VYATTA, INC. | Vyatta System

Bridging

REFERENCE GUIDE

Vyatta
Suite 200
1301 Shoreway Road
Belmont, CA 94002
vyatta.com
650 413 7200
1 888 VYATTA 1 (US and Canada)

COPYRIGHT

PROPRIETARY NOTICES

RELEASE DATE: March 2012

DOCUMENT REVISION. R6.4 v01

RELEASED WITH: R6.4.0

PART NO. A0-0247-10-0001

Contents

Quick List of Commands

Use this list to help you quickly locate commands.

List of Examples

Use this list to help you locate examples you'd like to look at or try.

Preface

This document describes the various deployment, installation, and upgrade options for Vyatta software.

This preface provides information about using this guide. The following topics are presented:

Intended Audience

This guide is intended for experienced system and network administrators. Depending on the functionality to be used, readers should have specific knowledge in the following areas:

- Networking and data communications
- TCP/IP protocols
- General router configuration
- Routing protocols
- Network administration
- Network security
- IP services

Organization of This Guide

This guide has the following aid to help you find the information you are looking for:

- Quick List of Commands

  Use this list to help you quickly locate commands.

- List of Examples

  Use this list to help you locate examples you'd like to try or look at.

This guide has the following chapters:

Chapter	Description	Page
Chapter 1: Bridging Overview	This chapter provides a brief introduction to the Vyatta system's support for Layer 2 bridging.	1
Chapter 2: Bridging Configuration Examples	This chapter provides configuration examples for bridging.	4
Chapter 3: Bridge Group Commands	This chapter lists the commands used to create the bridge group (the bridge interface) and define its characteristics.	28
Chapter 4: Bridge Interface Commands	This chapter describes commands for adding interfaces to a bridge group.	62
Glossary of Acronyms		82

Document Conventions

This guide uses the following advisory paragraphs, as follows.

WARNING *Warnings alert you to situations that may pose a threat to personal safety.*

CAUTION *Cautions alert you to situations that might cause harm to your system or damage to equipment, or that may affect service.*

NOTE *Notes provide information you might need to avoid problems or configuration errors.*

This document uses the following typographic conventions.

`Monospace`	Examples, command-line output, and representations of configuration nodes.
`bold Monospace`	Your input: something you type at a command line.
bold	Commands, keywords, and file names, when mentioned inline. Objects in the user interface, such as tabs, buttons, screens, and panes.
italics	An argument or variable where you supply a value.
<key>	A key on your keyboard, such as <Enter>. Combinations of keys are joined by plus signs (“+”), as in <Ctrl>+c.
[key1 | key2]	Enumerated options for completing a syntax. An example is [enable | disable].
num1–numN	A inclusive range of numbers. An example is 1–65535, which means 1 through 65535, inclusive.
arg1..argN	A range of enumerated values. An example is eth0..eth3, which means eth0, eth1, eth2, or eth3.
arg[*arg*...] *arg*[,*arg*...]	A value that can optionally represent a list of elements (a space-separated list and a comma-separated list, respectively).

Vyatta Publications

Full product documentation is provided in the Vyatta technical library. To see what documentation is available for your release, see the *Guide to Vyatta Documentation*. This guide is posted with every release of Vyatta software and provides a great starting point for finding the information you need.

Additional information is available on www.vyatta.com and www.vyatta.org.

Chapter 1: Bridging Overview

This chapter provides a brief introduction to the Vyatta system's support for Layer 2 bridging.

This chapter presents the following topics:

Layer 2 Bridging

Bridging allows you to connect multiple network segments (typically LAN segments) at the Layer 2 level.

Since bridging occurs at Layer 2 (the data link layer) and IP addresses are relevant only on Layer 3 (the network layer), IP addresses are not allowed on the interfaces being bridged.

To create a bridge, use the following workflow:

1. Create the bridge group. You create a bridge group by defining a bridge interface and setting its characteristics.
2. Add the interfaces to the bridge group. You do with within the configuration node for the interface itself.

The following interface types can be added directly to bridge groups:

- Physical Ethernet interfaces
- Ethernet bonded links
- VLAN interfaces configured under physical Ethernet interfaces or Ethernet bonded links
- OpenVPN interfaces
- Tunnel interfaces
- Wireless interfaces in access mode (not in station mode)

RFC 1483 Bridged Ethernet

ADSL interfaces cannot be added to bridge groups, but the Vyatta system supports the mechanisms described in RFC 1483 bridging ADSL traffic over an ATM network. You can bridge ADSL traffic over Ethernet using the interfaces adsl <adslx> pvc <pvc-id> bridged-ethernet command.

Supported Interface Types

MTU for Bridge Groups

The effective MTU (maximum transmission unit) size for a bridge group is the minimum MTU of all the interfaces that belong to the bridge group. So, the maximum frame size of frames transmitted by the bridged interfaces will be this effective MTU size.

Chapter 2: Bridging Configuration Examples

This chapter provides configuration examples for bridging.

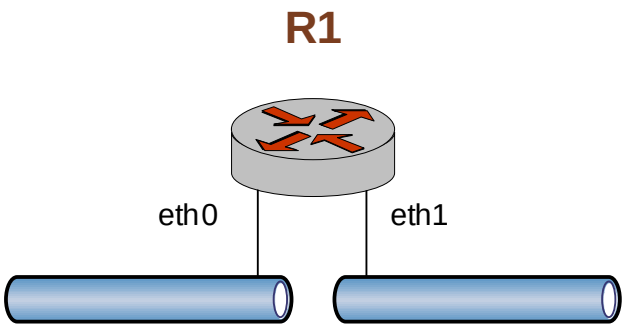
This chapter presents the following topics:

Basic Bridging Configuration

This section presents a sample configuration for a basic bridge between two Ethernet segments on a Vyatta system.

When you have finished, the system will be configured as shown in Figure 2-1.

Figure 2-1 Basic bridging



In this example, you create a bridge interface and assign the Ethernet interfaces to the bridge group.

Example 2-1 creates the bridge interface and adds the Ethernet interfaces to the bridge group. To do this, perform the following steps on R1 in configuration mode

Example 2-1 Configuring a bridge between two Ethernet interfaces

Step	Command
Create the bridge interface.	`vyatta@R1# set interfaces bridge br0`
Add eth0 to the bridge group.	`vyatta@R1# set interfaces ethernet eth0 bridge-group bridge br0`
Add eth1 to the bridge group.	`vyatta@R1# set interfaces ethernet eth1 bridge-group bridge br0`
Commit the configuration.	`vyatta@R1# commit`

Example 2-1 Configuring a bridge between two Ethernet interfaces

View the configuration.

```
vyatta@R1# show interfaces
   bridge br0 {
   }
   ethernet eth0 {
       bridge-group {
           bridge br0
       }
   }
   ethernet eth1 {
       bridge-group {
           bridge br0
       }
   }
```

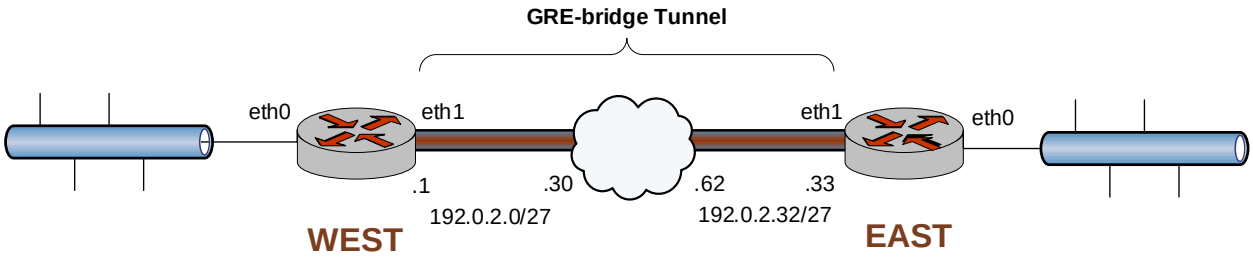
Bridging Across a WAN Using a GRE Tunnel

This section presents a sample configuration for bridging remote network segments using a GRE-bridge encapsulated tunnel between Vyatta systems WEST and EAST. First WEST is configured, and then EAST.

This basic tunnel is not protected by a key: this means it is not secure.

When you have finished, these systems will be configured as shown in Figure 2-2 with bridged network segments connected to eth0 on each of the two systems.

Figure 2-2 Bridging across a WAN using a GRE-bridge encapsulated tunnel



Configure WEST

GRE tunnels are explained in detail in the *Vyatta Tunnels Reference Guide*. Please see that guide for further details.

The GRE-bridge tunnel in the example configuration extends from eth1 on WEST through the wide-area network to eth1 on EAST. In this example, you create the bridge interface, add eth0 to the bridge group, and then create a tunnel interface and add it to the bridge group.

- The source IP address of the tunnel endpoint (the **local-ip**) is the same as the address associated with eth1 in this example.
- The destination IP address of the tunnel endpoint (the **remote-ip**) is 192.0.2.33 on EAST.
- The tunnel encapsulation is **gre-bridge**.
- The tunnel is added to the bridge group.

Example 2-2 creates the bridge and tunnel interfaces and adds eth0 and the tunnel interface to the bridge group. To do this, perform the following steps on WEST in configuration mode.

