunneling to use, ask your upstream ISP which IPv6 services they provide and how they provide them. We recommend the following guidelines:

- If your ISP provides only dual-stack IPv6, which is Internet Protocol Control Protocol (IPCP) and IPv6CP, you should configure run dual-stack, native IPv6. 6to4 addressing format is not appropriate in this case.
- If your ISP provides manual tunnel IPv6 (IPv6-over-IPv4 tunnel), you should use manual tunneling.
- If your ISP does not provide IPv6, go to www.6bone.net to find an upstream IPv6 provider, and follow their posted instructions.

If you do not find an IPv6 provider, use 6to4 tunneling. This option, however, is only feasible if the next-hop router is configured for 6to4 tunneling.

Chapter 2

IPv6 Configuration

This chapter explains how to enable IPv6 features on the security device and how to configure the security device to act as an IPv6 router or IPv6 host.

It contains the following sections:

- “Overview” on page 11
 - “Address Autoconfiguration” on page 11
 - “Neighbor Discovery” on page 13
- “Enabling an IPv6 Environment” on page 18
 - “Enabling IPv6 at the Device Level” on page 18
 - “Disabling IPv6 at the Device Level” on page 19
- “Configuring an IPv6 Host” on page 19
 - “Binding the IPv6 Interface to a Zone” on page 20
 - “Enabling IPv6 Host Mode” on page 20
 - “Setting an Interface Identifier” on page 20
 - “Configuring Address Autoconfiguration” on page 21

- “Configuring an IPv6 Router” on page 22
 - “Binding the IPv6 Interface to a Zone” on page 22
 - “Enabling IPv6 Router Mode” on page 22
 - “Setting an Interface Identifier” on page 23
 - “Setting Address Autoconfiguration” on page 23
 - “Disabling Address Autoconfiguration” on page 24
 - “Setting Advertising Time Intervals” on page 25
 - “Advertising Packet Characteristics” on page 27
 - “Advertising Router Characteristics” on page 28
 - “Neighbor Unreachability Detection” on page 29
 - “MAC Session-Caching” on page 29
 - “Static Neighbor Cache Entries” on page 30
 - “Base Reachable Time” on page 30
 - “Probe Time” on page 31
 - “Retransmission Time” on page 31
- “Configuration Examples” on page 33
 - “IPv6 Router” on page 33
 - “IPv6 Host” on page 33

Overview

ScreenOS allows you to configure a security device to be an IPv6 router or an IPv6 host.

This overview explains the following topics:

- Address autoconfiguration
- Neighbor discovery

The sections following the overview explain how to configure an IPv6 host or router.

Address Autoconfiguration

Address autoconfiguration allows local hosts to autoconfigure IPv6 addresses from their extended unique identifier (EUI) values. A security device configured for address autoconfiguration advertises an IPv6 prefix to local IPv6 hosts. The local hosts use this prefix to autoconfigure IPv6 addresses from their EUI values.

Address autoconfiguration, reduces the need to manually assign addresses to individual hosts. Ideally, IPv6 hosts have address autoconfiguration enabled. An interface, configured to operate as an IPv6 router, can enable local on-link IPv6 hosts to perform autoconfiguration. Autoconfiguration does not require a stateful configuration protocol, such as Dynamic Host Configuration Protocol version 6 (DHCPv6).

Extended Unique Identifier

An EUI address is a 64-bit hex interface identifier. If you do not specify an EUI value explicitly, the security device autogenerates it from the MAC address of the IPv6 interface. This usually happens immediately the first time you define an IPv6 interface.

A device configured for address autoconfiguration advertises an IPv6 prefix to local IPv6 hosts. The local hosts use this prefix to autoconfigure IPv6 addresses from their EUI-ID values.

NOTE: For more information about EUI, see *Guidelines for 64-Bit Global Identifier (EUI-64) Registration Authority* at <http://standards.ieee.org/regauth/oui/tutorials/EUI64.html>.

Router Advertisement Messages

A Router Advertisement (RA) is a message sent by a router to on-link hosts periodically or in response to a Router Solicitation (RS) request from another host. The autoconfiguration information in an RA includes the following:

- IPv6 prefixes of the IPv6 router, which allow the on-link hosts to access the router
- Maximum Transmission Unit (MTU), which informs the on-link hosts the maximum size (in bytes) of exchanged packets
- Specific routes to the router, which allow the on-link hosts to send packets through the router
- Whether or not to perform IPv6 address autoconfiguration and, when appropriate, a prefix list
- Period that autoconfigured addresses remain valid and preferred

Router Solicitation Messages

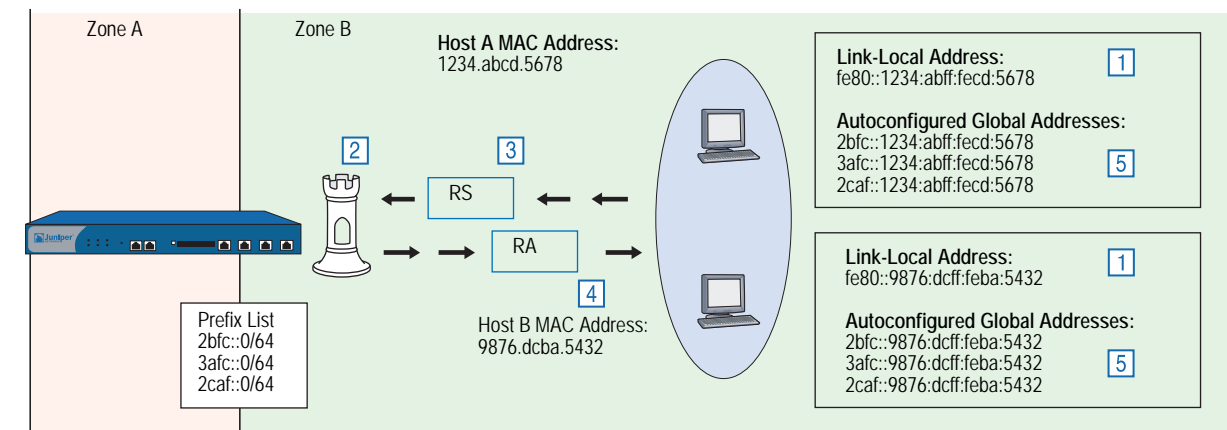
A Router Solicitation (RS) is a message sent by hosts to discover the presence and properties of on-link routers. When an IPv6 router receives an RS request from a host, it responds by transmitting an RA message back to the host. An RA announces the existence of the router and provides the host with the information it needs to perform autoconfiguration tasks.

Each RS contains the link-local address of the source host. The host derives the link-local address from its MAC address. When the IPv6 router receives the RS, it uses the link-local address to transmit an RA back to the host.

Prefix Lists

A *prefix list* is a table containing IPv6 prefixes. When entries are present in the list, the router includes them in the RAs it sends to on-link hosts. Each time a host receives an RA, it can use the prefixes to perform address autoconfiguration. Figure 4 shows Host A and Host B using three prefixes to generate unique global addresses.

Figure 4: Address Autoconfiguration



1. On startup, IPv6 Hosts A and B generate link-local addresses from their MAC addresses.
2. Each host broadcasts RS messages. Each message uses the host link-local address as the source address for the RS packets.
3. The IPv6 router receives the RS message.
4. The IPv6 router transmits confirming RA messages to the hosts. These messages contain a prefix list.
5. The hosts use the prefixes to perform autoconfiguration.

Neighbor Discovery

Neighbor Discovery (ND) is the process of tracking the reachability status for neighbors in a local link. A device views a neighbor as reachable when the device receives recent confirmation that the neighbor received and processed IP traffic or Neighbor Solicitation (NS) requests. Otherwise, it considers the neighbor unreachable. Although not explicitly required, IPv6 host might have ND enabled. An IPv6 router with ND enabled can send ND information downstream.

Neighbor Cache Table

The Neighbor Cache table contains information about neighbors to which hosts have recently sent traffic. In addition, the table tracks the current reachability status of neighbors on the local link. Each entry contains the following information:

- IPv6 address of the neighbor
- MAC address of the neighbor
- Current neighbor reachability state
- Age of the neighbor entry
- Number of packets currently queued for transmission to the destination neighbor

Table entries are keyed on the IPv6 address.

Neighbor Unreachability Detection

Neighbor Unreachability Detection (NUD) works by building and maintaining a Neighbor Cache table, which contains the address for each neighbor to which a host has recently sent traffic. The device uses these entries to record changes in the reachability status of the neighbors. NUD allows the device to track the changing reachability state of each neighbor and to make traffic-forwarding decisions.

Neighbor Entry Categories

A Neighbor Cache table entry can belong to any of four categories. The category of the entry determines how the device generates the entry initially and manages reachability states thereafter.

- **Endpoint host entries** When an endpoint host makes an initial attempt to send traffic to a neighbor, the device automatically generates a corresponding entry in the Neighbor Cache table. The device uses this entry for further communication and to track the reachability state of the endpoint host.
- **Next-hop gateway router entries** When you create a virtual routing table entry in a device for a gateway router, the device automatically generates a corresponding entry in the Neighbor Cache table. The device uses this entry for further communication and to track the reachability state of the gateway router.
- **Manual tunnel gateway interface entries** When you set up a manual IPv6 over IPv4 tunnel interface, the device automatically generates an entry for the interface. This entry has an IPv6 link-local address. The device uses this entry to monitor the reachability state of the tunnel.
 - For information about IPv6 over IPv4 tunneling, see “IPSec 6in4 Tunneling” on page 120.
 - For information about link-local addresses, see “Configuring Manual Tunneling” on page 99.
 - For information about tunnel gateway transitions, see “Tunnel Gateway State Transitions” on page 17.
- **Static entries** When you create a Neighbor Cache entry statically, the device does not use it to perform ND or NUD operations. Instead, it assigns the entry a special reachability state called Static. This enables the entry in all circumstances. While the entry exists, the device forwards any traffic sent to the represented neighbor.

Neighbor Reachability States

No Neighbor Cache entry exists for a neighbor until the device sends the neighbor an initial NS request. Until this happens, the device does not recognize the existence of the neighbor. When the device sends the initial request, it creates a table entry and sets it to the Incomplete state.

The reachability states are as follows:

- **Incomplete:** A host attempted to send traffic to a neighbor currently unknown to the device, and initial address resolution is still in progress. The device broadcasts an NS request (using a solicited node multicast address) to find the neighbor, but has not yet received a confirming Neighbor Advertisement (NA).

The Incomplete state has different characteristics when the neighbor is a next-hop gateway router. For more information, see “Next-Hop Gateway Router State Transitions” on page 16.

- **Reachable:** The device currently considers the neighbor reachable because it received a confirming NA reply from the neighbor. While the entry state is

Reachable, the device forwards any traffic sent to the neighbor. The entry state remains Reachable until the Reachable Time interval (expressed in seconds) elapses. Then the state changes to Stale.

- **Stale:** The device considers the neighbor unreachable because the Reachable Time interval has elapsed since the most recent NA from the neighbor. However, the device makes no attempt to verify reachability until a host attempts to send more traffic to the neighbor.
- **Delay:** A host attempted to send traffic to the neighbor while the state was Stale. The device makes no active attempt to verify neighbor reachability. Instead, it waits for upper-layer protocols to provide reachability confirmation. The device maintains the Delay state for five seconds. If the device receives confirmation during this delay period, the state changes to Reachable. Otherwise, the state changes to Probe.
- **Probe:** The Delay period elapsed, and the device received no confirmation from the upper-layer application. The device sends up to two unicast NS probes to verify reachability. If the device receives an NA message from the neighbor, the state changes to Reachable. Otherwise, the device deletes the reachability entry from the table. In effect, removal of a neighbor entry makes the device view the neighbor as nonexistent.
- **Probe Forever:** The device no longer considers the neighbor reachable, has made an attempt to forward traffic to the neighbor, and is sending unicast NS probes to verify reachability. The device continues to retransmit the probes indefinitely or until it receives a reachability confirmation from the neighbor.

The device uses the Probe Forever reachability state *only* when the entry represents a next-hop gateway router.

NOTE: The Neighbor Cache entry might also exist in Active and Inactive states but only when the neighbor is a manual IPv6in4 tunnel gateway interface.

How Reachability State Transitions Occur

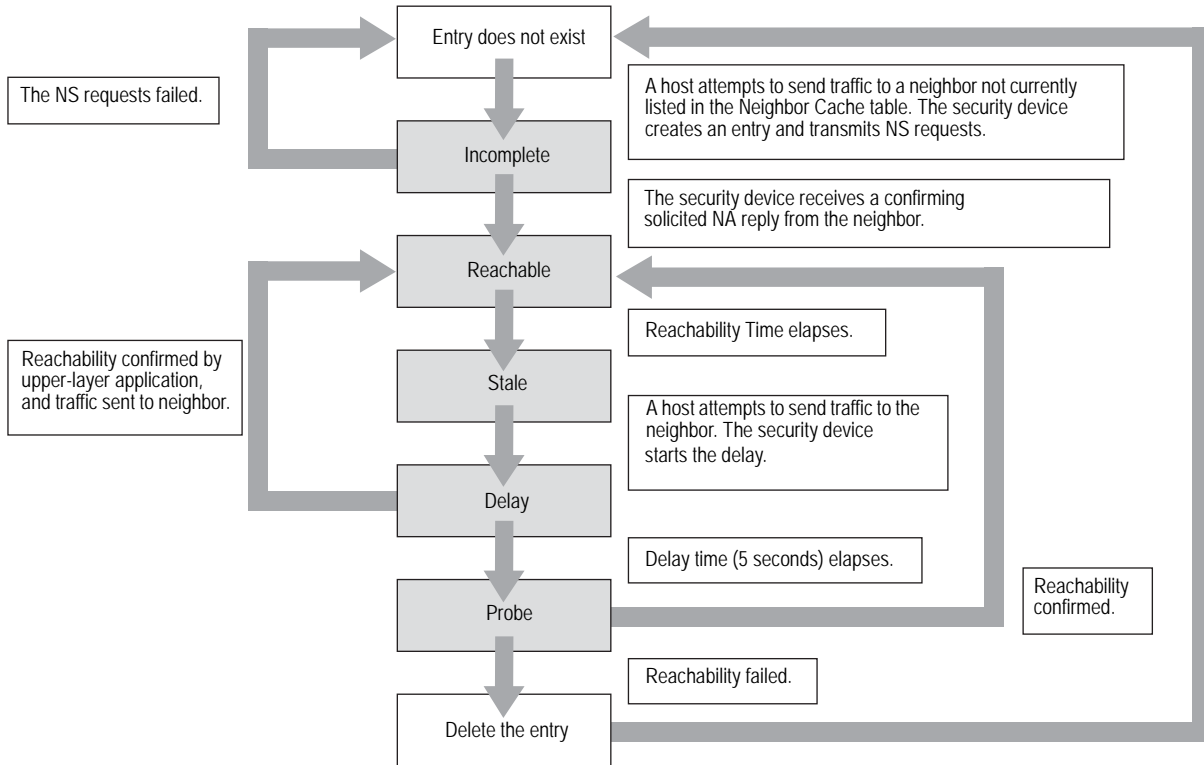
A device changes the reachability state of a Neighbor Cache entry depending on the neighbor category, the current state of the entry, and whether on-link hosts attempt to send traffic to the neighbor.

Endpoint Host State Transitions

When an on-link host attempts to send traffic to a neighbor, the device searches the Neighbor Cache table for a corresponding Neighbor Cache table entry. If no entry exists, the device broadcasts an NS message for the neighbor. It then creates a new table entry and assigns it an Incomplete state.

Figure 5 shows how the device handles reachability state transitions after it generates a Neighbor Cache entry.

Figure 5: Endpoint Host Reachability Transitions

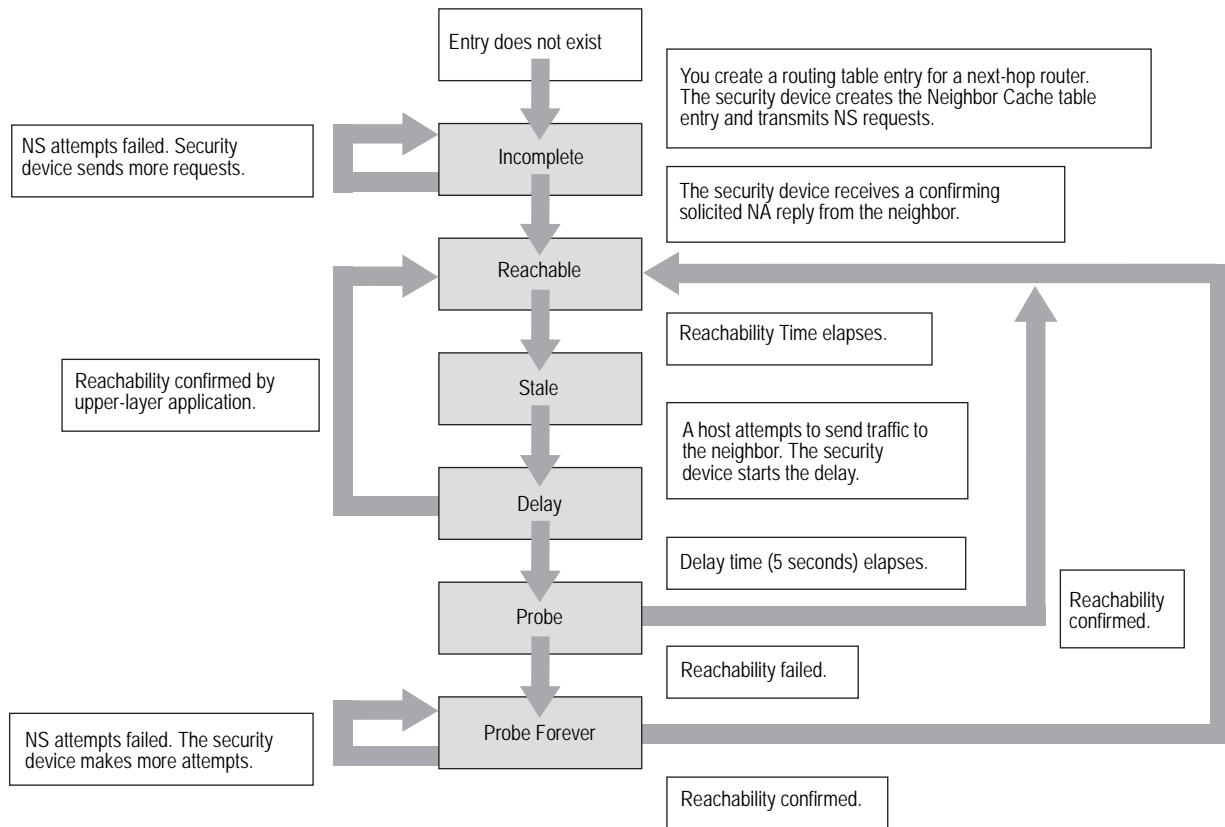


Next-Hop Gateway Router State Transitions

When you create a routing table entry to an IPv6 next-hop gateway router, the device automatically generates a corresponding Neighbor Cache table entry and assigns it an Incomplete state.

Figure 6 shows how the device handles reachability state transitions after it generates the Neighbor Cache entry for the first time.

Figure 6: Next-Hop Gateway Router Reachability Transitions



When you remove a router from which the device generated a Neighbor Cache table entry, the device deletes the entry automatically.

Tunnel Gateway State Transitions

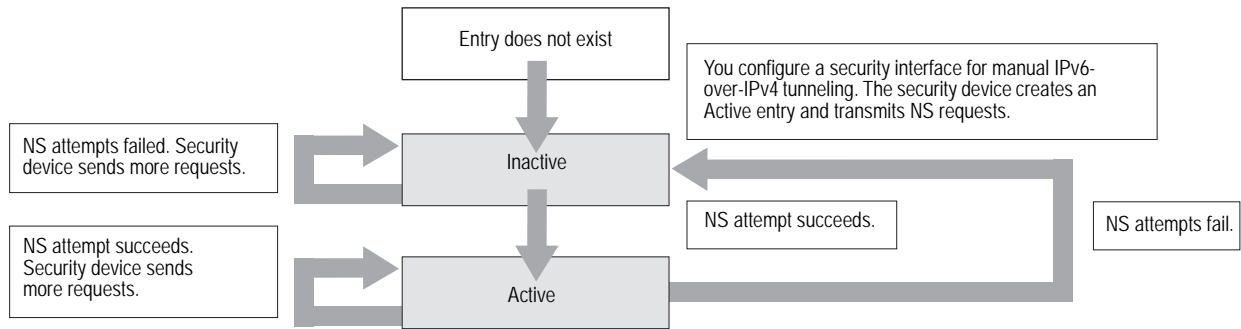
When you configure an interface for manual IPv6-in-IPv4 tunneling, the device automatically generates a corresponding entry in the Neighbor Cache table.

A device uses two entry states (also known as heartbeat states).

- **Inactive:** The device does not consider the neighbor reachable and has sent an NS message to test for reachability.
- **Active:** The device considers the neighbor reachable and periodically sends NS messages to confirm reachability.

When you create the IPv6-over-IPv4 tunnel, the device generates the Neighbor Cache entry and assigns it the Inactive state. Figure 7 shows how the device handles reachability state transitions after initial generation of the entry.

Figure 7: Tunnel Gateway State Transitions



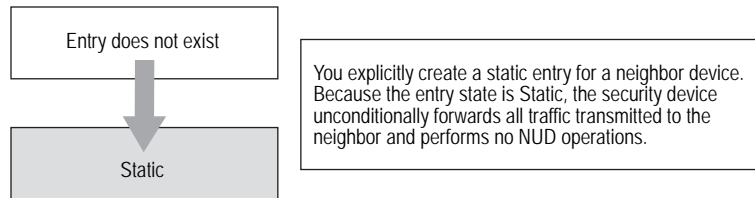
When you remove the manual tunnel from which the device generated the Neighbor Cache Table entry, the device deletes the entry automatically.

For information about manual IPv6 over IPv4 tunneling, see “IPSec Tunneling” on page 111.

Static Entry Transitions

When you create a static entry, the device always forwards traffic transmitted to the neighbor because no NUD operations apply.

Figure 8: Static Entry State Transitions



Enabling an IPv6 Environment

To set up a security device for IPv6 operation, you must first enable an IPv6 environment variable on the device. You must complete this step otherwise you cannot view IPv6 features or options in the WebUI or the CLI.

Enabling IPv6 at the Device Level

To enable a device for IPv6, you must start a CLI session with the device. You can establish CLI sessions using software that emulates a VT100 terminal, such as Telnet or Secure Command Shell (SSH). If you make a direct connection through the console port, you can use HyperTerminal. (For more information about establishing CLI sessions, see the hardware document for the security device.

To check the IPv6 status of the security device, enter the following command:

```
get envvar
```

If the device is currently IPv6-enabled, the following appears in the console output:

```
ipv6=yes
```

If this output does not appear, the device is not IPv6-enabled (default). To enable IPv6, enter the following commands:

```
set envvar ipv6=yes  
save  
reset save-config yes
```

When the confirmation prompt appears, enter **y**.

Disabling IPv6 at the Device Level

You must start a CLI session with the device by establishing a console connection with HyperTerminal or another terminal emulation software.

To disable IPv6, enter the following commands:

```
unset envvar ipv6  
save  
reset
```

When the confirmation prompt appears, enter **y**.

Configuring an IPv6 Host

After enabling the device for IPv6 operation, you can configure the device to be an IPv6 host by performing the following steps:

1. Bind the interface to a zone (such as Trust, Untrust, or a user-defined zone).
2. Enable the mode and interface.
3. Configure address autoconfiguration.
4. Configure neighbor discovery.

The following sections describe ScreenOS settings pertinent to IPv6 host configuration.

NOTE: Optionally, in addition to an IPv6 address, the security device can have an IPv4 IP address associated with the same interface. For information about IPv4 interface configuration, see *Volume 2: Fundamentals*.

Binding the IPv6 Interface to a Zone

You can bind an interface to a custom or preset security zone with the WebUI or the CLI. Interface naming varies by platform. To view the interfaces on your security device you can use the **get interface** command .

In the following example, you bind an interface named ethernet1/2 to the trust zone.

WebUI

Network > Interfaces: Select **Edit** to change the zone binding for an existing interface entry or click **New** to configure a new interface entry.

CLI

```
set interface ethernet1/2 zone trust
save
```

Enabling IPv6 Host Mode

You can enable IPv6 modes from the WebUI or the CLI. The mode options are: none (not using IPv6), host, or router.

WebUI

Network > Interfaces: Select **Edit** to change the IPv6 mode for an existing interface entry or click **New** to configure a new interface entry. Select **Host** mode.

Network > Interfaces > Edit (for ethernet1) > IPv6

CLI

```
set interface ethernet1/2 ipv6 mode host
set interface ethernet1/2 ipv6 enable
save
```

Setting an Interface Identifier

You can configure the Extended Unique Identifier (EUI) for the interface. The EUI is a 64-bit hexadecimal extension of the Ethernet Media Access Control (MAC) address. The device uses this value to autoconfigure an IPv6 link-local IP address for the interface.

WebUI

Network > Interfaces: Select **Edit** to change the zone binding for an existing interface entry, or click **New** to configure a new interface entry. Enter an **Interface ID**.

CLI

```
set interface ethernet1/2 ipv6 interface-id 0210dbffe7ac108
save
```

Configuring Address Autoconfiguration

When you define a prefix list entry for address autoconfiguration, IPv6 on-link hosts can use the prefix to generate unique IPv6 addresses. In the following example, you define prefix list entry 2bfc::0/64.

WebUI

Network > Interfaces > Edit (for IPv6 interface) > Prefix lists: Enter the following, then click **OK**:

New IPV6 Prefix/Length: 2bfc::0/64
Prefix Flags
Autonomous: (select)
Onlink (select)

CLI

```
set interface ethernet3 ipv6 ra prefix 2bfc::0/64 autonomous onlink
```

After you make this setting, the device automatically includes the prefix in any RAs sent to on-link hosts.

Configuring Neighbor Discovery

To direct the interface to learn of the existence and identity of other routers, you can enable the Accept Incoming RAs setting for the IPv6 interface. With this setting enabled, the interface accepts RA messages from other IPv6 peer devices.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

Accept Incoming Router Advertisements: (select)

CLI

```
set interface ethernet3 ipv6 ra accept
```

After you enable this setting, the interface accepts any route advertisement it receives from another host in the link. When the interface receives such an advertisement, it stores the advertised IPv6 address and MAC address in the Neighbor Cache table.

To see if the interface received and stored any advertised routes, you can view the contents of the NDP table by executing the following command:

```
get ndp
```

Configuring an IPv6 Router

To configure an IPv6 router:

1. Enable the IPv6 environment on the device.
2. Bind the interface to a zone (such as Trust, Untrust, or a user-defined zone).
3. Enable the mode and interface.
4. Configure address autoconfiguration.
5. Configure router advertisement (RA) parameters
6. Configure neighbor discovery (ND) parameters.

The following sections describe ScreenOS settings pertinent to IPv6 router configuration.

NOTE: Optionally, in addition to an IPv6 address, the security device can have an IPv4 IP address associated with the same interface. For information about IPv4 interface configuration, see *Volume 2: Fundamentals*.

Binding the IPv6 Interface to a Zone

You can bind an interface to a custom or preset security zone with the WebUI or the CLI. Interface naming varies by platform.

In the following example, you bind an interface named ethernet1/2 to the trust zone.

WebUI

Network > Interfaces: Select **Edit** to change the zone binding for an existing interface entry or click **New** to configure a new interface entry.

CLI

```
interface ethernet1/2 zone trust
save
```

Enabling IPv6 Router Mode

You can enable IPv6 modes from the WebUI or the CLI. The mode options are: none (not using IPv6), host, or router.

WebUI

Network > Interfaces: Select **Edit** to change the IPv6 mode for an existing interface entry or click **New** to configure a new interface entry. Select **Router** mode.

```
Network > Interfaces > Edit (for ethernet1) > IPv6
```

CLI

```
set interface ethernet1/2 ipv6 mode router
set interface ethernet1/2 ipv6 enable
save
```

Setting an Interface Identifier

An IPv6 interface identifier sets the Extended Unique Identifier (EUI) for the interface. The EUI is a 64-bit hexadecimal extension of the Ethernet Media Access Control (MAC) address. The device uses this value to autoconfigure an IPv6 link-local IP address for the interface.

WebUI

Network > Interfaces: Select **Edit** to change the zone binding for an existing interface entry, or click **New** to configure a new interface entry. Enter an **Interface ID**.

CLI

```
set interface ethernet1/2 ipv6 interface-id 0210dbffe7ac108
save
```

Setting Address Autoconfiguration

To set host autoconfiguration for an IPv6 router, you must do all of the following:

- Enable the Outgoing Router Advertisements setting.
- Disable the Managed Configuration Flag.
- Disable the Other Parameters Configuration Flag.

Outgoing Router Advertisements Flag

Enabling the Outgoing Router Advertisements flag allows the interface to send Router Advertisement (RA) messages to on-link hosts. After enabling this setting, the interface immediately broadcasts a route advertisement to hosts in the link. It also broadcasts an RA automatically when it receives a Router Solicitation (RS) from a host or when you change any RA setting on the interface.

In the following example, you enable the Allow RA Transmission setting.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Allow RA Transmission: (select)

CLI

```
set interface ethernet3 ipv6 ra transmit
```

Managed Configuration Flag

Enabling the Managed Configuration flag directs local hosts to use a stateful address autoconfiguration protocol, such as DHCPv6, to generate host addresses.

Local hosts cannot perform stateless address autoconfiguration while this setting is enabled.

In the following example, you disable the Managed Configuration flag to allow autoconfiguration.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Managed Configuration Flag: (deselect)

CLI

```
set interface ethernet3 ipv6 ra managed
```

Other Parameters Configuration Flag

Enabling the Other Parameters Configuration flag directs local hosts to use a stateful address autoconfiguration protocol (DHCPv6) to configure parameters other than host addresses.

Local hosts cannot perform stateless address autoconfiguration while this setting is enabled.

In the following example, you disable the Other Parameters Configuration flag to allow autoconfiguration.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Other Parameters Configuration Flag: (deselect)

CLI

```
set interface ethernet3 ipv6 ra other
```

Disabling Address Autoconfiguration

To disable host autoconfiguration for an IPv6 router, you must do the following:

- Disable the Outgoing Router Advertisements setting so that on-link host can't send router advertisements.
- Enable the Managed Configuration Flag to force the hosts to use a stateful addressing protocol, such as DHCPv6.
- Enable the Other Parameters Configuration Flag.

In the following example, you disable address autoconfiguration on an IPv6 router.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Allow RA Transmission: (deselect)
Managed Configuration Flag: (select)
Other Parameters Configuration Flag: (select)

CLI

```
unset interface ethernet3 ipv6 ra transmit
unset interface ethernet3 ipv6 ra managed
unset interface ethernet3 ipv6 ra other
```

Setting Advertising Time Intervals

Address autoconfiguration uses several advertised time interval parameters for IPv6 routers. These intervals determine the frequency of events or the lifetime of identified objects.

Advertised Reachable Time Interval

Enabling the Reachable Time setting instructs the interface to include the Reachable Time interval in outgoing RA messages. This interval tells on-link hosts how long in seconds to consider the IPv6 interface reachable after they receive an RA from the interface.

The interface bases the Reachable Time interval on the current Base Reachable Time setting. For information about the Base Reachable Time, see “Base Reachable Time” on page 30.

The interface uses this value while performing Neighbor Unreachability Detection (NUD). The security device builds and maintains a Neighbor Cache table, which contains the address for each neighbor to which a host has recently sent traffic. The device uses these entries to record changes in the reachability status of the neighbors. NUD allows the device to track the changing reachability state of each neighbor and to make traffic-forwarding decisions accordingly.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

Reachable Time: (select)

CLI

```
set interface ethernet3 ipv6 ra reachable-time
```

Advertised Retransmit Time Interval

Enabling the Retransmission Time instructs the interface to include the Retransmission Time interval in outgoing RA messages. This interval (expressed in seconds) is the time that elapses between retransmissions of NS messages.

For information about the Retransmission Time interval, see “Retransmission Time” on page 31.

The interface uses this value while performing NUD.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Retransmission Time: (select)

CLI

```
set interface ethernet3 ipv6 ra retransmit-time
```

Maximum Advertisement Interval

The Maximum Advertisement interval specifies the maximum number of seconds allowed between transmission of unsolicited multicast RAs from the IPv6 interface.

In the following example, you set the interval to 500 seconds.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Maximum Advertisement Interval: 500

CLI

```
set interface ethernet3 ipv6 ra max-adv-int 500
```

Minimum Advertisement Interval

The Minimum Advertisement interval setting specifies the minimum number of seconds allowed between transmission of unsolicited multicast RAs from the IPv6 interface.

In the following example, you set the interval to 100 seconds.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Minimum Advertisement Interval: 100

CLI

```
set interface ethernet3 ipv6 ra min-adv-int 100
```

Advertised Default Router Lifetime

The Default Router Lifetime setting specifies the number of seconds that hosts can identify the interface to be the default router, after the hosts receive the last RA from the interface.

In the following example, you set the lifetime to 1500 seconds.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Minimum Advertisement Interval: 1500

CLI

```
set interface ethernet3 ipv6 ra default-life-time 1500
```

Advertising Packet Characteristics

An RA can provide on-link host devices with information about packets exchanged through the IPv6 interface, including the link MTU and the hop limit.

