Bridging Configuration Guide, 17.2.0

November 2017

Supporting AT&T Vyatta Network Operating System

Contents

Copyright Statement

The training materials and other content provided herein for assistance in training on the Vyatta vRouter may have references to Brocade as the Vyatta vRouter was formerly a Brocade product prior to AT&T's acquisition of Vyatta. Brocade remains a separate company and is not affiliated to AT&T.

About This Guide

This guide describes how to configure Layer 2 bridging on AT&T products that run on the AT&T Vyatta Network Operating System (referred to as a virtual router, vRouter, or router in the guide).

Bridging Overview

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:

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 this within the configuration node for the interface itself.

   Info:

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

   - Physical data plane interfaces
   - VLAN interfaces

MTU for bridge groups

The effective maximum transmission unit (MTU) 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.

Spanning Tree Protocol

Spanning Tree Protocol (STP) is a network protocol that ensures a loop-free topology for Ethernet networks. The basic function of STP is to prevent bridge loops. Spanning tree also allows a network design to include redundant links to provide automatic backup paths if an active link fails, thus, eliminating the need to manually enable or disable the backup links.

AT&T Vyatta vRouter supports Rapid Spanning Tree Protocol (RSTP) that is an enhancement of the STP and provides the following advantages.

- Rapid convergence—Convergence in a standard STP network can take 30 to 50 seconds. The transition of a port to the forwarding state is passive and is based on various timers that are timing out. RSTP provides significantly faster spanning tree convergence after the topology changes by introducing new convergence behaviors and bridge port roles. RSTP responds to topology changes within 3 x hello times (default 3 x 2 seconds) or within a few milliseconds of a physical link failure.
- 802.1D legacy interoperability—RSTP interoperates fully with older STP switches. Although the introduction of a 802.1D STP switch means that the network loses its fast convergence benefit, the network is still able to run in a loop-free topology because RSTP interoperates with STP. Typically, a network is designed with all-RSTP bridges, but the backward compatibility ensures that the accidental introduction of an old STP bridge does not cause an outage.

The difference between the STP and RSTP spanning tree versions follow.

Table 1: Difference between the STP and RSTP versions

STP	RSTP
In a stable topology, only the root sends Bridge Protcol Data Units (BPDU)s that are relayed by others.	In a stable topology, all bridges generate BPDUs every hello interval (2 seconds). These are used as keep-alive mechanisms.

STP	RSTP
The following port states are supported: • Disabled • Blocking • Listening • Learning • Forwarding	The following port states are supported: • Disabled • Discarding • Learning • Forwarding Blocking and listening states are replaced with discarding state. The disabled state is not a part of the Rapid STP specification, but is used when the vRouter interface is set to the down state.
The following port roles are supported: • Root (Forwarding) • Designated (Forwarding) • Blocking The port that receives the best BPDU on a bridge, is the root port, that is, the port closest to the root bridge in terms of path cost. A port is a designated port if it can send the best BPDU on the segment to which it is connected. On a given segment, there can be only one path toward the root bridge. A blocking port is defined as not being the designated or root port.	The following port roles are supported: • Root (Forwarding) • Designated (Forwarding) • Alternate (Discarding) • Backup (Discarding) The blocking port role is split into the backup and alternate port roles. A port is a designated port if it can send the best BPDU on the segment to which it is connected. On a given segment, there can only be one path towards the root bridge. An alternate port is a port that receives more useful BPDUs from another bridge and is a blocked port. A backup port receives more useful BPDUs from the same bridge that it is on and is a blocked port.
STP uses the following timers for convergence (advertised by the root bridge): • Hello—2 seconds • Max Age—20 seconds (10 missed hellos) • Forward Delay—15 seconds	The proposal-and-agreement process for synchronization is less than 1 second. Hello, max age, and forward delay timers are used only for backward compatibility with standard STP. Only RSTP port receiving STP (802.1d) messages behave as standard STP.
Slow transition that is 50 seconds, which is as follows: • Blocking (20 seconds) • Listening (15 seconds) • Learning (15 seconds) • Forwarding	Faster transition on point-to-point and edge ports only. There are fewer states and no learning state. RSTP actively looks for possible failure by Request Link Query (RLQ), a feedback mechanism.
Uses only two bits in the BPDU flag octet. • Bit 7—Topology Change Acknowledgement (TCA) Bit • Bit 0—Topology Change	Uses other six bits of the flag octet (for BPDU type 2 or version 2): • Bit 1: Proposal bit • Bits 2 and 3 : Port role bit • Bit 4 : Learning bit • Bit 5 : Forwarding bit • Bit 6 : Agreement bit • Bits 0 and 7 : TCA and TCN for backward compatibility

STP	RSTP
The bridge that discovers a change in the network informs the root, which in turn informs all others by sending BPDUs with the TCA bit set and instructs them to clear their data base entries after the short timer (~Forward delay) expires.	A topology Change (TC) is flooded through the network, every bridge generates a TC and informs its neighbors when it is aware of a TC and immediately deletes old data base entries.
If a nonroot bridge does not receive a hello for a max-age interval of time on a root port, the STP starts claiming the root role by generating its own BPDU.	Waits for 3 times the hello on a root port before deciding to act.