Example 2-2 Creating a basic GRE-bridge tunnel endpoint and bridge on WEST

Step	Command
Create the bridge interface.	`vyatta@WEST# set interfaces bridge br0`
Add eth0 to the bridge group.	`vyatta@WEST# set interfaces ethernet eth0 bridge-group bridge br0`
Configure an address on eth1.	`vyatta@WEST# set interfaces ethernet eth1 address 192.0.2.1/27`
Create the tunnel interface and specify the source IP address for the tunnel.	`vyatta@WEST# set interfaces tunnel tun0 local-ip 192.0.2.1`
Specify the IP address of the other end of the tunnel.	`vyatta@WEST# set interfaces tunnel tun0 remote-ip 192.0.2.33`
Specify the GRE-bridge encapsulation mode for the tunnel.	`vyatta@WEST# set interfaces tunnel tun0 encapsulation gre-bridge`
Add tun0 to the bridge group.	`vyatta@WEST# set interfaces tunnel tun0 bridge-group bridge br0`
Commit the configuration.	`vyatta@WEST# commit`

Example 2-2 Creating a basic GRE-bridge tunnel endpoint and bridge on WEST

View the configuration.

```
vyatta@WEST# show interfaces
   bridge br0 {
   }
   ethernet eth0 {
       bridge-group {
           bridge br0
       }
   }
   ethernet eth1 {
       address 192.0.2.1/27
   }
   tunnel tun0 {
       bridge-group {
           bridge br0
       }
       encapsulation gre-bridge
       local-ip 192.0.2.1
       remote-ip 192.0.2.33
   }
```

Configure EAST

EAST is configured similarly to WEST. The differences are as follows:

- The address assigned to eth1 is 192.0.2.33/2.
- The local IP address (**local-ip**) is 192.0.2.33.
- The remote IP address (**remote-ip**) is 192.0.2.1.

Example 2-3 shows the completed configuration.

Example 2-3 Configuration for a basic GRE-bridge tunnel endpoint and bridge on EAST

Step	Command

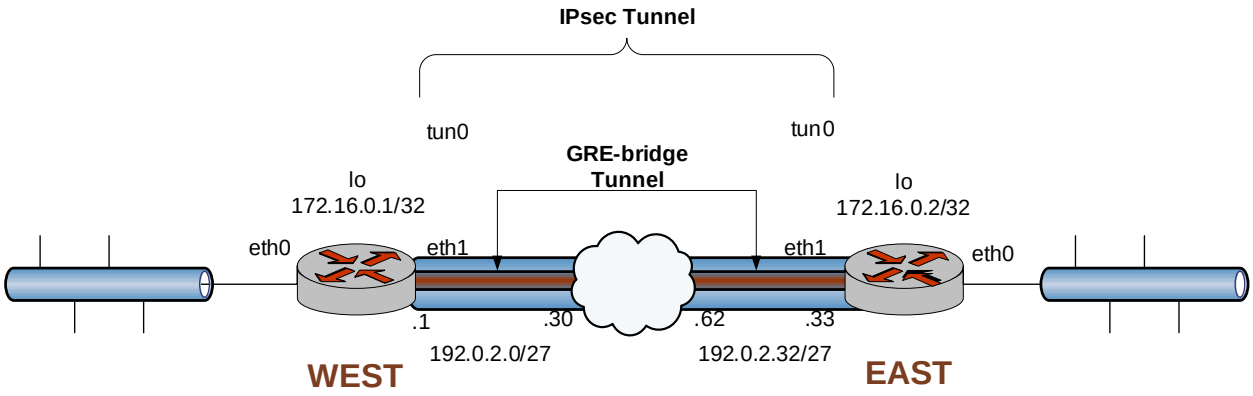
View the configuration.	vyatta@EAST# show interfaces
	bridge br0 {
	}
	ethernet eth0 {
	bridge-group {
	bridge br0
	}
	}
	ethernet eth1 {
	address 192.0.2.33/27
	}
	tunnel tun0 {
	bridge-group {
	bridge br0
	}
	encapsulation gre-bridge
	local-ip 192.0.2.33
	remote-ip 192.0.2.1
	}

Bridging across a WAN Using a GRE Tunnel over IPsec VPN

This example configures a GRE-bridge tunnel between WEST and EAST and protects it within an IPsec tunnel between the same endpoints.

When you have finished, WEST and EAST will be configured as shown in Figure 2-3.

Figure 2-3 GRE-bridge tunnel protected by an IPsec tunnel



Configure WEST

This section presents the following examples:

- Example 2-4 Defining the bridge, Ethernet, and loopback interfaces on WEST
- Example 2-5 Defining the GRE-bridge tunnel from WEST to EAST
- Example 2-6 Defining the IPsec tunnel from WEST to EAST

Define the Bridge, Ethernet, and Loopback Interfaces on "WEST"

Example 2-4 defines the bridge, Ethernet, and loopback interfaces on WEST. In this example:

- The bridge interface br0 is created.
- Ethernet interface eth0 is added to the bridge group
- Ethernet interface eth1 is configured with IP address 192.0.2.1/27.
- Loopback interface lo is configured with IP address 172.16.0.1/32.

To create the bridge, Ethernet, and loopback interfaces on WEST, perform the following steps in configuration mode:

Example 2-4 Defining the bridge, Ethernet, and loopback interfaces on WEST

Step	Command
Create the bridge interface.	`vyatta@WEST# set interfaces bridge br0`

Example 2-4 Defining the bridge, Ethernet, and loopback interfaces on WEST

Add eth0 to the bridge group.	`vyatta@WEST# set interfaces ethernet eth0 bridge-group bridge br0`
Configure an address on eth1.	`vyatta@WEST# set interfaces ethernet eth1 address 192.0.2.1/27`
Configure an address on lo.	`vyatta@WEST# set interfaces loopback lo address 172.16.0.1/32`

Define the GRE Tunnel on "WEST"

NOTE *This example deals with GRE tunnels in the context of a bridge. GRE tunnels themselves are explained in detail in the Vyatta Tunnels Reference Guide.*

Example 2-5 defines WEST's end of the GRE-bridge tunnel. In this example:

- The IP address on the local side of the GRE tunnel (**local-ip**) is assigned the local loopback address 172.16.0.1.
- The IP address of the other end of the GRE tunnel (**remote-ip**) is assigned the loopback address of the remote system 172.16.0.2.
- The tunnel encapsulation is **gre-bridge**.
- The tunnel is added to the bridge group.

To create the tunnel interface and the tunnel endpoint on WEST, perform the following steps in configuration mode:

Example 2-5 Defining the GRE-bridge tunnel from WEST to EAST

Step	Command
Specify the local IP address for the GRE tunnel.	`vyatta@WEST# set interfaces tunnel tun0 local-ip 172.16.0.1`
Specify the remote IP address for the GRE tunnel.	`vyatta@WEST# set interfaces tunnel tun0 remote-ip 172.16.0.2`
Specify the encapsulation mode for the tunnel.	`vyatta@WEST# set interfaces tunnel tun0 encapsulation gre-bridge`
Add tun0 to the bridge group.	`vyatta@WEST# set interfaces tunnel tun0 bridge-group bridge br0`
Commit the configuration.	`vyatta@WEST# commit`

Example 2-5 Defining the GRE-bridge tunnel from WEST to EAST

View the modified configuration.

```
vyatta@WEST# show interfaces
    bridge br0 {
    }
    ethernet eth0 {
        bridge-group {
            bridge br0
        }
    }
    ethernet eth1 {
        address 192.0.2.1/27
    }
    loopback lo{
        address 172.16.0.1/32
    }
    tunnel tun0 {
        bridge-group {
            bridge br0
        }
        encapsulation gre-bridge
        local-ip 172.16.0.1
        remote-ip 172.16.0.2
    }
```

Define the IPsec Tunnel on “WEST”

Example 2-6 creates the IPsec tunnel from WEST to EAST.

- WEST uses IP address 192.0.2.1 on eth1.
- EAST uses IP address 192.0.2.33 on eth1.
- The IKE group is IKE-1W
- The preshared secret is “test_key_1”.
- The IPsec tunnel is between subnet 172.16.0.1/32 on WEST and 172.16.0.2/32 on EAST, using ESP group ESP-1W.

This examples assumes that you have already configured the following:

- IKE group IKE-1W
- ESP group ESP-1W

NOTE *If you need more information about IKE and ESP groups, they are explained in detail in the Vyatta VPN Reference Guide.*

To create the IPsec tunnel from WEST to EAST, perform the following steps on WEST in configuration mode.

Example 2-6 Defining the IPsec tunnel from WEST to EAST

Step	Command
Enable VPN on eth1.	`vyatta@WEST# set vpn ipsec ipsec-interfaces interface eth1`
Define the site-to-site connection to EAST. Set the authentication mode.	`vyatta@WEST# set vpn ipsec site-to-site peer 192.0.2.33 authentication mode pre-shared-secret`
Navigate to the node for the peer for easier editing.	`vyatta@WEST# edit vpn ipsec site-to-site peer 192.0.2.33` `[edit vpn ipsec site-to-site peer 192.0.2.33]`
Provide the string that will be used to authenticate the peers.	`vyatta@WEST# set authentication pre-shared-secret test_key_1` `[edit vpn ipsec site-to-site peer 192.0.2.33]`
Specify the IKE group.	`vyatta@WEST# set ike-group IKE-1W` `[edit vpn ipsec site-to-site peer 192.0.2.33]`
Identify the IP address on this system to be used for this connection.	`vyatta@WEST# set local-ip 192.0.2.1` `[edit vpn ipsec site-to-site peer 192.0.2.33]`
Create a tunnel configuration, and provide the local subnet for this tunnel.	`vyatta@WEST# set tunnel 1 local subnet 172.16.0.1/32` `[edit vpn ipsec site-to-site peer 192.0.2.33]`
Specify the remote subnet for the tunnel.	`vyatta@WEST# set tunnel 1 remote subnet 172.16.0.2/32` `[edit vpn ipsec site-to-site peer 192.0.2.33]`
Specify the ESP group for this tunnel.	`vyatta@WEST# set tunnel 1 esp-group ESP-1W` `[edit vpn ipsec site-to-site peer 192.0.2.33]`
Return to the top of the configuration hierarchy.	`vyatta@WEST# top`
Commit the configuration.	`vyatta@WEST# commit`

Example 2-6 Defining the IPsec tunnel from WEST to EAST

View the modified configuration.

```
vyatta@WEST# show vpn ipsec ipsec-interfaces
 interface eth1
vyatta@WEST# show vpn ipsec site-to-site peer 192.0.2.33
 authentication
    mode pre-shared-secret
    pre-shared-secret test_key_1
 }
 ike-group IKE-1W
 local-ip 192.0.2.1
 tunnel 1 {
    esp-group ESP-1W
    local {
        subnet 172.16.0.1/32
    }
    remote {
        subnet 172.16.0.2/32
    }
 }
```

Configure EAST

EAST is configured similarly to WEST. The differences in the interface configuration are as follows:

- The address assigned to eth1 is 192.0.2.33/27.
- The address assigned to the loopback interface is 172.16.0.2/32.
- The IP address on the local side is 172.16.0.2.
- The on the remote side is 172.16.0.1.