Link MTU Value

Enabling the Link MTU flag directs the IPv6 interface to include the Link MTU field in RA messages. The Link MTU is the maximum size (in bytes) of any IPv6 packet sent by a host over a link.

NOTE: The default Link MTU value for Ethernet is 1500 bytes. The default for PPPoE is 1490 bytes. This value must be from 1280 to 1500. You can change the MTU value for some platforms. See the hardware documentation for your security device to see if you can configure the MTU value.

In the following example, you enable advertisement of the link MTU.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Link MTU: (select)

CLI

```
set interface ethernet3 ipv6 ra link-mtu
```

Current Hop Limit

The Current Hop Limit setting specifies the hop limit for packets sent by any local IPv6 host that uses RAs from this interface for address autoconfiguration. Setting the Current Hop Limit value to zero denotes an unspecified number of hops.

In the following example, you set the Current Hop Limit value to 50 hops.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

Current Hop Limit: 50

CLI

```
set interface ethernet3 ipv6 ra hop-limit 50
```

Advertising Router Characteristics

An IPv6 interface can use RAs to inform on-link hosts of router attributes, including the Link Layer (MAC) address of the interface and the router preference level.

Link Layer Address Setting

Enabling the Link Layer Address flag directs the IPv6 interface to include the Link Layer (MAC) address of the interface in outgoing RA messages.

In the following example, you enable the Link Layer Address flag.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

Link Layer Address: (select)

CLI

```
set interface ethernet3 ipv6 ra link-address
```

Advertised Router Preference

The Advertised Router Preference setting specifies the preference level for the router.

When a host receives the RA from the IPv6 interface, and other IPv6 routers are present, the preference level determines whether the host views the interface as a primary router or a secondary router.

In the following example, you set the preference to High, which designates the interface as the primary router.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

Advertised Router Preference: High (select)

CLI

```
set interface ethernet3 ipv6 ra preference high
```

Configuring Neighbor Discovery Parameters

When you configure an interface for IPv6 operation, the interface must locate and confirm the existence of neighbors on the same link as the interface. Neighbor Discovery (ND) allows an device to track the reachability status for neighbors in a local link.

Neighbor Unreachability Detection

Enabling Neighbor Unreachability Detection (NUD) directs the interface to perform NUD. Each entry in the table represents a neighbor and contains the current reachability status of the neighbor.

In the following example, you enable NUD for IPv6 interface ethernet3.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click OK:

NUD (Neighbor Unreachability Detection): (select)

CLI

```
set interface ethernet3 ipv6 nd nud
```

MAC Session-Caching

By default, the interface uses MAC address session-caching to increase the speed and efficiency of address resolution. Enabling the always-on-dest setting instructs the interface to bypass this process. Before transmitting traffic to a neighbor, the interface always consults the Neighbor Cache table instead of caching the MAC address in the session.

To bypass MAC session caching, enter the following CLI command:

```
set ndp always-on-dest
```

Static Neighbor Cache Entries

To create a static entry in the Neighbor Cache table, set the Neighbor Discovery Parameter (NDP) using the CLI command **set ndp**.

```
set ndp ip_addr mac_addr interface
```

where:

- *ip_addr* is the neighbor IPv6 address.
- *mac_addr* is the neighbor MAC address.
- *interface* is the device interface.

In the following example, you start a new entry in the Neighbor Cache table.

- Neighbor IP address 32f1::250:af:34ff:fe27
- MAC address 1234abcd1234
- Device interface ethernet3

```
set ndp 32f1::250:af:34ff:fe27 1234abcd1234 ethernet3
```

To delete the entry, enter the **unset ndp ip_addr interface** command.

Base Reachable Time

When an IPv6 interface transmits a Neighbor Solicitation (NS) message to a neighbor and receives a Neighbor Advertisement (NA) message in reply, the device sets the neighbor reachability status to Reachable. The Base Reachable Time setting specifies the approximate length of time (expressed in seconds) that the interface maintains the Reachable status.

After this time interval passes, the status goes to Stale mode.

NOTE: The Base Reachable Time setting only specifies an approximation of the actual time that the status remains Reachable. The exact time interval is called Reachable Time. The interface determines Reachable Time randomly, using the Base Reachable Time as a baseline value. The resulting Reachable Time is usually within 50 to 150 percent of the Base Reachable Time setting.

In the following example, you set the Base Reachable Time (for IPv6 interface ethernet3) to 45 seconds.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

Base Reachable Time: 45

CLI

```
set interface ethernet3 ipv6 nd base-reachable-time 45
```

Probe Time

While an entry status is Incomplete or Probe Forever (as when the neighbor is a next-hop gateway), the interface attempts to confirm the reachability of the neighbor. Each attempt is called a *probe*. During a probe, the interface transmits NS requests to the neighbor. The endpoint host state mode Probe Time setting specifies the interval of time (expressed in seconds) between probes.

In the following example, you set the Probe Time (for IPv6 interface ethernet3) to 3 seconds.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

Retransmission Time: 3

CLI

```
set interface ethernet3 ipv6 nd probe-time 3
```

Retransmission Time

When an IPv6 interface begins a probe, it transmits NS requests to the neighbor. The Retransmission Time setting specifies the time interval (expressed in seconds) that elapses between NS requests.

The following example sets the Retransmission Time (for IPv6 interface ethernet3) to 2 seconds.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

Retransmission Time: 2

CLI

```
set interface ethernet3 ipv6 nd retransmit-time 2
```

Duplicate Address Detection Retry Count

Duplicate Address Detection (DAD) determines if more than one on-link device has the same unicast address. The DAD Retry Count setting specifies the number of consecutive NS messages to send while performing DAD for the IPv6 interface.

The following example sets the DAD Retry Count (for IPv6 interface ethernet3) to 2 retries.

WebUI

Network > Interfaces > Edit (for IPv6 interface): Enter the following, then click **OK**:

DAD (Duplicate Address Detection) Retry Count: 2

CLI

```
set interface ethernet3 ipv6 nd dad-count 2
```

Viewing IPv6 Interface Parameters

You can view IPv6 configuration information in the WebUI or the CLI.

WebUI

Network > Interfaces > IPv6

CLI

get interface

Viewing Neighbor Discovery Configurations

You can view the current Neighbor Discovery (ND) settings using the WebUI or the CLI. The following example displays the current ND settings for an IPv6 interface (ethernet3).

WebUI

Network > Interfaces > Edit (for IPv6 interface)

CLI

get interface ethernet3 ipv6 config

Viewing the Current RA Configuration

Before configuring an IPv6 interface to function as an IPv6 router, we recommend that you check the current RA configuration.

In the following example, you display the RA configuration settings for IPv6 interface ethernet3.

WebUI

Network > Interfaces > Edit (for IPv6 interface ethernet3): View the current settings.

CLI

get interface ethernet3 ipv6 ra

Configuration Examples

This section contains examples of IPv6 router and host configuration. Only CLI commands are shown.

IPv6 Router

This example shows a security device with two interfaces that operate in IPv6 Router mode:

CLI

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 111111111111111111
set interface ethernet2 ipv6 ip 2eee::1/64
set interface ethernet2 ipv6 ra transmit

set interface ethernet3 zone trust
set interface ethernet3 ipv6 mode router
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 interface-id 222222222222222222
set interface ethernet3 ipv6 ip 3eee::1/64
set interface ethernet3 ipv6 ra transmit
```

IPv6 Host

The following commands configure an IPv6 host:

CLI (Device B)

```
set interface ethernet2 zone trust
set interface ethernet2 ip 20.1.1.2/24
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 333333333333333333
set interface ethernet2 ipv6 ra accept
```


Chapter 3

Connection and Network Services

IPv6-enabled interfaces support Dynamic Host Configuration Protocol version 6 (DHCPv6) and Point-to-Point Protocol over Ethernet (PPPoE).

This chapter contains the following sections:

- “Overview” on page 36
- “Dynamic Host Configuration Protocol Version 6” on page 36
 - “Device-Unique Identification” on page 36
 - “Identity Association Prefix Delegation-Identification” on page 37
 - “Prefix Features” on page 37
 - “Server Preference” on page 38
 - “Configuring a DHCPv6 Server” on page 38
 - “Configuring a DHCPv6 Client” on page 40
 - “Viewing DHCPv6 Settings” on page 41
- “Configuring Domain Name System Servers” on page 42
 - “Requesting DNS and DNS Search List Information” on page 43
 - “Setting Proxy DNS Address Splitting” on page 44
- “Configuring PPPoE” on page 46
- “Setting Fragmentation” on page 47

Overview

This chapter explains the following network and connection services:

- Dynamic Host Configuration Protocol version 6 (DHCPv6)
- Domain Name System (DNS)
- Point-to-Point over Ethernet (PPPoE)
- Fragmentation

The next sections provide an overview of each network or connection service.

Dynamic Host Configuration Protocol Version 6

An IPv6 router can only be a DHCPv6 server. An IPv6 host can only be a DHCP client.

As a DHCPv6 client, the interface can request (from a DHCPv6 server):

- Delegation of long-lived prefixes across an administrative boundary. The server does not have to know the topology of the targeted local network. For example, an ISP can use DHCPv6 to assign prefixes to downstream networks through downstream DHCP clients. To speed up the client/server interaction, the client can request rapid commit (if enabled). Rapid commit reduces the number of messages from four to two.

NOTE: For more information about the impact of using rapid commit options, refer to RFCs 3315 and 3633.

- IP addresses of available DNS servers. The interface can also request DNS search-list information. This list contains partial domain names, which assist DNS searches by concatenating entered usernames to the domain names.

As a DHCPv6 server, the interface can provide both of these services to a DHCPv6 client. To speed up prefix delegation, an IPv6 router configured to be a DHCPv6 server can support a rapid commit option. You can also set a server preference option.

NOTE: DHCPv6 is complementary to address autoconfiguration. The services are not interchangeable.

Device-Unique Identification

A Device-Unique Identification (DUID) identifies a network device such as a host or a router. You can use the DUID (or a name associated with the DUID) to determine how the device performs prefix delegation for a particular router or host that has a particular DUID.

Identity Association Prefix Delegation-Identification

An Identity Association Prefix Delegation-Identification (IAPD-ID) is a positive integer that identifies a certain prefix on the server. The client can use this value to request a specific prefix from the server. If you specify a nonzero IAPD-ID value, it maps statically to a single prefix that has the same IAPD-ID on the server. If you specify a zero IAPD-ID value or omit the value (which assigns it a zero value by default) the IAPD-ID maps it dynamically to a pool of prefixes on the server. The device treats all prefixes with a zero IAPD-ID as belonging to this pool.

A DHCPv6 server can contain up to 16 prefixes per client DUID.

Prefix Features

Prefixes delegated by a DHCPv6 server contain the following elements:

- The Top-Level Aggregator (TLA) identifies the highest level in the routing hierarchy (the left most portion of the IPv6 address). The TLA usually specifies owned address space for an organization, such as an ISP. You specify the length of the TLA using the prefix length.
- The Site-Local Aggregator (SLA) identifies a network or subnet used by the organization.

For example, a server might delegate the following prefix:

2001:908e:1::/48

where:

- TLA is 2001:908e
- SLA is 1

The same server might also delegate the following prefix:

2001:908e:2::/48

where:

- TLA is 2001:908e
- SLA is 2

The SLAs differ in each example, which means that each prefix maps to a different network but belongs to the same owned address space.

Server Preference

You can optionally send a server preference value in the DHCP advertise messages that the device sends to DHCPv6 clients. A DHCPv6 client uses this value to choose one server instead of another.

The default value is -1 and indicates that no preference appears in the advertise messages sent to clients. You can set the value from 0 (lowest priority) to 255 (highest priority).

In the following example, you set the server preference for interface ethernet1/2 to 255, the highest priority.

WebUI

Network > DHCPV6 > Edit (Server): Enter 255 in the Server Preference Number field, then click **Apply**.

CLI

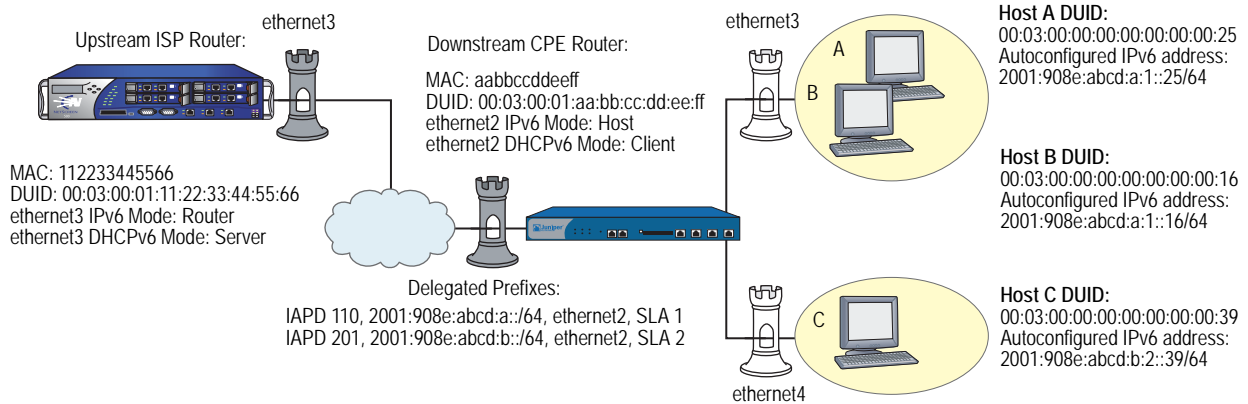
```
set interface ethernet1/2 dhcp6 server preference 255
```

Configuring a DHCPv6 Server

In environments, such as an ISP, where you allow end users access to outside networks, you can set up security devices to delegate different prefixes with different SLAs. Figure 9 shows an upstream IPv6 router set to delegate DHCPv6 prefixes to downstream routers.

The IPv6 router delegates two prefixes, each with a different SLA. Downstream CPE routers are able to use the two different SLAs to assign prefixes to two different networks.

Figure 9: DHCPv6 Prefix Delegation



In this example, you configure interface ethernet3 as a DHCPv6 server and specify two prefixes on the upstream ISP router. The prefixes are then available for delegation to a downstream Customer Premises Equipment (CPE) router.

- Specifies downstream client DUID 00:03:00:01:aa:bb:cc:dd:ee:ff. This allows the client to request prefix delegation from the server.
- Delegates the following prefixes:
 - 2001:908e:abcd:a::/64, SLA 1, IAPD 110, preferred lifetime 259200 seconds, valid lifetime 345600 seconds
 - 2001:908e:abcd:b::/64, SLA 2, IAPD 201, preferred lifetime 172800 seconds, valid lifetime 345600 seconds
- Because of the nonzero IAPD numbers, the server delegates both prefixes statically (not dynamically).

NOTE: The WebUI section lists only the navigational paths to the device configuration pages. For specific values, see the CLI section that follows it.

WebUI

1. Server Interface

Network > Interfaces > Edit (for ethernet3) > IPv6

Network > DHCPV6 > Edit (for ethernet3)

2. Downstream Client Identification

Network > DHCPv6 > DUID/Prefix Delegation List (for ethernet3) > New DUID

3. Delegated Prefixes

Network > DHCPv6 > DUID/Prefix Delegation List (for ethernet3) > Add Prefix Entry

CLI

1. Server Interface

```
set interface ethernet3 ipv6 mode router
set interface ethernet3 ipv6 enable
set interface ethernet3 dhcp6 server
```

2. Downstream Client Identification

```
set interface ethernet3 dhcp6 server options client-duid
00:03:00:01:aa:bb:cc:dd:ee:ff
```

3. Delegated Prefixes

```
set interface ethernet3 dhcp6 server options pd duid
00:03:00:01:aa:bb:cc:dd:ee:ff iapd-id 110 prefix 2001:908e:abcd:a::/64
259200 345600
set interface ethernet3 dhcp6 server options pd duid
00:03:00:01:aa:bb:cc:dd:ee:ff iapd-id 201 prefix 2001:908e:abcd:b::/64
172800 345600
save
```

Configuring a DHCPv6 Client

In the following example, you configure a downstream CPE router to be a DHCPv6 client. The router can then request prefixes from the upstream router, and automatically push IPv6 addresses derived from the prefixes to local hosts. See Figure 9.

- Specifies preferred server with DUID 00:03:00:01:11:22:33:44:55:66
- Requests the following preferred prefixes from the server:
 - 2001:908e:abcd:a::/64, SLA 1, IAPD 110, preferred lifetime 259200 seconds (3 days), valid lifetime 345600 seconds (4 days)
 - 2001:908e:abcd:b::/64, SLA 2, IAPD 201, preferred lifetime 172800 seconds (2 days), valid lifetime 345600 seconds (4 days)
- Designates ethernet3 and ethernet4 as downstream interfaces.
 - Prefixes with SLA of 1 go to ethernet3
 - Prefixes with SLA of 2 go to ethernet4

NOTE: The WebUI section lists only the navigational paths to the device configuration pages. For specific values, see the CLI section that follows it.

WebUI

1. Downstream CPE Interfaces

Network > Interfaces > Edit (for ethernet4)

Network > Interfaces > Edit (for ethernet4) > IPv6

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

2. Client Interface (to Upstream Router)

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > DHCPv6 > Edit (for ethernet2)

3. Preferred Prefixes (from Upstream Router)

Network > DHCPv6 > Prefix Assignment List (for ethernet2) > Learned (Suggested) Prefix add

4. Downstream Interfaces

Network > DHCPv6 > Prefix Assignment List (for ethernet2) > Prefix Distribution add

CLI

1. Downstream CPE Interfaces

```
set interface ethernet4 zone trust
set interface ethernet4 ipv6 mode router
set interface ethernet4 ipv6 enable
set interface ethernet3 zone trust
set interface ethernet3 ipv6 mode router
set interface ethernet3 ipv6 enable
```

2. Client Interface (to Upstream Router)

```
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 dhcp6 client
set interface ethernet2 dhcp6 client options request pd
set interface ethernet2 dhcp6 client prefer-server
    00:03:00:01:11:22:33:44:55:66
```

3. Preferred Prefixes (from Upstream Router)

```
set interface ethernet2 dhcp6 client pd iapd-id 110 prefix 2001:908e:abcd:a::/64
    259200 345600
set interface ethernet2 dhcp6 client pd iapd-id 201 prefix 2001:908e:abcd:b::/64
    172800 345600
```

4. Downstream Interfaces

```
set interface ethernet2 dhcp6 client pd iapd-id 110 ra-interface ethernet3 sla-id 1
    sla-len 2
set interface ethernet2 dhcp6 client pd iapd-id 201 ra-interface ethernet4 sla-id 2
    sla-len 2
set interface ethernet2 dhcp6 client enable
save
```

Viewing DHCPv6 Settings

You can view DHCPv6 server and client settings from the command line interface.

To view DHCPv6 settings, enter the following command:

```
get interface interface_name dhcp6
```

Sample output:

```
device-> get interface ethernet1/2 dhcp6
DHCPv6 Configuration, enabled.
```

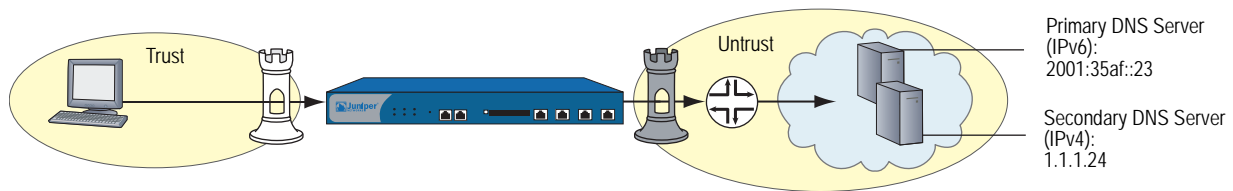
```
-----
Mode           : server
DUID           : 00:03:00:01:00:10:db:7a:c1:08
Interface      : ethernet1/2
Allow rapid-commit: yes
Preference     : 255
-----
```

Configuring Domain Name System Servers

In order to use Domain Name System (DNS) for domain name-to-address resolution, you must configure the device by entering an IP address for a primary and secondary DNS server. The DNS servers you choose to configure might use IPv4 or IPv6 addresses.

Figure 10 shows a primary and secondary DNS server in the Untrust zone behind a router.

Figure 10: Domain Name System Servers



In the following example, you designate an IPv6 server as the primary server and an IPv4 server as the secondary server.

- Primary server residing at IPv6 address 2001:35af::23
- Secondary server residing at IPv4 address 1.1.1.24
- DNS refresh every day at 11:00pm

WebUI

Network > DNS > Host

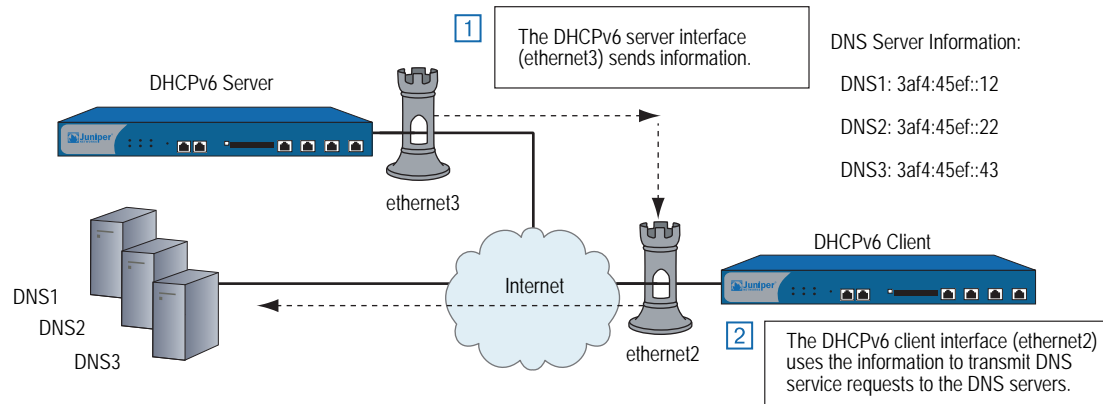
CLI

```
set dns host dns1 2001:35af::23
set dns host dns2 1.1.1.24
set dns host schedule 23:00
save
```

Requesting DNS and DNS Search List Information

Figure 11 shows a DHCPv6 server and a DHCPv6 client. The DHCPv6 client receives information about the DNS devices from interface ethernet3 on the DHCPv6 server. This interface automatically transmits information about the primary, secondary, and tertiary DNS servers. The DHCPv6 client can then send DNS service requests to any of the DNS servers.

Figure 11: DNS Servers and DHCPv6 Client



WebUI (Server)

Network > Interfaces > Edit (for ethernet3) > IPv6

Network > DHCPv6 > Edit (for ethernet3)

WebUI (Client)

Network > Interfaces > Edit (for ethernet3) > IPv6

Network > DHCPv6 > Edit (for ethernet3)

CLI (Server)

```
set interface ethernet3 ipv6 mode router
set interface ethernet3 ipv6 enable
set interface ethernet3 dhcp6 server
set interface ethernet3 dhcp6 server enable
set interface ethernet3 dhcp6 server options dns dns1 3af4:45ef::12
set interface ethernet3 dhcp6 server options dns dns2 3af4:45ef::22
set interface ethernet3 dhcp6 server options dns dns3 3af4:45ef::43
save
```

CLI (Client)

```
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 dhcp6 client
set interface ethernet2 dhcp6 client enable
set interface ethernet2 dhcp6 client options request dns
set interface ethernet2 dhcp6 client options request search-list
save
```

Setting Proxy DNS Address Splitting

ScreenOS supports proxy DNS address splitting, by which you direct selected domain name queries to specific DNS servers. Address splitting has two advantages:

- Reduces overhead. You can relieve DNS servers from processing irrelevant service requests.
- Protects traffic. You can perform IPsec tunneling on queries sent to DNS servers.

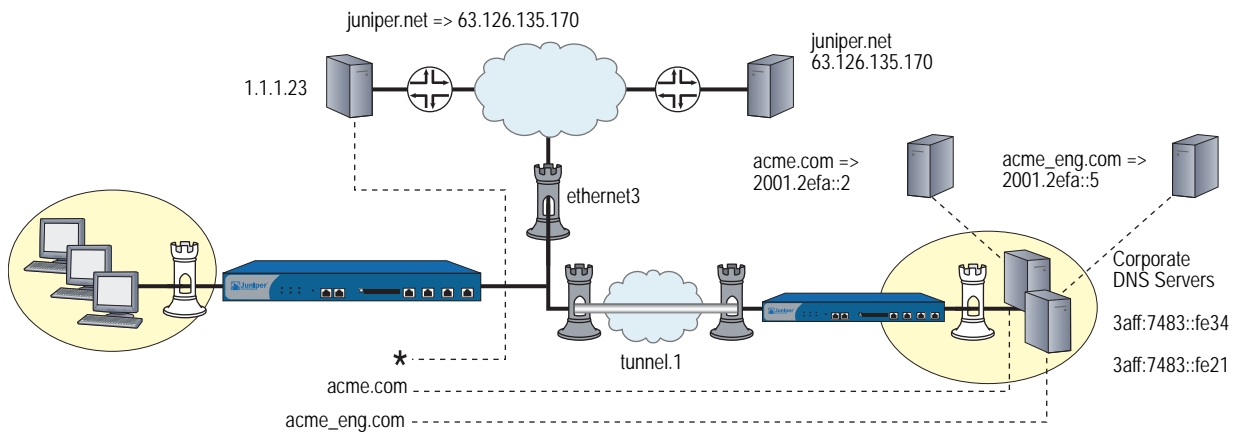
The DNS proxy selectively redirects the DNS queries to specific DNS servers, according to partial or complete domain names. This is useful when VPN tunnels or PPPoE virtual links provide multiple network connectivity, and it is necessary to direct some DNS queries to one network while directing other queries to another network.

Some advantages of a DNS proxy are as follows:

- Domain lookups are usually more efficient. For example, DNS queries meant for the corporate domain (such as marketing.acme.com) could go to the corporate DNS server exclusively, while all others go to the ISP DNS server, which reduces the load on the corporate server. In addition, this can prevent corporate domain information from leaking into the Internet.
- DNS proxy allows you to transmit selected DNS queries through a tunnel interface, which prevents malicious users from learning about internal network configuration. For example, DNS queries bound for the corporate server can pass through a tunnel interface and use security features such as authentication, encryption, and anti-replay.

In the following example, you create two proxy-DNS entries that selectively forward DNS queries to different servers. See Figure 12.

Figure 12: Proxy DNS Using Split Servers



- Any DNS query with an FQDN containing the domain name `acme.com` goes out through tunnel interface `tunnel.1` to the corporate DNS server at IPv6 address `3aff:7483::fe34`.

For example, if a host sends a DNS query to the `www.acme.com`, the device automatically directs the query to this server. (For this example, assume that the server resolves the query to IPv6 address `2001.2efa::2`.)

- Any DNS query with an FQDN containing the domain name `acme_engineering.com` goes out through tunnel interface `tunnel.1` to the DNS server at IPv6 address `3aff:7483::fe21`.

For example, if a host sends a DNS query to the `intranet.acme_eng.com`, the device directs the query to this server. (For this example, assume that the server resolves the query to IPv6 address `2001.2efa::5`.)

- All other DNS queries (denoted by an asterisk) bypass the corporate servers and go out through interface `ethernet3` to the DNS server at IPv4 address `1.1.1.23`.

For example, if the host and domain name is `www.juniper.net`, the device automatically bypasses the corporate servers and directs the query to this server, which resolves the query to IPv4 address `63.126.135.170`.

WebUI

Network > DNS > Proxy

Network > DNS > Proxy > New

CLI

```
set dns proxy
set dns proxy enable
set dns server-select domain .acme.com outgoing-interface tunnel.1
  primary-server 3aff:7483::fe34
set dns server-select domain .acme_eng.com outgoing-interface tunnel.1
  primary-server 3aff:7483::fe21
set dns server-select domain * outgoing-interface ethernet3 primary-server
  1.1.1.23
```

Configuring PPPoE

An IPv6-enabled interface can be a Point-to-Point Protocol over Ethernet (PPPoE) client, which allows members of an IPv6 Ethernet LAN to make individual PPP connections with their ISP. As with IPv4 implementations, the device encapsulates each outgoing IP packet within a PPP payload, then encapsulates the PPP payload inside a PPPoE payload. PPPoE allows devices to operate compatibly on digital subscriber lines (DSL), Ethernet Direct, and cable networks run by ISPs using PPPoE for their clients' Internet access.

You set PPPoE at the interface level and then enable the interface. The interface negotiates with an access concentrator for an Internet Protocol Control Protocol (IPCP) prefix, then an IPv6 Control Protocol (IPv6CP) prefix.

NOTE: For more information about PPPoE in dual-stack IPv6 environments, see RFC 4241.

Setting up an interface for PPPoE (dual-stack mode) consists of the following steps:

1. Create a PPPoE instance with username and password.
2. Specify that the PPPoE instance obtains IPv4 and IPv6 prefixes from the PPPoE access concentrator (AC).
3. Configure the device for IPv6 Host mode.

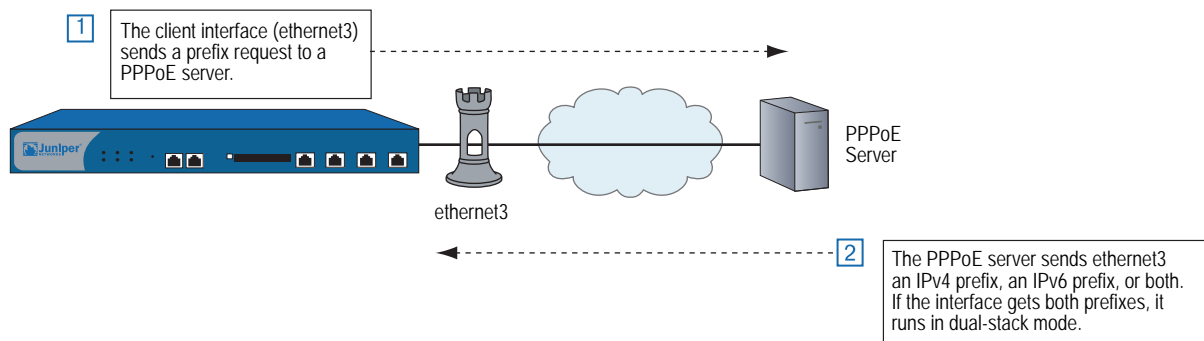
After you enable PPPoE for the interface, the interface negotiates with an AC for an IPCP prefix, then an IPv6CP prefix. Either or both of these attempts could be successful. If both are successful, the interface operates in dual-stack mode. Otherwise, it operates in single-stack mode.

In the following example, you configure interface ethernet3 for PPPoE.

- PPPoE instance named NY_Office and bind it to ethernet3
- PPPoE username Richard_B, password er5cmdj
- PPPoE instance obtains an IPv4 address first then an IPv6 address.

Figure 13 shows a PPPoE client and server exchange.

Figure 13: PPPoE Client and Server



NOTE: The WebUI section lists only the navigational paths to the device configuration pages. For specific values, see the CLI section that follows it.

WebUI

1. PPPoE Instance

Network > PPPoE > New

2. Client Interface

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

CLI

1. PPPoE Instance

```
set pppoe name NY_Office username richard_b password er5cmdj
set pppoe name NY_Office ppp ipcp ipv6cp
set pppoe name NY_Office interface ethernet3
```

2. Client Interface

```
set interface ethernet3 zone untrust
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
```

Setting Fragmentation

An IPv6 router running ScreenOS uses the Path MTU in combination with the MTU to determine fragmentation.

NOTE: For more information about Path MTU for IPv6, see RFC 1981.

An IPv6 interface initially determines the Path MTU to equal the MTU of the first hop in the path. If any packet sent on that path is too large to be forwarded by a router along the path, that router discards the packet and returns an Internet Control Message Protocol version 6 (ICMPv6) Packet Too Big (PTB) message to the source router. Upon receipt of a PTB message, the source router reduces its Path MTU to match the MTU reported in the PTB message. This process might occur several times before the Path MTU discovery process ends because the packet might have to pass through other routers with smaller MTUs further along the path.

The device applies the Path MTU to traffic going through the device and to traffic originating in the device. For through traffic, if any fragmentation is required before forwarding a packet out of an interface at the flow level, the interface sends back an ICMPv6 PTB message with the MTU set to the outgoing interface's MTU.

NOTE: For devices that do not allow you to configure Path MTU, the value is 1500.

For fragmentation to occur, you must enable Path MTU for the interface. For devices that accept a configured value, you can set a value from 1280 to 1500.

WebUI

Network > Interfaces > Edit

In the Maximum Transfer Unit (MTU) Admin MTU: Enter the MTU value, then click **OK**.

CLI