Bridging Configuration Examples

Basic bridging configuration

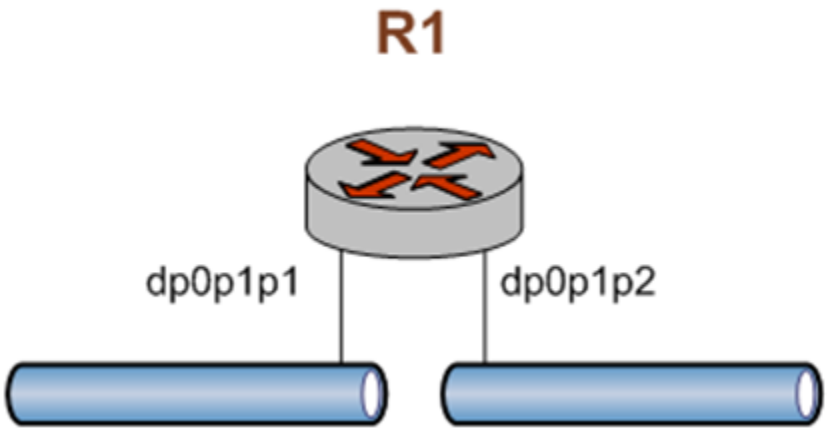
This section presents a sample configuration for a basic bridge between two Ethernet segments on an AT&T Vyatta vRouter.

Note: For information on bridging with GRE tunnels, see AT&T Vyatta Network Operating System Tunnels Configuration Guide.

Note: In the vRouter, a data plane interface is an abstraction that represents the underlying physical or virtual Ethernet interface of the system. The terms Ethernet interface and data plane interface are synonymous in this guide.

When you have finished, the system will be configured as shown in the following figure.

Figure 1: Basic bridging



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

The following steps create the bridge interface and adds the data plane interfaces to the bridge group. To do this, perform the following steps on R1 in configuration mode.

Table 2: Configuring a bridge between two data plane interfaces

Step	Command
Create the bridge interface.	`vyatta@R1# set interfaces bridge br0`
Add the dp0p1p1 interface to the bridge group.	`vyatta@R1# set interfaces dataplane dp0p1p1 bridge-group bridge br0`
Add the dp0p1p2 interface to the bridge group.	`vyatta@R1# set interfaces dataplane dp0p1p2 bridge-group bridge br0`

Step	Command
Commit the configuration.	`vyatta@R1# commit`

View the configuration.	`vyatta@R1# show interfaces`
	`  bridge br0`
	`  dataplane dp0p1p1 {`
	`  bridge-group {`
	`  bridge br0`
	`  }`
	`  }`
	`  dataplane dp0p1p2 {`
	`  bridge-group {`
	`  bridge br0`
	`  }`
	` }

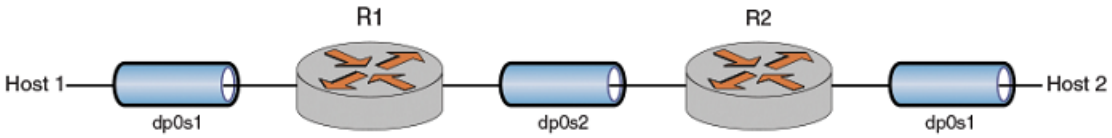
Configuring bridge ports

This section provides a sample configuration for two bridges between Ethernet segments on two AT&T Vyatta vRouters.

> **Note:** In the vRouter, a data plane interface is an abstraction that represents the underlying physical or virtual Ethernet interface of the system. The terms Ethernet interface and data plane interface are synonymous in this guide.

The following example shows two main types of bridge port configurations: a bridge-to-bridge port and an edge port. The same configuration applies to both bridges.

Figure 2: Configuring bridge ports



Ports are configured as edge ports if they are attached to a LAN that has no other bridges attached. These edge ports transition directly to the forwarding state. RSTP still continues to monitor the port for BPDUs if a bridge is connected. RSTP can also be configured to automatically detect edge ports. As soon as the bridge detects a BPDU coming to an edge port, the port becomes a nonedge port.

When you have finished, the system is configured as shown in the following figure.

Configuring bridge ports shows how to create the bridge interface and add the data plane interfaces and port configurations to both bridge groups.

To do this, perform the following steps on both R1 and R2 in configuration mode.