Example 2-7 shows the completed interfaces configuration.

Example 2-7 Configuration for interfaces on EAST

Step	Command

View the modified configuration.	vyatta@EAST# show interfaces
	bridge br0 {
	}
	ethernet eth0 {
	bridge-group {
	bridge br0
	}
	}
	ethernet eth1 {
	address 192.0.2.33/27
	}
	loopback lo{
	address 172.16.0.2/32
	}
	tunnel tun0 {
	bridge-group {
	bridge br0
	}
	encapsulation gre-bridge
	local-ip 172.16.0.2
	remote-ip 172.16.0.1
	}

The differences in the IPsec VPN configuration are as follows:

- The peer address is 192.0.2.1.
- The IKE group is IKE-1E.
- The IP address on the local side is 192.0.2.33.
- The ESP group is ESP-1E.
- The local subnet is 172.16.0.2/32.
- The remote subnet is 172.16.0.1/32.

Example 2-8 shows the completed IPsec VPN configuration.

Example 2-8 Configuration for IPsec VPN on EAST

Step	Command

View the modified configuration.	`vyatta@EAST# show vpn ipsec ipsec-interfaces`
	`  interface eth1`
	`vyatta@EAST# show vpn ipsec site-to-site peer 192.0.2.1`
	`  authentication`
	`    mode pre-shared-secret`
	`    pre-shared-secret test_key_1`
	`  }`
	`  ike-group IKE-1E`
	`  local-ip 192.0.2.33`
	`  tunnel 1 {`
	`    esp-group ESP-1E`
	`    local {`
	`        subnet 172.16.0.2/32`
	`    }`
	`    remote {`
	`        subnet 172.16.0.1/32`
	`    }`
	`  }`

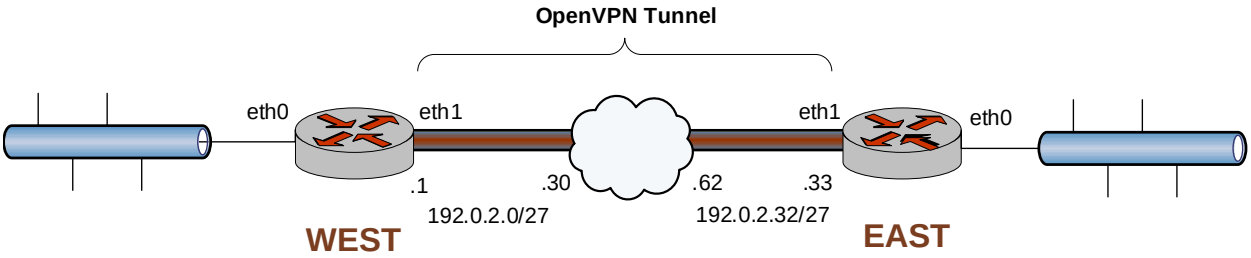
Bridging Across a WAN Using Site-to-Site OpenVPN

This example configures a bridge across a site-to-site OpenVPN tunnel between WEST and EAST.

NOTE *If you need more information about OpenVPN tunnels, they are explained in detail in the Vyatta VPN Reference Guide.*

When you have finished, WEST and EAST will be configured as shown in Figure 2-4 and the LANs connected to WEST and EAST will be bridged.

Figure 2-4 Bridging remote LANs across a site-to-site OpenVPN tunnel



This section presents the following examples:

- Example 2-9 Defining the bridge and Ethernet interfaces on WEST
- Example 2-10 Defining the OpenVPN tunnel from WEST to EAST
- Example 2-11 Configuration for all interfaces on EAST

Configure WEST

Configuring WEST involves defining the bridge, Ethernet, and OpenVPN interfaces and adding the LAN-facing Ethernet interface as well as the OpenVPN interface to the bridge group.

Define the Bridge and Ethernet Interfaces on "WEST"

Example 2-9 defines the bridge and Ethernet interfaces on WEST. In this example:

- The bridge interface br0 is created.
- Ethernet interface eth0 is added to the bridge group.
- Ethernet interface eth1 is configured with IP address 192.0.2.1/27.

To create the bridge and Ethernet interfaces on WEST, perform the following steps in configuration mode.

Example 2-9 Defining the bridge and Ethernet interfaces on WEST

Step	Command
Create the bridge interface.	`vyatta@WEST# set interfaces bridge br0`
Add eth0 to the bridge group.	`vyatta@WEST# set interfaces ethernet eth0 bridge-group bridge br0`
Configure an address on eth1.	`vyatta@WEST# set interfaces ethernet eth1 address 192.0.2.1/27`

Define the OpenVPN Tunnel on "WEST"

Example 2-10 defines WEST's end of the OpenVPN tunnel. In this example:

- The OpenVPN tunnel mode is set to **site-to-site**.
- The IP address of the other end of the OpenVPN tunnel (**remote-host**) is assigned the address of the remote system 192.0.2.33.
- The location of the shared secret file is specified.
- The OpenVPN tunnel is added to the bridge group.

To create the OpenVPN interface on WEST, perform the following steps in configuration mode.

Example 2-10 Defining the OpenVPN tunnel from WEST to EAST

Step	Command
Create the OpenVPN tunnel, and specify the mode to be used.	`vyatta@WEST# set interfaces openvpn vtun0 mode site-to-site`
Specify the remote host address for the OpenVPN tunnel.	`vyatta@WEST# set interfaces openvpn vtun0 remote-host 192.0.2.33`
Specify the file containing the shared secret.	`vyatta@WEST# set interfaces openvpn vtun0 shared-secret-key-file /config/auth/secret`
Add vtun0 to the bridge group.	`vyatta@WEST# set interfaces openvpn vtun0 bridge-group bridge br0`
Commit the configuration.	`vyatta@WEST# commit`

Example 2-10 Defining the OpenVPN tunnel from WEST to EAST

View the modified configuration.

```
vyatta@WEST# show interfaces
    bridge br0 {
    }
    ethernet eth0 {
        bridge-group {
            bridge br0
        }
    }
    ethernet eth1 {
        address 192.0.2.1/27
    }
    openvpn vtun0 {
        bridge-group {
            bridge br0
        }
        mode site-to-site
        remote-host 192.0.2.33
        shared-secret-key-file /config/auth/secret
    }
```

Configure EAST

EAST is configured similarly to WEST. The differences in the interface configuration are as follows:

- The address assigned to eth1 is 192.0.2.33/27.
- The address of the remote host is 192.0.2.1.

Example 2-11 shows the completed interfaces configuration.

Example 2-11 Configuration for all interfaces on EAST

Step	Command

View the modified configuration.	vyatta@EAST# show interfaces
	bridge br0 {
	}
	ethernet eth0 {
	bridge-group {
	bridge br0
	}
	}
	ethernet eth1 {
	address 192.0.2.33/27
	}
	openvpn vtun0 {
	bridge-group {
	bridge br0
	}
	mode site-to-site
	remote-host 192.0.2.1
	shared-secret-key-file /config/auth/secret
	}

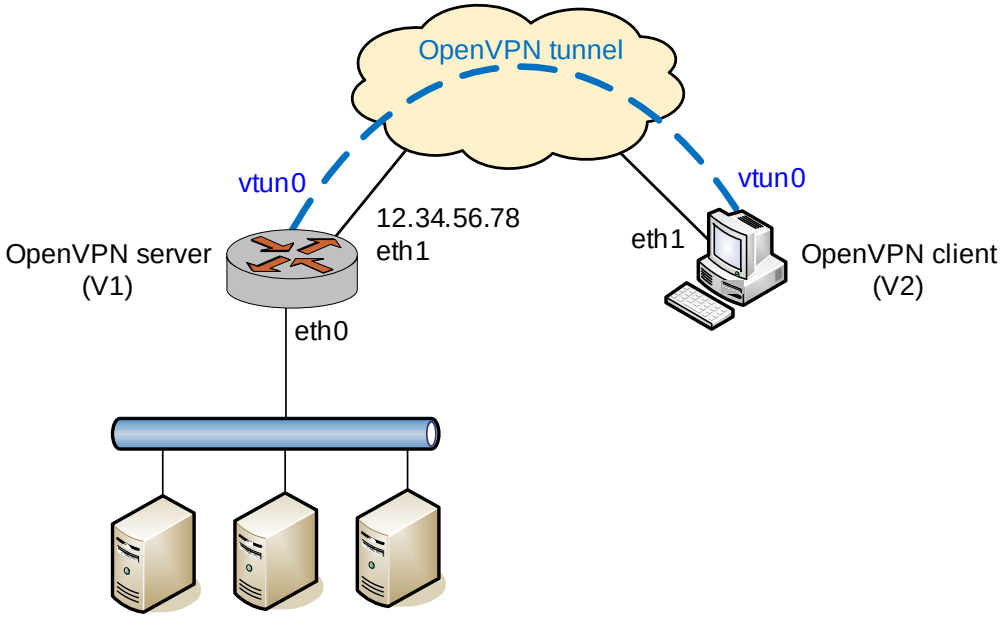
Bridging Across a WAN Using Client-Server OpenVPN

This example configures a bridge across a client-server mode OpenVPN tunnel between an OpenVPN client (V2) and an OpenVPN server (V1).