```
set interface trust trust-vr mtu 1280
save
```

NOTE: For advanced configuration examples for manual tunneling with fragmentation enabled, see “Manual Tunneling with Fragmentation Enabled” on page 124.

Chapter 4

Static and Dynamic Routing

ScreenOS supports static routing and dynamic routing with Routing Information Protocol next-generation (RIPng).

This chapter contains the following sections:

- “Overview” on page 50
 - “Dual Routing Tables” on page 50
 - “Static and Dynamic Routing” on page 51
 - “Upstream and Downstream Prefix Delegation” on page 51
- “Static Routing” on page 52
- “RIPng Configuration” on page 53
 - “Creating and Deleting a RIPng Instance” on page 54
 - “Enabling and Disabling RIPng on Interfaces” on page 55
- “Global RIPng Parameters” on page 56
 - “Advertising the Default Route” on page 56
 - “Rejecting Default Routes” on page 57
 - “Configuring Trusted Neighbors” on page 57
 - “Redistributing Routes” on page 58
 - “Protecting Against Flooding by Setting an Update Threshold” on page 59
- “RIPng Interface Parameters” on page 60
 - “Route, Interface, and Offset Metrics” on page 60
 - “Configuring Split Horizon with Poison Reverse” on page 64

- “Viewing Routing and RIPng Information” on page 64
 - “Viewing the Routing Table” on page 65
 - “Viewing the RIPng Database” on page 65
 - “Viewing RIPng Details by Virtual Router” on page 66
 - “Viewing RIPng Details by Interface” on page 14-67
 - “Viewing RIPng Neighbor Information” on page 68
- “Configuration Examples” on page 69
 - “Enabling RIPng on Tunnel Interfaces” on page 69
 - “Avoiding Traffic Loops to an ISP Router” on page 71
 - “Setting a Null Interface Redistribution to OSPF” on page 79
 - “Redistributing Discovered Routes to OSPF” on page 80
 - “Setting Up OSPF-Summary Import” on page 80

Overview

Dual-stack architecture allows the security device to receive, process, and forward IPv6 traffic and supports assignment of an IPv4 address, at least one IPv6 prefix, or both. This allows the device to exchange packets between dissimilar networks and supports security policies between remote networks over dissimilar WAN backbones.

The next sections explain IPv4/IPv6 routing tables, and static and dynamic routing concepts.

Dual Routing Tables

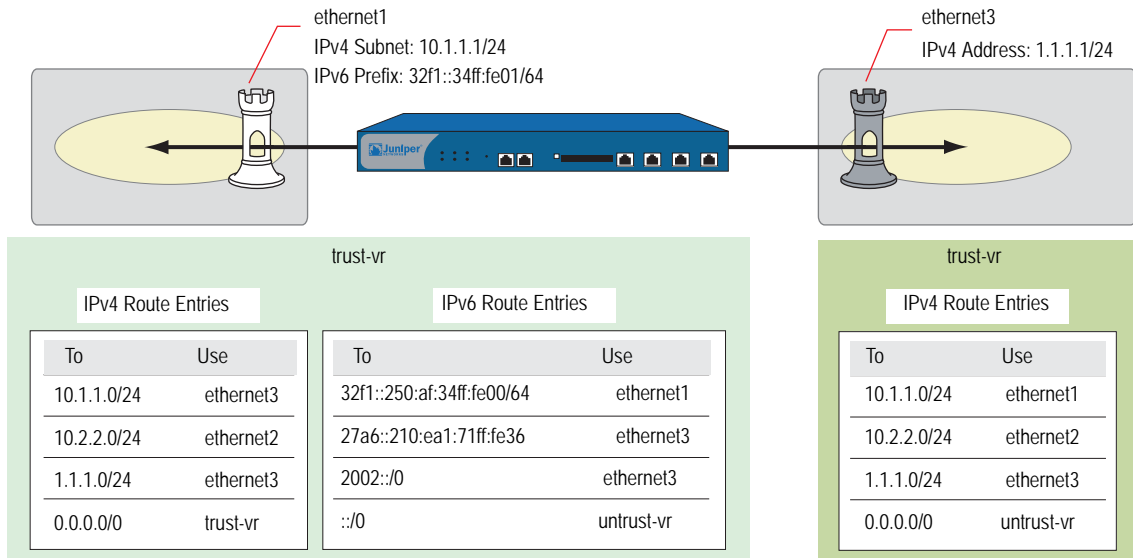
If you configure an interface to have both IPv4 and IPv6 addresses, the virtual router to which the interface is bound contains two separate routing tables: one for IPv4 entries and one for IPv6 entries. Both tables reside in the same virtual router.

For example, if you bind an interface to a zone bound to the trust-vr virtual router, and then you configure the interface for dual-stack mode, the trust-vr virtual router builds and maintains an IPv4 routing table and an IPv6 routing table.

Figure 14 shows a security device with two Ethernet interfaces:

- Ethernet1 runs in dual-stack mode and maintains two routing tables.
- Ethernet3 runs in IPv4 mode only. The trust-vr for ethernet3 maintains only one routing table.

Figure 14: Dual-Stack Router Behavior



Static and Dynamic Routing

ScreenOS supports static routing and dynamic routing with Routing Information Protocol next-generation (RIPng), an Interior Gateway Protocol (IGP) that uses a distance-vector algorithm to determine the best route to a destination, using the hop count as the metric. RIPng supports route redistribution to import known routes, from a router running a different protocol, into the current RIPng routing instance. For example, you can import static routes from a virtual router into a RIPng instance. RIPng is intended only for use in IPv6 networks.

NOTE: For more information about RIPng, see RFCs 2080 and 2081.

Upstream and Downstream Prefix Delegation

In some network topologies, devices delegate prefixes to downstream devices, which can use the prefixes to autoconfigure other devices. For example, a corporate WAN might use multiple security devices used in the following ways:

- Customer Premises Equipment (CPE) routers to delegate IP addresses to local hosts
- Gateway routers to delegate subnetwork prefixes to the CPE routers
- Internet Service Provider (ISP) routers further upstream to delegate network prefixes to gateway routers

Static Routing

The process for configuring static routes for IPv6 interfaces is the same as for IPv4 interfaces. The difference is the address notation. For more information about IPv4 static routing, see “Static Routing” on page 7-1.

In the following example, you configure two interfaces.

- ethernet2 IPv4 address 1.1.2.1/24 (optional)
- ethernet2 configured in Router mode with IPv6 address 32f1:250a::02/48
- ethernet2 bound to trust zone and enabled for route advertising to local hosts
- ethernet3 configured in Router mode with IPv6 address 32f1:250a::03/48
- Two static routing entries:
 - 32f1:250a::01/48 for ethernet2, advertised to local hosts
 - ::/0 for ethernet3, external gateway 27a6::210:ea1:71ff:fe36

WebUI

1. Interfaces

Network > Interfaces > Edit (ethernet2)

Network > Interfaces > Edit (ethernet2) Static IP (select)

Network > Interfaces > Edit (ethernet2) > IPv6

Network > Interfaces > Edit (ethernet2) > IPv6 > ND/RA Settings

Network > Interfaces > Edit (ethernet3)

2. Routes

Network > Routing > Routing Entries New (trust-vr)

Network > Routing > Routing Entries New (trust-vr) > Gateway: (select)

CLI

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 1.1.2.1/24
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 32f1:250a::02/48
set interface ethernet2 ipv6 ra transmit
set interface ethernet2 ipv6 ra link-mtu
set interface ethernet2 ipv6 ra link-address
```

```
set interface ethernet3 zone untrust
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet3 ipv6 32f1:250a::03/48
```

2. Routes

```
set vrouter trust-vr route 32f1:250a::01/48 interface ethernet2
set vrouter trust-vr route ::/0 interface ethernet3 gateway
27a6::210:ea1:71ff:fe36
```

RIPng Configuration

You create RIPng on a per-virtual router basis on a security device. If you have multiple virtual routers (VRs) within a system, you can enable multiple instances of RIPng.

NOTE: Before you configure a dynamic routing protocol on a security device, you should assign a VR ID. For more information, see *Volume 2: Fundamentals*.

This section describes the following basic steps to configure RIPng on a security device:

1. Create the RIPng routing instance in a VR.
2. Enable the RIPng instance.
3. Enable RIPng on interfaces that connect to other RIPng routers.
4. Redistribute routes learned from different routing protocols (such as OSPF, BGP, or statically configured routes) into the RIPng instance.

This section describes how to perform each of these tasks using the CLI or the WebUI.

Optionally, you can configure RIPng parameters such as the following:

- Global parameters, such as timers and trusted RIPng neighbors, that are set at the VR level for RIPng (see “Global RIPng Parameters” on page 56)
- Interface parameters that are set on a per-interface basis for RIPng (see “RIPng Interface Parameters” on page 60)

Creating and Deleting a RIPng Instance

You create and enable a RIPng routing instance on a specific virtual router (VR) on a security device. When you create and enable a RIPng routing instance on a VR, RIPng transmits and receives packets on all RIPng-enabled interfaces in the VR.

Deleting a RIPng routing instance in a VR removes the corresponding RIPng configurations for all interfaces that are in the VR.

For more information about VRs and configuring a VR on security devices, see *Volume 7: Routing*.

Creating a RIPng Instance

You create a RIPng routing instance on the *trust-vr* and then enable RIPng.

WebUI

Network > Routing > Virtual Router (trust-vr) > Edit: Enter a Virtual Router ID, then select **Create RIPng Instance**.

Select Enable RIPng, then click **OK**.

CLI

1. **Router ID**
set vrouter trust-vr router-id 10
2. **RIPng Routing Instance**
set vrouter trust-vr protocol ripng
set vrouter trust-vr protocol ripng enable
save

Deleting a RIPng Instance

In this example, you disable the RIPng routing instance in the trust-vr. RIPng stops transmitting and processing packets on all RIPng-enabled interfaces of the trust-vr.

WebUI

Network > Routing > Virtual Router (trust-vr) > Edit > Edit RIPng Instance: Deselect Enable RIPng and then click **OK**.

Network > Routing > Virtual Router (trust-vr) > Edit > Delete RIPng Instance and then click **OK** at the confirmation prompt.

CLI

```
unset vrouter trust-vr protocol ripng enable
unset vrouter trust-vr protocol ripng
save
```

Enabling and Disabling RIPng on Interfaces

By default, RIPng is disabled on all interfaces in the virtual router (VR) and you must explicitly enable it on an interface. When you disable RIPng at the interface level, RIPng does not transmit or receive packets on the specified interface. Interface configuration parameters are preserved when you disable RIPng on an interface.

NOTE: If you disable the RIPng routing instance in the VR (see “Deleting a RIPng Instance” on page 54), RIPng stops transmitting and processing packets on all interfaces in the VR.

Enabling RIPng on an Interface

In this example, you enable RIPng on the Trust interface.

WebUI

Network > Interface > Edit (for Trust) > RIPng: Select Protocol RIPng **Enable**, then click **Apply**.

CLI

```
set interface trust protocol ripng enable
save
```

Disabling RIPng on an Interface

In this example, you disable RIPng on the Trust interface. The **unset interface *interface_name* protocol ripng** command unbinds the existing instance of RIPng from the interface. To disable RIP for the interface enter the **unset interface *interface_name* protocol ripng enable** command before saving.

WebUI

Network > Interface (for Trust) > RIPng: Clear Protocol RIPng **Enable**, then click **Apply**.

CLI

```
unset interface trust protocol ripng
unset interface trust protocol ripng enable
save
```

Global RIPng Parameters

This section describes RIPng global parameters that you can configure at the virtual router (VR) level. When you configure a RIPng parameter at the VR level, the parameter setting affects operations on all RIPng-enabled interfaces. You can modify global parameter settings through the RIPng routing protocol context in the CLI or by using the WebUI.

Table 2 lists the RIPng global parameters and their default values.

Table 2: Global RIPng Parameters and Default Values

RIPng Global Parameter	Description	Default Value(s)
Advertise default route	Specifies whether the default route (::/0) is advertised.	Disabled
Default metric	Default metric value for routes imported into RIPng from other protocols, such as OSPF and BGP.	10
Flush timer	Specifies, in seconds, when a route is removed from the time the route is invalidated.	120 seconds
Invalid timer	Specifies, in seconds, when a route becomes invalid from the time a neighbor stops advertising the route.	180 seconds
Maximum neighbors	The maximum number of RIPng neighbors allowed. To list the neighbors for a particular interface, enter the get interface interface_name protocol ripng command.	256
Redistribute routes	Provides a way to route traffic coming from another dynamic routing protocol, such as Open Shortest Path First (OSPF), or static routes.	Disabled
Reject default routes	Specifies whether RIPng rejects a default route learned from another protocol. See "Rejecting Default Routes" on page 57.	Disabled
Incoming route map	Specifies the filter for routes to be learned by RIPng.	None
Outgoing route map	Specifies the filter for routes to be advertised by RIPng.	None
Update threshold	Specifies the number of updates the device processes before the update timer expires. If the threshold is set as 5 and if it receives 10 updates (not number of routes) before the update timer expires, the device drops the last 5 updates. After the update timer is reset another 5 updates are accepted and so on.	Zero (0) seconds
Trusted neighbors	Specifies an access list that defines RIPng neighbors. If no neighbors are specified, RIPng uses multicasting or broadcasting to detect neighbors on an interface. See "Configuring Trusted Neighbors" on page 57.	All neighbors are trusted
Update timer	Specifies, in seconds, when to issue updates of RIPng routes to neighbors.	30 seconds

Advertising the Default Route

You can change the RIPng configuration to include the advertisement of the default route (a non-RIPng route) and change the metric associated with the default route present in a particular VR routing table.

By default, the default route (::/0) is not advertised to RIPng neighbors. The following command advertises the default route to RIPng neighbors in the trust-vr VR with a metric of 5 (you must enter a metric value). The default route must exist in the routing table.

WebUI

Network > Routing > Virtual Router (trust-vr) > Edit > Edit RIPng Instance:
Enter the following, then click **OK**:

Advertising Default Route: (select)
Metric: 5

CLI

```
set vrouter trust-vr protocol ripng adv-default-route metric number 5  
save
```

Rejecting Default Routes

In a Route Detour Attack, a router inserts a default route (::/0) into the routing domain in order to detour packets to itself. The router can then drop the packets, causing service disruption, or it can obtain sensitive information in the packets before forwarding them. On Juniper Networks security devices, RIPng by default accepts any default routes that are learned in RIPng and adds the default route to the routing table.

In the following example, you configure the RIPng routing instance running in trust-vr to reject any default routes that are learned in RIPng.

WebUI

Network > Routing > Virtual Router (trust-vr) > Edit > Edit RIPng Instance:
Enter the following, then click **OK**:

Reject Default Route Learnt by RIPng: (select)

CLI

```
set vrouter trust-vr protocol ripng reject-default-route  
save
```

Configuring Trusted Neighbors

Multi-access environments can allow devices, including routers, to be connected into a network relatively easily. This can cause stability or performance issues if the connected device is not reliable. To prevent this problem, you can use an access list to filter the devices that are allowed to become RIPng neighbors. By default, RIPng neighbors are limited to devices that are on the same subnet as the virtual router (VR).

In this example, you configure the following global parameters for the RIPng routing instance running in the trust-vr:

- Maximum number of RIPng neighbors is 1.
- The IP address of the trusted neighbor, 2eee::5/64, is specified in an access-list.

WebUI

Network > Routing > Virtual Router (trust-vr) > Access List > New: Enter the following, then click **OK**:

Access List ID: 10
 Sequence No.: 1
 IP/Netmask: 2eee::5/64
 Action: Permit (select)

Network > Routing > Virtual Router (trust-vr) > Edit > Edit RIPng Instance:
 Enter the following, then click **OK**:

Trusted Neighbors: (select), 10
 Maximum Neighbors: 1

CLI

```
set vrouter trust-vr
device(trust-vr)-> set access-list 10 permit ip 2eee::5/64 1
device(trust-vr)-> set protocol ripng
device(trust-vr/ripng)-> set max-neighbor-count 1
device(trust-vr/ripng)-> set trusted-neighbors 10
device(trust-vr/ripng)-> exit
device(trust-vr)-> exit
save
```

Redistributing Routes

Route redistribution is the exchange of route information between routing protocols. For example, you can redistribute the following types of routes into the RIPng routing instance in the same virtual router (VR):

- Routes learned from BGP
- Routes learned from OSPF
- Directly connected routes
- Imported routes
- Statically configured routes

You need to configure a route map to filter the routes that are redistributed. For more information about creating route maps for route redistribution, see *Volume 7: Routing*.

Routes imported into RIPng from other protocols have a default metric of 10. You can change the default metric (see “Global RIPng Parameters” on page 56).

In this example, you redistribute static routes that are in the subnetwork 2eee::/64 to RIPng neighbors in the trust-vr. To do this, you first create an access list to permit addresses in the 2eee::/64 subnetwork. Then, configure a route map that permits addresses that match the access list you configured. Use the route map to specify the redistribution of static routes into the RIPng routing instance.

WebUI

Network > Routing > Virtual Router (trust-vr) > Access List > New: Enter the following, then click **OK**:

Access List ID: 20
Sequence No.: 1
IP/Netmask: 2eee::/64
Action: Permit (select)

Network > Routing > Virtual Router (trust-vr) > Route Map > New: Enter the following, then click **OK**:

Map Name: rmap1
Sequence No.: 1
Action: Permit (select)
Match Properties:
Access List: (select), 20 (select)

Network > Routing > Virtual Router (trust-vr) > Edit > Edit RIPng Instance > Redistributable Rules: Enter the following, then click **Add**:

Route Map: rmap1 (select)
Protocol: Static (select)

CLI

```
set vrouter trust-vr access-list 20 permit ip 2eee::/64 1
set vrouter trust-vr route-map name rmap1 permit 1
set vrouter trust-vr route-map rmap1 1 match ip 20
set vrouter trust-vr protocol ripng redistribute route-map rmap1 protocol static
save
```

Protecting Against Flooding by Setting an Update Threshold

A malfunctioning or compromised router can flood its neighbors with RIPng routing update packets. On virtual router (VRs), you can configure the maximum number of update packets that can be received on a RIPng interface within an update interval to avoid flooding of update packets. All update packets that exceed the configured update threshold are dropped. If you do not set an update threshold, all update packets are accepted.

In networks where neighbors have large routing tables, specifying an update threshold can impair the security device's ability to learn valid routes. Large routing tables accept and generate a large number of routing updates including flash updates within a given period. Update packets to the security device that exceed the changed threshold will be dropped and important routes might not be learned.

In this example, you set the maximum number of routing update packets that RIPng can receive on an interface to 4.

WebUI

Network > Routing > Virtual Router (trust-vr) > Edit > Edit RIPng Instance:
Enter the following, then click **OK**:

Maximum Number Packets per Update Time: (select), 4

CLI

```
set router trust-vr protocol ripng threshold-update 4
save
```

RIPng Interface Parameters

This section describes RIPng parameters that you configure at the interface level. When you configure a RIPng parameter at the interface level, the parameter setting affects the RIPng operation only on the specific interface. You can modify interface parameter settings with **interface** commands in the CLI or by using the WebUI.

Table 3 lists the RIPng interface parameters and their default values.

Table 3: RIPng Interface Parameters and Default Values

RIPng Interface Parameter	Description	Default Value
RIPng metric	Specifies the RIPng metric for the interface.	1
Passive mode	Specifies that the interface is to receive but not transmit RIPng packets.	No
Incoming route map	Specifies the filter for routes to be learned by RIPng.	None.
Outgoing route map	Specifies the filter for routes to be advertised by RIPng.	None.
Split-horizon	Specifies whether to enable split-horizon (do not advertise routes learned from an interface in updates sent to the same interface). If split horizon is enabled with the poison-reverse option, routes that are learned from an interface are advertised with a metric of 16 in updates sent to the same interface.	Split-horizon is enabled. Poison reverse is disabled.

Route, Interface, and Offset Metrics

A RIPng uses a positive integer to indicate the number of hops needed to reach a router. A device uses three values to calculate this number:

- The *route* metric is an integer associated with a particular route prefix.
- The *interface cost* metric is an integer associated with a particular interface configured for RIPng operation.
- The *offset* metric is a value added to the total.

The method by which the device derives the total metric depends on the interface through which the traffic flows.

- If the interface receives incoming traffic, the total metric is:
route metric + interface cost metric + offset metric
- If the interface transmits outgoing traffic, the interface cost metric does not apply, so the total metric is:
route metric + offset metric

Access Lists and Route Maps

In this example, you set up an IPv6 access list for the trust-vr with an access control list (ACL) ID of 10 and create a route map with prefix ::/0.

NOTE: To create an IPv4 access list and route map, see “Configuring a Route Map” on page 7-38.

WebUI

Network > Routing > Virtual Routers > Access Lists > New (trust-vr)

Network > Routing > Virtual Routers > Route Maps (trust-vr) > New

CLI

```
set vrtr trust-vr
set access-list ipv6 10
set access-list 10 permit ip ::/0 10
set route-map ipv6 name rm_six permit 10
set match ip 10
end
```

Static Route Redistribution

In the following example, you configure two devices for RIPng, Device A and Device B. For Device B, you add 2 to the metric value received by a route map (default is 1).

You configure the devices as follows:

- Device A to transmit outgoing traffic to Device B with these metrics:
 - Route metric (5) +
 - Offset metric (1 by default)

The total RIPng metric for outgoing packets is 6.

- Device B receives traffic from Device A with these metrics:
 - Route metric (4) +
 - Interface cost metric (7) +
 - Offset metric (2)

The total RIPng metric for incoming packets is 13.

WebUI (Device A)

1. IPv6 Interface

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet2) > IPv6 > ND/RA Settings

Network > Interfaces > Edit (for ethernet2) > RIPng

2. RIPng Virtual Routing

Network > Routing > Virtual Routers > Edit (for trust-vr) > Create RIPng Instance

3. RIPng Interface Metric

Network > Interfaces > Edit (for ethernet2) > IPv6

CLI (Device A)

1. IPv6 Interface

```
set interface ethernet3 zone trust
set interface ethernet3 ipv6 mode router
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 ip 2eee::1/64
set interface ethernet3 ipv6 interface-id 111111111111111111
set interface ethernet3 ipv6 ra transmit
set interface ethernet3 protocol ripng enable
```

2. RIPng Virtual Routing

```
set vrouter trust-vr protocol ripng
set vrouter trust-vr protocol ripng enable
set vrouter trust-vr protocol ripng advertise-def-route always metric 5
```

3. RIPng Interface Metric

```
set interface ethernet3 protocol ripng metric 1
```

WebUI (Device B)

1. IPv6 Interface

Network > Interfaces > Edit (for Trust)

Network > Interfaces > Edit (for Trust) > IPv6

Network > Interfaces > Edit (for Trust) > IPv6 > ND/RA Settings

2. Static Routes

Network > Routing > Routing Entries

3. RIPng Virtual Router

Network > Routing > Virtual Routers > Edit (for trust-vr) > Create RIPng Instance

Network > Interfaces > Edit (for Trust) > RIPng

Network > Routing > Virtual Routers > Access List > New (for trust-vr)

4. IPv6 Route-Map

Network > Routing > Virtual Routers > Route Map > New (for trust-vr)

Network > Routing > Virtual Router > Edit (for trust-vr) > Create RIPng Instance

5. Static Route Redistribution

Network > Routing > Virtual Router > Edit (for trust-vr) > Edit RIPng Instance > Redistributable Rules

CLI (Device B)

1. IPv6 Interface

```
set interface trust zone trust
set interface trust ipv6 mode host
set interface trust ipv6 enable
set interface trust ipv6 interface-id 2222222222222222
set interface trust ipv6 ra accept
```

2. Static Routes

```
set route 3a51:94ef::1/48 interface trust metric 2 tag 23
set route 3a51:ee45::1/48 interface trust metric 3 tag 37
```

3. RIPng Virtual Router

```
set vrouter trust-vr protocol ripng
set vrouter trust-vr protocol ripng enable
set interface trust protocol ripng enable
set interface trust protocol ripng metric 7
set vrouter trust-vr access-list ipv6 10
set vrouter trust-vr access-list ipv6 10 permit ip 3a51::1/16 10
```

4. IPv6 Route-Map

```
set vrouter trust-vr route-map ipv6 name abc permit 25
set vrouter trust-vr route-map abc 25 match ip 10
set vrouter trust-vr route-map abc 25 metric 4
set vrouter trust-vr route-map abc 25 offset-metric 2
set vrouter trust-vr protocol ripng route-map abc out
set interface trust protocol ripng route-map abc out
```

5. Static Route Redistribution

```
set vrouter trust-vr protocol ripng redistribute route-map abc protocol static
save
```

Configuring Split Horizon with Poison Reverse

In the following example, you configure split horizon with poison reverse for the interface.

WebUI

Network > Interfaces > Edit (for Trust) > RIPng: Enter the following, then click **OK**:

Split Horizon: Enabled with poison reverse (select)

CLI

```
set interface trust protocol ripng split-horizon poison-reverse
save
```

Viewing Routing and RIPng Information

After modifying RIPng parameters, you can view the following types of RIPng details:

- Database, which shows routing information
- Protocol, which gives RIPng and interface details for a virtual router (VR)
- Neighbor

Viewing the Routing Table

You can view route information from the CLI.

WebUI

NOTE: You must use the CLI to view the complete routing table.

CLI

get route v6

Sample output:

```
untrust-vr (0 entries)
```

```
-----  
C - Connected, S - Static, A - Auto-Exported, I - Imported, R - RIP  
iB - IBGP, eB - EBGP, O - OSPF, E1 - OSPF external type 1  
E2 - OSPF external type 2  
trust-vr (6 entries)
```

```
-----  
ID          IP-Prefix      Interface      Gateway      P Pref  Mtr  Vsys  
-----  
* 5          ::/0          trust fe80::210:dbff:fe22:3017  U 251   2   none  
* 2          3abc::1/64    trust fe80::210:dbff:fe22:3017  R 100  11  Root  
* 3          2abc::1/64    trust fe80::210:dbff:fe22:3017  R 100  11  Root  
* 6 2eee::210:dbff:fe20  
1          2eee::1/64    trust fe80::210:dbff:fe22:3017  R 100   2  Root
```

Viewing the RIPng Database

You can verify RIPng routing information from the CLI. You can choose to view a complete list of all RIPng database entries or a single entry.

In this example, you view detailed information from the RIPng database. You can choose to view all database entries or limit the output to a single database entry by appending the IP address and mask of the desired VR.

In this example, you specify the trust-vr and append the prefix and IP address 10.10.10.0/24 to view only a single table entry.

WebUI

NOTE: You must use the CLI to view the RIPng database.

CLI

```
get vrouter trust-vr protocol ripng database prefix 10.10.10.0/24
```

After you enter the following CLI command, you can view the RIPng database entry:

```
device-> get vrouter trust-vr protocol ripng database 10.10.10.0/24
```

The RIPng database contains the following fields:

- **DBID**, the database identifier for the entry
- **Prefix**, the IP address and prefix
- **Nexthop**, the address of the next-hop (router)
- **If**, the type of connection (Ethernet or tunnel)
- Cost metric assigned to indicate the distance form the source

Flags can be one or more of the following: multipath (M), RIPng (R), Redistributed (I), Advertised default (D), Permanent (P), Summary (S), Unreachable (U), or Hold (H).

In this example, the database identifier is 7, the IP address and prefix is 2eee::/64, and the next hop is 2eee::100/64. It is an Ethernet connection with a cost of 2. The flags are M and R and indicate that this route is multipath and uses RIPng.

Viewing RIPng Details by Virtual Router

You can view complete RIPng information for a virtual router to check a configuration or verify that saved changes are active. You can limit output to only the interface summary table by appending *interface* to the CLI command.

WebUI

NOTE: You must use the CLI to view the RIPng details.

CLI

```
get vrouter trust-vr protocol ripng
```

Sample output:

```
device-> get vrouter trust-vr protocol ripng
VR: trust-vr
-----
State: enabled
Version: 1
Default metric for routes redistributed into RIPng: 10
Maximum neighbors per interface: 256
Next RIPng update scheduled after: 7 sec
Advertising default route: disabled
Default routes learnt by RIPng will be accepted
Incoming routes filter and offset-metric: not configured
Outgoing routes filter and offset-metric: not configured
Update packet threshold is not configured
Total number of RIPng interfaces created on vr(trust-vr): 1

Update Invalid Flush (Timers in seconds)
-----
      30    180   120

Flags: Split Horizon - S, Split Horizon with Poison Reverse - P, Passive - I

Interface  IP-Prefix      Admin    State    Flags  NbrCnt  Metric  Ver-Rx/Tx
-----
ethernet3/4 fe80::210:dbff:fe8d enabled   enabled S          1      1    1/1
```

You can view RIPng settings, packet details, RIPng timer information, and a summarized interface table.

Viewing RIPng Details by Interface

You can view complete RIPng information for a specific interface to check a configuration or verify that saved changes are active.

WebUI

NOTE: You must use the CLI to view the RIPng details.

CLI

```
get interface ethernet3/4 protocol ripng
```

Sample output:

```
VR: trust-vr
-----
Interface: ethernet3/4, IP: fe80::210:dbff:fe8d:3ea0/64, RIPng: enabled, Router:
enabled
Receive version 1, Send Version 1
State: Up, Passive: No
Metric: 1, Split Horizon: enabled, Poison Reverse: disabled
Incoming routes filter and offset-metric: not configured
Outgoing routes filter and offset-metric: not configured
Current neighbor count: 1
Next update after: 16 sec
Transmit Updates: 30 (3 triggered), Receive Updates: 68
Update packets dropped because flooding: 0
Bad packets: 0, Bad routes: 0
Neighbors on interface ethernet3/4
-----
IpAddress      Version  Age           Expires      BadPackets  BadRoutes
-----
fe80::210:dbf.. 1         00:12:13     00:02:39      0           0
```

Viewing RIPng Neighbor Information

You can view details about RIPng neighbors for a virtual router (VR). You can retrieve a list of information about all neighbors or an entry for a specific neighbor by appending the IP address of the desired neighbor. You can check the status of a route and verify the connection between the neighbor and the security device from these statistics.

In the following example you view RIPng neighbor information for the trust-vr.

WebUI

NOTE: You must use the CLI to view RIPng neighbor information.

CLI

```
get vrouter trust-vr protocol ripng neighbors
```

This command produces output similar to the following output:

```
n2-> get vr trust protocol ripng neighbors
VR: trust-vr
-----
Neighbors on interface trust
-----
IpAddress      Version  Age           Expires      BadPackets  BadRoutes
-----
fe80::210:dbf.. 1         00:06:52     00:02:29      0           0
```

In addition to viewing the IP address, you can view the following RIPng neighbor information:

- Age of the entry
- Expiration time
- Number of bad packets
- Number of bad routes
- Flags: static (S), demand circuit (T), NHTB (N), down (D), up (U), poll (P), or demand circuit init (I)

Configuration Examples

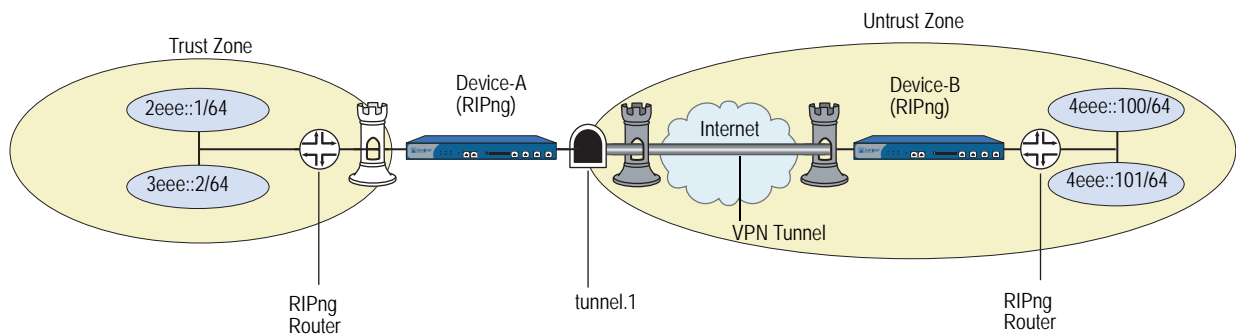
This section contains examples for using RIPng and static routing with tunnels with multiple security devices that behave as IPv6 hosts or IPv6 routers placed upstream or downstream from other devices.

Enabling RIPng on Tunnel Interfaces

The following example creates and enables a RIPng routing instance in trust-vr, on the Device-A device. You enable RIPng on both the VPN tunnel interface and the Trust zone interface. Only routes that are in the subnet `2eee::/64` are advertised to the RIPng neighbor on Device-B. This is done by first configuring an access list that permits only addresses in the subnet `2eee::/64`, then specifying a route map `abcd` that permits routes that match the access list. You then specify the route map to filter the routes that are advertised to RIPng neighbors.

Figure 15 shows the described network scenario.

Figure 15: Tunnel Interface with RIPng Example



WebUI (Device A)

Network > Routing > Virtual Router > Edit (for trust-vr) > Create RIPng Instance: Select **Enable RIPng**, then click **OK**.