Table 3: Configuring bridge ports

Step	Command
Create the bridge interface.	`vyatta@R1# set interfaces bridge br0`
Enable spanning tree.	`vyatta@R1# set interfaces bridge br0` `  spanning-tree`

Step	Command
Add dp0s1 and dp0s2 interfaces to the bridge.	`vyatta@R1# set interfaces dataplane dp0s1 bridge-group bridge br0` `vyatta@R1# set interfaces dataplane dp0s2 bridge-group bridge br0`
Configure host-facing interfaces as edge ports.	`vyatta@R1# set interfaces dataplane dp0s1 bridge-group admin-edge` `vyatta@R1# set interfaces dataplane dp0s1 bridge-group auto-edge`
Commit the configuration.	`vyatta@R1# commit`

Use the `show brief` command to verify the following configurations.
• The bridge-to-bridge port is of point-to-point type.
• The host-facing port is of edge type .
• Both ports are RSTP version.
	`vyatta@R1:~$ show bridge br0 spanning-tree brief`
	`Bridge br0`
	`Designated Root 8.000.52:54:00:00:01:01`
	`Designated Root Cost 1000`
	`Designated Root Port dp0s2 (2)`
	`Bridge ID 8.000.52:54:00:00:02:02`
	
	`Port State Role Cost Prio Type Ver`
	`dp0s1 (1) forwarding Desg 2000 8 edge rstp`
	`dp0s2 (2) forwarding Root 1000 8 p2p rstp`

Bridge Group Commands

Related bridge group commands

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

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 by bridge interfaces. Commands for working with ARP are described in AT&T Vyatta Network Operating System Basic System Configuration Guide.
DHCP commands	DHCP is supported by bridge groups. For DHCP-related commands, refer to AT&T Vyatta Network Operating System Services Configuration Guide.
Firewall	Firewall is supported by bridge groups. Commands for configuring firewall are described in AT&T Vyatta Network Operating System Firewall Configuration Guide.
Policy-based routing	Policy-based routing is supported by bridge groups. Commands for configuring policy-based routing are described in AT&T Vyatta Network Operating System Policy-based Routing Configuration Guide.
OSPF	OSPF is supported by bridge groups. Commands for configuring OSPF are described in AT&T Vyatta Network Operating System OSPF Configuration Guide.
QoS	Quality-of-service (QoS) traffic policies are supported by bridge groups. Commands for configuring QoS are described in AT&T Vyatta Network Operating System QoS Configuration Guide.
RIP	RIP is supported by bridge groups. Commands for configuring RIP are described in AT&T Vyatta Network Operating System RIP Configuration Guide.
RIPng	RIPng is supported by bridge groups. Commands for configuring RIPng are described in AT&T Vyatta Network Operating System RIPng Configuration Guide.

clear bridge <brx> macs

Clears the FDB for bridge MAC address for a bridge group.

Syntax:

```
clear bridge brx macs
```

brx

The ID of a bridge group.

Operational mode

Use this command to clear the forwarding database (FDB) for bridge MAC address for a bridge group.

Note: After clearing the FDB for bridge MAC address, a short period of unicast-packet flooding may last until the source MAC addresses are relearned.

clear bridge <brx> macs mac-address <mac-address>

Clears from the FDB for bridge MAC address the record that matches the MAC address.

Syntax:
`clear bridge` *brx* **`macs mac-address`** *mac-address*

brx
The ID of a bridge group.
mac-address
A MAC address for which information is to be cleared from the database. The format of the address is *hh*:*hh*:*hh*:*hh*:*hh*:*hh*, where *h* is a hexadecimal number.

Operational mode

Use this command to clear from the FDB for bridge MAC address the record that matches the MAC address.

clear bridge <brx> macs port <port>

Clears from the FDB for bridge MAC address entries that match a bridge interface port.

Syntax:
`clear bridge` *brx* **`macs port`** *port*

brx
The ID of a bridge group.
port
A port for which information is cleared from the database.

Operational mode

Use this command to clear from the FDB for bridge MAC address entries that match a bridge interface port.

clear bridge <brx> macs port <port> mac-address <mac-address>

Clears records from the FDB for bridge MAC address entries of a bridge port that matches with the MAC address.

Syntax:
`clear bridge` *brx* **`macs port`** *port* **`mac-address`** *mac-address*

brx
The ID of a bridge group.
mac-address
A MAC address for which information is cleared from the database. The format of the addresss is *hh*:*hh*:*hh*:*hh*:*hh*:*hh*, where *h* is a hexadecimal number.
port
A port for which information is cleared from the FDB for bridge MAC address.

Operational mode

Use this command to clear records from the FDB for bridge MAC address entries of a bridge port that matches with the MAC address.

interfaces bridge <brx>

Defines a bridge group.

Syntax:
`set interfaces bridge` *brx*

Syntax:
`delete interfaces bridge` *brx*

Syntax:
`show interfaces bridge` *brx*

brx

Multinode. The identifier for the bridge group. The identifier ranges from **br**0 through **br** *xxxxxxxxxxxxx* (the letters **br** followed by as many as 13 decimal digits, each digit represented here as an *x*).

You can define multiple bridge groups by creating more than one **bridge** configuration node.

Configuration mode

```
interfaces {
 bridge brx
}
```

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*

Syntax:
`delete interfaces bridge` *brx* **address** *address*

Syntax:
`show interfaces bridge` *brx* **address**

brx

Bridge group ID.

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.