NOTE *If you need more information about OpenVPN tunnels, they are explained in detail in the Vyatta VPN Reference Guide.*

When you have finished, V1 and V2 will be configured as shown in Figure 2-5 and V2 will be a host on the LAN connected to V1 through the bridge across the client-server OpenVPN tunnel.

Figure 2-5 Bridging to a remote client using client-server OpenVPN



Configure V1

This section presents the following topics:

- Define the Bridge Interface on "V1"
- Define the Ethernet Interfaces on "V1"
- Define the OpenVPN Server on "V1"
- Define the DHCP Server on "V1"
- Commit and Display the Configuration on "V1"

Define the Bridge Interface on "V1"

To configure V1 for bridging, perform the following steps in configuration mode. In this example:

- The **address** command creates the bridge group (the bridge interface br0) and assigns IP address 192.168.200.1/24 to the bridge interface.

Example 2-12 V1 - Bridge configuration

Step	Command
Create the bridge interface and assign it an IP address.	`vyatta@V1# set interfaces bridge br0 address 192.168.200.1/24`

Define the Ethernet Interfaces on "V1"

In this example:

- The eth0 interface is not assigned an IP address, because it is made part of the bridge group. Bridge groups operate at Layer 2, not Layer 3, and so an IP address is not configured.
- The eth1 interface is assigned a static IP address: 12.34.56.78.

Example 2-13 V1 - Ethernet configuration

Step	Command
Create the Ethernet interface eth0 and assign it to the bridge group.	`vyatta@V1# set interfaces ethernet eth0 bridge-group bridge br0`
Create the Ethernet interface eth1 and assign it a static IP address.	`vyatta@V1# set interfaces ethernet eth1 address 12.34.56.78`

Define the OpenVPN Server on "V1"

To configure the OpenVPN server with TLS on V1, perform the following steps in configuration mode. In this example:

- The vtun0 interface is not assigned an IP address. It is made part of the bridge group.
- The **mode** option specifies that this endpoint is to operate in server mode.
- The **server subnet** option indicates that the client's tunnel IP address is allocated from the 192.168.200.0/24 subnet.
- The **remote-host** option is not set, since clients will be actively contacting the server.

Example 2-14 V1 - OpenVPN server configuration

Step	Command
Create the OpenVPN interface vtun0 and assign it to the bridge group.	`vyatta@V1# set interfaces openvpn vtun0 bridge-group bridge br0`
Set the OpenVPN mode.	`vyatta@V1# set interfaces openvpn vtun0 mode server`
Set the subnet for the OpenVPN tunnel.	`vyatta@V1# set interfaces openvpn vtun0 server subnet 192.168.200.0/24`

Example 2-14 V1 - OpenVPN server configuration

Specify the location of the CA certificate file.	`vyatta@V1# set interfaces openvpn vtun0 tls ca-cert-file /config/auth/ca.crt`
Specify the location of the host certificate file.	`vyatta@V1# set interfaces openvpn vtun0 tls cert-file /config/auth/V1.crt`
Specify the location of the DH file.	`vyatta@V1# set interfaces openvpn vtun0 tls dh-file /config/auth/dh1024.pem`
Specify the location of the host key file.	`vyatta@V1# set interfaces openvpn vtun0 tls key-file /config/auth/V1.key`

Define the DHCP Server on "V1"

A DHCP server is required in order to provide IP addresses to remote clients as they connect. To configure the DHCP server on V1, perform the following steps in configuration mode. In this example:

- The vtun0 interface is not assigned an IP address. It is made part of the bridge group.
- The **mode** option specifies that this endpoint will operate in server mode.
- The **server subnet** option indicates that the client's tunnel IP address is allocated from the 192.168.200.0/24 subnet.
- The **remote-host** option is not set since, clients will be actively contacting the server.

Example 2-15 V1 - DHCP server configuration

Step	**Command**
Create the DHCP server service.	`vyatta@V1# set service dhcp-server`
Create a shared network.	`vyatta@V1# set service dhcp-server shared-network-name ovpn-test`
Specify the subnet.	`vyatta@V1# set service dhcp-server shared-network-name ovpn-test subnet 192.168.200.0/24`
Specify the default router.	`vyatta@V1# set service dhcp-server shared-network-name ovpn-test subnet 192.168.200.0/24 default-router 192.168.200.1`
Specify the beginning of the range of addresses that the DHCP server will provide.	`vyatta@V1# set service dhcp-server shared-network-name ovpn-test subnet 192.168.200.0/24 start 192.168.200.100`

Example 2-15 V1 - DHCP server configuration

Specify the end of the range of addresses that the DHCP server will provide.	`vyatta@V1# set service dhcp-server shared-network-name ovpn-test subnet 192.168.200.0/24 start 192.168.200.100 stop 192.168.200.150`

Commit and Display the Configuration on “V1”

Example 2-16 V1 - Commit and display the configuration

Step	Command
Commit the change.	`vyatta@V1# commit`

Show the configuration.	`vyatta@V1# show interfaces`
	`bridge br0 {`
	`  address 192.168.200.1/24`
	`}`
	`ethernet eth0 {`
	`  bridge-group {`
	`  bridge br0`
	`  }`
	`}`
	`ethernet eth1 {`
	`  address 12.34.56.78`
	`}`
	`openvpn vtun0 {`
	`  bridge-group {`
	`  bridge br0`
	`  }`
	`  mode server`
	`  server {`
	`  subnet 192.168.200.0/24`
	`  }`
	`  tls {`
	`  ca-cert-file /config/auth/ca.crt`
	`  cert-file /config/auth/V1.crt`
	`  dh-file /config/auth/dh1024.pem`
	`  key-file /config/auth/V1.key`
	`  }`
	`}`

Example 2-16 V1 - Commit and display the configuration

```
[edit]
vyatta@V1# show service
 dhcp-server {
    shared-network-name ovpn-test {
        subnet 192.168.200.0/24 {
            default-router 192.168.200.1
            start 192.168.200.100 {
                stop 192.168.200.150
            }
        }
    }
 }
```

Configure V2

This section presents the following topics:

Define the Bridge Interface on “V2”

To configure V2 for bridging, perform the following step in configuration mode. In this example:

- The **address** command sets the IP address of the bridge interface (br0) to be assigned by the DHCP server on V1.

Example 2-17 V2 - Bridge configuration

Step	Command
Create the bridge interface and specify that the IP address is to be provided by a DHCP server.	`vyatta@V2# set interfaces bridge br0 address dhcp`

Define the Ethernet Interface on “V2”

In this example:

- eth1 will be assigned an IP address by the Internet provider’s DHCP server.

To configure the Ethernet interface on V2, perform the following step in configuration mode.

Example 2-18 V2 - Ethernet configuration

Step	Command
Create the Ethernet interface eth1 and specify that the IP address is to be provided by a DHCP server.	`vyatta@V2# set interfaces ethernet eth1 address dhcp`

Define the OpenVPN Client on “V2”

To configure the OpenVPN server with TLS on V2, perform the following steps in configuration mode. In this example:

- The vtun0 interface is not assigned an IP address. It is made part of the bridge group.
- The **mode** option specifies that this endpoint will operate in client mode.
- The **remote-host** option specifies the IP address of the OpenVPN server.

Example 2-19 V2 - OpenVPN client configuration

Step	Command
Create the OpenVPN interface vtun0 and assign it to the bridge group.	`vyatta@V2# set interfaces openvpn vtun0 bridge-group bridge br0`
Set the OpenVPN mode.	`vyatta@V2# set interfaces openvpn vtun0 mode client`
Specify the IP address of the OpenVPN host.	`vyatta@V2# set interfaces openvpn vtun0 remote-host 12.34.56.78`
Specify the location of the CA certificate file.	`vyatta@V2# set interfaces openvpn vtun0 tls ca-cert-file /config/auth/ca.crt`
Specify the location of the host certificate file.	`vyatta@V2# set interfaces openvpn vtun0 tls cert-file /config/auth/V2.crt`
Specify the location of the host key file.	`vyatta@V2# set interfaces openvpn vtun0 tls key-file /config/auth/V2.key`

Commit and Display the Configuration on “V2”

Example 2-20 V2 - Commit and display the configuration

Step	Command
Commit the change.	`vyatta@V2# commit`

Show the configuration.	`vyatta@V2# show interfaces`
	`bridge br0 {`
	`address dhcp`
	`}`
	`ethernet eth1 {`
	`address dhcp`
	`}`
	`openvpn vtun0 {`
	`bridge-group {`
	`bridge br0`
	`}`
	`mode client`
	`remote-host 12.34.56.78`
	`tls {`
	`ca-cert-file /config/auth/ca.crt`
	`cert-file /config/auth/V1.crt`
	`key-file /config/auth/V1.key`
	`}`
	`}`
	`[edit]`

Chapter 3: Bridge Group Commands

This chapter lists the commands used to create the bridge group (the bridge interface) and define its characteristics.

Configuration Commands	
interfaces bridge <brx>	Defines a bridge group.
interfaces bridge <brx> address <address>	Assigns an address to a bridge group.
interfaces bridge <brx> aging <age>	Specifies the MAC address aging timeout for a bridge group.
interfaces bridge <brx> description <desc>	Specifies a description for a bridge group.
interfaces bridge <brx> dhcpv6-options	Specifies the way in which a DHCPv6 client is to acquire an address and/or parameters from a DHCPv6 server.
interfaces bridge <brx> disable	Disables a bridge group without discarding configuration.
interfaces bridge <brx> disable-link-detect	Directs a bridge group not to detect physical link-state changes.
interfaces bridge <brx> forwarding-delay <delay>	Specifies the amount of time a bridge group keeps listening after a topology change.
interfaces bridge <brx> hello-time <interval>	Specifies the hello packet interval for a bridge group.
interfaces bridge <brx> ipv6 address	Assigns an IPv6 address to a bridge interface.
interfaces bridge <brx> ipv6 disable-forwarding	Disables IPv6 forwarding on a bridge interface.
interfaces bridge <brx> ipv6 dup-addr-detect-transmits <num>	Specifies the number of times to transmit NS packets as part of the DAD process.
interfaces bridge <brx> ipv6 router-advert	Specifies the router advertisements to be sent from the bridge interface.
interfaces bridge <brx> max-age <interval>	Specifies how long a bridge group waits for a hello packet from the spanning tree root.
interfaces bridge <brx> priority <priority>	Specifies the forwarding priority of a bridge group in the spanning tree.
interfaces bridge <brx> stp <state>	Enables IEEE 802.1D Spanning Tree Protocol on a bridge group.
Operational Commands	
show bridge	Displays the information for active bridge groups.