Network > Routing > Virtual Router > Access List (for trust-vr) > New: Enter the following, then click **OK**:

Access List ID: 10
 Sequence No.: 10
 IP/Netmask: 2eee::/64
 Action: Permit

Network > Routing > Virtual Router > Route Map (for trust-vr) > New: Enter the following, then click **OK**:

Map Name: abcd
 Sequence No.: 10
 Action: Permit
 Match Properties:
 Access List: (select), 10

Network > Routing > Virtual Router > Edit (for trust-vr) > Edit RIPng Instance: Select the following, then click **OK**:

Outgoing Route Map Filter: abcd

Network > Interfaces > Edit (for tunnel.1) > RIPng: Enter the following, then click **Apply**:

Enable RIPng: (select)

Network > Interfaces > Edit (for trust) > RIPng: Enter the following, then click **Apply**:

Enable RIPng: (select)

CLI (Device A)

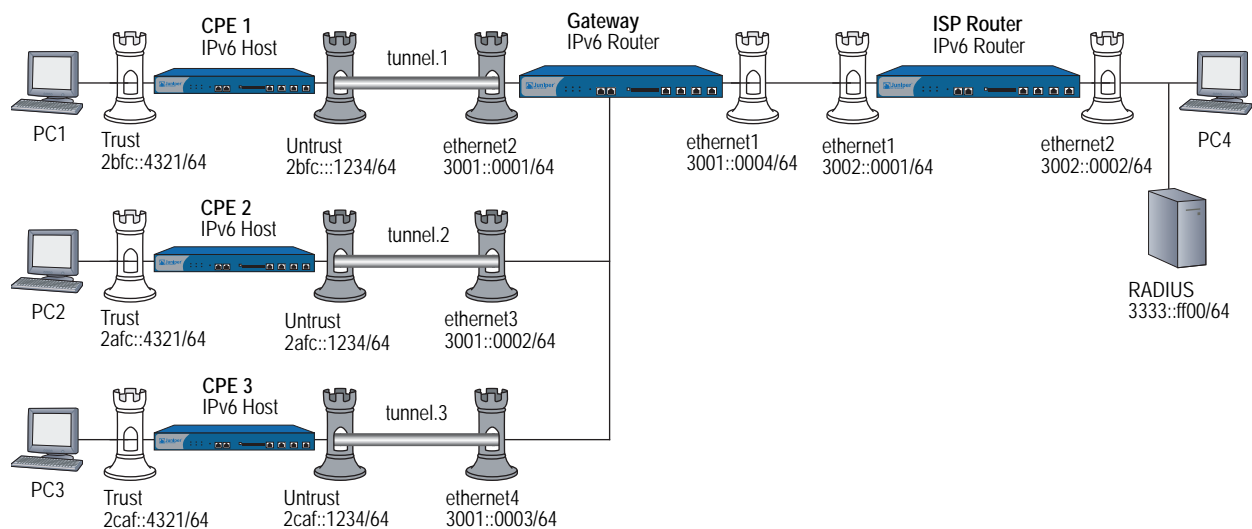
```
set vrouter trust-vr protocol ripng
set vrouter trust-vr protocol ripng enable
set interface tunnel.1 protocol ripng enable
set interface trust protocol ripng enable
set vrouter trust-vr access-list 10 permit ipv6 2eee::/64 10
set vrouter trust-vr route-map ipv6 name abcd permit 10
set vrouter trust-vr route-map ipv6 abcd 10 match ip 10
set vrouter trust-vr protocol ripng route-map abcd out
save
```

Avoiding Traffic Loops to an ISP Router

Figure 16 shows multiple devices configured to avoid traffic loops to the upstream ISP router. The ISP and gateway routers are in IPv6 Router mode. The three customer premises equipment (CPE) devices are IPv6 hosts and receive delegated IPv6 prefixes from the ISP router through the gateway router. The RADIUS server resides in the Trust zone of the Gateway device.

In this example, you enable the **route-deny** feature on the interface of a gateway connected to the ISP router. This feature prevents the default route on the gateway from creating a traffic loop to the ISP router.

Figure 16: RADIUSv6 IKE Example



Configuring the Customer Premises Equipment

This section lists the configuration steps for setting up the CPE 1, CPE 2, and CPE 3. The CPE devices are IPv6 hosts. The IPv6 addresses are placeholders for the values that each device could autoconfigure.

1. Configure IPv6 interfaces for all CPE devices. When configuring a CPE device interface in IPv6 Host mode, you don't have to explicitly assign the IPv6 prefix. The device accepts the prefix from the gateway router. For each CPE device configuration, we show and recommend assigning each IPv6 interface an interface ID.
2. Configure the following tunnels:
 - a. CPE 1 with interface tunnel.1.
 - b. CPE 2 with interface tunnel.2.
 - c. CPE 3 with interface tunnel.3.
3. Define a VPN for each CPE.
4. Define an XAuth client user with username and password for hosts.

5. Configure the Gateway router.

NOTE: In your network, if one or more CPE devices don't specify an IPv6 prefix and use one or more IPv4 addresses, then you need to assign an IPv4 address to the gateway router interface.

6. Configure the ISP router.

NOTE: The WebUI section lists the navigational paths to the device configuration pages for CPE 1, CPE 2, and CPE3. For specific values, see the specific CPE CLI sections that follow it.

WebUI (CPE 1, CPE 2, CPE 3)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.n) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

5. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

6. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (CPE 1)

1. Interfaces

```
set interface trust zone trust
set interface trust ipv6 mode host
set interface trust ipv6 enable
set interface trust ipv6 interface-id 3333333333333333
set interface trust ipv6 ra accept
```

```
set interface trust manage
set interface trust route
```

```
set interface untrust zone trust
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 4444444444444444
set interface untrust ipv6 ra accept
```

```
set interface untrust manage
set interface untrust route
```

2. Tunnel

```
set interface tunnel.1 zone untrust
```

3. Static Routes

```
set vrouter trust-vr route 3002::0001/64 interface tunnel.1
set vrouter trust-vr route ::/0 interface untrust gateway 3001::0001/64
```

4. IKE

```
set ike gateway CPE1_G add 3001::0001/64 main outgoing-interface untrust
  preshare abc123 sec-level standard
set ike gateway CPE1_G xauth client any username PC1 password PC1
```

5. VPN

```
set vpn CPE1toG gateway CPE1_G sec-level standard
set vpn CPE1toG bind interface tunnel.1
```

6. Policies

```
set policy from trust to untrust any any any permit
set policy from untrust to trust any any any permit
```

CLI (CPE 2)

1. Interfaces

```
set interface trust zone trust
set interface trust ipv6 mode host
set interface trust ipv6 enable
set interface trust ipv6 interface-id 3333333333333333
set interface trust ipv6 ra accept
```

```
set interface trust manage
set interface trust route
```

```
set interface untrust zone trust
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 4444444444444444
set interface untrust ipv6 ra accept
```

```
set interface untrust manage
set interface untrust route
```

2. Tunnel

```
set interface tunnel.2 zone untrust
```

3. Static Routes

```
set vrouter trust-vr route 3002::0001/64 interface tunnel.2
set vrouter trust-vr route ::/0 interface untrust gateway 3001::0001/64
```

4. IKE

```
set ike gateway CPE2_G add 3001::0001/64 main outgoing-interface untrust
  preshare abc123 sec-level standard
set ike gateway CPE2_G xauth client any username PC1 password PC1
```

5. VPN

```
set vpn CPE2toG gateway CPE2_G sec-level standard
set vpn CPE2toG bind interface tunnel.2
```

6. Policies

```
set policy from trust to untrust any any any permit
set policy from untrust to trust any any any permit
```

CLI (CPE 3)

1. Interfaces

```
set interface trust zone trust
set interface trust ipv6 mode host
set interface trust ipv6 enable
set interface trust ipv6 interface-id 5555555555555555
set interface trust ipv6 ra accept
```

```
set interface trust manage
set interface trust route
```

```
set interface untrust zone trust
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 6666666666666666
set interface untrust ipv6 ra accept
```

```
set interface untrust manage
set interface untrust route
```

2. Tunnel

```
set interface tunnel.3 zone untrust
```

3. Static Routes

```
set vrouter trust-vr route 3002::0001/64 interface tunnel.3
set vrouter trust-vr route ::/0 interface untrust gateway 3001::0001/64
```

4. IKE

```
set ike gateway CPE3_G add 3001::0001/64 main outgoing-interface untrust
  preshare abc123 sec-level standard
set ike gateway CPE3_G xauth client any username PC3 password PC3
```

5. VPN

```
set vpn CPE3toG gateway CPE3_G sec-level standard
set vpn CPE3toG bind interface tunnel.3
```

6. Policies

```
set policy from trust to untrust any any any permit
set policy from untrust to trust any any any permit
```

Configuring the Gateway

This section lists the configuration steps for setting up the gateway router, an IPv6 router.

1. Configure the ethernet1 interface of gateway with the route-deny feature.
2. Configure the gateway with three tunnels (tunnel.1, tunnel.2, and tunnel.3).
3. Configure the XAuth server with users for the three IKE gateways (from CPE 1, CPE 2, and CPE 3). Specify RADIUS shared secret (juniper) and port number (1812).
4. Define static routes between the RADIUS server and gateway and define a default route from the gateway to the ISP router.
5. Define an IP pool (P1) with an IP range from 2fba::5555 to 2fba::8888 on the gateway (Device 2).

WebUI (Gateway)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

5. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

6. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (Gateway)

1. Interfaces

```
set interface ethernet1 zone trust
set interface ethernet1 ipv6 mode router
set interface ethernet1 ipv6 enable
set interface ethernet1 ipv6 interface-id 111111111111111111
set interface ethernet1 ipv6 ip 2eee::1/64
set interface ethernet1 ipv6 ra transmit
set interface ethernet1 manage
set interface ethernet1 route
set interface ethernet1 route-deny
```

```
set interface ethernet2 zone untrust
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 111111111111111111
set interface ethernet2 ipv6 ip 2eee::1/64
set interface ethernet2 ipv6 ra transmit
set interface ethernet2 manage
set interface ethernet2 route
```

```
set interface ethernet3 zone untrust
set interface ethernet3 ipv6 mode router
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 interface-id 111111111111111111
set interface ethernet3 ipv6 ip 2eee::1/64
set interface ethernet3 ipv6 ra transmit
set interface ethernet3 zone untrust
set interface ethernet3 manage
set interface ethernet3 route
```

```
set interface ethernet4 zone untrust
set interface ethernet4 ipv6 mode router
set interface ethernet4 ipv6 enable
set interface ethernet4 ipv6 interface-id 111111111111111111
set interface ethernet4 ipv6 ip 2eee::1/64
set interface ethernet4 ipv6 ra transmit
set interface ethernet4 manage
set interface ethernet4 route
```

2. Tunnels

```
set interface tunnel.1 zone untrust
set interface tunnel.2 zone untrust
set interface tunnel.3 zone untrust
```

3. Routes

```
set vrouter trust-vr route 2fbc::4321/64 interface tunnel.1
set vrouter trust-vr route 2afc::4321/64 interface tunnel.2
set vrouter trust-vr route 2caf::4321/64 interface tunnel.3
set vrouter trust-vr route ::/0 interface ethernet1 gateway 3002::0001
```

4. RADIUS

```
set ippool P1 2fba::5555 2fba::8888
set auth-server RAD1 id 1
set auth-server RAD1 server-name 3333::ff00
set auth-server RAD1 account-type xauth
set auth-server RAD1 radius secret juniper
set auth-server RAD1 radius port 1812
set xauth default ippool P1
```

5. IKE

```
set ike gateway G_CPE1 add 20.1.1.1 main outgoing-interface ethernet2 preshare
  abc123 sec-level standard
set ike gateway G_CPE1 xauth server RAD1 query-config user PC4
set ike gateway G_CPE1 modecfg server action add-route

set ike gateway G_CPE2 add 60.1.1.1 main outgoing-interface ethernet3 preshare
  abc123 sec-level standard
set ike gateway G_CPE2 xauth server RAD1 query-config user PC4
set ike gateway G_CPE2 modecfg server action add-route

set ike gateway G_CPE3 add 80.1.1.1 main outgoing-interface ethernet4 preshare
  abc123 sec-level standard
set ike gateway G_CPE3 xauth server RAD1 user PC4
set ike gateway G_CPE3 modecfg server action add-route
```

6. VPN

```
set vpn GtoCPE1 gateway NS2_NS1 sec-level standard
set vpn G2toCPE1 bind interface tunnel.1
```

```
set vpn GtoCPE2 gateway NS2_NS3 sec-level standard
set vpn GtoCPE2 bind interface tunnel.2
```

```
set vpn GtoCPE3 gateway NS2_NS4 sec-level standard
set vpn GtoCPE3 bind interface tunnel.3
```

7. Policies

```
set policy from trust to untrust any any any permit
set policy from untrust to trust any any any permit
```

Configuring the ISP Router

This section lists the configuration steps for setting up the ISP router, an IPv6 router.

1. Configure the ISP router to reach the gateway router and the RADIUS server.
2. Configure accounts for the RADIUS server.
3. Define a static route on the ISP router to reach the 100.100.100.0/24 network. Make the next hop the ethernet1 interface IP address of the gateway.

WebUI (ISP Router)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Route

Network > Routing > Routing Table > New (trust-vr)

CLI (ISP Router)

1. Interfaces

```
set interface ethernet1 zone trust
set interface ethernet1 ipv6 mode router
set interface ethernet1 ipv6 enable
set interface ethernet1 ipv6 interface-id 1111111111111111
set interface ethernet1 ipv6 ip 3002::0001/64
set interface ethernet1 ipv6 ra transmit
set interface ethernet1 manage
set interface ethernet1 route
```

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 2222222222222222
set interface ethernet2 ipv6 ip 3002::0002/64
set interface ethernet2 ipv6 ra transmit
set interface ethernet2 manage
set interface ethernet2 route
```

2. Route

```
set vrouter trust-vr route 2fba::5555/64 interface ethernet1 gateway
3001::0004/64
```

Setting a Null Interface Redistribution to OSPF

In this example, you set a static route to a null interface and redistribute the routes to the ISP router using Open Shortest Path First (OSPF) routing protocol. Any traffic sent to this route is subject to redistribution to OSPF and becomes available to outside OSPF devices.

1. Configure CPE 1, CPE 2, and CPE 3 as described on page 71.
2. Configure the gateway router as described on page 75.
3. Configure the ISP router as described on page 78.

WebUI (OSPF for Gateway Router)

Network > Routing > Virtual Router > Edit

CLI (Gateway)

```
set vrouter trust-vr route 2fba::5555/64 interface null

set vrouter trust-vr
set protocol ospf
set area 0
set enable
set interface ethernet1 protocol ospf area 0
set interface ethernet1 protocol ospf enable
set vrouter trust-vr
set route-map name abc permit 10
set match ip 10
set access-list 10 permit ip 2fba::5555/64 10
set vrouter trust-vr protocol ospf redistribute route-map abc protocol static
```

WebUI (ISP)

Network > Routing > Virtual Router > Edit

CLI (ISP)

```
set vrouter trust-vr
set protocol ospf
set area 0
set enable
set interface ethernet1 protocol ospf area 0
set interface ethernet1 protocol ospf enable
```

Redistributing Discovered Routes to OSPF

You can configure the gateway router to redistribute discovered routes to Open Shortest Path First (OSPF) routing protocol. To do this, append the following command information to the configuration that appears on page 71.

WebUI (Gateway)

Network > Routing > Virtual Router > Edit

CLI (Gateway)

```
set vrouter trust-vr protocol ospf redistribute route-map abc protocol discovered
```

Setting Up OSPF-Summary Import

In this example, you set up the gateway and ISP routers to redistribute discovered routes to OSPF routing protocol. See page 71 for the full configuration and illustration.

1. Configure CPE 1, CPE 2, and CPE 3 as described on page 71 and the ISP router as described on page 78.
2. Configure the gateway as described on page 75 and then redistribute the static routes to the ISP router. Append the added configuration below.
3. Configure the import summary feature for OSPF.

WebUI (Gateway)

Network > Routing > Routing Table > New (trust-vr)

Network > Routing > Virtual Router > Edit

CLI (Gateway)

3. OSPF

```
set vrouter trust-vr
set protocol ospf
set area 0
set enable
set interface ethernet1 protocol ospf area 0
set interface ethernet1 protocol ospf enable
set vrouter trust-vr
set route-map name abc permit 10
set match ip 10
exit
set access-list 10 permit ip 2fba::5555/64 10
set vrouter trust-vr protocol ospf redistribute route-map abc protocol discovered
set vrouter trust protocol ospf summary-import 2fba::5555/64
```

Chapter 5

Address Translation

When two hosts using different IP stacks exchange service requests through a security device, the device must perform address translation for all packets before it can transmit them across an IPv4/IPv6 boundary. This chapter describes the translation process. It contains the following sections:

- “Overview” on page 82
 - “Translating Source IP Addresses” on page 83
 - “Translating Destination IP Addresses” on page 84
- “Configuration Examples” on page 86
 - “Translating Source IP Addresses” on page 83
 - “Translating Destination IP Addresses” on page 84
 - “IPv6 Hosts to Multiple IPv4 Hosts” on page 86
 - “IPv4 Hosts to Multiple IPv6 Hosts” on page 90
 - “IPv4 Hosts to a Single IPv6 Host” on page 91
 - “Translating Addresses for Domain Name System Servers” on page 93

Overview

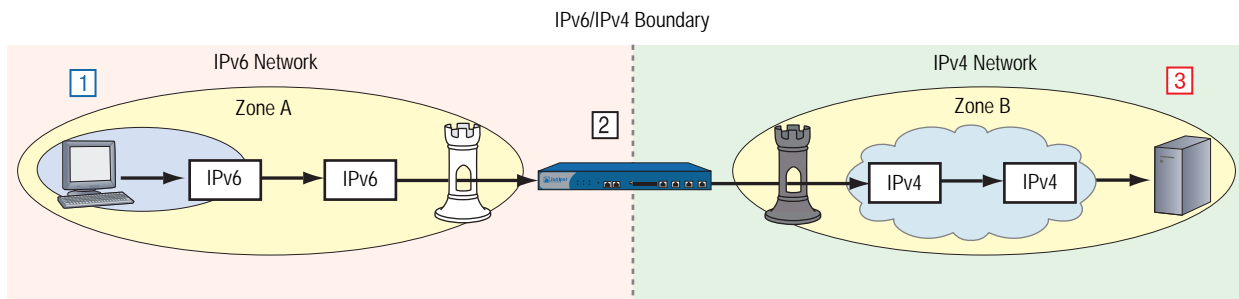
Network Address Translation with Port Translation (NAT-PT) is a mechanism that performs address translation for packets transmitted between hosts that use incompatible IP stacks. For example, a packet generated by an IPv6 host, transmitted over an IPv4-only backbone to an IPv4 host, must have IPv4 source and destination addresses. NAT-PT provides such addresses by translating the original IPv6 source and destination addresses to IPv4.

Devices perform address translations using the following mechanisms.

- Dynamic IP (DIP), which generates new source addresses from a defined address pool.
- Mapped IP (MIP), which translates destination addresses by performing direct mapping between one address and another. MIP can also map addresses to specified subnets.

Figure 17 shows how a device might use NAT-PT to transmit an IPv6 service request packet to an IPv4 host.

Figure 17: Network Address Translation (NAT) Across an IPv4/IPv6 Boundary



1. The IPv6 host transmits an IPv6 service request packet to the local IPv6 interface of a security device.
2. The security device translates the packet addresses to IPv4. It uses DIP to translate the source address and uses MIP to translate the destination address.
3. The destination host receives and processes the IPv4 packet.

NAT-PT converts IPv6 packets into IPv4 packets or IPv4 packets into IPv6 packets, which makes them routable across the IPv6/IPv4 boundary.

When an IPv4 host translates addresses between IPv4 and IPv6, it uses a MIP to generate an *IPv4-mapped* destination address. An IPv4-mapped address is a global unicast IPv6 address that contains an embedded IPv4 address. IPv4-mapped addresses are in the format *IPv6::a.b.c.d*, where:

- *IPv6* is the IPv6 subnet portion of the destination address.
- *a.b.c.d* is the embedded IPv4 address (expressed in decimal notation).

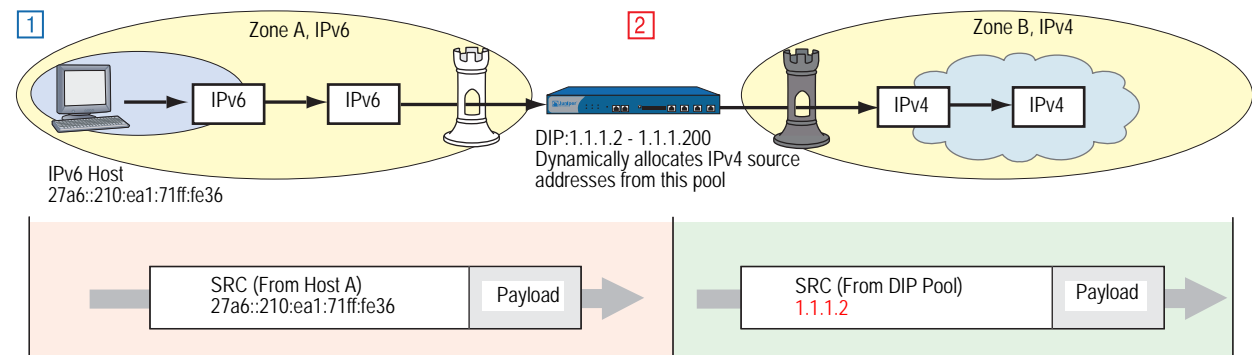
Translating Source IP Addresses

When two hosts that use dissimilar IP stacks exchange packets through a device, the device must translate the source address of each packet to an address compatible with the destination host. For example, an IPv6 packet transmitted into an IPv4 network needs an IPv4 source address; otherwise, the packet cannot successfully traverse the IPv4/IPv6 boundary. To translate source addresses, devices can use Dynamic IP (DIP), a mechanism that automatically generates source IP addresses from a defined pool of addresses.

DIP from IPv6 to IPv4

Figure 18 shows an IPv6 host sending a request packet to an IPv4 server over an IPv4 backbone. The security device, an IPv6 router, replaces the IPv6 source address in the packet header with an IPv4 address.

Figure 18: DIP from IPv6 to IPv4

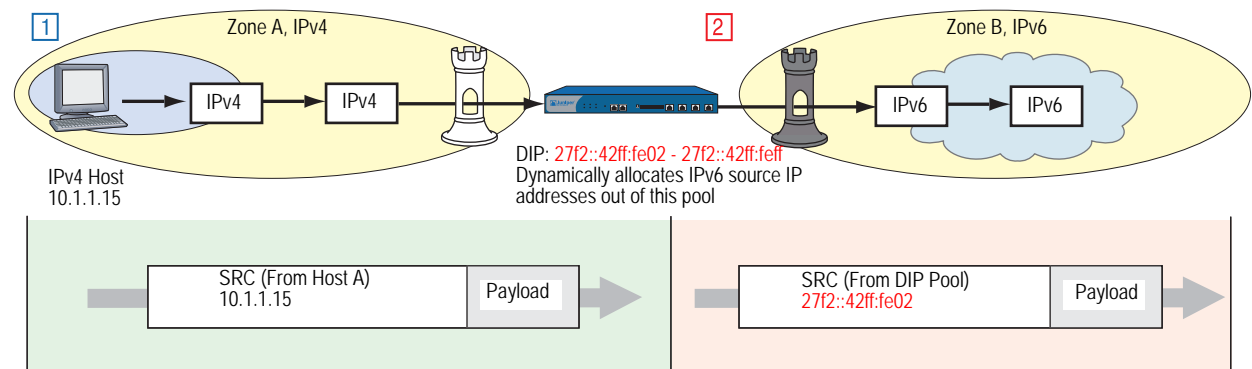


In this example, the device takes IPv4 source addresses 1.1.1.2 from the DIP pool and uses it to replace the original IPv6 source address (27a6::210:ea1:71ff:fe36). It then transmits the packet across the IPv6/IPv4 boundary.

DIP from IPv4 to IPv6

Figure 19 shows an IPv4 host sending a request packet to an IPv6 server in zone B over an IPv6 backbone. The security device replaces the IPv4 source address in the packet header with an IPv6 address.

Figure 19: DIP from IPv4 to IPv6



In this example, the device generates IPv6 source addresses 27f2::42ff:fe02 from the DIP pool and uses it to replace the original IPv4 source address (10.1.1.15). It then sends the packet across the IPv4/IPv6 boundary.

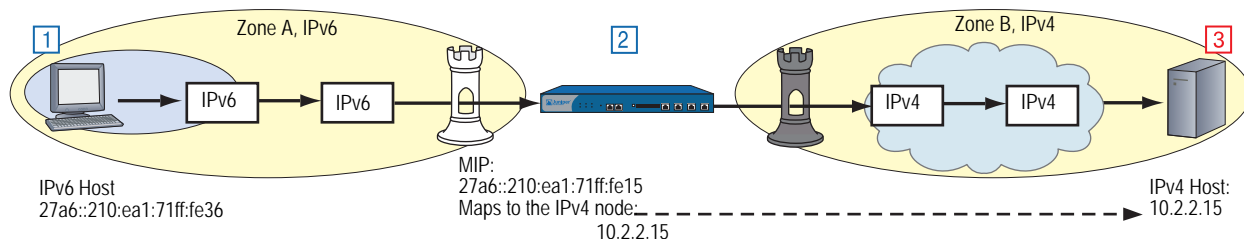
Translating Destination IP Addresses

When two hosts that use dissimilar IP stacks exchange packets through a device, the device must translate the destination address of each packet to an address compatible with the destination host. For example, an IPv6 packet transmitted into an IPv4 network needs an IPv4 destination address; otherwise, the packet cannot cross the IPv4/IPv6 boundary. To perform this translation, devices use mapped IP (MIP).

MIP from IPv6 to IPv4

Figure 20 shows an IPv6 host sending a request packet to an IPv4 server over an IPv4 backbone, the device must use a destination address that matches the IP address format of the destination host.

Figure 20: MIP from IPv6 to IPv4



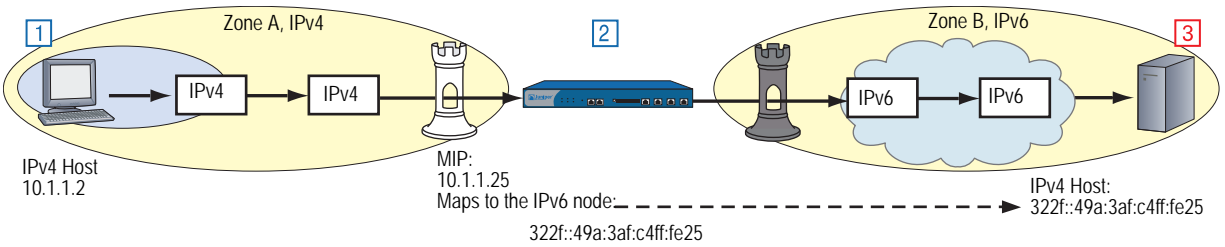
1. An IPv6 Host generates an IPv6 service request packet and sends it to the security device. The packet enters Zone A.
2. The device replaces the destination address with the MIP address and sends the packet to Zone B.
3. The destination device receives the packet somewhere in Zone B. To this device, the packet is an IPv4 packet.

In this example, the IPv6 host addresses the packet to the MIP address (27a6::210:ea1:71ff:fe15). The device translates this MIP to the address of the remote IPv4 host (10.2.2.15).

MIP from IPv4 to IPv6

Figure 21 shows an IPv4 host sending a request packet to an IPv6 server over an IPv6 backbone. The security device uses a destination address that matches the IP address format of the destination host.

Figure 21: MIP from IPv4 to IPv6



1. An IPv4 Host generates an IPv4 service request packet and sends it to the security device. The packet enters Zone A.
2. The security device replaces the destination address with the MIP address and sends the packet to Zone B.
3. The destination device receives the packet somewhere in Zone B. To this device, the packet is an IPv6 packet.

In this example, the IPv4 host addresses the packet to the MIP address (10.1.1.25). The device translates this MIP to the address of the remote IPv6 host (322f::49a:3af:c4ff:fe25).

A MIP can also map to IPv6 networks and subnets.

Configuration Examples

The following sections contain examples of NAT-PT scenarios and translation with domain name services.

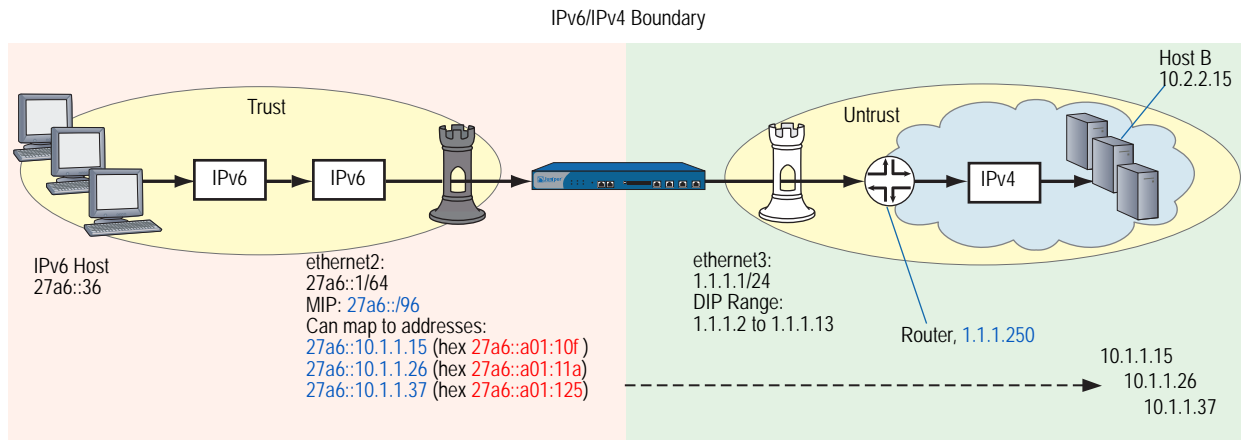
When a device transmits a service request packet from a host and forwards it to another host that uses a different IP stack, the device can use a NAT-PT policy to translate the IPv6 source and destination addresses of the outgoing packet. For example, a device residing at the border between an IPv6 network and an IPv4 WAN might transmit an outgoing IPv6 service request sent by an IPv6 host, using a NAT-PT policy to translate the source and destination addresses to IPv4.

IPv6 Hosts to Multiple IPv4 Hosts

When you need to send service requests from IPv6 hosts to multiple IPv4 destination hosts, define a policy that uses *IPv6-to-IPv4 network mapping* to translate the destination address.

Figure 22 shows the NAT-PT mechanism mapping local IPv6 addresses to IPv4 addresses in a remote IPv4 network. When a local IPv6 host transmits an outgoing service request packet through the device, the device uses MIP to generate an IPv6 destination address that uses IPv4-mapped format. When the device transmits packets across the IPv4/IPv6 boundary, it translates these destination addresses into IPv4 addresses.

Figure 22: IPv4-Mapped Addresses



IPv6 hosts can send HTTP requests through a device across an IPv6/IPv4 boundary. To send such a request, the user enters an IPv4-mapped address (such as 27a6::10.1.1.15) in the URL field of the browser. Such an entry might look like `http://[27ab::10.1.1.15]`. The device automatically translates this address into its hexadecimal equivalent. To IPv6 hosts the destinations appear to be IPv6-compatible devices. However, before transmitting the HTTP request packet over the boundary, the device translates the address to its IPv4 equivalent, which allows the packet to traverse the IPv4 WAN. To the target IPv4 servers, the packets are completely IPv4-compatible.

In the following example, you configure a device to allow hosts in an island IPv6 network to send HTTP service requests to hosts in a remote IPv4 network. Because the destination address is a network instead of a single host, the device uses *IPv4-mapped addresses* to represent individual remote nodes.

Host A generates an IPv6 service request packet and addresses it to Host B. It specifies an IPv6 destination address in the MIP table. The packet then goes from Zone A to Zone B.

Device A gets a source address from the DIP pool and translates the MIP address to the IPv4 address of the destination host. It then sends the packet out the interface to Device B.

The device translates the IPv4-mapped address to its IPv4 equivalent and then assigns it to the destination address of the outgoing packet.

WebUI

1. Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

2. MIP

Network > Interfaces > Edit (for ethernet2) > IPv6 > MIP > New

3. DIP

Network > Interfaces > Edit (for ethernet3) > DIP > New

4. Router

Network > Routing > Routing Entries New (for trust-vr)

5. Policy

Policies > New (for Trust to Untrust)

Policies > Edit (for policy) > Advanced

CLI

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 27a6::1/64
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
```

2. MIP

```
set interface ethernet2 mip 27a6::/96 ipv6 ipv4
```

3. DIP

```
set interface ethernet3 dip 7 1.1.1.2 1.1.1.13
```

4. Router

```
set vrouter trust-vr route 0.0.0.0/0 gateway 1.1.1.250
```

5. Policy

```
set policy from trust to untrust any-ipv6 mip(27a6::/96) http nat src dip-id 7 permit
save
```

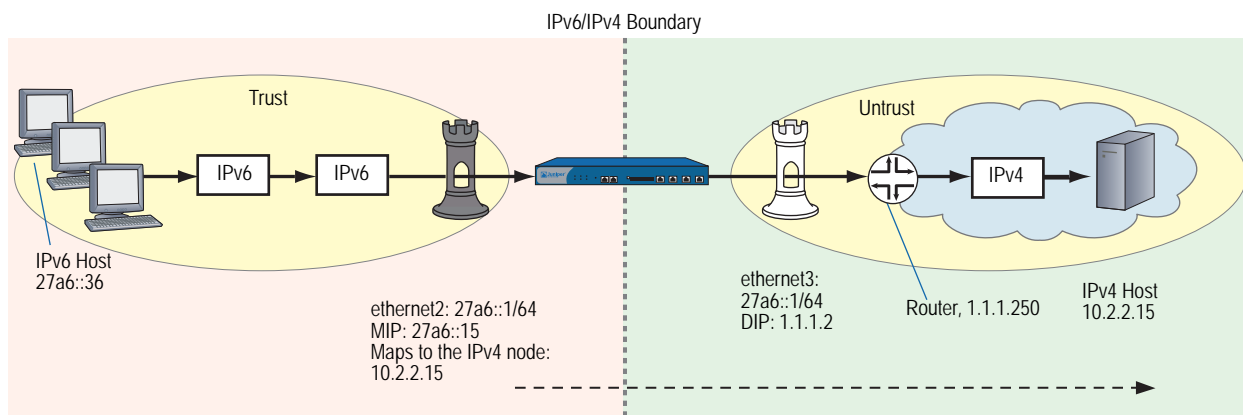
IPv6 Hosts to a Single IPv4 Host

When you need to send service requests from IPv6 hosts to a single IPv4 host, you can define a policy that uses *IPv6-to-IPv4 host mapping* to translate the destination address.