Configuration mode

```
interfaces {
 bridge brx {
```

```
  address address
 }
}
```

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

Specifies the MAC address aging timeout for a bridge group.

Syntax:
`set interfaces bridge` *brx* **`aging`** { *age* | **`0`** }

Syntax:
`delete interfaces bridge` *brx* **`aging`**

Syntax:
`show interfaces bridge` *brx* **`aging`**

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

brx
Bridge group ID.

`aging { `*`age`*` | ``0`` }`
Specifies the length of time, in seconds, that a MAC address is to be kept before being aged out. The range is 10 through 1,000,000. The default value is 300 seconds. Specifiying an **`aging`** value of **`0`** means that a MAC address is kept forever.

Configuration mode

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

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 value.

Use the `delete` form of this command to restore the default MAC address aging configuration.

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*

Syntax:
`delete interfaces bridge` *brx* **`description`**

Syntax:

```
show interfaces bridge brx description
```

brx

Bridge group ID.

desc

A brief description for the bridge group.

Configuration mode

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

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> disable

Disables a bridge group without discarding configuration.

Syntax:

```
set interfaces bridge brx disable
```

Syntax:

```
delete interfaces bridge brx disable
```

Syntax:

```
show interfaces bridge brx
```

Bridging is enabled.

brx

Bridge group ID.

Configuration mode

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

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 the bridge interface not to detect changes in link-states.

Syntax:

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

Syntax:
`delete interfaces bridge` *brx* **`disable-link-detect`**

Syntax:
`show interfaces bridge` *brx*

Detects changes in link-states.

brx
Bridge group ID.

Configuration mode.

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

Use this command to disable a bridge group.

Use the `set` form of this command to specify whether to ignore changes in link-states.

Use the `delete` form of this command to detect changes in link-states.

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

interfaces bridge <brx> ipv6 address

Assigns an IPv6 address to a bridge interface.

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

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

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

brx
Bridge group ID.

`autoconf`
Generates an IPv6 address using the Stateless Address Autoconfiguration (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.

Configuration mode

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

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

You can use the **autoconf** keyword to direct the system to autoconfigure the address, using the SLAAC 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 to 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 bridge brx ipv6 disable-forwarding *(page 19)* 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`**

Syntax:
`delete interfaces` **`bridge`** *brx* **`ipv6 disable-forwarding`**

Syntax:
`show interfaces bridge` *brx* **`ipv6 disable-forwarding`**

IPv6 packets are forwarded.

brx
Bridge group ID.

Configuration mode

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

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*

Syntax:

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

Syntax:

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

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

brx

Bridge group ID.

num

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

Configuration mode

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

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 ]
```

Syntax:

```
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 ]
```

Syntax:

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

Router advertisements are not sent on an interface.

brx

Bridge group ID.

`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 is 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 be 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 interface 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 through 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 time 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**.

Configuration mode

```
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
    }
  }
}
```

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 *(page 19)* command), router advertisements are not sent out.

Most router advertisement parameters are required by either the Neighbor Discovery (ND) protocol or the Stateless Address Autoconfiguration (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> mac <mac-addr>

Specifies the mac address of a bridge interface.

Syntax:
`set interfaces bridge` *brx* **`mac`** *mac-addr*

Syntax:
`delete interfaces bridge` *brx* **`mac`**

Syntax:
`show interfaces bridge` *brx* **`mac`**

brx
Bridge group ID.

`mac-addr`
The MAC address to be set for the bridge interface. The format should be appropriate for the interface type. For an Ethernet interface, this is six colon-separated 8-bit numbers in hexadecimal; for example, 00:0a:59:9a:f2:ba.

Configuration mode

```
interfaces {
 bridge brx {
  mac mac-addr
  }
 }
}
```

Use this command to set the media access control (MAC) address of the bridge interface. The MAC address for a bridge interface defaults to the MAC address of the lowest numbered data plane interface, that is a member of the bridge group. The bridge MAC address is the least significant part of the bridge ID. The bridge priority is the most significant part of the bridge ID. The bridge with the lowest numerical bridge ID is selected as the route bridge.

Use the `set` form of this command to set the MAC address of the bridge interface.

Use the `delete` form of this command to remove a configured MAC address for the bridge interface.

Use the `show` form of this command to view MAC address configuration for the bridge interface.

interfaces bridge <brx> spanning-tree

Enables spanning tree protocol on a bridge group.

Syntax:
`set interfaces bridge` *brx* **`spanning-tree`**

Syntax:
`delete interfaces bridge` *brx* **`spanning-tree`**

Syntax:
`show interfaces bridge` *brx* **`spanning-tree`**

Spanning tree protocol is disabled.

brx
The ID of bridge group.

`spanning-tree`
Specifies the Spanning Tree Protocol.

Configuration mode