Commands for using other system features with bridge groups can be found in the following locations.

Related Commands Documented Elsewhere	
ARP commands	ARP is supported on bridge interfaces. Commands for working with ARP are described in *Vyatta Basic System Reference Guide*.
Firewall	Firewall is supported on bridge groups. Commands for configuring firewall are described in the *Vyatta Firewall Reference Guide*.
IPS	Content inspection through the Intrusion Prevention System is supported on bridge groups. Commands for configuring the Intrusion Prevention System are described in *Vyatta Security Reference Guide*.
OSPF	OSPF is supported for bridge groups. Commands for configuring OSPF are described in the *Vyatta OSPF Reference Guide*.
QoS	Quality of service traffic policies are supported on bridge groups. Commands for configuring QoS are described in the *Vyatta QoS Reference Guide*.
RIP	RIP is supported on bridge groups. Commands for configuring RIP are described in the *Vyatta RIP Reference Guide*.
RIPng	RIPng is supported on bridge groups. Commands for configuring RIPng are described in the *Vyatta RIPng Reference Guide*.

interfaces bridge <brx>

Defines a bridge group.

Syntax

set interfaces bridge *brx*

delete interfaces bridge *brx*

show interfaces bridge *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
    }
}
```

Parameters

brx	Multi-node. The identifier for the bridge group. Supported identifiers are **br0** through **br999**. You can define multiple bridge groups by creating more than one **bridge** configuration node.

Default

None.

Usage Guidelines

Use this command to define a bridge group. Note that you must create the bridge group (using this command) before you can assign interfaces to it.

Use the **set** form of this command to create the bridge group and define bridge settings.

Use the **delete** form of this command to remove all configuration for a bridge group.

Use the **show** form of this command to view bridge group configuration.

interfaces bridge <brx> address <address>

Assigns an address to a bridge group.

Syntax

set interfaces bridge *brx* **address** *address*

delete interfaces bridge *brx* **address** *address*

show interfaces bridge *brx* **address**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        address address
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
address	Multi-node. The IP address and network prefix for the interface. The address must either be in the form *ip-address/prefix*, or the keywords **dhcp** or **dhcpv6**. If **dhcp** is specified, an IPv4 address and network prefix is assigned using the Dynamic Host Configuration Protocol (DHCP). If **dhcpv6** is specified, an IPv6 address and network prefix are set using the DHCP for IPv6 (DHCPv6). You can assign multiple addresses to a bridge group by creating multiple **address** configuration nodes.

Default

None.

Usage Guidelines

Use this command to assign an address to a bridge group.

Use the **set** form of this command to set the address for the bridge group.

Use the **delete** form of this command to remove address configuration for the bridge group

Use the **show** form of this command to view bridge group address configuration.

interfaces bridge <brx> aging <age>

Specifies the MAC address aging timeout for a bridge group.

Syntax

set interfaces bridge *brx* **aging** *age*

delete interfaces bridge *brx* **aging**

show interfaces bridge *brx* **aging**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        aging age
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
age	The length of time, in seconds, that a MAC address is to be kept before being aged out. The range is 10 to 1000000 or 0, where 0 means the address is kept forever. The default is 300.

Default

MAC addresses are aged out of the forwarding database after 300 seconds (5 minutes).

Usage Guidelines

Use this command to specify the length of time that a dynamic MAC address entry is kept in a bridge's forwarding database. If this interval expires without the entry being updated, the entry is aged out of the table.

Use the **set** form of this command to set the MAC address aging timeout.

Use the **delete** form of this command to restore the default MAC address aging timeout.

Use the **show** form of this command to view the MAC address aging configuration.

interfaces bridge <brx> description <desc>

Specifies a description for a bridge group.

Syntax

set interfaces bridge *brx* **description** *desc*

delete interfaces bridge *brx* **description**

show interfaces bridge *brx* **description**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        description desc
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
desc	A brief description for the bridge group.

Default

None.

Usage Guidelines

Use this command to specify a description for the bridge group.

Use the **set** form of this command to specify a description for the bridge group.

Use the **delete** form of this command to remove the bridge group description.

Use the **show** form of this command to view the bridge group description.

interfaces bridge <brx> dhcpv6-options

Specifies the way in which a DHCPv6 client is to acquire an address and/or parameters from a DHCPv6 server.

Syntax

set interfaces bridge *brx* **dhcpv6-options** [**parameters-only** | **temporary**]

delete interfaces bridge *brx* **dhcpv6-options** [**parameters-only** | **temporary**]

show interfaces bridge *brx* **dhcpv6-options**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        dhcpv6-options [parameters-only | temporary]
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
parameters-only	Acquires only configuration parameters (and not an IPv6 address) from the DHCPv6 server. Only one of the **parameters-only** and the **temporary** parameter may be specified.
temporary	Acquires a temporary IPv6 address as described for IPv6 privacy addressing in RFC 4941. Only one of the **parameters-only** and the **temporary** parameter may be specified.

Default

None.

Usage Guidelines

Use this command to specify in what way the DHCPv6 client is to acquire an IPv6 address and/or parameters from a DHCPv6 server.

Note that these parameters are only relevant if the **dhcpv6** option has been set for the interfaces bridge <brx> address <address> command. Otherwise, they are ignored.

The **parameters-only** option is typically used in conjunction with SLAAC or static address configuration. It and the **temporary** parameter are mutually exclusive.

Use the **set** form of this command to specify the DHCPv6 options.

Use the **delete** form of this command to remove the DHCPv6 options.

Use the **show** form of this command to view DHCPv6 option configuration.

interfaces bridge <brx> disable

Disables a bridge group without discarding configuration.

Syntax

set interfaces bridge *brx* **disable**

delete interfaces bridge *brx* **disable**

show interfaces bridge *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        disable
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.

Default

Bridging is enabled.

Usage Guidelines

Use this command to disable a bridge group.

Use the **set** form of this command to specify whether to disable bridging on the interface.

Use the **delete** form of this command to restore the default value for the bridge group.

Use the **show** form of this command to view bridge group configuration.

interfaces bridge <brx> disable-link-detect

Directs a bridge group not to detect physical link-state changes.

Syntax

set interfaces bridge *brx* **disable-link-detect**

delete interfaces bridge *brx* **disable-link-detect**

show interfaces bridge *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        disable-link-detect
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.

Default

The interface detects physical link state changes.

Usage Guidelines

Use this command to direct a bridge group to not detect physical state change to the link (for example, when the cable is unplugged).

Use the **set** form of this command to disable detection of physical state changes.

Use the **delete** form of this command to enable detection of physical state changes.

Use the **show** form of this command to view bridge group configuration.

interfaces bridge <brx> forwarding-delay <delay>

Specifies the amount of time a bridge group keeps listening after a topology change.

Syntax

set interfaces bridge *brx* **forwarding-delay** *delay*

delete interfaces bridge *brx* **forwarding-delay**

show interfaces bridge *brx* **forwarding-delay**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        forwarding-delay delay
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
delay	The amount of time, in seconds, the bridge keeps learning about the topology of the spanning tree after a topology change. The range is 0 to 200. The default is 15.

Default

The the bridge listens for 15 seconds before transitioning to Forwarding state.

Usage Guidelines

Use this command to specify the amount of time the bridge will keep listening after a topology change.

After a topology change, the bridge remains in a listening state for the forward delay period, learning about the topology of the spanning tree for this interval. During this period, no traffic is forwarded. After the forward delay interval has passed, the bridge transitions to the forwarding state and begins to forward traffic again.

Use the **set** form of this command to specify the amount of time the bridge will keep listening after a topology change.

Use the **delete** form of this command to restore the forwarding-delay to its default.

Use the **show** form of this command to view the forwarding-delay configuration.

interfaces bridge <brx> hello-time <interval>

Specifies the hello packet interval for a bridge group.

Syntax

set interfaces bridge *brx* **hello-time** *interval*

delete interfaces bridge *brx* **hello-time**

show interfaces bridge *brx* **hello-time**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        hello-time interval
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
interval	The interval in seconds at which this bridge will transmit hello packets. The range is 0 to 30. The default is 2.

Default

The default is 2.

Usage Guidelines

Use this command to specify the “hello packet” interval.

Hello packets are Bridge Protocol Data Units (BPDUs) used as messages to communicate the state of the spanning tree topology. On a spanning tree, hello packets are sent by the bridge that assumes itself to be the root bridge.

Use the **set** form of this command to specify the hello packet interval.

Use the **delete** form of this command to restore the hello packet interval to the default value.

Use the **show** form of this command to view the hello-time configuration.

interfaces bridge <brx> ipv6 address

Assigns an IPv6 address to a bridge interface.

Syntax

set interfaces bridge *brx* **ipv6 address** [**autoconf** | **eui64** *ipv6prefix*]

delete interfaces bridge *brx* **ipv6 address** [**autoconf** | **eui64** *ipv6prefix*]

show interfaces bridge *brx* **ipv6 address** [**autoconf** | **eui64**]

Command Mode

Configuration mode.

Configuration Statement