This NAT-PT mechanism maps an IPv6 address to the IPv4 address of a single remote IPv4 host. When a local IPv6 host transmits an outgoing service request packet through the device, the device uses the mapped address for the IPv4 destination address in the packet header. This translation allows the outgoing packet to traverse the IPv6/IPv4 boundary and establish communication with the remote host.

Figure 23 shows IPv6 hosts sending HTTP requests to a single IPv4 server through a device across an IPv6/IPv4 boundary.

Figure 23: IPv6-to-IPv4 Host Mapping



To send a request, the user enters an IPv6 address (such as 27a6::15) in the URL field of a browser. To IPv6 hosts the destination appears to be an IPv6-compatible device. However, before transmitting the HTTP request packet over the IPv6/IPv4 boundary, the device translates the address to its mapped IPv4 address, which allows the packet to traverse the IPv4 WAN. To the target IPv4 server, the packet is completely IPv4-compatible.

In the following example, you configure a device to allow hosts in an island IPv6 network to send HTTP requests to a single remote IPv4 host (10.2.2.15).

WebUI

1. Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

2. MIP

Network > Interfaces > Edit (for ethernet2) > IPv6 > MIP > New

3. DIP

Network > Interfaces > Edit (for ethernet3) > DIP > New

4. Router

Network > Routing > Routing Entries New (for trust-vr)

5. Policy

Policies > New (for Trust to Untrust)

Policies > Edit (for policy) > Advanced

CLI

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 27a6::1/64
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
```

2. MIP

```
set interface ethernet2 mip 27a6::15 ipv6 host 10.2.2.15
```

3. DIP

```
set interface ethernet3 dip 7 1.1.1.2 1.1.1.13
```

4. Routers

```
set vrouter trust-vr route 0.0.0.0/0 gateway 1.1.1.250
```

5. Policy

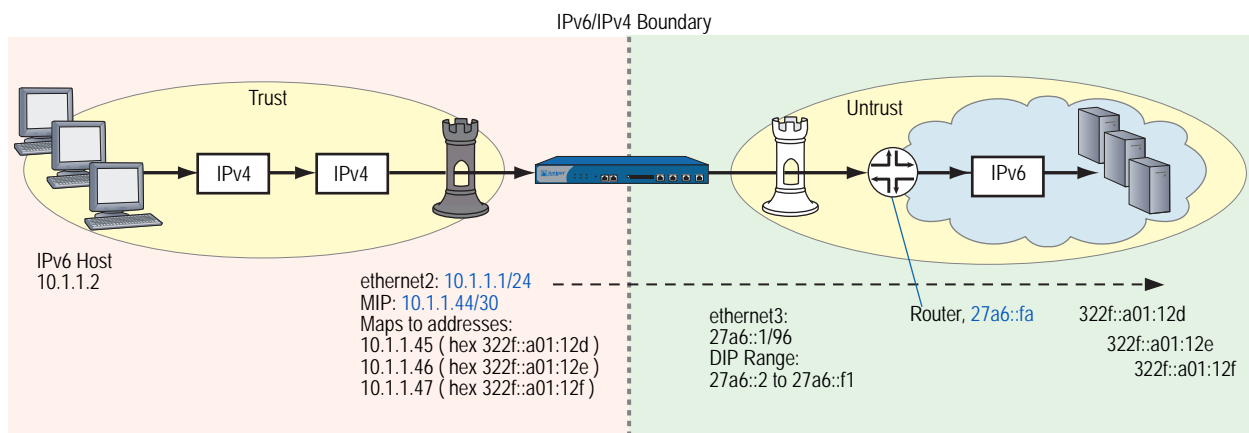
```
set policy from trust to untrust any-ipv6 mip(27a6::15) http nat src dip-id 7 permit
```

IPv4 Hosts to Multiple IPv6 Hosts

When you need to send service requests from local IPv4 hosts to multiple remote IPv6 hosts, you can define a policy that uses *IPv4-to-IPv6 network mapping*.

Figure 24 shows NAT-PT mapping a local IPv4 address to an IPv6 network (or subnet). When a local IPv4 host transmits an outgoing IPv4 service request packet through the device, the device translates the mapped address to the IPv6 address and assigns it to the destination address of the packet header. This translation allows the outgoing packet to traverse the IPv4/IPv6 boundary to establish communication with the remote IPv6 host.

Figure 24: IPv4-to-IPv6 Network Mapping



The mapped IPv4 address indirectly represents the remote IPv6 host. In effect, the local IPv4 hosts can view the remote host as part of the local IPv4 network.

Because the MIP belongs to the same subnet as the local IPv4 hosts, the hosts can view the remote host as part of the local IPv4 network.

WebUI

1. Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

2. MIP

Network > Interfaces > Edit (for ethernet2) > IPv6 > MIP > New

3. DIP

Network > Interfaces > Edit (for ethernet3) > DIP > New

4. Router

Network > Routing > Routing Entries New (for trust-vr)

5. Policy

Policies > New (for Trust to Untrust)

Policies > Edit (for policy) > Advanced

CLI

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.1.1/24
set interface ethernet3 zone untrust
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 ip 27a6::1/64
```

2. MIP

```
set interface ethernet2 mip 10.1.1.44 ipv6 prefix 322f::44/96 netmask
255.255.255.252
```

3. DIP

```
set interface ethernet3 dip 7 27a6::2 27a6::f1
```

4. Routers

```
set vrouter trust-vr route ::/0 interface ethernet3 gateway 27a6::fa
```

5. Policy

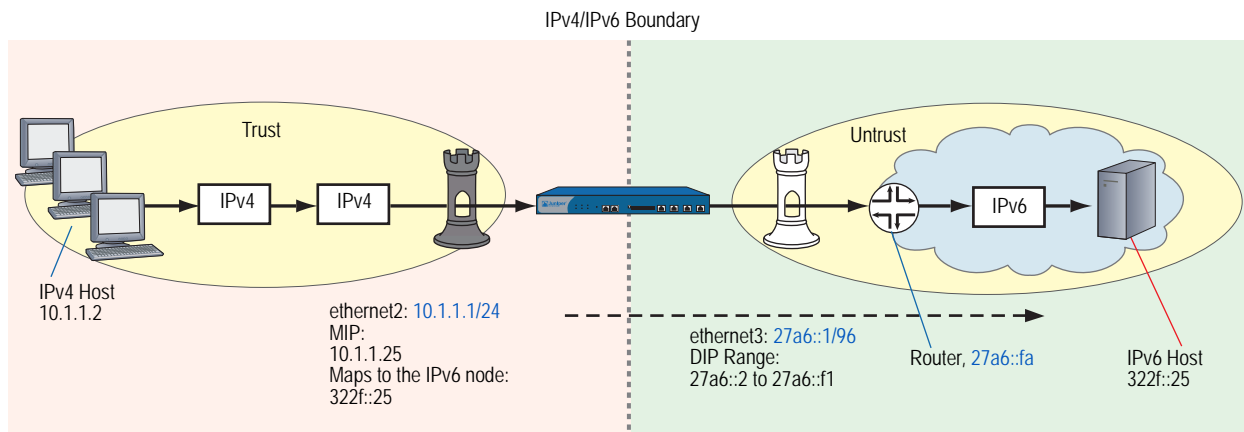
```
set policy from trust to untrust any-ipv4 mip(10.1.1.44/30) http nat src dip-id 7
permit
```

IPv4 Hosts to a Single IPv6 Host

When you need to send service requests from local IPv4 hosts to a single remote IPv6 host, define a policy that uses *IPv4-to-IPv6 host mapping*.

Figure 25 shows an IPv4 host transmitting an outgoing IPv4 service request packet through the device. The device translates the mapped address to an IPv6 address, and assigns it to the destination address of the packet header. This translation allows the outgoing packet to traverse the IPv4/IPv6 boundary to establish communication with the remote IPv6 host.

Figure 25: IPv4-to-IPv6 Host Mapping



The MIP belongs to the same subnet as the local IPv4 hosts, so hosts view the remote host as part of the local IPv4 network.

WebUI

1. Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

2. MIP

Network > Interfaces > Edit (for ethernet2) > IPv6 > MIP > New

3. DIP

Network > Interfaces > Edit (for ethernet3) > DIP > New

4. Router

Network > Routing > Routing Entries New (for trust-vr)

5. Policy

Policies > New (for Trust to Untrust)

Policies > Edit (for policy) > Advanced

CLI

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.1.1/24
set interface ethernet3 zone untrust
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 ip 27a6::1/96
```

2. MIP

```
set interface ethernet2 mip 10.1.1.25 ipv6 host 322f::25
```

3. DIP

```
set interface ethernet3 dip 7 27a6::2 27a6::f1
```

4. Routers

```
set vrouter trust-vr route ::/0 interface ethernet3 gateway 27a6::fa
```

5. Policy

```
set policy from trust to untrust any-ipv4 mip(10.1.1.25) http nat src dip-id 7 permit
```

Translating Addresses for Domain Name System Servers

Domain Name System (DNS) allows network devices to identify each other using domain names instead of IP addresses. An external DNS server keeps a table of domain names, each name having at least one associated IP address. You can use these domain names (as well as IP addresses) for identifying endpoints in policy definitions.

Some domain names can have both IPv4 and IPv6 addresses. For example, the hypothetical domain names `acme.com` and `juniper.net` could have the following addresses:

Domain Name	IPv4 Addresses	IPv6 Addresses
www.acme.com	1.2.2.15	27a6::e02:20f
	1.2.2.62	27a6::23ee:51a
	2.3.5.89	27a6::5542:225
www.juniper.net	3.1.1.21	3090::e02:20f
	3.20.7.96	3090::23ee:51a
	4.6.5.89	2e66::3354:722
	4.200.7.88	2e66::3354:722

When you define a policy that uses domain names as endpoints, and both domain names have IPv4 and IPv6 addresses, the device can establish secure communication between the endpoints using either protocol. For example, the following commands define a policy between the domain names `juniper.net` and `acme.com`:

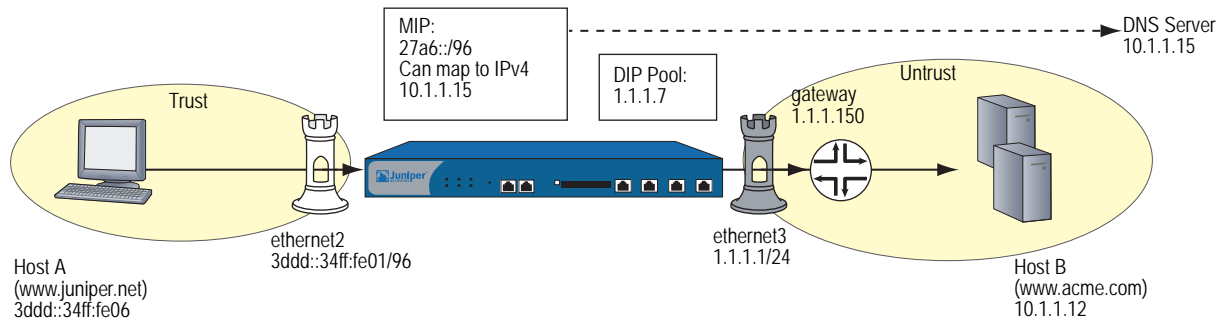
```
set address trust juniper www.juniper.net
set address untrust acme www.acme.com
set policy from trust to untrust juniper acme any permit
```

The two endpoints can exchange either type of traffic because the domain names each have IPv4 and IPv6 addresses.

However, if one domain name uses an IP stack that the other domain name does not, you must use NAT-PT to translate the source and destination addresses for transmitted DNS requests. For example, if `juniper.net` has only IPv6 addresses and `acme.com` has only IPv4 addresses, any service request sent from one to the other must undergo NAT-PT address translation.

In the following example, an IPv6 host (`www.juniper.net`) sends service requests to an IPv4 host (`www.acme.com`). It obtains the IPv4 address of the destination host from an IPv4 DNS server.

Figure 26: NAT-PT DNS Example



The device requires two policies:

- A policy that permits outgoing service requests and that performs NAT-PT on the source and destination addresses
- A policy that permits outgoing service requests from www.juniper.net to www.acme.com

In the following example, you configure a device to allow a Host A (www.juniper.net) to send service requests to a Host B (www.acme.com). The host domain names serve as the communication endpoints.

WebUI

1. Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

2. MIP

Network > Interfaces > Edit (for ethernet2) > IPv6 > MIP > New

3. DIP

Network > Interfaces > Edit (for ethernet3) > DIP > New

4. Routers

Network > Routing > Routing Entries New (for trust-vr)

5. Addresses

Objects > Addresses > New (for Trust)

Objects > Addresses > New (for Untrust)

6. Policy

Policies > New (for Trust to Untrust)

Policies > Edit (for policy) > Advanced > Advanced

CLI

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 3ddd::34ff:fe01/96
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
```

2. MIP

```
set interface ethernet2 mip 3ddd::/96 ipv6 ipv4 vrouter trust-vr
```

3. DIP

```
set interface ethernet3 dip 7 1.1.1.7
```

4. Addresses

```
set address trust juniper.net 3ddd::34ff:fe06/96
set address untrust acme www.acme.com
```

5. Routers

```
set vrouter trust-vr route 0.0.0.0/0 gateway 1.1.1.250
```

6. Policy

```
set policy from trust to untrust any-ipv6 mip(3ddd::/96) any nat dip-id 7 permit
set policy from trust to untrust any-ipv4 acme any permit
save
```


Chapter 6

IPv6 in an IPv4 Environment

6over4 tunneling is the process of encapsulating IPv6 packets within IPv4 packet headers so that IPv6 packets can traverse an IPv4 wide area network (WAN).

This chapter contains the following sections:

- “Overview” on page 98
- “Configuring Manual Tunneling” on page 99
- “Configuring 6to4 Tunneling” on page 102
 - “6to4 Routers” on page 102
 - “6to4 Relay Routers” on page 103
 - “Tunnels to Remote Native Hosts” on page 104
 - “Tunnels to Remote 6to4 Hosts” on page 107

Overview

6over4 tunneling is a way to send IPv6 traffic over an IPv4 wide area network (WAN) when you don't need authentication and encryption for the exchanged traffic. For example, your organization might need to exchange traffic between island IPv6 networks over an IPv4 WAN. In this case, the security device provides only firewall services.

There are two kinds of 6over4 tunneling:

- *Manual tunneling* uses a manually configured, static remote-end tunnel termination point.
- *6to4 tunneling* derives the remote-end tunnel termination point dynamically.

NOTE: For more information about 6to4 tunneling, refer to RFC 3056.

In most cases, the kind of tunneling to use depends on the kinds of routers and other devices present in a WAN infrastructure. For example, some ISPs (Internet Service Providers) provide manual tunnels to their customers as addendum services, and might have no need for 6to4 tunneling. In addition, there might be insufficient 6to4-configured relays in the IPv6 backbone to support 6to4 tunneling for your organization over the WAN.

6to4 tunneling can be appropriate if your organization needs to set up a one-to-many configuration, where one IPv6 network can access unspecified, multiple island IPv6 networks through multiple 6to4 relay routers. In such cases, 6to4 tunneling might be the best solution.

Configuring Manual Tunneling

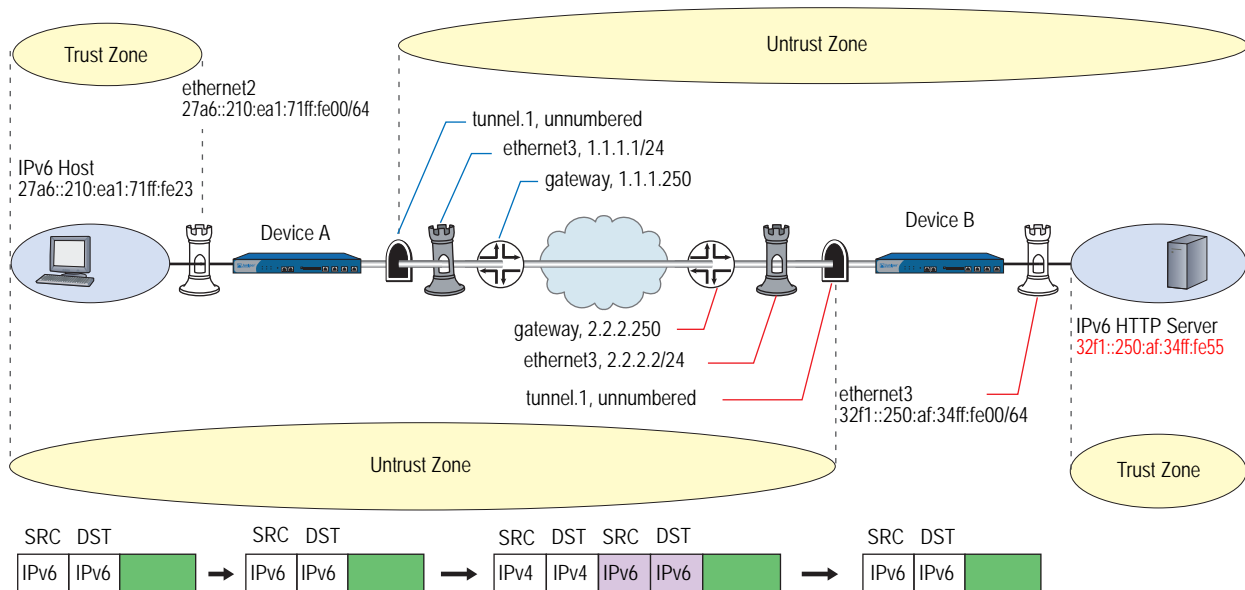
Manual tunneling is best suited for applications that require strict control and usually provide more security than 6to4 tunneling. Manual tunneling does not require address translation because you specify explicitly the IPv4 address of the destination gateway.

IPv6 hosts use the tunnel to communicate with an IPv6 server over an IPv4 WAN backbone. The endpoints of the tunnel are interfaces on devices A and B.

When you configure a device for manual tunneling, the device encapsulates each outgoing IPv6 packet inside an IPv4 packet before transmitting it into IPv4 network space.

Figure 27 shows a manual tunnel between two static, defined end points.

Figure 27: IPv6 Tunneling Using IPv4 Encapsulation Example



In the following example, you set up two devices (A and B) as endpoints for a manual 6to4 tunnel.

WebUI (Device A)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnel Interfaces

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

Network > Interfaces > Edit (for tunnel.1) > Encapsulation

Routers

Network > Routing > Routing Table > New (trust-vr)

Network > Routing > Route Table > New (trust-vr):

Address Book Entries

Objects > Addresses > List (Trust) > New

Policy

Policies > (From: Untrust, To: Trust) > New

WebUI (Device B)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnel Interfaces

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

Network > Interfaces > Edit (for tunnel.1) > Encapsulation

Routers

Network > Routing > Routing Table > New (trust-vr)

Network > Routing > Route Table > New (trust-vr)

Address Book Entries

Objects > Addresses > List (Trust) > New

Policy

Policies > (From: Untrust, To: Trust) > New

CLI (Device A)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.1.1/24
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 2abc::1/64
set interface ethernet2 ipv6 ra link-mtu
set interface ethernet2 ipv6 ra link-address
set interface ethernet2 ipv6 ra transmit
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
```

2. Tunnel Interface

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
set interface tunnel.1 tunnel encap ip6in4 manual
set interface tunnel.1 tunnel local-if ethernet3 dst-ip 2.2.2.2
```

3. Routers

```
set vrouter trust-vr route 3abc::/64 interface tunnel.1
set vrouter trust-vr route 0.0.0.0/0 interface ethernet3 gateway 1.1.1.250
```

4. Addresses

```
set address trust L_Hosts 2abc::/64
set address untrust R_Server 3abc::100/128
```

5. Policy

```
set policy from trust to untrust L_Hosts R_Server http permit
```

CLI (Device B)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 1.100.2.1/24
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 3abc::1/64
set interface ethernet3 zone untrust
set interface ethernet3 ip 2.2.2.2/24
```

2. Tunnel Interface

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
set interface tunnel.1 tunnel encap ip6in4 manual
set interface tunnel.1 tunnel local-if ethernet3 dst-ip 1.1.1.1
```

3. Routers

```
set vrouter trust-vr route 2abc::/64 interface tunnel.1
set vrouter trust-vr route 0.0.0.0/0 interface ethernet3 gateway 2.2.2.250
```

4. Addresses

```
set address trust L_Server 3abc::100/128
set address untrust R_Clients 2abc::/64
```

5. Policy

```
set policy from untrust to trust R_Clients L_Server http permit
```

Configuring 6to4 Tunneling

When a device uses *6to4 tunneling*, the device determines remote-end tunnel termination points (gateways) dynamically from routing table entries. In contrast with manual tunneling, which explicitly designates a single termination point for a VPN tunnel, 6to4 tunneling can allow any number of devices to serve as remote gateways for the tunnel. This allows one-to-many communication between a protected island IPv6 network and multiple external IPv6 networks.

For a network device to function as a 6to4 host, it must have an interface configured with a 6to4 address. Any IPv6 host without such an address is said to be *native*, or non-6to4. For example, if a host has interfaces with global aggregate IPv6 addresses but none with a 6to4 address, it is a native host. By contrast, if the host has an interface configured with a 6to4 address (whether or not it has other kinds of IPv6 addresses), it can function as a 6to4 host.

To set up a 6to4 tunnel between two devices, you must configure each communicating interface with a 6to4 address. The interfaces then serve as virtual border routers that handle the transition between IPv4 space and IPv6 space.

There are two kinds of 6to4 virtual router:

- **6to4** routers function as border routers between IPv4 WANs and 6to4 networks.
- **6to4 relay** routers function as border routers between IPv4 WANs and native networks.

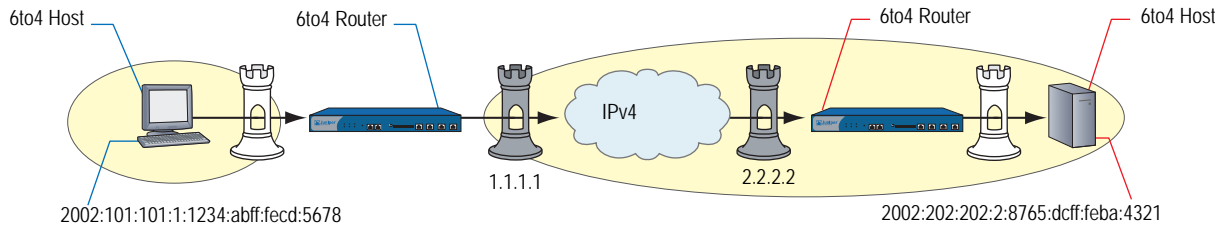
6to4 Routers

A *6to4 router* is a router configured to support exchange of packets between 6to4 hosts and other 6to4 hosts over an IPv4 WAN. You can make a device function as a 6to4 router by configuring an IPv4 interface for 6to4.

When a packet from a 6to4 host passes through the device, the 6to4 router encapsulates the packet inside an IPv4 packet, then transmits it into the IPv4 network space. 6to4 routers can also receive such encapsulated packets, decapsulate them, and forward them to 6to4 devices.

Figure 28 shows a 6to4 host sending a service request across an IPv4 WAN to another 6to4 host. The 6to4 router on Device A encapsulates the outgoing packets, and the 6to4 router on Device B decapsulates them.

Figure 28: 6to4 Routers

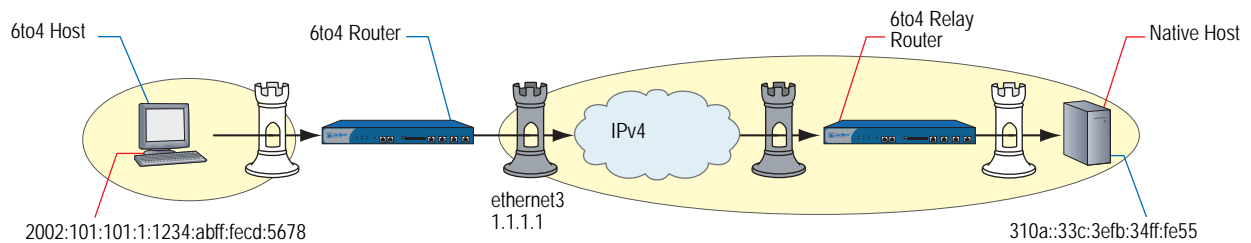


6to4 Relay Routers

A *6to4 relay router* is a router configured to support exchange of packets between 6to4 hosts and native (non-6to4) hosts over an IPv4 WAN. As with 6to4 routers, you can make a device function as a 6to4 router by configuring an IPv4 interface to handle 6to4 tunneling. When a 6to4 relay router receives an incoming encapsulated 6to4 packet, the router decapsulates the packet and forwards it to the native destination host. When a native host transmits an outgoing packet, the 6to4 relay router encapsulates the packet inside an IPv4 packet, then transmits it into the IPv4 network space.

In Figure 29, a 6to4 host sends a service request across an IPv4 WAN to a native (non-6to4) host. Device B decapsulates incoming packets, using the native IPv6 address as the destination address.

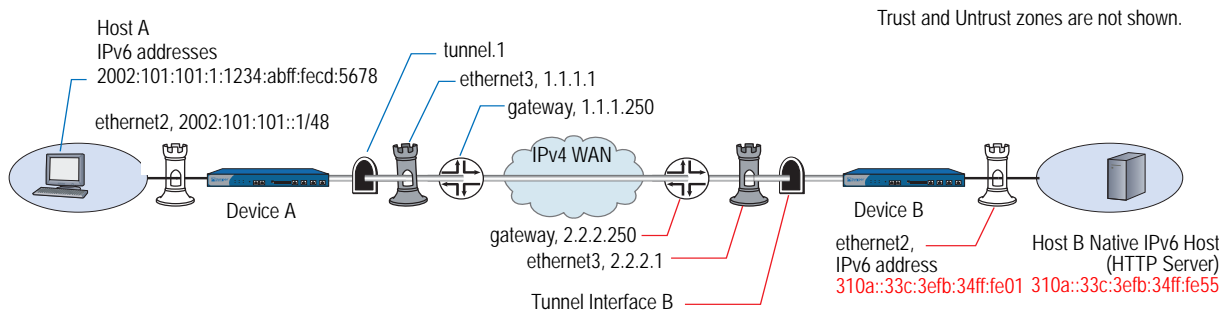
Figure 29: 6to4 Routers with Native Addresses



Tunnels to Remote Native Hosts

When a 6to4 host sends a service request to a remote native host, the device derives the remote termination point (gateway IPv4 address) from a routing table entry. Figure 30 shows Device A sending an HTTP service request to the native server protected by Device B.

Figure 30: 6over6 Manual Tunneling



The 6to4 router on Device A derives the remote gateway IPv4 address (2.2.2.1) from a configured routing table gateway entry 2002:0202:201:2:9876:abff:fe01:5432. (For this example, the MAC address of the remote gateway interface is 9876abcd5432.)

A device addresses packets that pass between a 6to4 host and a native host over an IPv4 WAN in the following manner:

1. The IPv6 host (Host A) generates an IPv6 packet and sends it to Device A.
2. Device A, a 6to4 router, encapsulates the outgoing IPv6 packet inside an IPv4 packet and sends it out the tunnel interface over the IPv4 WAN to Device B.
3. Device B decapsulates the packet and forwards it to Host B.
4. Host B generates a reply packet. Device B encapsulates it and forwards it to Device A.
5. Device A receives the encapsulated reply packet through the tunnel interface. Device A decapsulates the packet and forwards it to Host A.

In this example, you configure a device to send HTTP requests from IPv6 hosts to a native host (an HTTP server) over an IPv4 WAN infrastructure.

WebUI (Device A)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnel Interfaces

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

Network > Interfaces > Edit (for tunnel.1) > Encapsulation

Routers

Network > Routing > Routing Table > New (trust-vr)

Address Book Entries

Objects > Addresses > List (Trust) > New

Policy

Policies > (From: Untrust, To: Trust) > New

WebUI (Device B)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnel Interfaces

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

Network > Interfaces > Edit (for tunnel.1) > Encapsulation

Routers

Network > Routing > Routing Table > New (trust-vr)

Network > Routing > Routing Table > trust-vr New: Enter the following, then click **OK**:

Network > Routing > Routing Table > trust-vr Edit > Gateway: (select)

Address Book Entries

Objects > Addresses > List (Trust) > New

Objects > Addresses > List (Trust) > New

Policy

Policies > (From: Untrust, To: Trust) > New: Enter the following, then click **OK**:

Policies > (From: Untrust, To: Trust) > Edit > Advanced:

CLI (Device A)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.1.1/24
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 2002:101:101::1/48
set interface ethernet2 ipv6 ra link-mtu
set interface ethernet2 ipv6 ra link-address
set interface ethernet2 ipv6 ra transmit
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
```

2. Tunnel Interface

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
set interface tunnel.1 tunnel encap ip6in4 6to4
set interface tunnel.1 tunnel local-if ethernet3
```

3. Routers

```
set vrouter trust-vr route 0.0.0.0/0 gateway 1.1.1.250
set vrouter trust-vr route 3abc::/16 interface tunnel.1 gateway 2002:0202:201::1
```

4. Address Book Entries

```
set address trust L_Hosts 2002:101:101::/48
set address untrust R_Server 3abc::100/128
```

5. Policy

```
set policy from trust to untrust L_Hosts R_Server http permit
```

CLI (Device B)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.2.2.1/24
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 310a::33c:3efb:34ff:fe01/64
set interface ethernet2 ipv6 ra link-mtu
set interface ethernet2 ipv6 ra link-address
set interface ethernet2 ipv6 ra transmit
set interface ethernet3 zone untrust
set interface ethernet3 ip 2.2.2.1/24
```

2. Tunnel Interface

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
set interface tunnel.1 tunnel encap ip6in4 6to4
set interface tunnel.1 tunnel local-if ethernet3
```

3. Routers

```
set vrouter trust-vr route 0.0.0.0/0 gateway 2.2.2.250
set vrouter trust-vr route 2002::/16 interface tunnel.1
```

4. Address Book Entries

```
set address trust L_Server 310a::33c:3efb:34ff:fe55/128
set address untrust R_Clients 2002:101:101::/48
```

5. Policy

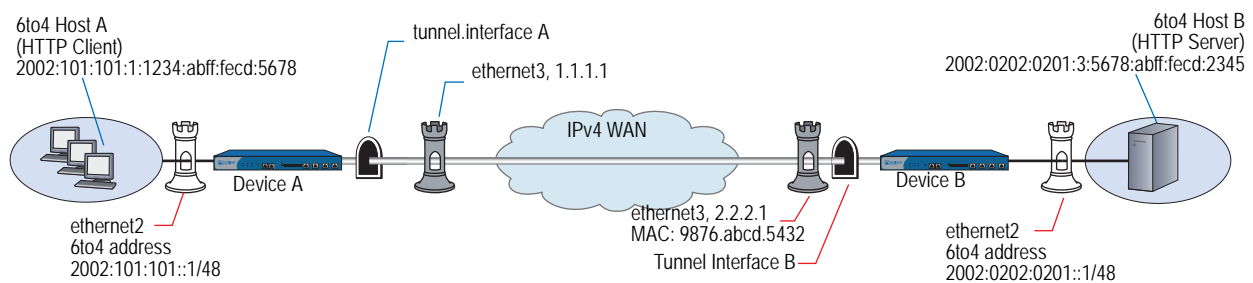
```
set policy from untrust to trust R_Clients L_Server http permit
```

Tunnels to Remote 6to4 Hosts

When a 6to4 host sends a service request to remote a 6to4 host, the device derives the remote termination point (gateway IPv4 address) from the destination IP address of the outgoing request packet.

Figure 31 shows Host A sending an HTTP service request to a server protected by Device B.

Figure 31: 6to4 Tunnel



Trust and Untrust zones are not shown.

In this example, the 6to4 destination address of the outgoing packet is 2002:0202:0201:3:5678:abff:fece:2345, which is the address of 6to4 Host B. The 6to4 router (on Device A) derives the remote gateway IPv4 address (2.2.2.1) from this destination address.

A device addresses packets that pass between a 6to4 host and another 6to4 host over an IPv4 WAN in the following manner:

1. The IPv6 Host A generates an IPv6 packet and sends it to Device A.
2. Device A derives the IPv4 destination address of the remote gateway (2.2.2.1) from the destination address of the outgoing packet.
3. Device A encapsulates the outgoing 6to4 packet inside an IPv4 packet and sends it out the tunnel interface into Zone B.
4. Device B decapsulates the packet and forwards it to Host B.
5. Host B generates a reply packet.
6. Device B encapsulates it and forwards it to Device A.

In this example, you configure a device to send HTTP requests from IPv6 hosts to an IPv6 server over an IPv4 WAN infrastructure.