```
interfaces {
   bridge brx {
       spanning-tree
   }
}
```

Use this command to enable the IEEE 802.1D Spanning Tree Protocol (STP) on a bridge group. When STP is enabled on a bridge group, it is enabled for all interfaces and vifs assigned to the bridge group.

Use the `set` form of this command to enable STP on a bridge group.

Use the `delete` form of this command to restore the default state of STP on a bridge group; that is, STP is disabled..

Use the `show` form of this command to view the STP configuration on a bridge group.

interfaces bridge <brx> spanning-tree forwarding-delay <delay>

Specifies the amount of time a bridge group spends in the listening and learning state after a topology change.

Syntax:
`set interfaces bridge` *brx* **`spanning-tree forwarding-delay`** *delay*

Syntax:
`delete interfaces bridge` *brx* **`spanning-tree forwarding-delay`**

Syntax:
`show interfaces bridge` *brx* **`spanning-tree forwarding-delay`**

The forwarding delay is 15 seconds.

brx
The ID of a bridge group.

`spanning-tree`
Specifies spanning tree configuration.

`forwarding-delay` ***delay***
Specifies the length of time, in seconds, that an STP bridge port spends in the listening and learning state, before changing to the forwarding state after a topology change. The delay time ranges from 4 through 30 seconds.

The default value is 15 seconds, that is, the bridge group spends 15 seconds in the listening state, where it listens for spanning tree bridge protocol data unit (BPDU) packets, before changing to the learning state. It then spends 15 seconds in the learning state where it learns the source MAC addresses of other devices on the network, before changing to the forwarding state.

> **Note:** The **`forwarding-delay`** applies to ports in STP mode and RSTP ports that are neither edge ports nor point-to-point ports.

Configuration mode

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

Note: The configuration enforces the following relationship between the **forwarding-delay** **forwarding-delay** and **max-age** spanning tree protocol timers:

2 * (**forwarding-delay** - 1 second) >= **max-age**

Use the **set** form of this command to specify the length of time that the bridge spends in the listening and learning state after a topology change.

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

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

interfaces bridge <brx> spanning-tree hello-time <interval>

Specifies the hello packet advertisement interval for a bridge group.

Syntax:
set interfaces bridge *brx* **spanning-tree hello-time** *interval*

Syntax:
delete interfaces bridge *brx* **spanning-tree hello-time**

Syntax:
show interfaces bridge *brx* **spanning-tree hello-time**

Hello packets are transmitted at two-second intervals.

brx
Bridge group ID.

spanning-tree
Specifies spanning tree configuration.

hello-time ***interval***
Specifies the hello packet advertisement interval in seconds. The range is 1 through 10. The default value is 2 seconds.

Configuration mode

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

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

Use the **delete** form of this command to restore the default **hello-time** configuration.

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

interfaces bridge <brx> spanning-tree max-age <interval>

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

Syntax:

```
set interfaces bridge brx spanning-tree max-age interval
```

Syntax:

```
delete interfaces bridge brx spanning-tree max-age
```

Syntax:

```
show interfaces bridge brx spanning-tree max-age
```

The bridge group removes neighbor bridges at 20-second intervals.

brx
Bridge group ID.

spanning-tree
Specifies spanning tree configuration.

max-age *interval*
Specifies the maximum age, in seconds, of received bridge BPDUs. The range is 6 through 40. The default value is 20 seconds.

Configuration mode

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

Note: The configuration enforces the following relationship between the **forwarding-delay** and **max-age** spanning tree protocol timers:

2 * (**forwarding-delay** - 1 second) >= **max-age**

Use this command to specify the interval at which neighbor bridges are removed.

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

Use the `delete` form of this command to restore the default maximum age interval configuration.

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

interfaces bridge <brx> spanning-tree priority <priority>

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

Syntax:

```
set interfaces bridge brx spanning-tree priority priority
```

Syntax:

```
delete interfaces bridge brx spanning-tree priority
```

Syntax:

```
show interfaces bridge brx spanning-tree priority
```

The priority value is 8.

brx
Bridge group ID.

spanning-tree
Specifies spanning tree configuration.

priority

Specifies the forwarding priority of this bridge group in the spanning tree. The lower the number, the higher the priority. The range is from 0 through 15. The default value is 8.

Configuration mode

```
interfaces {
    bridge brx {
        spanning-tree {
            priority priority
        }
    }
}
```

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 default priority configuration.

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

interfaces bridge <brx> spanning-tree tx-hold-count <count>

Specifies the maximum number of BPDUs that a bridge can send each second.

Syntax:
`set interfaces bridge` *brx* **`spanning-tree tx-hold-count`** *count*

Syntax:
`delete interfaces bridge` *brx* **`spanning-tree tx-hold-count`** *count*

Syntax:
`show interfaces bridge` *brx* **`spanning-tree tx-hold-count`**

None

brx
The ID of a Bridge group.

count
The maximum number of BPDUs transmitted during one hello time period. The number ranges from 1 through 10.

Configuration mode