```
interfaces bridge brx {
    ipv6 {
        address {
            autoconf
            eui64 ipv6prefix
        }
    }
}
```

Parameters

brx	The interface identifier.
autoconf	Generates an IPv6 address using the SLAAC protocol. Set this value if the interface is performing a “host” function rather than a “router” function. This value can be specified in addition to specifying static IPv6, static IPv4, or IPv4 DHCP addresses on the interface.
ipv6prefix	The 64-bit IPv6 address prefix used to configure an IPv6 address, in EUI-64 format. The system concatenates this prefix with a 64-bit EUI-64 value derived from the 48-bit MAC address of the interface.

Default

None.

Usage Guidelines

Use this command to assign an IPv6 address to an interface.

You can using the **autoconf** keyword to direct the system to autoconfigure the address, using the SLAAC (Stateless Address Auto-Configuration) protocol defined in RFC 4862. Alternatively, you can provide an EUI-64 IPv6 address prefix so that the system constructs the IPv6 address.

If you want the system to use SLAAC to acquire addresses on this interface, then in addition setting this parameter, you must also disable IPv6 forwarding, either globally (using the system ipv6 disable-forwarding command) or specifically on this interface (using the interfaces <interface> ipv6 disable-forwarding command).

Use the **set** form of this command to specify an IPv6 address for the interface.

Use the **delete** form of this command to delete an IPv6 address from the interface.

Use the **show** form of this command to view IPv6 address configuration settings.

interfaces bridge <brx> ipv6 disable-forwarding

Disables IPv6 forwarding on a bridge interface.

Syntax

set interfaces bridge *brx* **ipv6 disable-forwarding**

delete interfaces bridge *brx* **ipv6 disable-forwarding**

show interfaces bridge *brx* **ipv6 disable-forwarding**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces bridge brx {
    ipv6 {
        disable-forwarding
    }
}
```

Parameters

brx	The interface identifier.

Default

IPv6 packets are forwarded.

Usage Guidelines

Use this command to disable IPv6 packet forwarding on an interface.

You can also disable IPv6 forwarding globally (that is, for all interfaces) using the system ipv6 disable-forwarding command.

Use the **set** form of this command to disable IPv6 packet forwarding on an interface.

Use the **delete** form of this command to enable IPv6 packet forwarding on an interface.

Use the **show** form of this command to display IPv6 packet forwarding interface configuration.

interfaces bridge <brx> ipv6 dup-addr-detect-transmits <num>

Specifies the number of times to transmit NS packets as part of the DAD process.

Syntax

set interfaces bridge *brx* **ipv6 dup-addr-detect-transmits** *num*

delete interfaces bridge *brx* **ipv6 dup-addr-detect-transmits**

show interfaces bridge *brx* **ipv6 dup-addr-detect-transmits**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces bridge brx {
    ipv6 {
        dup-addr-detect-transmits num
    }
}
```

Parameters

brx	The interface identifier.
num	The number of times to transmit NS packets as part of the DAD process. The default is 1.

Default

One NS packet is transmitted as part of the DAD process.

Usage Guidelines

Use this command to specify the number of times to transmit Neighbor Solicitation (NS) packets as part of the Duplicate Address Detection (DAD) process.

Use the **set** form of this command to specify the number of times to transmit Neighbor Solicitation (NS) packets as part of the Duplicate Address Detection (DAD) process.

Use the **delete** form of this command to delete the parameter from the interface and use the default value.

Use the **show** form of this command to view NS packet configuration for DAD.

interfaces bridge <brx> ipv6 router-advert

Specifies the router advertisements to be sent from the bridge interface.

Syntax

set interfaces bridge *brx* **ipv6 router-advert** [**cur-hop-limit** *limit*] [**default-lifetime** *lifetime*] [**default-preference** *preference*] [**link-mtu** *mtu*] [**managed-flag** *state*] [**max-interval** *interval*] [**min-interval** *interval*] [**other-config-flag** *state*] [**prefix** *ipv6net* [**autonomous-flag** *state* | **on-link-flag** *state* | **preferred-lifetime** *lifetime* | **valid-lifetime** *lifetime*]] [**reachable-time** *time*] [**retrans-timer** *time*] [**send-advert** *state*]

delete interfaces bridge *brx* **ipv6 router-advert** [**cur-hop-limit**] [**default-lifetime**] [**default-preference**] [**link-mtu**] [**managed-flag**] [**max-interval**] [**min-interval**] [**other-config-flag**] [**prefix** *ipv6net* [**autonomous-flag** | **on-link-flag** | **preferred-lifetime** | **valid-lifetime**]] [**reachable-time**] [**retrans-timer** [**send-advert**]

show interfaces bridge *brx* **ipv6 router-advert**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces bridge brx {
    ipv6 {
        router-advert {
            cur-hop-limit limit
            default-lifetime lifetime
            default-preference preference
            link-mtu mtu
            managed-flag state
            max-interval interval
            min-interval interval
            other-config-flag state
            prefix ipv6net {
                autonomous-flag state
                on-link-flag state
                preferred-lifetime lifetime
                valid-lifetime lifetime
            }
            reachable-time time
            retrans-timer time
            send-advert state
        }
```

```
    }
}
```

Parameters

brx	The interface identifier.
cur-hop-limit *limit*	Specifies the Hop Count field of the IP header for outgoing (unicast) IP packets. This value is placed in the Hop Count field of the IP header for outgoing (unicast) IP packets. The range is 0 to 255. The default is 64. A value of 0 means unspecified by the router.
default-lifetime lifetime	Specifies the lifetime , in seconds, associated with the default router. Supported values are 0, which indicates that the router is not a default router, and the range from the value configured for the **max-interval** option to 9000 (18.2 hours). If not configured, the value for this timer is three times **max-interval**.
default-preference *preference*	The preference associated with the default router. Supported values are as follows: **low**: The default router is low preference. **medium**: The default router is medium preference. **high**: The default router is high preference. The default is **medium**.
link-mtu *mtu*	The MTU value to be advertised for the link. The range of values is 0, or 1280 to the maximum MTU for the type of link, as defined in RFC 2464. The default is 0, which means the MTU is not specified in the router advertisement message. That is because it is expected that the MTU will configured directly on the interface itself and not for routing advertisements. You can configure this option in cases where the link MTU is not well known. If the value set here does not match the MTU configured on the interface, the system issues a warning but does not fail.

managed-flag *state*	Whether to use the administered protocol for address autoconfiguration. Supported values are as follows: **true**: Hosts use the administered (stateful) protocol for address autoconfiguration in addition to any addresses autoconfigured using stateless address autoconfiguration. **false**: Hosts use only stateless address autoconfiguration. The default is **false**.
max-interval *interval*	The maximum time, in seconds, allowed between sending unsolicited multicast router advertisements from the interface. The range of supported values is 4 to 1800. The default is 600 (10 minutes).
min-interval *interval*	The minimum time, in seconds, allowed between sending unsolicited multicast router advertisements from the interface. The range of supported values is 3 to 0.75 * **max-interval**. The default is 0.33 * **max-interval**.
other-config-flag *state*	The interfaces uses the administered (stateful) protocol for autoconfiguration of non-address information, as defined in RFC 4862. Supported values are as follows: **true**: Hosts use the administered protocol for autoconfiguration of non-address information. **false**: Hosts use stateless autoconfiguration of non-address information. The default is **false**.
prefix *ipv6net*	Multi-node. The IPv6 prefix to be advertised on the IPv6 interface, in the format *ipv6-address/prefix*. You can define more than one IPv6 prefix by configuring multiple **prefix** configuration nodes.
autonomous-flag *state*	Specifies whether the prefix can be used for autonomous address configuration as defined in RFC 4862. Supported values are as follows: **true**: The prefix can be used for autonomous address configuration. **false**: The prefix cannot be used for autonomous address configuration. The default is **true**.

on-link-flag *state*	Specifies whether the prefix can be used for on-link determination, as defined in RFC 4862. Supported values are as follows: **true**: The prefix can be used for on-link determination. **false**: The advertisement makes no statement about on-link or off-link properties of the prefix. For instance, the prefix might be used for address configuration with some addresses belonging to the prefix being on-link and others being off-link. The default is **true**.
preferred-lifetime *lifetime*	The length of time, in seconds, that the addresses generated from the prefix via stateless address autoconfiguration (SLAAC) is to remain preferred, as defined in RFC 4862. The interval is with respect to the time the packet is sent. The range is 1 to 4294967296 plus the keyword **infinity**, which represents forever. (The actual value of **infinity** is a byte where all bits are set to ones: 0XFFFFFFFF.) The default is 604800 (seven days).
valid-lifetime *lifetime*	The length of time, in seconds, that the prefix is valid for the purpose of on-link determination, as defined in RFC 4862. The interval is with respect to the time the packet is sent. The range is 1 to 4294967296 plus the keyword **infinity**, which represents forever. (The actual value of **infinity** is a byte where all bits are set to ones: 0XFFFFFFFF.) The default is 2592000 (30 days).
reachable-time *time*	The length of time, in milliseconds, for which the system assumes a neighbor is reachable after having received a reachability confirmation. This value is used by address resolution and the Neighbor Unreachability Detection algorithm (see Section 7.3 of RFC 2461). The range is 0 to 3600000, where a value of 0 means the reachable time is not specified in the router advertisement message. The default is 0.
retrans-timer *time*	The length of time, in milliseconds, between retransmitted NS messages. This value is used by address resolution and the Neighbor Unreachability Detection algorithm (see Sections 7.2 and 7.3 of RFC 2461). The range of supported values is 0 to 4294967295, where a value of 0 means the retransmit tme is not specified in the router advertisement message. The default is 0.

send-advert *state*	Specifies whether router advertisements are to be sent from this interface. Supported values are as follows: **true**: Sends router advertisements from this interface. **false**: Does not send router advertisements from this interface. If this value is in effect, parameters in this configuration subtree are still used to configure the local implementation parameters. The default is **true**.

Default

Router advertisements are not sent on an interface.