WebUI (Device A)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnel Interfaces

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

Network > Interfaces > Edit (for tunnel.1) > Encapsulation

Routers

Network > Routing > Routing Table > New (trust-vr)

Network > Routing > Routing Table > New (untrust-vr)

Address Book Entries

Objects > Addresses > List (Trust) > New

Policy

Policies > (From: Untrust, To: Trust) > New

WebUI (Device B)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnel Interfaces

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

Network > Interfaces > Edit (for tunnel.1) > Encapsulation

Routers

Network > Routing > Routing Table > untrust-vr New

Address Book Entries

Objects > Addresses > List (Trust) > New

Objects > Addresses > List (Untrust) > New

Policy

Policies > (From: Untrust, To: Trust) > New

CLI (Device A)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.1.1/24
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 2002:101:101::1/48
set interface ethernet2 ipv6 ra link-mtu
set interface ethernet2 ipv6 ra link-address
set interface ethernet2 ipv6 ra transmit
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
```

2. Tunnel Interface

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
set interface tunnel.1 tunnel encap ip6in4 6to4
set interface tunnel.1 tunnel local-if ethernet3
```

3. Routers

```
set vrouter trust-vr route 2002:0202:0201::/48 interface tunnel.1
```

4. Address Book Entries

```
set address trust L_Hosts 2002:101:101::/48
set address untrust R_Server 2002:202:201:0:5678:abff:fezd:2345/128
```

5. Policy

```
set policy from trust to untrust L_Hosts R_Server any permit
```

CLI (Device B)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.2.2.1/24
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 2002:202:0201::1/48
set interface ethernet2 ipv6 ra link-mtu
set interface ethernet2 ipv6 ra link-address
set interface ethernet2 ipv6 ra transmit
set interface ethernet3 zone untrust
set interface ethernet3 ip 2.2.2.1/24
```

2. Tunnel Interface

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
set interface tunnel.1 tunnel encap ip6in4 6to4
set interface tunnel.1 tunnel local-if ethernet3
```

3. Routers

```
set vrouter trust-vr route 2002::/16 interface tunnel.1
```

4. Address Book Entries

```
set address untrust R_Clients 2002:101:101::/48
set address trust L_Server 2002:202:0201:3:5678:abff:fezd:2345/128
```

5. Policy

```
set policy from untrust to trust R_Clients L_Server http permit
```

Chapter 7

IPSec Tunneling

6in6, 4in6, and 6in4 tunneling allow you to create virtual private networks. These mechanisms automatically perform packet encapsulation and ensure secure packet exchange over the WAN.

This chapter contains the following sections:

- “Overview” on page 112
- “IPSec 6in6 Tunneling” on page 112
- “IPSec 4in6 Tunneling” on page 115
- “IPSec 6in4 Tunneling” on page 120
- “Manual Tunneling with Fragmentation Enabled” on page 124
 - “IPv6 to IPv6 Route-Based VPN Tunnel” on page 125
 - “IPv4 to IPv6 Route-Based VPN Tunnel” on page 127

Overview

ScreenOS allow you to create virtual private networks (VPNs). A VPN connection can link two local area networks (LANs) or a remote dialup user with a LAN. The traffic that flows between these two points passes through shared resources such as routers, switches, and other network equipment that make up the public WAN. To secure VPN communication traffic, the two participants create an IP Security (IPSec) tunnel.

You create policies to provide IPSec security services such as authentication and encryption or use Network Address Translation (NAT) to define private address space. For more information about NAT, see *Volume 8: Address Translation*. For information about security concepts and VPNs, see *Volume 2: Fundamentals* and *Volume 5: Virtual Private Networks*.

This chapter includes configuration examples that use VPN tunnels in a purely IPv6 environment and other scenarios for IPv4 or IPv6 island networks or hosts.

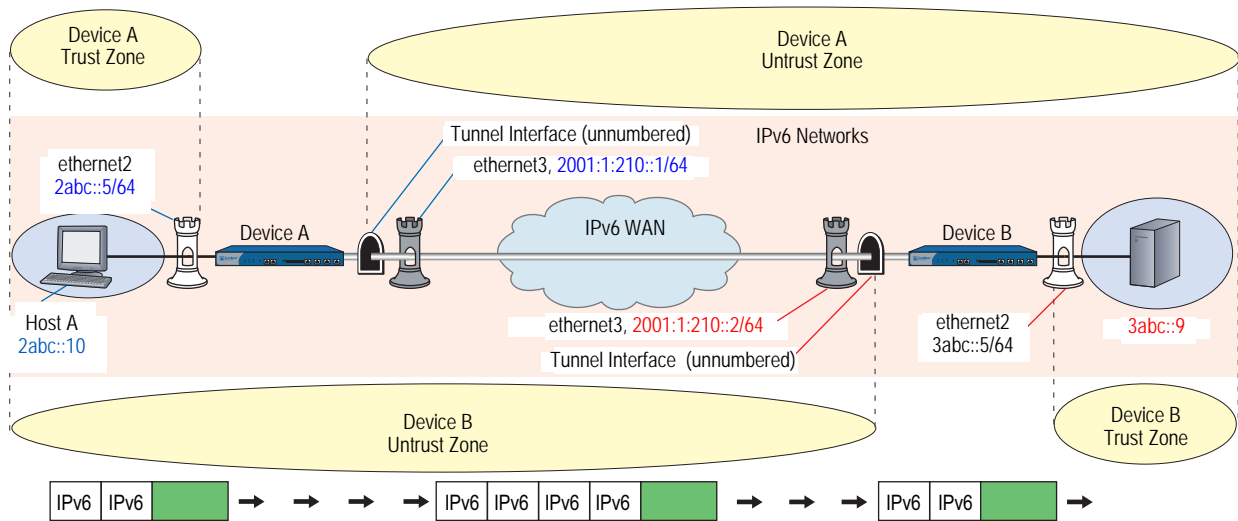
IPSec 6in6 Tunneling

6in6 tunneling is a method to encapsulates IPv6 packets inside IPv6 packets and is used where the exchanged information must travel within an IPv6 domain and requires protection to ensure data confidentiality and integrity.

Use IPSec in combination with 6in6 tunneling when you need to establish secure communication between IPv6 hosts and other IPv6 hosts over an IPv6 infrastructure. For example, you might need to exchange secured traffic between an IPv6 network and an IPv6 server over an IPv6 WAN.

Figure 32 shows an IPv6 host sending an HTTP service request to an IPv6 webserver over an IPv6 WAN infrastructure. Both devices have policies that use IPSec security.

Figure 32: IPSec with 6in6 Tunnel Example



The following lists the steps to how a device configured for IPv4-to-IPv6 host mapping translates the source and destination addresses of an outgoing service request packet.

1. Host A generates the service request packet and assigns it the IPv6 destination address (3abc::9).
2. Device A encapsulates the entire IPv6 packet inside of another IPv6 packet. It then sends it out the tunnel interface to Device B.
3. The remote IPv6 host device sends a reply packet.
4. Device A decapsulates the replay packet and sends it to the IPv6 host.

In the following example, you set up IPSec policies that allow HTTP communication between an IPv6 network and a remote IPv6 network or a subnet. The traffic passes over an IPv6 WAN environment.

WebUI (Device A)

1. Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (For tunnel.1) > IPv6

3. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit (for gw-test) > Advanced

4. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit (For vpn-test) > Advanced

5. Route

Network > Routing > Routing Entries > New

6. Policies

Policies > New (Trust to Untrust)

Policies > New (Untrust to Trust)

CLI (Device A)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 2abc::5/64
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 ip 2001:1:210::1/64
```

2. Tunnel Interface

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
```

3. IKE

```
set ike gateway gw-test address 2001:1:210::2 main outgoing-interface ethernet3
local-address 2001:1:210::1 preshare abcd1234 proposal pre-g2-des-md5
```

4. VPN

```
set vpn vpn-test gateway gw-test proposal g2-esp-des-md5
set vpn vpn-test bind interface tunnel.1
set vpn vpn-test proxy-id local-ip ::/0 remote-ip ::/0 any
```

5. Route

```
set route 3abc::/16 interface tunnel.1
```

6. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

WebUI (Device B)

1. Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

2. Tunnel Interface

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (For tunnel.1) > IPv6

3. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit (for gw-test) > Advanced

4. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit (For vpn-test) > Advanced

5. **Route**
Network > Routing > Routing Entries > New

6. **Policies**
Policies > New (Trust to Untrust)

Policies > New (Untrust to Trust)

CLI (Device B)

1. **Interfaces**

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 3abc::5/64
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 ip 2001:1:210::2/64
```

2. **Tunnel Interface**

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
```

3. **IKE**

```
set ike gateway gw-test address 2001:1:210::1 main outgoing-interface ethernet3
local-address 2001:1:210::2 preshare abcd1234 proposal pre-g2-des-md5
```

4. **VPN**

```
set vpn vpn-test gateway gw-test proposal g2-esp-des-md5
set vpn vpn-test bind interface tunnel.1
set vpn vpn-test proxy-id local-ip ::/0 remote-ip ::/0 ANY
```

5. **Route**

```
set route 2abc::/16 interface tunnel.1
```

6. **Policies**

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

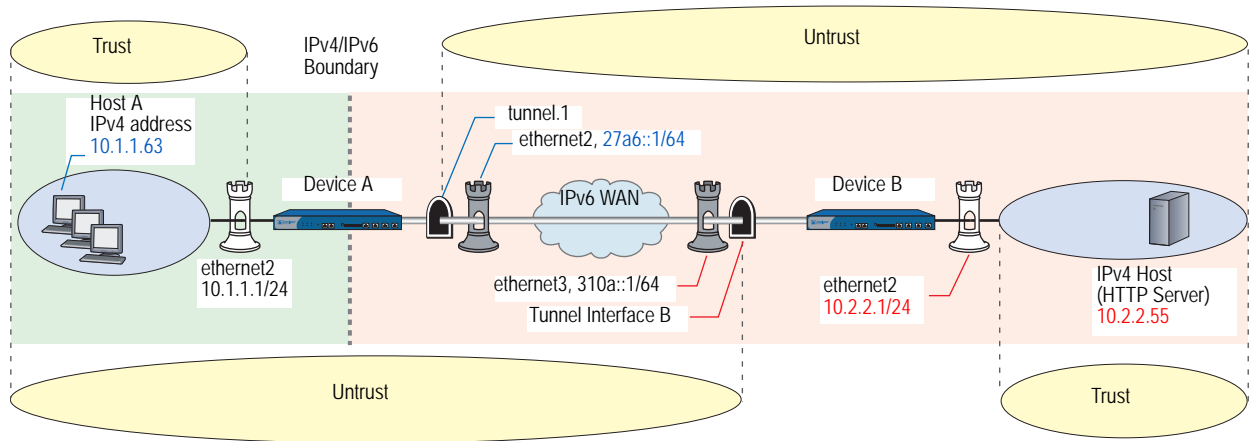
IPSec 4in6 Tunneling

4in6 tunneling allows you to establish communication between IPv4 networks over an IPv6 backbone. This transition mechanism uses IPSec to encapsulate packets exchanged between IPv4 networks over an IPv6 backbone. Encapsulation makes exchanged packets routable across the IPv6 network space.

In addition to encapsulating the packets, IPSec can perform authentication and encryption.

Figure 33 shows an IPSec tunnel between remote IPv4 networks. In this example, you define a tunnel interface on each gateway device and bind the tunnel to a zone that borders IPv6 network space.

Figure 33: IPSec 4in6 Tunnel Example



A device addresses packets that pass between an IPv4 host in an IPv4 island network and another IPv4 host over an IPv6 WAN backbone in the following manner:

1. Host A generates an IPv6 service request packet and addresses it to Host B. The packet goes from Zone A to Zone B.
2. Device A encapsulates the outgoing IPv4 packet inside an IPv6 packet and sends it out the tunnel interface.
3. Device B decapsulates the packet and forwards it to Host B.
4. Device B generates a reply packet, encapsulates it, and forwards it to Device A.
5. Device A receives the encapsulated reply packet through the tunnel interface. Device A decapsulates the packet and forwards it to Host A.

The steps to set up 4in6 tunneling are as follows:

1. Configure an interface for communication with the protected IPv4 network.
 - Bind the interface to a zone (typically the Trust zone).
 - Assign the interface an IPv4 address and subnet mask.
2. Configure an interface for communication over the IPv6 WAN.
 - Bind the interface to a zone (typically the Untrust zone).
 - Configure the interface for IPv6 Host mode.
 - Assign the interface an IPv6 address and subnet mask.
 - Create a tunnel interface (unnumbered) in the zone and bind it to the IPv6 interface.
 - Configure the tunnel interface for IPv6 Host mode.

3. Set up interfaces on the peer device in a similar manner.
4. Set up IPSec between the peer devices. For more examples of IPSec, see Volume 5: *Virtual Private Networks*.
5. On each device, create address book entries that identify the IPv6 host, subnet, or network.
6. Set up routing entries that allow the hosts to access each other.
7. Set up security policies.

In the following example, you create an IPSec tunnel between IPv6 endpoints. IPv4 devices behind Device A transmit service requests hosts behind Device B. Device A use IPSec to encapsulate the service request packets inside IPv6 packets. Device B receives and decapsulates the packets.

WebUI (Device A)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

IKE

VPNs > Autokey Advanced > Gateway > New

VPN

VPNs > Autokey IKE > New

Routers

Network > Routing > Routing Table > New (trust-vr)

Network > Routing > Routing Table > New (untrust-vr)

Policy

Policies > (From: Trust, To: Untrust) > New

CLI (Device A)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.1.1/24
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 ip 27a6::1/64
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
```

3. IKE

```
set ike gateway IPSec_Servers ip 310a::1 main outgoing-interface ethernet3
local-address 27a6::1 proposal rsa-g2-aes128-sha
```

4. VPN

```
set vpn Tunnel_Servers gateway IPSec_Servers no-replay tunnel sec-level standard
set vpn Tunnel_Servers bind interface tunnel.1
```

5. Routes

```
set vrouter trust-vr route 0.0.0.0/0 interface ethernet3
set vrouter trust-vr route 310a::0/64 interface tunnel.1 gateway 310a::1
```

6. Policy

```
set policy from trust to untrust any any any permit
```

WebUI (Device B)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet3)

Network > Interfaces > Edit (for ethernet3) > IPv6

Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.1) > IPv6

IKE

VPNs > Autokey Advanced > Gateway > New

VPN

VPNs > Autokey IKE > New

Routers

Network > Routing > Routing Table > New (trust-vr)

Addresses

Objects > Addresses > List (Trust) > New

Objects > Addresses > List (Untrust) > New

Policy

Policies > (From: Trust, To: Untrust) > New

CLI (Device B)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.2.2.1/24
set interface ethernet3 zone untrust
set interface ethernet3 ip 2.2.2.1/24
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 ip 310a::1/96
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
```

3. IKE

```
set ike gateway IPSec_Clients ip 27a6::1 main outgoing-interface ethernet3
local-address 310a::1 proposal rsa-g2-aes128-sha
```

4. VPN

```
set vpn Tunnel_Clients id 1 gateway IPSec_Clients no-replay tunnel sec-level
standard
set vpn Tunnel_Clients id 2 bind interface tunnel.1
```

5. Routes

```
set vrouter trust-vr route 0.0.0.0/0 vrouter untrust-vr
set vrouter untrust-vr route 27a6::0/64 interface tunnel.1 gateway 27a6::1
```

6. Addresses

```
set address trust L_Server 10.2.2.55/32
set address untrust R_Clients 10.1.1.1/24
```

7. Policy

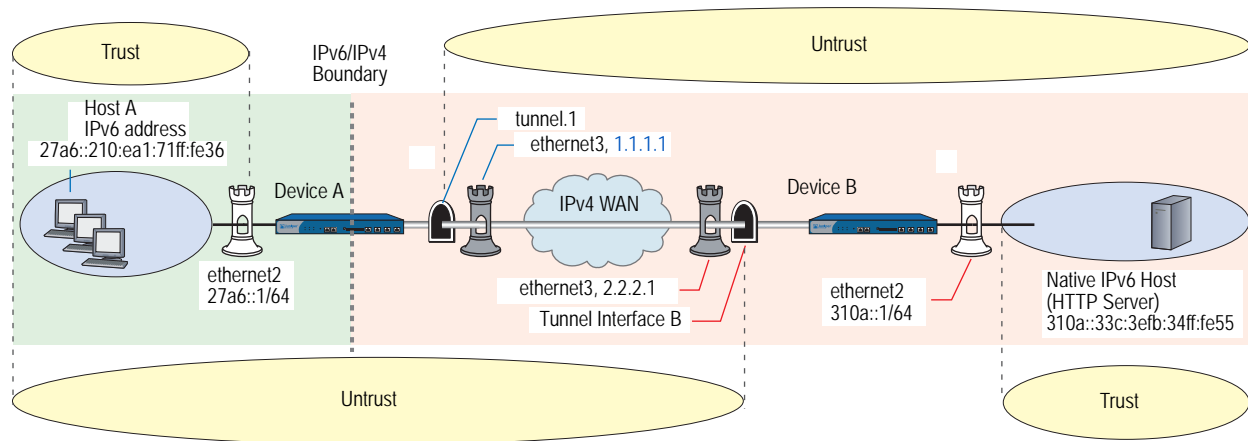
```
set policy from untrust to trust R_Clients L_Server any permit
```

IPSec 6in4 Tunneling

You can use IPSec tunneling, which supports authentication and encryption, to encapsulate packets as the security device transmits them between remote IPv6 island networks over an IPv4 WAN.

Figure 34 shows an IPSec tunnel between remote IPv6 island networks. In this example, you first define a tunnel interface on each gateway device; then you bind the tunnel to a zone that borders IPv4 network space.

Figure 34: Tunnel Interface and Zone Example



A device addresses packets that pass between a host in an IPv6 island network and another IPv6 host over an IPv4 WAN backbone in the following manner:

1. Host A generates an IPv6 service request packet and addresses it to Host B. The packet goes from Zone A to Zone B.
2. Device A encapsulates the outgoing IPv6 packet inside an IPv4 packet and sends it out the tunnel interface into Zone B.
3. Device B decapsulates the packet and forwards it to Host B.
4. Device B generates a reply packet, encapsulates it, and forwards it to Device A.
5. Device A receives the encapsulated reply packet through the tunnel interface. Device A decapsulates the packet and forwards it to Host A.

In most cases, the necessary setup tasks are as follows:

1. Configure an interface for communication with the protected island IPv6 network.
 - Bind the interface to a zone (typically the Trust zone).
 - Configure the interface for IPv6, Host mode.
 - Assign the interface a global unicast prefix. (For information about global unicast addresses, see “Address Types” on page 3.)

2. Configure an interface for communication over the IPv4 WAN.
 - Bind the interface to a zone (typically the Untrust zone).
 - Assign the interface an IPv4 address and subnet mask.
 - Create a tunnel interface (unnumbered) in the zone, and bind it to the IPv4 interface.
3. Follow steps 1 and 2 to set up interfaces on the peer device.
4. Set up IPSec between the peer devices. For more information about IPSec, see Volume 5: *Virtual Private Networks*.
5. On each device, create address book entries that identify the IPv6 host, subnet, or network.
6. Set up routing entries that allow the hosts to access each other.
7. Set up security policies.

In the following example, you create an IPSec tunnel between IPv4 endpoints. IPv6 devices behind Device A transmit service requests to hosts behind Device B. Device A use IPSec to encapsulate the service request packets inside IPv4 packets. Device B receives and decapsulates the packets. See Figure 34 on page 120.

The devices perform Phase 1 of the AutoKey IKE tunnel negotiation as follows:

- RSA authentication
- Diffie-Hellman Group 2
- AES128 encryption algorithm
- SHA hashing algorithm

The devices perform Phase 2 of the tunnel negotiation uses the Standard proposal.

- Diffie-Hellman Group 2
- ESP (Encapsulating Security Payload) tunneling
- 3DES encryption algorithm
- SHA hashing algorithm

WebUI (Device A)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnels

Network > Interfaces > New (Tunnel IF)

IKE

VPNs > Autokey Advanced > Gateway > New

VPN

VPNs > Autokey IKE > New

Routers

Network > Routing > Routing Table > New (trust-vr)

Addresses

Objects > Addresses > List (Trust) > New

Objects > Addresses > List (Untrust) > New

Policy

Policies > (From: Trust, To: Untrust) > New

CLI (Device A)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.1.1/24
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 27a6::1/64
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
```

3. IKE

```
set ike gateway IPSec_Servers ip 2.2.2.1 main outgoing-interface ethernet3
proposal rsa-g2-aes128-sha
```

4. VPN

```
set vpn Tunnel_Servers gateway IPSec_Servers no-replay tunnel sec-level standard
set vpn Tunnel_Servers bind interface tunnel.1
```

5. Routes

```
set vrouter trust-vr route 0.0.0.0/0 interface ethernet3
set vrouter trust-vr route ::/0 interface tunnel.1
```

6. Addresses

```
set address trust L_Clients 27a6::210:ea1:71ff:fe36/64
set address untrust R_Servers 32f1::250:af:34ff:fe34/64
```

7. Policy

```
set policy from trust to untrust L_Clients R_Servers any permit
```

WebUI (Device B)

Interfaces

Network > Interfaces > Edit (for ethernet2)

Network > Interfaces > Edit (for ethernet2) > IPv6

Network > Interfaces > Edit (for ethernet3)

Tunnels

Network > Interfaces > New (Tunnel IF)

IKE

VPNs > Autokey Advanced > Gateway > New

VPN

VPNs > Autokey IKE > New

Routers

Network > Routing > Routing Table > New (trust-vr)

Network > Routing > Route Table > New (untrust-vr)

Addresses

Objects > Addresses > List (Trust) > New

Objects > Addresses > List (Untrust) > New

Policy

Policies > (From: Untrust, To: Trust) > New

CLI (Device B)

1. Interfaces

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.2.2.1/24
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 ip 310a::1/64
set interface ethernet3 zone untrust
set interface ethernet3 ip 2.2.2.1/24
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ip unnumbered interface ethernet3
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
```

3. **IKE**

```
set ike gateway IPSec_Clients ip 1.1.1.1 main outgoing-interface ethernet3
proposal rsa-g2-aes128-sha
```
4. **VPN**

```
set vpn Tunnel_Clients id 1 gateway IPSec_Clients no-replay tunnel sec-level
standard
set vpn Tunnel_Clients id 2 bind interface tunnel.1
```
5. **Routes**

```
set vrouter trust-vr route 0.0.0.0/0 interface ethernet3
set vrouter trust-vr route ::/0 interface tunnel.1
```
6. **Addresses**

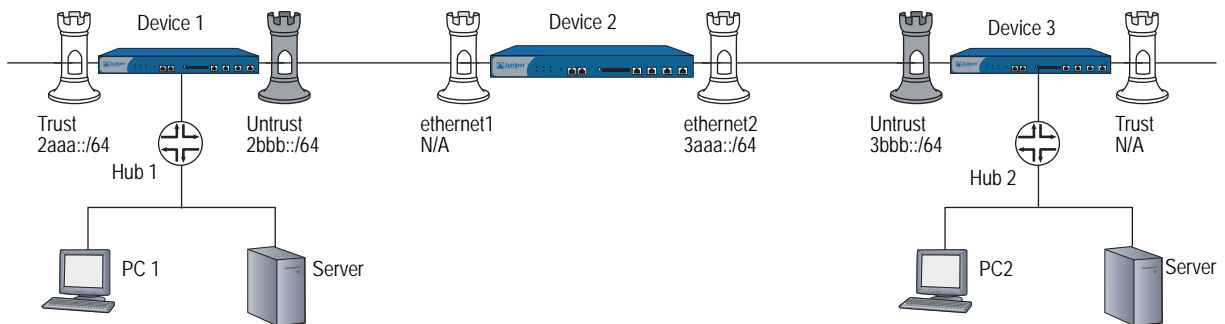
```
set address trust L_Server 310a::33c:3efb:34ff:fe55/128
set address untrust R_Clients 27a6::210:ea1:71ff:fe36/64
```
7. **Policy**

```
set policy from untrust to trust R_Clients L_Server any tunnel vpn Tunnel_Clients
```

Manual Tunneling with Fragmentation Enabled

Figure 35 shows the configuration of the network used in the following tunneling examples.

Figure 35: Manual Tunneling Example



IPv6 to IPv6 Route-Based VPN Tunnel

In the following example, you send IPv6 packets through an IPv6 tunnel, with a route-based VPN that uses 3DES encryption, and you enable fragmentation.

CLI (Device 1)

1. General

```
set console time 0
unset zone untrust block
```

2. Interfaces

```
set interface ethernet1/1 zone trust
set interface ethernet1/1 ipv6 mode router
set interface ethernet1/1 ipv6 ip 2aaa::/64
set interface ethernet1/1 ipv6 enable
set interface ethernet1/1 ipv6 interface-id 0000000000000001
set interface ethernet1/1 ipv6 ra transmit
set interface ethernet1/1 route
set interface ethernet1/1 manage
```

```
set interface ethernet1/2 zone untrust
set interface ethernet1/2 ipv6 mode host
set interface ethernet1/2 ipv6 ip 2bbb::/64
set interface ethernet1/2 ipv6 enable
set interface ethernet1/2 ipv6 interface-id 0000000000000002
set interface ethernet1/2 route
set interface ethernet1/2 manage
```

3. IKE

```
set ike gateway ton6 address 3aaa::5 outgoing-interface ethernet1/2 local-address
2bbb::2 preshare abc sec-level standard
set vpn vpn4 gateway ton6 sec-level standard
```

4. Tunnel

```
set interface tunnel.6 zone untrust
set interface tunnel.6 ipv6 mode host
set interface tunnel.6 ipv6 enable
set interface tunnel.6 ip unnumbered interface ethernet1/2
set vpn vpn4 bind interface tunnel.6
```

5. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

6. Routes

```
set vrouter trust-vr route ::/0 interface ethernet1/2 gateway 2bbb::3
set vrouter trust-vr route 3bbb::/64 interface tunnel.6
```

CLI (Device 2)

1. General

set console time 0
unset zone untrust block

2. Interfaces

set interface untrust zone untrust
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 ip 2bbb::/64
set interface untrust ipv6 interface-id 0000000000000003
set interface untrust manage
set interface untrust route

set interface trust zone trust
set interface trust ipv6 mode host
set interface trust ipv6 ip 3aaa::/64
set interface trust ipv6 enable
set interface trust ipv6 interface-id 0000000000000004
set interface trust route
set interface trust manage

3. Routes

set vrouter trust-vr route 3bbb::/64 interface trust gateway 3aaa::5
set vrouter trust-vr route 2aaa::/64 interface untrust gateway 2bbb::2

4. Policies

set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit

CLI (Device 3)

1. General

set console time 0
unset zone untrust block

2. Interfaces

set interface untrust zone untrust
set interface untrust ipv6 mode host
set interface untrust ipv6 ip 3aaa::/64
set interface untrust ipv6 interface-id 0000000000000005
set interface untrust ipv6 enable
set interface untrust route
set interface untrust manage

set interface trust zone trust
set interface trust ipv6 mode router
set interface trust ipv6 ip 3bbb::/64
set interface trust ipv6 interface-id 0000000000000006
set interface trust ipv6 enable
set interface trust ipv6 ra transmit
set interface trust route
set interface trust manage

3. Tunnel

```
set interface tunnel.6 zone untrust
set interface tunnel.6 ipv6 mode host
set interface tunnel.6 ipv6 enable
set interface tunnel.6 ip unnumbered interface untrust
```

4. IKE

```
set ike gateway ton4 address 2bbb::2 outgoing-interface untrust local-address 3aaa::5
  preshare abc sec-level standard
```

5. VPN

```
set vpn vpn6 gateway ton4 sec-level standard
set vpn vpn6 bind interface tunnel.6
```

6. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

7. Routes

```
set vrouter trust-vr route ::/0 interface untrust gateway 3aaa::4
set vrouter trust-vr route 2aaa::/64 interface tunnel.6
```

IPv4 to IPv6 Route-Based VPN Tunnel

In the following example, you send IPv4 packets through an IPv4 tunnel, with a route-based VPN that uses 3DES encryption, and you enable fragmentation.

CLI (Device 1)

1. General

```
set console time 0
unset zone untrust block
```

2. Interfaces

```
set interface ethernet1/1 zone trust
set interface ethernet1/1 ipv6 mode router
set interface ethernet1/1 ipv6 ip 2aaa::/64
set interface ethernet1/1 ipv6 enable
set interface ethernet1/1 ipv6 interface-id 0000000000000001
set interface ethernet1/1 ipv6 ra transmit
set interface ethernet1/1 route
set interface ethernet1/1 manage
```

```
set interface ethernet1/2 zone untrust
set interface ethernet1/2 ipv6 mode router
set interface ethernet1/2 ipv6 ip 2bbb::/64
set interface ethernet1/2 ipv6 enable
set interface ethernet1/2 ipv6 interface-id 0000000000000002
set interface ethernet1/2 ipv6 ra transmit
set interface ethernet1/2 route
set interface ethernet1/2 manage
```

3. IKE

```
set ike gateway ton6 address 3aaa::5 outgoing-interface ethernet1/2 local-address
  2bbb::2 preshare abc sec-level standard
set vpn vpn4 gateway ton6 sec-level standard
```

4. Tunnel

```
set interface tunnel.6 zone untrust
set interface tunnel.6 ipv6 mode router
set interface tunnel.6 ipv6 enable
set interface tunnel.6 ip unnumbered interface ethernet1/2
set interface tunnel.6 ipv6 ra transmit
```

5. VPN

```
set vpn vpn4 bind interface tunnel.6
```

6. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

7. Routes

```
set vrouter trust-vr route ::/0 interface ethernet1/2 gateway 2bbb::3
set vrouter trust-vr route 3bbb::/64 interface tunnel.6
set interface ethernet1/1 ip 1.1.1.1/24
set route 4.1.1.0/24 interface tunnel.6
```

8. Policies

```
set policy from trust to untrust any any any permit
set policy from untrust to trust any any any permit
```

CLI (Device 2)

1. General

```
set console time 0
unset zone untrust block
```

2. Interfaces

```
set interface untrust zone untrust
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 0000000000000003
set interface untrust ipv6 ra accept
set interface untrust manage
set interface untrust route
```

```
set interface trust zone trust
set interface trust ipv6 mode router
set interface trust ipv6 ip 3aaa::/64
set interface trust ipv6 enable
set interface trust ipv6 interface-id 0000000000000004
set interface trust ipv6 ra transmit
set interface trust route
set interface trust manage
```

3. Routes

```
set vrouter trust-vr route 3bbb::/64 interface trust gateway 3aaa::5
set vrouter trust-vr route 2aaa::/64 interface untrust gateway 2bbb::2
```

4. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

CLI (Device 3)

1. General

```
set console time 0
unset zone untrust block
```

2. Interfaces

```
set interface untrust zone untrust
set interface untrust ipv6 mode host
set interface untrust ipv6 interface-id 0000000000000005
set interface untrust ipv6 enable
set interface untrust ipv6 ra accept
set interface untrust route
set interface untrust manage
```

```
set interface trust zone trust
set interface trust ipv6 mode router
set interface trust ipv6 ip 3bbb::/64
set interface trust ipv6 interface-id 0000000000000006
set interface trust ipv6 enable
set interface trust ipv6 ra transmit
set interface trust route
set interface trust manage
```

3. Tunnel

```
set interface tunnel.6 zone untrust
set interface tunnel.6 ipv6 mode host
set interface tunnel.6 ipv6 enable
set interface tunnel.6 ip unnumbered interface untrust
set interface tunnel.6 ipv6 ra accept
```

4. IKE

```
set ike gateway ton4 address 2bbb::2 outgoing-interface untrust local-address 3aaa::5
  preshare abc sec-level standard
set vpn vpn6 gateway ton4 sec-level standard
set vpn vpn6 bind interface tunnel.6
```

5. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

```
set policy from trust to untrust any any any permit
set policy from untrust to trust any any any permit
```

6. Routes

```
set vrouter trust-vr route ::/0 interface untrust gateway 3aaa::4
set vrouter trust-vr route 2aaa::/64 interface tunnel.6
set interface trust ip 4.1.1.1/24
set route 1.1.1.0/24 interface tunnel.6
```


Chapter 8

IPv6 XAuth User Authentication

This chapter describes IPv6 user authentication and Dead Peer Detection (DPD) and provides configuration examples.

It contains the following sections:

- “Overview” on page 132
 - “RADIUSv6” on page 132
 - “IPSec Access Session Management” on page 134
 - “Dead Peer Detection” on page 137
- “Configuration Examples” on page 138
 - “XAuth with RADIUS” on page 138
 - “RADIUS with XAuth Route-Based VPN” on page 139
 - “RADIUS with XAuth and Domain Name Stripping” on page 143
 - “IP Pool Range Assignment” on page 147
 - “RADIUS Retries” on page 153
 - “Calling-Station-Id” on page 153
 - “IPSec Access Session” on page 154
 - “Dead Peer Detection” on page 163

Overview

ScreenOS XAuth for IPv6 allows you to authenticate individual users and user groups after Phase 1 IKE negotiations.

You can use XAuth to authenticate remote virtual private network (VPN) users (or user groups) individually instead of only authenticating VPN gateways and/or devices or to assign TCP/IP configuration information, such as IP address, netmask, DNS server, and WINS server assignments.

RADIUSv6

ScreenOS supports RADIUSv6, the next-generation version of RADIUS. When a device functions as an XAuth server, it sends a username and password to a RADIUS server enabled for IPv4, IPv6, or both.

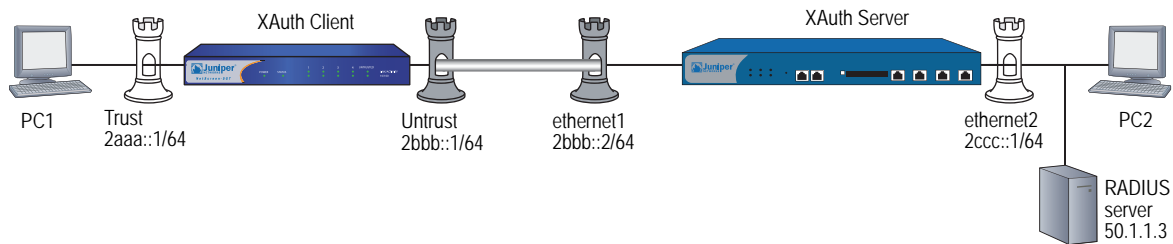
NOTE: For more information about RADIUS and IPv6, refer to RFC 3162 and RFC 2882.