```
interfaces {
    bridge brx {
        spanning-tree {
            tx-hold-count count
        }
    }
}
```

Use the `set` form of this command to specify the maximum number of BPDUs that a bridge can send each second.

Use the `delete` form of this command to delete the number of BPDUs that the bridge can send each second.

Use the `show` form of this command to display the number of BPDUs that the bridge can send each second.

interfaces bridge <brx> spanning-tree version <stp | rstp>

Specifies the version of the spanning tree that the bridge must use.

Syntax:
```
set interfaces bridge brx spanning-tree version [ stp | rstp ]
```

Syntax:
```
delete interfaces bridge brx spanning-tree version { stp | rstp }
```

Syntax:
```
show interfaces bridge brx spanning-tree version
```

The version of spanning tree is RSTP.

brx
The ID of a Bridge group.

stp
Specifies that the bridge must use the STP (IEEE 802.1D) version of spanning tree.

rstp
Specifies that the bridge must use the RSTP (IEEE 802.1w) version of spanning tree.

Configuration mode

```
interfaces {
    bridge brx {
        spanning-tree {
            version stp
            version rstp
        }
    }
}
```

Use the `set` form of this command to specify the version of spanning tree that a bridge must use.

Use the `delete` form of this command to restore to the default version of the spanning tree, which is RSTP.

Use the `show` form of this command to view the version of spanning tree for a bridge.

> **Note:** The bridge reverts to STP even after the version of spanning tree is configured to RSTP on those ports where it receives STP (STP version 0) BPDUs. A port remains in this state even after the STP bridge is removed. To revert a port to RSTP mode, use the `clear bridge brx spanning-tree version` command.

monitor interfaces bridge <brx>

Monitors traffic to the vRouter.

Syntax:
```
monitor interfaces bridge brx
```

brx
Bridge group ID.

Operational mode

```
monitor interfaces bridge
```

Use this command to monitor traffic to the vRouter and traffic forwarding.

show bridge

Displays information about a bridge group.

Syntax:
`show bridge [ brx ]`

brx
The ID of a bridge group.

Operational mode

Use this command to display information about the bridge group.

When a bridge group is not specified, the command displays information about all active bridge groups. When a bridge group is specified, the command displays information about the specified bridge group.

The following example shows how to display the details about the br0 bridge group.

```
vyatta@R$ show bridge br0
bridge name  bridge id                 STP enabled    interfaces (port)
br0          8.000.52:54:00:00:01:01     yes            dp0p1s1 (2)
                                                        dp0p1s2 (1)
```

show bridge <brx> macs

Displays FDB of bridge MAC address for a bridge group.

Syntax:
`show bridge brx macs`

brx
The ID of a bridge group.

Operational mode

Use this command to display the FDB of bridge MAC address for a bridge group. Returned entries are sorted by MAC address. The Age column in the output displays the number of seconds since a bridge port received a packet with that source MAC address.

The following example shows how to display the FDB of MAC addresses for the br0 bridge group.

```
vyatta@R$ show bridge br0 macs
Interface (port)   MAC Address             Type   Age
dp0p1s3 (1)        12:f7:71:97:4d:43       dynamic 0
dp0p1s1 (2)        5a:d2:19:f7:f9:3d       dynamic 0
```

show bridge <brx> macs mac-address <mac-address>

Displays from the FDB of bridge MAC addresses the record that matches the MAC address.

Syntax:
`show bridge brx macs mac-address mac-address`

brx
The ID of a bridge group.
mac-address

The MAC address for which information is displayed. The format of the address is *hh:hh:hh:hh:hh:hh,* where *h* is a hexadecimal number.

Operational mode

Use this command to display from the forwarding database (FDB) of bridge MAC addresses the record that matches the MAC address. The age column in the output displays the number of seconds since a bridge port received a packet with that source MAC address.

The following example shows how to display the record from the FDB of bridge MAC addresses the record that matches the 26:ef:f7:6d:4b:5f MAC address of the br0 bridge.

```
show bridge br0 macs mac-address 26:ef:f7:6d:4b:5f

Interface (port)   MAC Address             Type            Age
dp0p1s4 (1)        26:ef:f7:6d:4b:5f       dynamic         2
```

show bridge <brx> macs port <port>

Displays from the FDB of bridge MAC address the records that match a port.

Syntax:
`show bridge` *brx* **`macs port`** *port*

brx
The ID of the bridge group.

port
A port for which information is displayed.

Operational mode

Use this command to display from the FDB of bridge MAC address the records that match a port. Returned entries are sorted by MAC address. The Age column in the output displays the number of seconds since a bridge port received a packet with that source MAC address.

The following example shows how to display from the FDB of bridge MAC address the records that match a port dp0p1s4 in the br0 bridge.