Usage Guidelines

Use this command to configure router advertisements (RAs) to be sent out of the interface being configured.

Router advertisements are sent out by IPv6 routers in order to advertise their existence to hosts on the network. IPv6 hosts do not send out router advertisements.

If the **router-advert** node of the configuration tree is missing, router advertisements are not sent out. Also, if IPv6 forwarding is disabled either globally (using the **system ipv6 disable-forwarding** command) or on the interface (using the interfaces bridge <brx> ipv6 disable-forwarding command), router advertisements are not sent out.

Most router advertisement parameters are required by either the Neighbor Discovery (ND) protocol or the Stateless Address Auto-Configuration (SLAAC) protocol. These parameters are used both locally for the IPv6 implementation and become part of the RA messages sent to hosts on the network so that they can be configured appropriately.

Use the **set** form of this command to create the **router-advert** configuration node and begin to send router advertisements.

Use the **delete** form of this command to remove **router-advert** configuration node and stop sending router advertisements.

Use the **show** form of this command to view router advertisement configuration.

interfaces bridge <brx> max-age <interval>

Specifies how long a bridge group waits for a hello packet from the spanning tree root.

Syntax

set interfaces bridge *brx* **max-age** *interval*

delete interfaces bridge *brx* **max-age**

show interfaces bridge *brx* **max-age**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        max-age interval
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
interval	The interval a bridge group waits to receive a hello packet before recomputing the spanning-tree topology. The range is 6 to 200. The default is 20.

Default

The bridge group waits 20 seconds for a hello packet before recomputing the spanning-tree topology.

Usage Guidelines

Use this command to specify the interval a bridge group will wait to receive a hello packet from the spanning tree root. If this interval expires without the bridge group having received the hello packet, the bridge group considers the network topology to have changed and recomputes the spanning-tree topology.

Use the **set** form of this command to specify the maximum age interval.

Use the **delete** form of this command to restore the maximum age interval to its default value.

Use the **show** form of this command to view maximum age interval configuration.

interfaces bridge <brx> priority <priority>

Specifies the forwarding priority of a bridge group in the spanning tree.

Syntax

set interfaces bridge *brx* **priority** *priority*

delete interfaces bridge *brx* **priority**

show interfaces bridge *brx* **priority**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        priority priority
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
priority	The forwarding priority of this bridge in the spanning tree. The higher the number, the lower the priority. The default is 0, which is the highest priority.

Default

The default is 0.

Usage Guidelines

Use this command to specify the forwarding priority of this bridge in the spanning tree.

The Spanning Tree Protocol uses the bridge priority to determine the spanning tree root. The lower the number assigned to the bridge group, the higher its priority, and the more likely it is to be selected as the root of the spanning tree.

Use the **set** form of this command to specify the forwarding priority of this bridge in the spanning tree.

Use the **delete** form of this command to restore the priority to its default.

Use the **show** form of this command to view the priority configuration.

interfaces bridge <brx> stp <state>

Enables IEEE 802.1D Spanning Tree Protocol on a bridge group.

Syntax

set interfaces bridge *brx* **stp** *state*

delete interfaces bridge *brx* **stp**

show interfaces bridge *brx* **stp**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bridge brx {
        stp state
    }
}
```

Parameters

brx	The identifier for the bridge group. Supported identifiers are **br0** through **br999**.
stp	Allows you to enable or disable the Spanning Tree Protocol on a per-bridge basis. Supported values are as follows: **true**: Enables Spanning Tree Protocol on this bridge. **false**: Disables Spanning Tree Protocol on this bridge. The default is **false**.

Default

Spanning Tree Protocol is disabled.

Usage Guidelines

Use this command to specify whether or not the IEEE 802.1D Spanning Tree Protocol (STP) is enabled on a bridge group. When STP is enabled on bridge group, it is enabled for all interfaces and vifs assigned to the bridge group.

Use the **set** form of this command to specify whether or not the Spanning Tree Protocol is enabled on the interface.

Use the **delete** form of this command to restore the default.

Use the **show** form of this command to view the configuration.

show bridge

Displays the information for active bridge groups.

Syntax

show bridge [*bridge-group* [**macs** | **spanning-tree**]]

Command Mode

Operational mode.

Parameters

bridge-group	Displays information for the specified bridge group: one of **br0** through **br999**.
macs	Shows the MAC table for the specified bridge group.
spanning-tree	Shows spanning tree information for the specified bridge groups.

Usage Guidelines

Use this command to display information about configured bridge groups.

When used with no option, this command displays information about all active bridge groups. When the identifier of a bridge group is provided, this command displays information for the specified bridge group. You can display the media access control (MAC) table and Spanning Tree Protocol information for a bridge group.

Chapter 4: Bridge Interface Commands

This chapter describes commands for adding interfaces to a bridge group.

Configuration Commands	
interfaces adsl <adslx> pvc <pvc-id> bridged-ethernet	Adds a PVC with RFC 1483 Bridged Ethernet encapsulation on an ADSL interface to a bridge group.
interfaces bonding <bondx> bridge-group bridge <brx>	Adds an Ethernet link bonding interface to a bridge group.
interfaces bonding <bondx> vif <vlan-id> bridge-group bridge <brx>	Adds a bonding vif to a bridge group.
interfaces ethernet <ethx> bridge-group bridge <brx>	Adds an Ethernet interface to a bridge group.
interfaces ethernet <ethx> vif <vlan-id> bridge-group bridge <brx>	Adds an Ethernet vif to a bridge group.
interfaces openvpn <vtunx> bridge-group bridge <brx>	Adds an OpenVPN interface to a bridge group.
interfaces tunnel <tunx> bridge-group bridge <brx>	Adds a tunnel interface to a bridge group.
interfaces wireless <wlanx> bridge-group bridge <brx>	Adds a wireless interface to a bridge group.
Operational Commands	
clear interfaces bridge counters	Clears bridge interface statistics.
show interfaces bridge	Shows bridge interface information.

clear interfaces bridge counters

Clears bridge interface statistics.

Syntax

clear interfaces bridge [*if-name*] **counters**

Command Mode

Operational mode.

Parameters

if-name	The identifier for the interface whose bridging counters you wish to clear. This can be any interface on which bridging is supported.

Default

Statistics are cleared on all bridge interfaces.

Usage Guidelines

Use this command to clear statistics on bridge interfaces.

If no interface is specified then brdige statistics are cleared on all interfaces.

interfaces adsl <adslx> pvc <pvc-id> bridged-ethernet

Adds a PVC with RFC 1483 Bridged Ethernet encapsulation on an ADSL interface to a bridge group.

Syntax

set interfaces adsl *adslx* **pvc** *pvc-id* **bridged-ethernet** [**cost** *cost* | **priority** *priority*]

delete interfaces adsl *adslx* **pvc** *pvc-id* **bridged-ethernet** [**cost** | **priority**]

show interfaces adsl *adslx* **pvc** *pvc-id* **bridged-ethernet**

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    adsl adslx {
        pvc pvc-id {
            bridged-ethernet {
                cost cost
                priority priority
            }
        }
    }
}
```

Parameters

adslx	Multi-node. The identifier for the ADSL interface you are defining. This may be **adsl0** to **adslx**, depending on what physical ADSL ports are actually available on the system.
pvc-id	The identifier for the PVC. It can either be the *vpi*/*vci* pair (e.g., 0/35) or the keyword **auto**, where *vpi* is a Virtual Path Index from 0 to 255, *vci* is a Virtual Circuit Index from from 0 to 65535, and **auto** directs the system to detect the Virtual Path Index and Virtual Circuit Index automatically.
cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.

priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to enable RFC 1483 ADSL-to-Ethernet bridging on an ADSL PVC, and to set the cost and priority values for bridging on the interface.

Use the **set** form of this command to enable bridging on the PVC, or to specify cost or priority.

Use the **delete** form of this command to disable bridging on the PVC, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

interfaces bonding <bondx> bridge-group bridge <brx>

Adds an Ethernet link bonding interface to a bridge group.

Syntax

set interfaces bonding *bondx* **bridge-group bridge** *brx* [**cost** *cost* | **priority** *priority*]

delete interfaces bonding *bondx* **bridge-group bridge** *brx* [**cost** | **priority**]

show interfaces bonding *bondx* **bridge-group bridge** *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bonding bondx {
        bridge-group {
            bridge brx {
                cost cost
                priority priority
            }
        }
    }
}
```

Parameters

bondx	The identifier for the bonding interface. Supported values are **bond0** through **bond99**.
brx	The bridge group ID. Supported identifiers are **br0** through **br999**.
cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.
priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to add an Ethernet link bonding interface to a bridge group, and to set the cost and priority values for the bridge on the interface.

Use the **set** form of this command to add the interface to the bridge group, or to specify cost or priority.

Use the **delete** form of this command to remove the interface from the bridge group, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

interfaces bonding <bondx> vif <vlan-id> bridge-group bridge <brx>

Adds a bonding vif to a bridge group.

Syntax

set interfaces bonding *bondx* **vif** *vlan-id* **bridge-group bridge** *brx* [**cost** *cost* | **priority** *priority*]

delete interfaces bonding *bondx* **vif** *vlan-id* **bridge-group bridge** *brx* [**cost** | **priority**]

show interfaces bonding *bondx* **vif** *vlan-id* **bridge-group bridge** *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    bonding bondx {
        vif vlan-id {
            bridge-group {
                bridge brx {
                    cost cost
                    priority priority
                }
            }
        }
    }
}
```

Parameters

bondx	The identifier for the bonding interface. Supported values are **bond0** through **bond99**.
vlan-id	The ID of the VLAN the vif is connected to. The range is 0 to 4094.
brx	The bridge group ID. Supported identifiers are **br0** through **br999**.

cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.
priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to add a bonding vif to a bridge group, and to set the cost and priority values for the bridge on the interface.

Use the **set** form of this command to add the interface to the bridge group, or to specify cost or priority.