The next sections show various scenarios. Configuration examples follow these scenarios.

Single Client, Single Server

Figure 36 shows a single client device and a single server interacting with a RADIUS server performing XAuth authentication.

Figure 36: RADIUS with a Single Client and Single Server

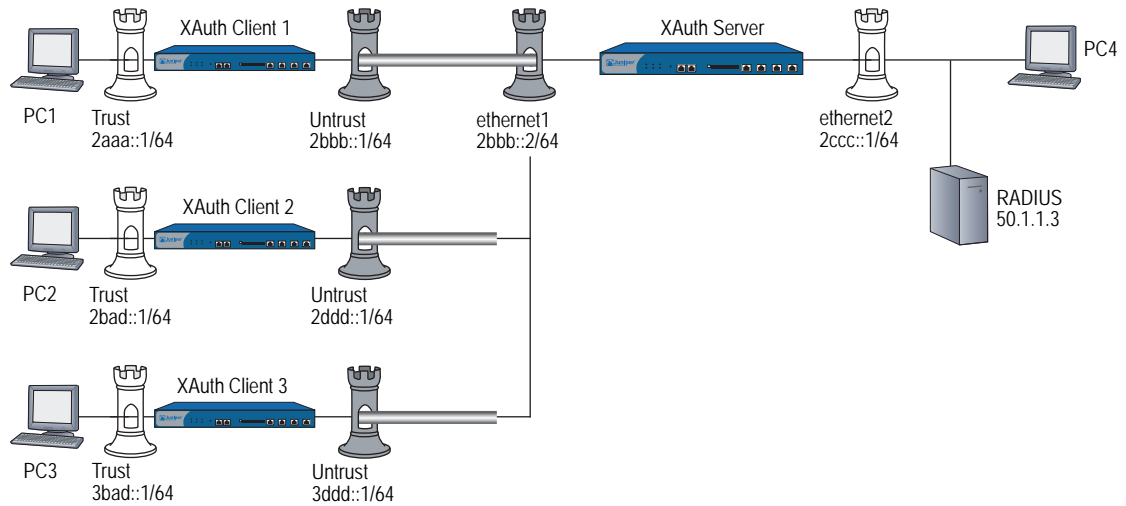


The RADIUS server resides in the Trust zone of the XAuth server. Each protected subnet can contain more than a single protected host. This figure shows only one.

Multiple Clients, Single Server

Figure 37 shows multiple client devices and a single server interacting with a RADIUS server performing XAuth authentication.

Figure 37: RADIUS with Multiple Clients and a Single Server

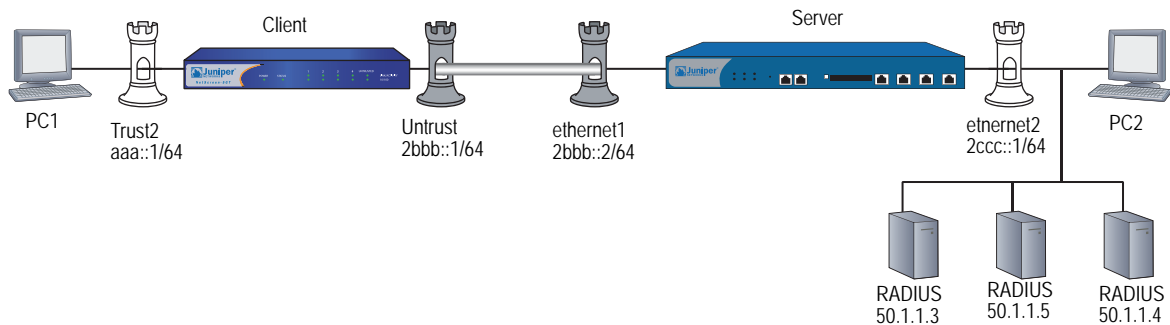


The RADIUS server resides in the Trust zone of the server. Each client device protects a different subnet. Each protected subnet can contain more than a single host, but the figure only shows one.

Single Client, Multiple Servers

Figure 38 shows a single client device and a single server interacting with multiple RADIUS servers.

Figure 38: RADIUS with a Single Client and Multiple Servers

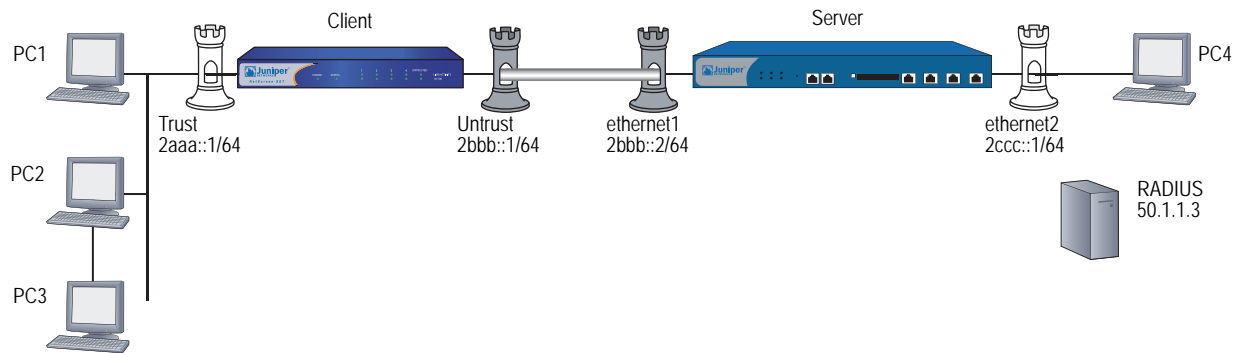


The RADIUS servers reside in the Trust zone of Device 2. Each protected subnet can contain more than a single host, but the figure only shows one.

Multiple Hosts, Single Server

Figure 39 shows a single client device (Device 1) protecting multiple hosts and a single server (Device 2) interacting with a RADIUS server.

Figure 39: RADIUS with Multiple Hosts and a Single Server



The RADIUS server resides in the Trust zone of the security server.

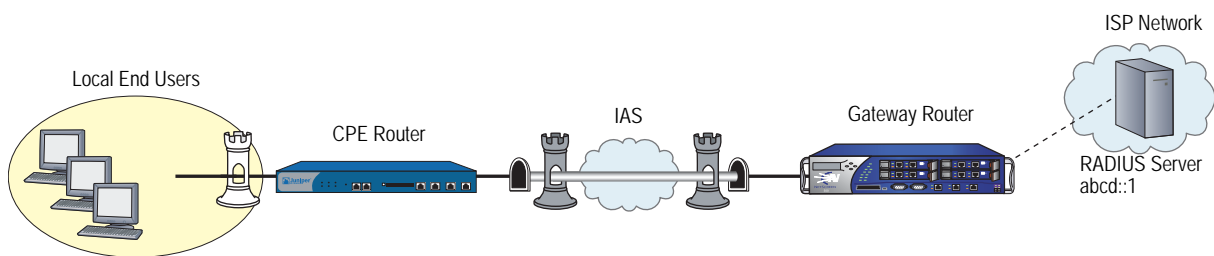
IPSec Access Session Management

You can use several security devices to allow user networks to access outside networks by setting up one or more local customer premises equipment (CPE) routers and one or more gateway routers. To protect packets transmitted over the access network, the CPE and gateway devices can establish an IPSec tunnel. The gateway, together with the RADIUS server manages and records accounting for the network access sessions.

IPSec Access Session

The time interval during which a network access session exists is called an IPSec Access Session (IAS). The IAS time interval begins when the first end user connects to the access network and ends when the last user disconnects from the network. Figure 40 shows how an ISP might use security devices as CPE and gateway routers.

Figure 40: IPSec Access Session with RADIUS Server



To initiate an IAS, the CPE device performs a Phase 1 Internet Key Exchange (IKE) and at least one Phase 2 IKE.

NOTE: For more information about ScreenOS security concepts and IKE, see “Internet Protocol Security” on page 5-1.

Each IAS has the following characteristics.

- A CPE device can have only one IAS Phase 1 Security Association (SA) for each IP address. For example, if the CPE device has three IPv6 addresses, it can host three Phase 1 SAs. However, there might be multiple Phase 2 SAs for each Phase 1 SA.
- During an IAS lifetime, the CPE and gateway devices might perform many IPsec operations, such as IKE Phase 1 and Phase 2 rekeying.
- During the IAS lifetime, the CPE initiates Phase 1 SA rekeys whenever the Phase 1 session expires.
- During the IAS lifetime, the CPE must initiate a new Phase 2 SA rekey *before* a Phase 2 session expires to ensure that SA is always on.

While it exists, each IAS contains and uses the following components:

Information Field	Description
CPE IPv6 Address	The address of the CPE device. The address is globally unique; that is, there cannot be more than one IAS with the same CPE IP address.
IKE Phase I ID	The ID that identifies the Phase 1 gateway for the IAS. This ID can consist of any of the following: <ul style="list-style-type: none"> ■ An alphanumeric string that identifies the gateway device ■ The ASN1 domain name of the gateway device ■ The Fully Qualified Domain Name (FQDN) of the gateway device ■ The IP address (IPv4 or IPv6) of the gateway device ■ The user-Fully Qualified Domain Name (u_FQDN) of the gateway device
XAuth User Name	The name of the XAuth user.
IKE Phase 1 Cookie Pair	The IKE Phase 1 SA cookies, which show the Phase 1 SA associated with the IAS.
Number of Phase 2 SAs	The number of Phase 2 SAs that currently exist in the IAS. For the IAS to exist, there must be one or more Phase 2 SAs.
Assigned IP address	The IP4 address assigned from the gateway to the CPE during the IKE mode configuration phase.
Phase 2 SA SPIs	Security Parameter Index (SPI) values, including both new and old SPIs.
IAS Start Time	The time the IAS began.

When a CPE gateway device initiates an IAS successfully, it starts RADIUS accounting for that IAS. When a Phase 2 SA expires or fails, the CPE device must establish another SA to continue secure communication with the gateway device. How the RADIUS server handles a Phase 2 rekey operation depends on whether IAS functionality is currently enabled or disabled.

When the IAS feature is disabled, the RADIUS accounting session starts over after the CPE establishes a new SA. Each time a Phase 2 session fails or times out, a new accounting session is necessary; and the RADIUS server must generate multiple billings (and, in some cases, other account-related features).

When the IAS feature is enabled, however, the original RADIUS accounting session resumes after each Phase 2 SA rekeying operation occurs. This allows the RADIUS server to maintain a single continuous accounting session to simplify billing and other accounting-related features.

Enabling and Disabling IAS Functionality

To enable the CPE gateway device for IAS functionality, enter the following command:

```
set ipsec-access-session enable
```

To disable the CPE gateway device for IAS functionality, enter the following command:

```
unset ipsec-access-session enable
```

Releasing an IAS Session

You can release an IPSec access session with the **clear ike all** command.

Limiting IAS Settings

The following settings limit the number of concurrently active IASs:

- The *maximum* specifies the maximum number of concurrent IASs the device allows. The default is 5000 sessions, and the range is from 0 to 5000 sessions.

To configure this limitation, enter the following command:

```
set ipsec access-session maximum maximum_number
```

- The *lower IAS threshold* specifies the minimum number of concurrent IASs the device allows before triggering a SNMP trap. The default is 1000 sessions, and the range is from 1 to 5000 sessions. This value must be less than the upper-threshold value.

To configure this limitation, enter the following command:

```
set ipsec access-session lower-threshold minimum_number
```

- The *upper IAS threshold* specifies the minimum number of concurrent IASs the device allows before triggering a SNMP trap. The default is 1000 sessions, and the range is from 1 to 5000 sessions. This value must be greater than the lower-threshold value.

To configure this limitation, enter the following command:

```
set ipsec access-session upper-threshold maximum_number
```

Dead Peer Detection

Dead Peer Detection (DPD) is a protocol that verifies the existence and liveliness of other IPsec peer devices when no incoming IPsec traffic from peers during a specified time interval exists.

A device performs DPD by sending encrypted IKE Phase 1 notification payloads (R-U-THERE) to peers and waiting for DPD acknowledgements (R-U-THERE-ACK) from its peers. The device sends an R-U-THERE request only if it has not received any traffic from the peer during a specified DPD interval. If a DPD-enabled device receives traffic on a tunnel, it resets the R-U-THERE counter for that tunnel to start a new interval. If the device receives an R-U-THERE-ACK from the peer device during this interval, it determines the peer to be alive. If the device does not receive an R-U-THERE-ACK response during the interval, it determines the peer to be dead.

The device removes the Phase 1 SA and all Phase 2 SAs for dead peers.

You can configure the following DPD parameters with the CLI or the WebUI:

- The interval parameter specifies the DPD interval. This interval is the amount of time (expressed in seconds) the device allows to pass before considering a peer to be dead. A setting of zero disables DPD.
- The always-send parameter instructs the device to send DPD requests regardless of whether there is IPsec traffic with the peer.
- The retry parameter specifies the maximum number of times to send the R-U-THERE request before considering the peer to be dead. As with an IKE heartbeat configuration, the default number of transmissions is 5 times, with a permissible range from 1 to 128 attempts.

NOTE: You can enable DPD or IKE heartbeat but not both. If you attempt to configure DPD after enabling the IKE heartbeat, the security device generates the following error message:

IKE: DPD cannot co-exist with IKE heartbeat.

The device generates a similar error if you attempt to configure an IKE heartbeat after enabling DPD.

Configuration Examples

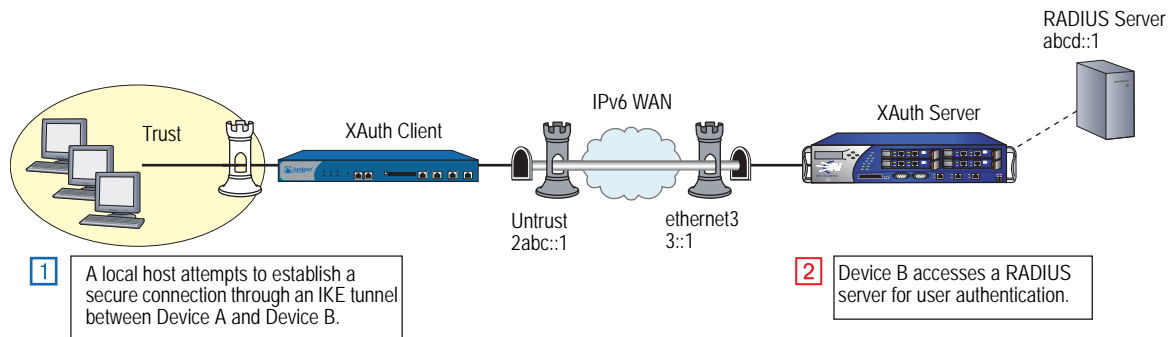
This section contains examples for RADIUS, IAS, and DPD.

NOTE: The WebUI section of each example lists only the navigational paths to the device configuration pages. For specific values, see the CLI section that follows it.

XAuth with RADIUS

Figure 41 shows device A performing XAuth authentication on users in its local network and a device B using a RADIUS server to perform XAuth in response to authentication through an IKE tunnel.

Figure 41: XAuth Example



WebUI (XAuth Client)

- VPNs > AutoKey IKE > New
- VPNs > AutoKey IKE > Edit
- VPNs > AutoKey IKE > Advanced

CLI (XAuth Client)

```
set ike gateway client_gw6 address 3::1 main outgoing-interface untrust
local-address 2abc::1 preshare 1234 proposal pre-g2-des-md5
set ike gateway client_gw6 cert peer-cert-type x509-sig
set ike gateway client_gw6 xauth client any username chris password swordfish
```

WebUI (XAuth Server)

1. **RADIUS**
Configuration > Auth > Auth Servers > Edit
2. **IKE**
VPNs > AutoKey IKE > New
VPNs > AutoKey IKE > Edit
VPNs > AutoKey IKE > Advanced

CLI (XAuth Server)

1. RADIUS

```
set auth-server xauth-rad id 1
set auth-server xauth-rad server-name 1.1.1.1
set auth-server xauth-rad account-type xauth
set auth-server xauth-rad radius secret "juniper"
set xauth default auth server xauth-rad query-config
```

2. IKE

```
set ike gateway server_gw6 address 2abc::1 aggressive outgoing-interface
  ethernet3 local-address 3::1 preshare 1234 proposal pre-g2-des-md5
set ike gateway server_gw6 cert peer-cert-type x509-sig
set ike gateway server_gw6 xauth server xauth-rad query-config
```

RADIUS with XAuth Route-Based VPN

In this example, an XAuth client performs authentication and authorization using XAuth and RADIUS, with prefix delegation and mode config, over a route-based VPN. This delegates the IP address obtained from the RADIUS server to a tunnel interface on XAuth client (unless the client specifies another interface).

NOTE: The VPN in this example is route-based. In a policy-based VPN, or if there is a framed netmask other than "/32", the device installs the IP address on the tunnel interface.

See Figure 36, "RADIUS with a Single Client and Single Server," on page 132.

The steps to configure this example are as follows:

1. Configure an XAuth client to perform IPv6 CPE operation (for VPN).
2. Configure an XAuth IPv6 gateway (for VPN) to perform server configuration.
3. Enable an XAuth server (RAD1) with IP address 50.1.1.3. Give RADIUS a shared secret, with port number 1812 for RADIUS.
4. Specify a valid username (PC1) and a valid password (PC1) for the user from the XAuth client.

WebUI (XAuth Client)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

5. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

6. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (XAuth Client)

1. Interfaces

```
set interface trust ipv6 mode router
set interface trust manage
set interface trust ipv6 enable
set interface trust ipv6 interface-id 111111111111111111
set interface trust ipv6 ip 2aaa::1/64
set interface trust ipv6 ra transmit
set interface trust route

set interface untrust ipv6 mode router
set interface untrust manage
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 122222222222222222
set interface untrust ipv6 ip 2bbb::1/64
set interface untrust route
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
set interface tunnel.1 ipv6 unnumbered
```

3. Routes

```
set vrouter trust route ::/0 interface untrust gateway 2bbb::2
set vrouter trust route 2ccc::0/64 interface tunnel.1
```

4. IKE

```
set ike gateway NS1_NS2 add 2bbb::2 outgoing-interface untrust local-address
2bbb::1 preshare abc123 sec-level standard
set ike gateway NS1_NS2 xauth client any username PC1 password PC1
```

5. VPN

```
set vpn NS1toNS2 gateway NS1_NS2 sec-level standard
set vpn NS1toNS2 bind interface tunnel.1
```

6. Policies

```
set policy from untrust to trust any-ipv6 any-ipv6 any permit
set policy from trust to untrust any-ipv6 any-ipv6 any permit
```

WebUI (XAuth Server)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. RADIUS

Configuration > Auth > Auth Servers > New

Configuration > Auth > Auth Servers > Edit

5. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

6. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

7. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (XAuth Server)

1. Interfaces

```
set interface ethernet1 zone untrust
set interface ethernet1 ipv6 mode router
set interface ethernet1 manage
set interface ethernet1 ipv6 enable
set interface ethernet1 ipv6 interface-id 211111111111111111
set interface ethernet1 ipv6 ip 2bbb::2/64
set interface ethernet1 route
```

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode router
set interface ethernet2 manage
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 222222222222222222
set interface ethernet2 ipv6 ip 2ccc::1/64
set interface ethernet2 ip 50.1.1.1/24
set interface ethernet2 route
set interface ethernet2 ipv6 ra transmit
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
```

3. Routes

```
set vrouter trust route ::/0 interface ethernet1 gateway 2bbb::1
set vrouter trust route 2aaa::0/64 interface tunnel.1
```

4. RADIUS

```
set auth-server RAD1 id 1
set auth-server RAD1 server-name 50.1.1.3
set auth-server RAD1 account-type xauth
set auth-server RAD1 radius secret juniper
set auth-server RAD1 radius port 1812
```

5. IKE

```
set ike gateway NS2_NS1 add 2bbb::1 outgoing-interface ethernet1 local-address
    2bbb::2 preshare abc123 sec-level standard
set ike gateway NS2_NS1 xauth server RAD1 query-config user PC1
```

6. VPN

```
set vpn NS2toNS1 gateway NS2_NS1 sec-level standard
set vpn NS2toNS1 bind interface tunnel.1
```

7. Policies

```
set policy from untrust to trust any-ipv6 any-ipv6 any permit
set policy from trust to untrust any-ipv6 any-ipv6 any permit
```

RADIUS with XAuth and Domain Name Stripping

In the following example, you use the domain name checking feature, which checks the domain name (starting from the right most character to the left most until it finds the separator character). This feature enables the device to allow users from a particular domain only.

See Figure 36, “RADIUS with a Single Client and Single Server,” on page 132.

The steps to configure this example are as follows:

1. Enable IPv6 CPE configuration (for VPN) and configure XAuth on the XAuth client.
2. Enable IPv6 gateway configuration (for VPN) on the XAuth server.
3. Enable the XAuth server as Radius RAD1, with IP address 50.1.1.3. Give it RADIUS shared secret (juniper) and port number (1812).
4. Enter commands that do the following:
 - On the XAuth client: Provide the username (PC1@juniper.net) and password (PC1) for that user.
 - On the XAuth server:
 - Configure the username that comes from the XAuth client to be PC1@juniper.net.
 - Allow only users from domain “juniper.net.”
 - Strip the username from right to left until one separator character (@) is found.

WebUI (XAuth Client)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. **IKE**
 VPNs > AutoKey Advanced > Gateway > New

VPN > AutoKey Advanced > Gateway > Edit

5. **VPN**
 VPNs > AutoKey IKE > New

VPN > AutoKey IKE > Edit

6. **Policies**
 Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (XAuth Client)

1. **Interfaces**

```
set interface trust ipv6 mode router
set interface trust manage
set interface trust ipv6 enable
set interface trust ipv6 interface-id 1111111111111111
set interface trust ipv6 ip 2aaa::1/64
set interface trust ipv6 ra transmit
set interface trust route

set interface untrust ipv6 mode router
set interface untrust manage
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 1222222222222222
set interface untrust ipv6 ip 2bbb::1/64
set interface untrust route
```

2. **Tunnel**

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
```

3. **IKE**

```
set ike gateway NS1_NS2 add 2bbb::2 outgoing-interface untrust local-address
  2bbb::1 preshare abc123 sec-level standard
set ike gateway NS1_NS2 xauth client any username PC1@juniper.net password
  PC1
```

4. **VPN**

```
set vpn NS1toNS2 gateway NS1_NS2 sec-level standard
set vpn NS1toNS2 bind interface tunnel.1
```

5. **Routers**

```
set vrouter trust route ::/0 interface untrust gateway 2bbb::2
set vrouter trust route 2ccc::0/64 interface tunnel.1
```

6. **Policies**

```
set policy from untrust to trust any-ipv6 any-ipv6 any permit
set policy from trust to untrust any-ipv6 any-ipv6 any permit
```

WebUI (XAuth Server)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. RADIUS

Configuration > Auth > Auth Servers > New

Configuration > Auth > Auth Servers > Edit

5. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

6. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

7. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (XAuth Server)**1. Interfaces**

```
set interface ethernet1 zone untrust
set interface ethernet1 ipv6 mode router
set interface ethernet1 manage
set interface ethernet1 ipv6 enable
set interface ethernet1 ipv6 interface-id 211111111111111111
set interface ethernet1 ipv6 ip 2bbb::2/64
set interface ethernet1 route
```

```
set interface ethernet2 zone trust
set interface ethernet2 ipv6 mode router
set interface ethernet2 manage
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 222222222222222222
set interface ethernet2 ipv6 ip 2ccc::1/64
set interface ethernet2 ip 50.1.1.1/24
set interface ethernet2 route
set interface ethernet2 ipv6 ra transmit
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
```

3. Routes

```
set vrouter trust route ::/0 interface ethernet1 gateway 2bbb::1
set vrouter trust route 2aaa::0/64 interface tunnel.1
```

4. RADIUS

```
set auth-server RAD1 id 1
set auth-server RAD1 server-name 50.1.1.3
set auth-server RAD1 account-type xauth
set auth-server RAD1 radius secret juniper
set auth-server RAD1 radius port 1812
set auth-server RAD1 username domain juniper.net
set auth-server RAD1 username separator @ number 1
```

5. IKE

```
set ike gateway NS2_NS1 add 2bbb::1 outgoing-interface ethernet1 local-address
  2bbb::2 preshare abc123 sec-level standard
set ike gateway NS2_NS1 xauth server RAD1 query-config user PC1@juniper.net
```

6. VPN

```
set vpn NS2toNS1 gateway NS2_NS1 sec-level standard
set vpn NS2toNS1 bind interface tunnel.1
```

7. Policies

```
set policy from untrust to trust any-ipv6 any-ipv6 any permit
set policy from trust to untrust any-ipv6 any-ipv6 any permit
```

IP Pool Range Assignment

A device assigns a user an IP address from its local IP pool when a RADIUS server returns a framed-ip-address of 255.255.255.254. If all the IP addresses in range contained in the default IP pool are used up, the device uses IP addresses from the next IP range.

See Figure 37, “RADIUS with Multiple Clients and a Single Server,” on page 133.

The steps to configure this example are as follows:

1. Enable IPv6 CPE configuration (for VPN) and XAuth client configuration on Client 1, Client 2, and Client 3.
2. On the XAuth server:
 - Enable the IPv6 gateway configuration (for VPN).
 - Configure a tunnel for Client 1, Client 2, and Client 3.
 - Enable XAuth server (RAD1) with IP address 50.1.1.3. Specify the RADIUS shared secret (juniper) and port number (1812).
 - Define a local IP pool (P1) containing two IP ranges.
 - 80.1.1.1 to 80.1.1.2
 - 80.1.1.3 to 80.1.1.4.
 - Enable the default XAuth IP pool (P1).
 - Specify valid usernames (PC4) and valid passwords (PC4) for those users from the XAuth clients (Device 1, Device 2, and Device 3).
3. On the RADIUS server, insert a framed IP address with a subnet mask of 255.255.255.254 for the user PC4.

WebUI (XAuth Client 1, XAuth Client 2, and XAuth Client 3)

NOTE: The WebUI paths are the same for each XAuth client. The values you enter for some fields differ. See the CLI commands for the correct values.

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

5. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

6. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (XAuth Client 1)

1. Interfaces

```
set interface trust ipv6 mode router
set interface trust ipv6 manage
set interface trust ipv6 enable
set interface trust ipv6 interface-id 111111111111111111
set interface trust ipv6 ip 2aaa::1/64
set interface trust ipv6 ra transmit
set interface trust route

set interface untrust ipv6 mode router
set interface untrust ipv6 manage
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 122222222222222222
set interface untrust ipv6 ip 2bbb::1/64
set interface untrust route
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
```

3. Routes

```
set vrouter trust route ::/0 interface untrust gateway 2bbb::2
set vrouter trust route 2ccc::0/64 interface tunnel.1
```

4. IKE

```
set ike gat NS1_NS2 add 2bbb::2 outgoing-interface untrust local-add 2bbb::1
  preshare abc123 sec-level standard
set ike gateway NS1_NS2 xauth client any username PC4 password PC4
```

5. VPN

```
set vpn NS1toNS2 gateway NS1_NS2 sec-level standard
set vpn NS1toNS2 bind interface tunnel.1
```

6. Policies

```
set policy id 1 from untrust to trust any-ipv6 any-ipv6 any permit
set policy id 2 from trust to untrust any-ipv6 any-ipv6 any permit
```

CLI (XAuth Client 2)

1. Interfaces

```
set interface trust ipv6 mode router
set interface trust manage
set interface trust ipv6 enable
set interface trust ipv6 interface-id 31111111111111111111
set interface trust ipv6 ip 2bad::1/64
set interface trust ipv6 ra transmit
set interface trust route

set interface untrust ipv6 mode router
set interface untrust manage
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 32222222222222222222
set interface untrust ipv6 ip 2ddd::1/64
set interface untrust route
```

2. Tunnel

```
set interface tunnel.2 zone untrust
set interface tunnel.2 ipv6 mode router
set interface tunnel.2 ipv6 enable
```

3. Routes

```
set vrouter trust route ::/0 interface untrust gateway 2ddd::2
set vrouter trust route 2ccc::0/64 interface tunnel.2
```

4. IKE

```
set ike gateway NS3_NS2 add 2ddd::2 outgoing-interface untrust local-add 2ddd::1
  preshare abc123 sec-level standard
set ike gateway NS3_NS2 xauth client any username PC4 password PC4
```

5. VPN

```
set vpn NS3toNS2 gateway NS3_NS2 sec-level standard
set vpn NS3toNS2 bind interface tunnel.2
```

6. Policies

```
set policy id 1 from untrust to trust any-ipv6 any-ipv6 any permit
set policy id 2 from trust to untrust any-ipv6 any-ipv6 any permit
```

CLI (XAuth Client 3)

1. Interfaces

```
set interface trust ipv6 mode router
set interface trust manage
set interface trust ipv6 enable
set interface trust ipv6 interface-id 41111111111111111111
set interface trust ipv6 ip 3bad::1/64
set interface trust ipv6 ra transmit
set interface trust route

set interface untrust ipv6 mode router
set interface untrust manage
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 42222222222222222222
set interface untrust ipv6 ip 3ddd::1/64
set interface untrust route
```

2. Tunnel

```
set interface tunnel.3 zone untrust
set interface tunnel.3 ipv6 mode router
set interface tunnel.3 ipv6 enable
```

3. Routes

```
set vrouter trust route ::/0 interface untrust gateway 3ddd::2
set vrouter trust route 2ccc::0/64 interface tunnel.3
```

4. IKE

```
set ike gateway NS4_NS2 add 3ddd::2 outgoing-interface untrust local-add 3ddd::1
  preshare abc123 sec-level standard
set ike gateway NS4_NS2 xauth client any username PC4 password PC4
```

5. VPN

```
set vpn NS4toNS2 gateway NS4_NS2 sec-level standard
set vpn NS4toNS2 bind interface tunnel.3
```

6. Policies

```
set policy id 1 from untrust to trust any-ipv6 any-ipv6 any permit
set policy id 2 from trust to untrust any-ipv6 any-ipv6 any permit
```

WebUI (XAuth Server)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Route

Network > Routing > Routing Table > New (trust-vr)

4. RADIUS

Configuration > Auth > Auth Servers > New

Configuration > Auth > Auth Servers > Edit

5. XAuth

Objects > IP Pools > New

Objects > IP Pools > Edit

VPNs > L2TP > Default Settings

6. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

7. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

8. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (XAuth Server)

1. Interfaces

```
set interface ethernet1 zone untrust
set interface ethernet1 ipv6 mode router
set interface ethernet1 manage
set interface ethernet1 ipv6 enable
set interface ethernet1 ipv6 interface-id 211111111111111111
set interface ethernet1 ipv6 ip 2bbb::2/64
set interface ethernet1 route
```

```
set interface ethernet2 zone trust
set interface ethernet2 manage
set interface ethernet2 ipv6 mode router
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 222222222222222222
set interface ethernet2 ipv6 ip 2ccc::1/64
set interface ethernet2 ipv6 ra transmit
set interface ethernet2 ip 50.1.1.1/24
set interface ethernet2 route
```

```
set interface ethernet3 zone untrust
set interface ethernet3 manage
set interface ethernet3 ipv6 mode router
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 interface-id 233333333333333333
set interface ethernet3 ipv6 ip 2ddd::2/64
set interface ethernet3 route
```

```
set interface ethernet4 zone untrust
set interface ethernet4 manage
set interface ethernet4 ipv6 mode router
set interface ethernet4 ipv6 enable
set interface ethernet4 ipv6 interface-id 244444444444444444
set interface ethernet4 ipv6 ip 3ddd::2/64
set interface ethernet4 route
```

2. Tunnels

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode router
set interface tunnel.1 ipv6 enable
set interface tunnel.2 zone untrust
set interface tunnel.2 ipv6 mode router
set interface tunnel.2 ipv6 enable
set interface tunnel.3 zone untrust
set interface tunnel.3 ipv6 mode router
set interface tunnel.3 ipv6 enable
```

3. Routes

```

set vrouter trust route 2aaa::0/64 interface ethernet1 gateway 2bbb::1 metric 2
set vrouter trust route 2aaa::0/16 interface tunnel.1
set vrouter trust route 2bad::0/64 interface ethernet3 gateway 2ddd::1 metric 2
set vrouter trust route 2bad::0/16 interface tunnel.2
set vrouter trust route 3bad::0/64 interface ethernet4 gateway 2ddd::1 metric 2
set vrouter trust route 3bad::0/16 interface tunnel.3

```

4. RADIUS

```

set auth-server RAD1 id 1
set auth-server RAD1 server-name 50.1.1.3
set auth-server RAD1 account-type xauth
set auth-server RAD1 radius secret juniper
set auth-server RAD1 radius port 1812

```

5. XAuth

```

set ippool P1 80.1.1.1 80.1.1.2
set ippool P1 80.1.1.3 80.1.1.4
set xauth default ippool P1

```

6. IKE