```
show bridge br0 macs port dp0p1s4

Interface (port)   MAC Address             Type            Age

dp0p1s4 (1)        26:ef:f7:6d:4b:5f       dynamic         2
dp0p1s4 (1)        72:ea:4b:ea:70:03       dynamic         10
dp0p1s4 (1)        ba:43:7f:c3:75:2c       dynamic         76
```

show bridge <brx> macs port <port> mac-address <mac-address>

Displays from the FDB of bridge MAC address the records that match a bridge interface port and MAC address.

Syntax:
`show bridge` *brx* **`macs mac-address`** *mac-address*

brx
The ID of bridge group.

port

A port for which information is displayed.

mac-address

The MAC address for which information is displayed. The format of the address is *hh:hh:hh:hh:hh:hh*, where *h* is a hexadecimal number.

Operational mode

Use this command to display from the FDB of bridge MAC address the records that match a bridge interface port and MAC address. The Age column in the output displays the number of seconds since a bridge port received a packet with that source MAC address.

The following example shows how to display from the FDB of bridge MAC address the records that match the dp0p1s4 bridge interface port and 26:ef:f7:6d:4b:5f MAC address for the br0 bridge.

```
show bridge br0 macs port dp0p1s4 mac-address 26:ef:f7:6d:4b:5f

Interface (port)   MAC Address             Type            Age
dp0p1s4 (1)        26:ef:f7:6d:4b:5f       dynamic         2
```

show bridge <brx> spanning-tree <brief>

Displays spanning tree information for a bridge group.

Syntax:

`show bridge` *brx* **`spanning-tree`** [**`brief`**]

brx

The ID of the bridge group.

`brief`

Displays a summary of spanning tree information for a bridge group.

Operational mode

Use this command to display spanning tree information for a bridge group. Use the `brief` parameter to display a summary of the spanning tree information. You see a detailed information when you do not use the `brief` parameter.

The following example shows how to display a summary of spanning tree information for the br0 bridge group.

```
vyatta@R1$ show bridge br0 spanning-tree brief

Bridge                  br0
Designated Root         8.000.52:54:00:00:01:01
Designated Root Cost    2000
Designated Root Port    dp0p1s1 (2)
Bridge ID               8.000.52:54:00:00:02:01

Port                State         Role   Cost       Prio  Type   Ver
dp0p1s1 (2)         forwarding    Root   2000       8     p2p    rstp
dp0p1s2 (3)         discarding    Altn   2000       8     p2p    rstp
dp0p1s3 (4)         discarding    Back   2000       8     p2p    rstp
dp0s8 (1)           forwarding    Desg   2000       8     p2p    rstp
```

The following example shows how to display a detailed output of spanning tree information for the br0 bridge group.

```
vyatta@R1$ show bridge br0 spanning-tree

br0
  link enabled    yes
  stp enabled     yes
  version         rstp
  bridge id       8.000.52:54:00:00:02:01
  designated root 8.000.52:54:00:00:01:01
  root port       dp0p1s1 (2)
  path cost     2000       internal path cost   0
  max age       20         bridge max age       20
  forward delay 15         bridge forward delay 15
  tx hold count 6          max hops             20
  hello time    2          ageing time          300
  time since topology change 12931
  topology change count      4
  topology change            no
  topology change port       dp0p1s3 (1)
  last topology change port  dp0p1s3 (1)

br0:dp0p1s1 (2)
  link enabled       yes                    role                Root
  port id            8.002                  state               forwarding
  port cost          2000                   admin cost          auto
  designated root    8.000.52:54:00:00:01:01 dsgn cost          0
  designated bridge  8.000.52:54:00:00:01:01 designated port     8.003
  admin edge port    no                     auto edge port      no
  oper edge port     no                     topology change ack  no
  point-to-point     no                     admin point-to-point no
  root block         no                     restricted TCN      no
  port hello time    2                      disputed            no
  bpdu guard port    no                     bpdu guard error    no
  network port       no                     BA inconsistent     no
  Num sent BPDU      9                      Num sent TCN        5
  Num rcvd BPDU      6487                   Num rcvd TCN        6
  Num Transition FWD 1                      Num Transition BLK  0
  Rcvd BPDU          none                   Send RSTP           yes

br0:dp0p1s2 (3)
  link enabled       yes                    role                Alternate
  port id            8.003                  state               discarding
  port cost          2000                   admin cost          auto
  designated root    8.000.52:54:00:00:01:01 dsgn cost          0
  designated bridge  8.000.52:54:00:00:01:01 designated port     8.004
  admin edge port    no                     auto edge port      no
  oper edge port     no                     topology change ack  no
  point-to-point     no                     admin point-to-point no
  root block         no                     restricted TCN      no
  port hello time    2                      disputed            no
  bpdu guard port    no                     bpdu guard error    no
  network port       no                     BA inconsistent     no
  Num sent BPDU      5                      Num send TCN        0
  Num rcvd BPDU      6487                   Num rcvd TCN        8
  Num Transition FWD 0                      Num Transition BLK  0
  Rcvd BPDU          none                   Send RSTP           yes
```

show bridge <brx> spanning-tree bridge <brief>

Displays bridge spanning tree information for a bridge group.

Syntax:

```
show bridge brx  spanning-tree bridge  [  brief ]
```

brx

The ID of the bridge group.

brief

Displays a summary of spanning tree information for a bridge group.

Operational mode

Use this command to display bridge spanning tree information for a bridge group. Use the `brief` parameter to display a summary of the spanning tree information for the bridge group. You see a detailed output when you do not use the `brief` parameter.

The following example shows how to display the spanning tree information for the br0 bridge-group.