Use the **delete** form of this command to remove the interface from the bridge group, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

interfaces ethernet <ethx> bridge-group bridge <brx>

Adds an Ethernet interface to a bridge group.

Syntax

set interfaces ethernet *ethx* **bridge-group bridge** *brx* [**cost** *cost* | **priority** *priority*]

delete interfaces ethernet *ethx* **bridge-group bridge** *brx* [**cost** | **priority**]

show interfaces ethernet *ethx* **bridge-group bridge** *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    ethernet ethx {
        bridge-group {
            bridge brx {
                cost cost
                priority priority
            }
        }
    }
}
```

Parameters

ethx	The identifier for the Ethernet interface. Supported values are **eth0** through **eth23**.
brx	The bridge group ID. Supported identifiers are **br0** through **br999**.
cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.
priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to add an Ethernet interface to a bridge group, and to set the cost and priority values for the bridge on the interface.

Use the **set** form of this command to add the interface to the bridge group, or to specify cost or priority.

Use the **delete** form of this command to remove the interface from the bridge group, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

interfaces ethernet <ethx> vif <vlan-id> bridge-group bridge <brx>

Adds an Ethernet vif to a bridge group.

Syntax

set interfaces ethernet *ethx* **vif** *vlan-id* **bridge-group bridge** *brx* [**cost** *cost* | **priority** *priority*]

delete interfaces ethernet *ethx* **vif** *vlan-id* **bridge-group bridge** *brx* [**cost** | **priority**]

show interfaces ethernet *ethx* **vif** *vlan-id* **bridge-group bridge** *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    ethernet ethx {
        vif vlan-id {
            bridge-group {
                bridge brx {
                    cost cost
                    priority priority
                }
            }
        }
    }
}
```

Parameters

ethx	The identifier for the Ethernet interface. Supported values are **eth0** through **eth23**.
vlan-id	The ID of the VLAN the vif is connected to. The range is 0 to 4094.
brx	The bridge group ID. Supported identifiers are **br0** through **br999**.

cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.
priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to add an Ethernet vif to a bridge group, and to set the cost and priority values for the bridge on the interface.

Use the **set** form of this command to add the interface to the bridge group, or to specify cost or priority.

Use the **delete** form of this command to remove the interface from the bridge group, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

interfaces openvpn <vtunx> bridge-group bridge <brx>

Adds an OpenVPN interface to a bridge group.

Syntax

set interfaces openvpn *vtunx* **bridge-group bridge** *brx* [**cost** *cost* | **priority** *priority*]

delete interfaces openvpn *vtunx* **bridge-group bridge** *brx* [**cost** | **priority**]

show interfaces openvpn *vtunx* **bridge-group bridge** *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    openvpn vtunx {
        bridge-group {
            bridge brx {
                cost cost
                priority priority
            }
        }
    }
}
```

Parameters

vtunx	The identifier for the OpenVPN interface. Supported values are **vtun0** through **vtun***x*, where *x* is a positive integer.
brx	The bridge group ID. Supported identifiers are **br0** through **br999**.
cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.
priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to add an OpenVPN interface to a bridge group, and to set the cost and priority values for the bridge on the interface.

Use the **set** form of this command to add the interface to the bridge group, or to specify cost or priority.

Use the **delete** form of this command to remove the interface from the bridge group, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

interfaces tunnel <tunx> bridge-group bridge <brx>

Adds a tunnel interface to a bridge group.

Syntax

set interfaces tunnel *tunx* **bridge-group bridge** *brx* [**cost** *cost* | **priority** *priority*]

delete interfaces tunnel *tunx* **bridge-group bridge** *brx* [**cost** | **priority**]

show interfaces tunnel *tunx* **bridge-group bridge** *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    tunnel tunx {
        bridge-group {
            bridge brx {
                cost cost
                priority priority
            }
        }
    }
}
```

Parameters

tunx	The identifier for the tunnel interface. Supported values are **tun0** through **tun23**.
brx	The bridge group ID. Supported identifiers are **br0** through **br999**.
cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.
priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to add a tunnel interface to a bridge group, and to set the cost and priority values for the bridge on the interface.

Use the **set** form of this command to add the interface to the bridge group, or to specify cost or priority.

Use the **delete** form of this command to remove the interface from the bridge group, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

interfaces wireless <wlanx> bridge-group bridge <brx>

Adds a wireless interface to a bridge group.

Syntax

set interfaces wireless *wlanx* **bridge-group bridge** *brx* [**cost** *cost* | **priority** *priority*]

delete interfaces wireless *wlanx* **bridge-group bridge** *brx* [**cost** | **priority**]

show interfaces wireless *wlanx* **bridge-group bridge** *brx*

Command Mode

Configuration mode.

Configuration Statement

```
interfaces {
    wireless wlanx {
        bridge-group {
            bridge brx {
                cost cost
                priority priority
            }
        }
    }
}
```

Parameters

tunx	The identifier for the tunnel interface. Supported values are **tun0** through **tun23**.
brx	The bridge group ID. Supported identifiers are **br0** through **br999**.
cost	The path cost for the interface within its bridge group. The Spanning Tree Protocol (STP) uses this value to calculate the shortest path from this bridge group to the spanning tree root. The range is 1 to 65535. The default is 19.
priority	The path priority for the interface within its bridge group. The range is 0 to 255. The default is 128.

Default

None.

Usage Guidelines

Use this command to add a tunnel interface to a bridge group, and to set the cost and priority values for the bridge on the interface.

Use the **set** form of this command to add the interface to the bridge group, or to specify cost or priority.

Use the **delete** form of this command to remove the interface from the bridge group, or to restore default values for cost and priority.

Use the **show** form of this command to view interface configuration for bridging.

show interfaces bridge

Shows bridge interface information.

Syntax

show interfaces bridge [*bridge-group* [**brief**] | **detail**]

Command Mode

Operational mode.

Parameters

bridge-group	Displays information for the specified bridge group: one of **br0** through **br999**.
brief	Shows a summary of information for a given bridge group.
detail	Shows detailed bridge interface information.

Usage Guidelines

Use this command to display information about configured bridge interfaces.

When used with no option, this command displays information about all active bridge interfaces. When the identifier of a bridge group is provided, this command displays information for the specified bridge group.

Glossary of Acronyms

ACL	access control list
ADSL	Asymmetric Digital Subscriber Line
AMI	Amazon Machine Image
API	Application Programming Interface
AS	autonomous system
ARP	Address Resolution Protocol
AWS	Amazon Web Services
BGP	Border Gateway Protocol
BIOS	Basic Input Output System
BPDU	Bridge Protocol Data Unit
CA	certificate authority
CCMP	AES in counter mode with CBC-MAC
CHAP	Challenge Handshake Authentication Protocol
CLI	command-line interface
DDNS	dynamic DNS
DHCP	Dynamic Host Configuration Protocol
DHCPv6	Dynamic Host Configuration Protocol version 6

DLCI	data-link connection identifier
DMI	desktop management interface
DMZ	demilitarized zone
DN	distinguished name
DNS	Domain Name System
DSCP	Differentiated Services Code Point
DSL	Digital Subscriber Line
eBGP	external BGP
EBS	Amazon Elastic Block Storage
EC2	Amazon Elastic Compute Cloud
EGP	Exterior Gateway Protocol
ECMP	equal-cost multipath
ESP	Encapsulating Security Payload
FIB	Forwarding Information Base
FTP	File Transfer Protocol
GRE	Generic Routing Encapsulation
HDLC	High-Level Data Link Control
I/O	Input/Ouput
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IGP	Interior Gateway Protocol
IPS	Intrusion Protection System
IKE	Internet Key Exchange
IP	Internet Protocol
IPOA	IP over ATM

IPsec	IP security
IPv4	IP Version 4
IPv6	IP Version 6
ISP	Internet Service Provider
KVM	Kernel-Based Virtual Machine
L2TP	Layer 2 Tunneling Protocol
LACP	Link Aggregation Control Protocol
LAN	local area network
LDAP	Lightweight Directory Access Protocol
LLDP	Link Layer Discovery Protocol
MAC	medium access control
MIB	Management Information Base
MLPPP	multilink PPP
MRRU	maximum received reconstructed unit
MTU	maximum transmission unit
NAT	Network Address Translation
ND	Neighbor Discovery
NIC	network interface card
NTP	Network Time Protocol
OSPF	Open Shortest Path First
OSPFv2	OSPF Version 2
OSPFv3	OSPF Version 3
P2P	peer-to-peer
PAM	Pluggable Authentication Module
PAP	Password Authentication Protocol
PAT	Port Address Translation

PCI	peripheral component interconnect
PKI	Public Key Infrastructure
PPP	Point-to-Point Protocol
PPPoA	PPP over ATM
PPPoE	PPP over Ethernet
PPTP	Point-to-Point Tunneling Protocol
PVC	permanent virtual circuit
QoS	quality of service
RADIUS	Remote Authentication Dial-In User Service
RHEL	Red Hat Enterprise Linux
RIB	Routing Information Base
RIP	Routing Information Protocol
RIPng	RIP next generation
Rx	receive
S3	Amazon Simple Storage Service
SLAAC	Stateless Address Auto-Configuration
SNMP	Simple Network Management Protocol
SMTP	Simple Mail Transfer Protocol
SONET	Synchronous Optical Network
SSH	Secure Shell
SSID	Service Set Identifier
STP	Spanning Tree Protocol
TACACS+	Terminal Access Controller Access Control System Plus
TCP	Transmission Control Protocol
TKIP	Temporal Key Integrity Protocol
ToS	Type of Service

Tx	transmit
UDP	User Datagram Protocol
vif	virtual interface
VLAN	virtual LAN
VPC	Amazon virtual private cloud
VPN	Virtual Private Network
VRRP	Virtual Router Redundancy Protocol
WAN	wide area network
WAP	wireless access point
WPA	Wired Protected Access