```

set ike gateway NS2_NS1 add 2bbb::1 outgoing-interface ethernet1 local-add
2bbb::2 preshare abc123 sec-level standard
set ike gateway NS2_NS3 add 2ddd::1 outgoing-interface ethernet3 local-add
2ddd::2 preshare abc123 sec-level standard
set ike gateway NS2_NS4 add 3ddd::1 outgoing-interface ethernet4 local-add
3ddd::2 preshare abc123 sec-level standard
set ike gateway NS2_NS1 xauth server RAD1 query-config user PC4
set ike gateway NS2_NS3 xauth server RAD1 query-config user PC4
set ike gateway NS2_NS4 xauth server RAD1 query-config user PC4

```

7. VPN

```

set vpn NS2toNS1 gateway NS2_NS1 sec-level standard
set vpn NS2toNS1 bind interface tunnel.1
set vpn NS2toNS3 gateway NS2_NS3 sec-level standard
set vpn NS2toNS3 bind interface tunnel.2
set vpn NS2toNS4 gateway NS2_NS4 sec-level standard
set vpn NS2toNS4 bind interface tunnel.3

```

8. Policies

```

set policy id 1 from untrust to trust any-ipv6 any-ipv6 any permit
set policy id 2 from trust to untrust any-ipv6 any-ipv6 any permit

```

RADIUS Retries

In an environment with a single XAuth client and XAuth server with multiple RADIUS servers, you can configure the number of RADIUS server retries before the XAuth server tries a backup server. The default number of retries is 3 (4 total). For example, if you configure the device to 20 retries, the device sends a total of 21 requests to the primary RADIUS server before trying the backup server.

this example, based on the topology shown in Figure 38 on page 133, only shows the RADIUS configuration for the XAuth server.

WebUI (XAuth Server, RADIUS Configuration)

Configuration > Auth > Auth Servers > New

Configuration > Auth > Auth Servers > Edit

CLI (XAuth Server, RADIUS Configuration)

```
set auth-server RAD1 id 1
set auth-server RAD1 server-name 50.1.1.3
set auth-server RAD1 backup1 50.1.1.5
set auth-server RAD1 account-type xauth
set auth-server RAD1 radius secret juniper
set auth-server RAD1 radius port 1812
set auth-server RAD1 radius retries 20
```

Calling-Station-Id

The Calling-Station-Id attribute cannot be seen in the Access-Request packet and in the Accounting-Request packet by default. You can configure this attribute to appear in packets. The ike-ip address will be sent in the access request packet.

In the following example based on Figure 36 on page 132, you configure an XAuth server to include the Calling-Station-Id attribute in packets. This example only shows the RADIUS configuration for the XAuth server.

WebUI (Device 2)

Configuration > Auth > Auth Servers > New

Configuration > Auth > Auth Servers > Edit

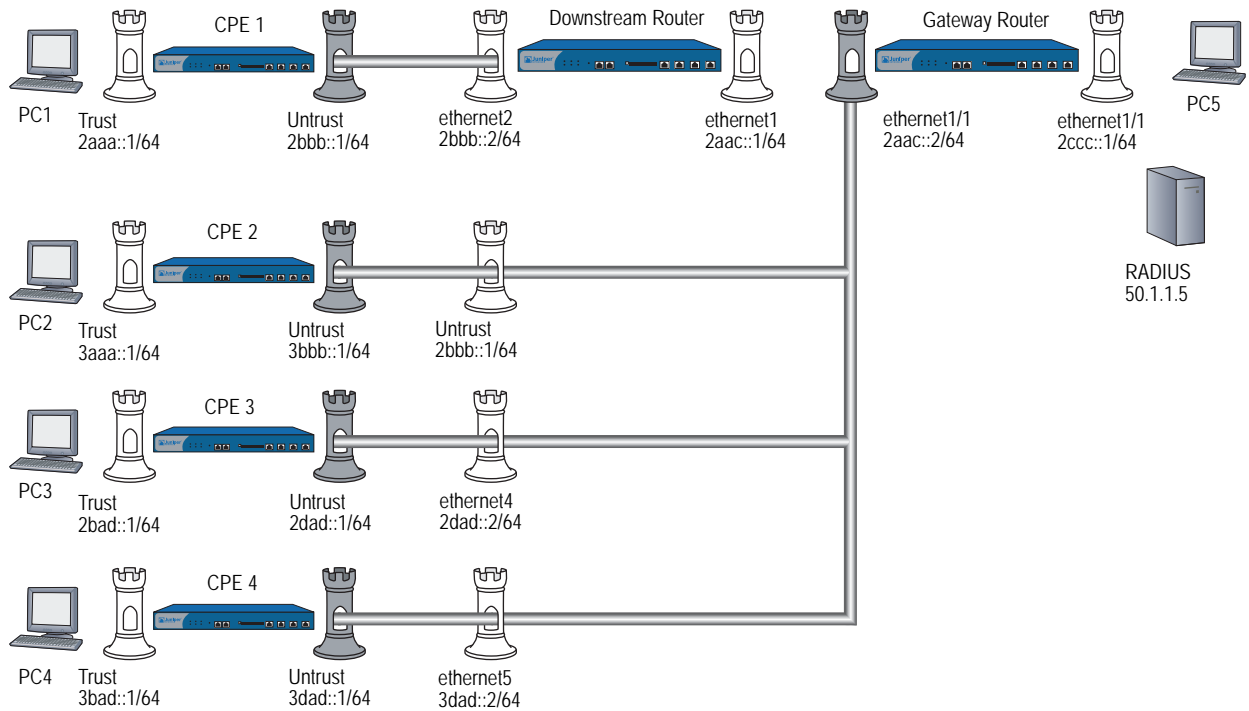
CLI (Device 2)

```
set auth-server RAD1 id 1
set auth-server RAD1 server-name 50.1.1.3
set auth-server RAD1 backup1 50.1.1.5
set auth-server RAD1 account-type xauth
set auth-server RAD1 radius secret juniper
set auth-server RAD1 radius port 1812
set auth-server RAD1 radius attribute calling-station-id
```

IPSec Access Session

Figure 42 shows four devices, configured as CPE routers, interacting with an upstream gateway router (Device 5) through a downstream router (Device 2). In all cases, ipsec-access-session feature is enabled, which allows the RADIUS server to maintain a single continuous RADIUS accounting session.

Figure 42: IPSec Access Session Example



In the following example, four CPEs establish tunnels to the same interface on the Gateway router (ethernet1, IP address 2aac::2). You initiate an IAS session on the Gateway, which allows the four different XAuth users (one on each CPE router), identifying themselves using separate usernames and passwords.

NOTE: The WebUI section lists only the navigational paths to the device configuration pages. For specific values, see the CLI section that follows it.

The basic steps to perform this example are as follows:

1. Configure all four CPE devices, the router, and the Gateway as described in the CLI below.
2. From each of the four CPEs, send an XAuth user (PC1, PC2, PC3, and PC11 with passwords PC1, PC2, PC3, and PC11, respectively). In addition, configure all the CPEs as XAuth clients.
3. From all the CPEs send domain names as “juniper.net” and configure VPN for aggressive mode negotiations.
4. On the Gateway:
 - a. Perform the XAuth server configuration.
 - b. Specify the RADIUS shared secret as juniper, and specify the port as 1812.
 - c. Add a user group (group1) and add user (n1) with domain name “juniper.net” and share limit 10.

On the RADIUS server:

- a. Add users PC1, PC2, PC3 and PC4 and specify passwords PC1, PC2, PC3 and PC4, respectively.
- b. Specify all the required attributes (such as framed-ip-address) to all the users.

WebUI (CPE 1, CPE 2, CPE 3, and CPE 4)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Routes

Network > Routing > Routing Table > New (trust-vr)

4. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

5. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

6. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (CPE 1)

1. Interfaces

```
set interface ethernet1 zone trust
set interface ethernet1 manage
set interface ethernet1 ipv6 mode router
set interface ethernet1 ipv6 enable
set interface ethernet1 ipv6 interface-id 611111111111111111
set interface ethernet1 ipv6 ip 2aaa::1/64
set interface ethernet1 ipv6 ra transmit
set interface ethernet1 route
```

```
set interface ethernet2 zone untrust
set interface ethernet2 manage
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 622222222222222222
set interface ethernet2 ipv6 ip 2bbb::1/64
set interface ethernet2 route
```

2. Tunnel

```
set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable
```

3. Routes

```
set vrouter trust-vr route ::/0 interface ethernet2 gateway 2bbb::2
set vrouter trust-vr route 2ccc::/64 interface tunnel.1
```

4. IKE

```
set ike gateway touniversal add 2aac::2 aggressive local-id juniper.net
    outgoing-interface ethernet2 local-address 2bbb::1 preshare abc123 sec-level
    standard
set ike gateway touniversal xauth client any username PC1 password PC1
```

5. VPN

```
set vpn vpn1 gateway touniversal sec-level standard
set vpn vpn1 bind interface tunnel.1
```

6. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

CLI (CPE 2)

1. Interfaces

```
set interface trust zone trust
set interface trust manage
set interface trust ipv6 mode router
set interface trust ipv6 enable
set interface trust ipv6 interface-id 31111111111111111111
set interface trust ipv6 ip 3aaa::1/64
set interface trust ipv6 ra transmit
set interface trust route
```

```
set interface untrust zone untrust
set interface untrust manage
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 32222222222222222222
set interface untrust ipv6 ip 3bbb::1/64
set interface untrust route
```

2. Tunnel

```
set interface tunnel.2 zone untrust
set interface tunnel.2 ipv6 mode host
set interface tunnel.2 ipv6 enable
```

3. Routes

```
set vrouter trust-vr route ::/0 interface untrust gateway 3bbb::2
set vrouter trust-vr route 2ccc::/64 interface tunnel.2
```

4. IKE

```
set ike gateway touniversal add 2aac::2 aggressive local-id juniper.net
    outgoing-interface untrust local-address 3bbb::1 preshare abc123 sec-level
    standard
set ike gateway touniversal xauth client any username PC2 password PC2
```

5. VPN

```
set vpn vpn2 gateway touniversal sec-level standard
set vpn vpn2 bind interface tunnel.2
```

6. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

CLI (CPE 3)

1. Interfaces

```

set interface trust zone trust
set interface trust manage
set interface trust ipv6 mode router
set interface trust ipv6 enable
set interface trust ipv6 interface-id 4111111111111111
set interface trust ipv6 ip 2bad::1/64
set interface trust ipv6 ra transmit
set interface trust route

set interface untrust zone untrust
set interface untrust manage
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 4222222222222222
set interface untrust ipv6 ip 2dad::1/64
set interface untrust route
    
```

2. Tunnel

```

set interface tunnel.3 zone untrust
set interface tunnel.3 ipv6 mode host
set interface tunnel.3 ipv6 enable
    
```

3. Routes

```

set vrouter trust-vr route ::/0 interface untrust gateway 2dad::2
set vrouter trust-vr route 2ccc::/64 interface tunnel.3
    
```

4. IKE

```

set ike gateway touniversal add 2aac::2 aggressive local-id juniper.net
    outgoing-interface untrust local-address 2dad::1 preshare abc123 sec-level
    standard
set ike gateway touniversal xauth client any username PC3 password PC3
    
```

5. VPN

```

set vpn vpn3 gateway touniversal sec-level standard
set vpn vpn3 bind interface tunnel.3
    
```

6. Policies

```

set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
    
```

CLI (CPE 4)

1. Interfaces

```
set interface trust zone trust
set interface trust manage
set interface trust ipv6 mode router
set interface trust ipv6 enable
set interface trust ipv6 interface-id 5111111111111111
set interface trust ipv6 ip 3bad::1/64
set interface trust ipv6 ra transmit
set interface trust route
```

```
set interface untrust zone untrust
set interface untrust manage
set interface untrust ipv6 mode host
set interface untrust ipv6 enable
set interface untrust ipv6 interface-id 5222222222222222
set interface untrust ipv6 ip 3dad::1/64
set interface untrust route
```

2. Tunnel

```
set interface tunnel.4 zone untrust
set interface tunnel.4 ipv6 mode host
set interface tunnel.4 ipv6 enable
```

3. Routes

```
set vrouter trust-vr route ::/0 interface untrust gateway 3dad::2
set vrouter trust-vr route 2ccc::/64 interface tunnel.4
```

4. IKE

```
set ike gateway touniversal add 2aac::2 aggressive local-id juniper.net
    outgoing-interface untrust local-address 3dad::1 preshare abc123 sec-level
    standard
set ike gateway touniversal xauth client any username PC11 password PC11
```

5. VPN

```
set vpn vpn4 gateway touniversal sec-level standard
set vpn vpn4 bind interface tunnel.4
```

6. Policies

```
set policy from trust to untrust any-ipv6 any-ipv6 any permit
set policy from untrust to trust any-ipv6 any-ipv6 any permit
```

WebUI (Device 2, Router)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Routes

Network > Routing > Routing Table > New (trust-vr)

CLI (Device 2, Router)**1. Internet**

```

set interface ethernet1 zone trust
set interface ethernet1 manage
set interface ethernet1 ipv6 mode host
set interface ethernet1 ipv6 enable
set interface ethernet1 ipv6 interface-id 111111111111111111
set interface ethernet1 ipv6 ip 2aac::1/64
set interface ethernet1 route

```

```

set interface ethernet2 zone trust
set interface ethernet2 manage
set interface ethernet2 ipv6 mode host
set interface ethernet2 ipv6 enable
set interface ethernet2 ipv6 interface-id 122222222222222222
set interface ethernet2 ipv6 ip 2bbb::2/64
set interface ethernet2 route

```

```

set interface ethernet3 zone trust
set interface ethernet3 manage
set interface ethernet3 ipv6 mode host
set interface ethernet3 ipv6 enable
set interface ethernet3 ipv6 interface-id 133333333333333333
set interface ethernet3 ipv6 ip 3bbb::2/64
set interface ethernet3 route

```

```

set interface ethernet4 zone trust
set interface ethernet4 manage
set interface ethernet4 ipv6 mode host
set interface ethernet4 ipv6 enable
set interface ethernet4 ipv6 interface-id 144444444444444444
set interface ethernet4 ipv6 ip 2dad::2/64
set interface ethernet4 route

```

```

set interface ethernet5 zone trust
set interface ethernet5 manage
set interface ethernet5 ipv6 mode host
set interface ethernet5 ipv6 enable
set interface ethernet5 ipv6 interface-id 155555555555555555
set interface ethernet5 ipv6 ip 3dad::2/64
set interface ethernet5 route
unset zone trust block

```

2. Routes

```

set vrouter trust-vr route 2ccc::/64 interface ethernet1 gateway 2aac::2
set vrouter trust-vr route 2aaa::/64 interface ethernet2 gateway 2bbb::1
set vrouter trust-vr route 3aaa::/64 interface ethernet3 gateway 3bbb::1
set vrouter trust-vr route 2bad::/64 interface ethernet4 gateway 2dad::1
set vrouter trust-vr route 3bad::/64 interface ethernet5 gateway 3dad::1

```

WebUI (Gateway Router)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnel

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Routes

Network > Routing > Routing Table > New (trust-vr)

4. Users

Objects > Users > Local > New

Objects > Users > Local > Edit

Objects > User > Local Groups > New

Objects > User > Local Groups > Edit

5. RADIUS

Configuration > Auth > Auth Servers > New

Configuration > Auth > Auth Servers > Edit

6. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

7. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

8. Policies

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (Gateway Router)**1. Interfaces**

```

set interface ethernet1/1 zone untrust
set interface ethernet1/1 manage
set interface ethernet1/1 ipv6 mode host
set interface ethernet1/1 ipv6 en
set interface ethernet1/1 ipv6 interface-id 211111111111111111

set interface ethernet1/1 ipv6 ip 2aac::2/64
set interface ethernet1/1 route
set interface ethernet1/2 zone trust
set interface ethernet1/2 manage
set interface ethernet1/2 ipv6 mode router
set interface ethernet1/2 ipv6 enable

set interface ethernet1/2 ipv6 interface-id 2222222222222222
set interface ethernet1/2 ipv6 ip 2ccc::1/64
set interface ethernet1/2 route
set interface ethernet1/2 ipv6 ra transmit
set interface ethernet1/2 ip 50.1.1.1/24

```

2. Tunnel

```

set interface tunnel.1 zone untrust
set interface tunnel.1 ipv6 mode host
set interface tunnel.1 ipv6 enable

```

3. Routes

```

set vrouter trust-vr route ::/0 interface ethernet1/1 gateway 2aac::1
set vrouter trust-vr route 2aaa::/64 interface tunnel.1
set vrouter trust-vr route 3aaa::/64 interface tunnel.1
set vrouter trust-vr route 2bad::/64 interface tunnel.1
set vrouter trust-vr route 3bad::/64 interface tunnel.1
set ipsec access-session enable

```

4. Users

```

set user n1 uid 1
set user n1 ike-id fqdn juniper.net share-limit 10
set user n1 type ike
set user n1 enable
set user-group group1 id 1
set user-group group1 user n1

```

5. RADIUS

```

set auth-server RAD1 id 1
set auth-server RAD1 server-name 50.1.1.5
set auth-server RAD1 account-type xauth
set auth-server RAD1 radius port 1812
set auth-server RAD1 radius secret juniper

```

6. IKE

```

set ike gateway universal dialup group1 aggressive outgoing-interface ethernet1/1
  local-address 2aac::2 preshare abc123 sec-level standard
set ike gateway universal xauth server RAD1 query-config
set ike gateway universal dpd interval 15
set ike gateway universal dpd retry 2

```

7. VPN

```
set vpn VPN1 gateway universal sec-level standard
set vpn VPN1 bind interface tunnel.1
```

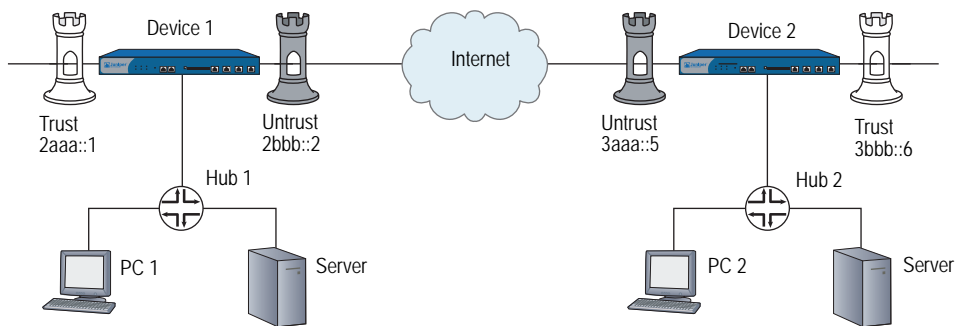
8. Policies

```
set policy id 1 from Untrust to Trust "Dial-up VPN ipv6" any-ipv6 any permit
set policy id 2 from trust to unTrust any-ipv6 "Dial-up VPN ipv6" any permit
set policy id 3 from untrust to Trust any-ipv6 any-ipv6 any permit
```

Dead Peer Detection

Figure 43 shows two devices configured for Dead Peer Detection (DPD).

Figure 43: Dead Peer Detection Example



NOTE: The WebUI section lists only the navigational paths to the device configuration pages. For specific values, see the CLI section that follows it.

On Device 1:

- Configure Trust and Untrust interfaces of Device 1 as an IPv6 router.
 - Configure the Trust interface to advertise prefix 2aaa::.
 - Configure the Untrust interface to advertise prefix 2bbb::.
- Configure an IKE gateway.
- Configure a VPN.
- Set the DPD interval time to 15 seconds and the retry setting to 10 retries.

On Device 2:

- Configure Untrust interface of as an IPv6 host so it can accept RAs.
- Configure the Trust interface as an IPv6 router, advertising prefix 3bbb::.
- Configure an IKE gateway.
- Configure a VPN.

NOTE: Optionally, you can set DPD Always Send by entering the **set ike gateway ton4 dpd always-send** command.

WebUI (Device 1)

1. Interfaces

Network > Interfaces > Edit (for trust)

Network > Interfaces > Edit (for trust) > IPv6

Network > Interfaces > Edit (for untrust)

Network > Interfaces > Edit (for untrust) > IPv6

2. Tunnels

Network > Interfaces > New (Tunnel IF)

Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Routes

Network > Routing > Routing Table > New (trust-vr)

4. IKE

VPNs > AutoKey Advanced > Gateway > New

VPNs > AutoKey Advanced > Gateway > Edit

5. VPN

VPNs > AutoKey IKE > New

VPNs > AutoKey IKE > Edit

6. Policy

Policies > (From: Untrust, To: Trust) > New

Policies > (From: Trust, To: Untrust) > New

CLI (Device 1)

1. Interfaces

```
unset zone untrust block
set interface trust ipv6 mode router
set interface trust ipv6 ip 2aaa::/64
set interface trust ipv6 interface-id 0000000000000001
set interface trust ipv6 enable
set interface trust ipv6 ra transmit
set interface trust route
set interface trust manage

set interface untrust ipv6 mode router
set interface untrust ipv6 ip 2bbb::/64
set interface untrust ipv6 interface-id 0000000000000002
set interface untrust ipv6 enable
set interface untrust ipv6 ra transmit
set interface untrust route
set interface untrust manage
```

2. Tunnel

```
set interface tunnel.6 zone untrust
set interface tunnel.6 ipv6 mode host
set interface tunnel.6 ipv6 enable
set interface tunnel.6 ipv6 ip 3ddd::/64
```

3. Routes

```
set vrouter trust-vr route 3bbb::/64 interface tunnel.6
set vrouter trust-vr route ::/0 interface untrust gateway 2bbb::3
```

4. IKE

```
set ike gateway ton4 address 3aaa::5 outgoing-interface untrust local-address
    2bbb::2 preshare abc sec-level standard
set ike gateway ton4 dpd interval 15
set ike gateway ton4 dpd retry 10
```

5. VPN

```
set vpn vpn1 gateway ton4 no-replay sec-level standard
set vpn vpn1 bind interface tunnel.6
```

6. Policies

```
set policy from untrust to trust any-ipv6 any-ipv6 any permit
set policy from trust to untrust any-ipv6 any-ipv6 any permit
```

WebUI (Device 2)

1. Interfaces

- Network > Interfaces > Edit (for untrust)
- Network > Interfaces > Edit (for untrust) > IPv6
- Network > Interfaces > Edit (for trust)
- Network > Interfaces > Edit (for trust) > IPv6

2. Tunnel

- Network > Interfaces > New (Tunnel IF)
- Network > Interfaces > Edit (for tunnel.6) > IPv6

3. Routes

- Network > Routing > Routing Table > New (trust-vr)

4. IKE

- VPNs > AutoKey Advanced > Gateway > New
- VPNs > AutoKey Advanced > Gateway > Edit

5. VPN

- VPNs > AutoKey IKE > New
- VPNs > AutoKey IKE > Edit

6. Policy

- Policies > (From: Untrust, To: Trust) > New
- Policies > (From: Trust, To: Untrust) > New

CLI (Device 2)

1. Interfaces

```
unset zone untrust block
set interface untrust ipv6 mode host
set interface untrust ipv6 interface-id 0000000000000005
set interface untrust ipv6 enable
set interface untrust ipv6 ra accept
set interface untrust route
set interface untrust manage

set interface trust ipv6 mode router
set interface trust ipv6 ip 3bbb::/64
set interface trust ipv6 interface-id 0000000000000006
set interface trust ipv6 enable
set interface trust ipv6 ra transmit
set interface trust route
set interface trust manage
```

2. Tunnel

```
set interface tunnel.6 zone untrust
set interface tunnel.6 ipv6 mode host
set interface tunnel.6 ipv6 ip 3ddd::/64
set interface tunnel.6 ipv6 enable
```

3. Routes

```
set vrouter trust-vr route ::/0 interface untrust gateway 3aaa::4
set vrouter trust-vr route 2aaa::/64 interface tunnel.6
```

4. IKE

```
set ike gateway ton1 address 2bbb::2 outgoing-interface untrust local-address
    3aaa::5 preshare abc sec-level standard
set ike gateway ton1 dpd interval 15
set ike gateway ton1 dpd retry 5
```

5. VPN

```
set vpn vpn4 gateway ton1 no-replay sec-level standard
set vpn vpn4 bind interface tunnel.6
```

6. Policies

```
set policy from untrust to trust any-ipv6 any-ipv6 any permit
set policy from trust to untrust any-ipv6 any-ipv6 any permit
```


Appendix A

Switching

You can use the security device as a switch to pass IPv6 traffic along a path. To do this, you must place the device in Transparent mode. For more information about Transparent mode, see *Volume 2: Fundamentals*.

Table 4 lists the ScreenOS commands that can allow IPv6 traffic and other non-IP traffic to pass through a security device operating in Transparent mode.

Table 4: Transparent Mode Commands to Bypass Non-IP Traffic

Command	Description
set interface vlan1 bypass-non-ip	Allows all Layer 2 non-IP traffic to pass through the security device.
unset interface vlan1 bypass-non-ip	Blocks all non-IP and non-ARP unicast traffic (default behavior).
unset interface vlan1 bypass-non-ip-all	Overwrites the unset interface vlan1 bypass-non-ip command when both commands appear in the configuration file. If you previously entered the unset interface vlan1 bypass-non-ip-all command, and you now want the device to revert to its default behavior of blocking only the non-IP and non-ARP unicast traffic, you first enter the set interface vlan1 bypass-non-ip command to allow all non-IP and non-ARP traffic, including multicast, unicast, and broadcast traffic to pass through the device. Then you enter the unset interface vlan1 bypass-non-ip command to block only the non-IP, non-ARP unicast traffic.
set interface vlan1 bypass-others-ipsec	Allows a device to pass IPSec traffic without attempting to terminate it, use the command. The device then allows the IPSec traffic to pass through to other VPN termination points.

Index

Numerics

3DES encryption.....	121
4in6 tunneling	
basic setup	115
definition.....	115
6in4 tunneling.....	111
basic setup	120
over IPv4 WAN.....	120
6over4 tunneling	
addresses, handling.....	99
definition.....	98
manual tunneling.....	99
types	98
when to use	98
6to4	
addresses	8, 102, 108
hosts	107
relay routers.....	102, 103
routers	102
tunneling	98, 102
tunneling, description.....	102

A

Access Concentrator (AC).....	46
addresses	
autoconfiguration.....	11
link-local	12
splitting.....	44
addresses, handling	
4in6 tunneling	116
6to4 tunneling	104
destination address translation	84
DIP from IPv4 to IPv6.....	84
DIP from IPv6 to IPv4.....	83
IPv4 hosts to a single IPv6 host.....	113
IPv6 hosts to multiple IPv4 hosts.....	87
manual tunneling.....	99
AES128 encryption.....	121
authentication.....	112, 115, 138
autoconfiguration	
address autoconfiguration.....	11
router advertisement messages	12
stateless.....	11

C

command line interface (CLI).....	30, 32
configuration examples	
6to4 host, tunneling to a.....	108
access lists and route maps	61
delegating prefixes.....	38, 40
DNS server information, requesting	43
IPv4 tunneling over IPv6 (autokey IKE).....	117
IPv6 requests to multiple IPv4 hosts.....	87
IPv6 to an IPv4 network over IPv4	113
IPv6 tunneling over IPv4 (autokey IKE).....	121
manual tunneling	100
native host, tunneling to a	104
PPPoE instance, configuring a.....	46
static route redistribution.....	61
customer premises equipment (CPE).....	39, 134

D

destination gateway.....	99
Device-Unique Identification (DUID).....	36
Diffie-Hellman Group	121
dissimilar IP stacks	84, 86
Domain Name System	
address splitting	44
address translation.....	93
addresses, splitting.....	45
DHCP client host	43
DHCPv6 search list.....	36
DNS refresh.....	42
DNS search list.....	43
DNS server	132
domain lookups	44
IPv4 or IPv6 addresses	42
partial domain names.....	36
proxy DNS.....	44
servers, tunneling to	44
dual-stack architecture	50
dissimilar networks.....	50
dissimilar WAN backbones	50
routing tables.....	50
Duplicate Address Detection (DAD)	
DAD Retry Count.....	31
function	31
Dynamic Host Configuration Protocol version 6 (DHCPv6)	

client and server.....	36	ISP	46
delegated prefixes.....	38		
purposes.....	35	L	
TLA and SLA	37	link-local addresses	12, 14
Dynamic IP	82		
from IPv6 to IPv4.....	83	M	
E		MAC address	13
encapsulation.....	103, 111, 117	MAC addresses.....	21, 29
encryption.....	112, 115	manual 6over4 tunneling	98
3DES.....	121	manual tunneling	99
AES128.....	121	Mapped IP (MIP)	82
endpoint host state mode		mapped IP (MIP).....	82
Base Reachable Time	30	IPv4 hosts to a single IPv6 host.....	91
Duplicate Address Detection (DAD).....	31	IPv4 hosts to multiple IPv6 hosts.....	90
Probe Forever state	31	IPv6 hosts to a single IPv4 host.....	88
Probe Time	31	IPv6 Hosts to multiple IPv4 hosts	86
Reachable Time.....	30	IPv6-to-IPv4 network mapping.....	86
Retransmission Time.....	31	MIP from IPv6 to IPv4	84
Stale mode.....	30	Maximum Transmission Unit (MTU)	12
G		N	
global unicast addresses.....	102, 120	NA - Neighbor Advertisement	30
H		native hosts	102, 104
hashing, Secure Hashing Algorithm (SHA)	121	NAT-PT	
Host mode.....	46, 116	See Network Address Translation-Port Translation	
I		NDP - Neighbor Discovery Parameter.....	21, 30
IAS - IPSec Access Session.....	134	Neighbor Cache table.....	13, 15, 30
Identity Association Prefix Delegation Identification		neighbor entry categories	14
(IAPD-ID).....	37, 39	neighbor cache table.....	13, 25
interfaces		Neighbor Discovery (ND).....	29
DHCPv6.....	35	Accept Incoming RAs	21
dual routing tables	50	age of the neighbor entry.....	13
Neighbor Discovery (ND)	29	bypassing MAC session-caching.....	29
Neighbor Discovery Parameter (NDP).....	30	definition	13
Neighbor Unreachability Detection (NUD).....	29	displaying.....	32
PPPoE.....	46	enabling.....	29
Interior Gateway Protocol (IGP)	51	Neighbor Cache table	13, 29
Internet Service Provider (ISP).....	36, 37, 44, 98	neighbor reachability state	13
IPv4 WAN	112	neighbor reachability status.....	30
IPv4/IPv6 boundaries.....	81-86	packets currently queued for transmission.....	13
IPv4/IPv6 boundary	90	reachability status	29
IPv4-mapped addresses.....	82, 87	Neighbor Discovery Parameter (NDP).....	30
IPv4-to-IPv6 host mapping.....	91	Network Address Translation-Port Translation	
IPv4-to-IPv6 network mapping.....	90	DIP addresses, translating.....	84
IPv6 addresses		DIP from IPv6 to IPv4	83
SLA - Site-Local Aggregator.....	37	Dynamic IP (DIP)	82
TLA - Top-Level Aggregator.....	37	IPv4 hosts to a single IPv6 host.....	91
IPv6 backbone	85, 115	IPv4 hosts to multiple IPv6 hosts.....	90
IPv6/IPv4 boundary	82, 83, 86, 88	IPv6 hosts to a single IPv4 host.....	88
IPv6-to-IPv4 host mapping.....	88	IPv6 hosts to multiple IPv4 hosts	86
island IPv6 networks.....	112	MIP.....	82
		MIP from IPv4 to IPv6.....	85
		outgoing service requests.....	82, 86
		source address translation	83

- when to use 82
- Network Address Translation-Port Translation (NAT-PT)
 - IPSec, when to use 112
- next-hop gateway 31
- NS - Neighbor Solicitation 14, 31
 - setting 30
- NUD - Neighbor Unreachability Detection 13
 - Neighbor Cache Table 13, 25
- O**
- outgoing request packets
 - from IPv6 to IPv4 84
- outgoing service requests 86, 88
- P**
- PPP - Point-to-Point Protocol 46
- PPPoE - Point-to-Point Protocol over Ethernet 46
- prefix list 12
- prefixes
 - prefix list 12
- probe 31
- Probe Time 31
- protocols
 - Interior Gateway Protocol (IGP) 51
- R**
- RA - Router Advertisement 12
- reachability states 14
 - transitions 15
- remote termination point 104, 107
- Retransmission Time 31
- RIP
 - filtering neighbors 57
 - flooding, protecting against 59
 - global parameters 56
 - instances, creating in VR 54
 - interface parameters 60
 - interfaces, enabling on 55
- RIP routes
 - redistributing 58
 - rejecting default 57
- RIP, configuring
 - steps 53
- RIP, viewing
 - database 66
 - neighbor information 68
 - protocol details 66
- RIPng - Routing Information Protocol next-generation.
 - 49, 51
 - interface cost 60
 - interface cost metric 62
 - metric calculation 62
 - offset metric 60, 62
 - route metric 60, 62
 - route redistribution 51
- RS - Router Solicitation 12
- RSA authentication 121
- S**
- SLA - Site-Local Aggregator 37, 39
- source address translation 83
- state transitions
 - endpoint host 15
 - next-hop gateway router 16
 - static entry 18
 - tunnel gateway 17
- stateless address autoconfiguration 11
- T**
- TLA - Top-Level Aggregator 37
- tunnel termination points 102
- U**
- upstream router 38
- V**
- virtual router 50, 102
- VRs
 - RIP 53–70
- W**
- Web user interface (WebUI) 32
- Web user interface (WebUI), on sample client,
 - downstream router 40
- WINS server 132
- X**
- XAuth
 - authentication 138
 - when to use 132