```
vyatta@R1$ show bridge br0 spanning-tree bridge

br0
   link enabled    yes
   stp enabled     yes
   version         rstp
   bridge id       8.000.52:54:00:00:02:01
   designated root 8.000.52:54:00:00:01:01
   root port       dp0p1s1 (2)
   path cost        2000       internal path cost   0
   max age          20         bridge max age       20
   forward delay    15         bridge forward delay 15
   tx hold count    6          max hops             20
   hello time       2          ageing time          300
   time since topology change 12931
   topology change count      4
   topology change            no
   topology change port       dp0p1s3 (1)
   last topology change port  dp0p1s3 (1)
```

The following example shows how to display a summary of spanning tree information for the br0 bridge-group.

```
vyatta@R1$ show bridge br0 spanning-tree bridge brief

Bridge                  br0
Designated Root         8.000.52:54:00:00:01:01
Designated Root Cost    8000
Designated Root Port    dp0p1s1 (2)
Bridge ID               8.000.52:54:00:00:02:01
```

show bridge <brx> spanning-tree port <port>

Displays spanning tree information for a bridge interface port.

Syntax:

```
show bridge brx  spanning-tree port  port
```

Syntax:

brx

Bridge group ID.

port

Specifies the port for which spanning tree information is displayed.

`brief`

Displays a summary of spanning tree information for a port.

Operational mode

Use this command to display spanning tree information for a bridge interface port. Use the `brief` parameter to display a summary of the spanning tree port information. You see detailed information when you do not use the `brief` parameter .

The following example shows how to display spanning tree dp0p1s1 port information for the br0 bridge group.

```
vyatta@R1$ show bridge br0 spanning-tree port dp0p1s1

br0:dp0p1s1 (2)
link enabled  yes   role          Root
port id       8.002 state         forwarding
port cost     2000  admin co      auto
designated root 8.000.52:54:00:00:01:01 dsgn cost  0
designated bridge 8.000.52:54:00:00:01:01 designated port  8.003
admin edge port    no             auto edge port    no
oper edge port     no             topology change ack  no
point-to-point     no             admin point-to-point no
root block         no             restricted TCN       no
port hello time    2 disputed     no
bpdu guard port    no             bpdu guard error     no
network port       no             BA inconsistent      no
Num sent BPDU      9              Num sent TCN         5
Num rcvd BPDU      6487           Num rcvd TCN         6
Num Transition FWD 1              Num Transition BLK   0
Rcvd BPDU          none           Send RSTP            yes
```

Bridge Interface Commands

clear interfaces bridge counters

Clears bridge interface statistics.

Syntax:

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

Statistics are cleared on all bridge interfaces.

if-name

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

Operational mode

Use this command to clear statistics on bridge interfaces.

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

interfaces dataplane <interface-name> bridge-group

Adds a data plane interface to a bridge group.

Syntax:

```
set interfaces dataplane interface-name bridge-group [ admin-edge | auto-edge | bpdu-guard | bridge brx
| cost cost | network-port | point-to-point status | priority priority | restrict-tcn | root-block ]
```

Syntax:

```
delete interfaces dataplane interface-name bridge-group [ admin-edge | auto-edge | bpdu-guard | bridge
| cost | network-port | point-to-point | priority | restrict-tcn | root-block ]
```

Syntax:

```
show interfaces dataplane interface-name bridge-group
```

interface-name

The name of a data plane interface. For more information about the supported name formats of a data plane interface, refer to Supported Interface Types *(page 39)*.

admin-edge

Sets the initial-edge state, specifying that the port connects to an end node instead of another spanning tree bridge. The default is off.

auto-edge

Allows the bridge to automatically determine the edge-port status. The default is off.

bpdu-guard

Enables the spanning tree BPDU guard. The BPDU guard is used at the network edge, where the port connects directly to an end node. The default is off.

bridge *brx*

The bridge group ID.

cost *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 value can be a numerical value that ranges from 1 through 200000000 or the auto keyword. The default is auto. If you use the auto keyword, the vRouter determines the port cost from the line speed.

network-port

Enables Spanning Tree uni-directional link detection.

point-to-point *status*
Sets the point-to-point operational status to one of the following values:

- auto: Determines the point-to-point operational status from the duplex setting.
- off: Disable the point-to-point operational status.
- on: Enable the point-to-point operational status.

priority
The path priority for the interface within its bridge group. The range is 0 to 63. The default is 0.

restrict-tcn
Restricts propagation of topology change notifications for the spanning tree.

root-block
Restricts the ability of ports to assume the spanning tree root role.

Configuration mode

```
interfaces {
       dataplane dpxpypz {
              bridge-group {
                    admin-edge
                    auto-edge
                    bpdu-guard
                    bridge brx
                    cost cost
                    network-port
                    point-to-point status
                    priority priority
                    restrict-tcn
                    root-block
              }
       }
}
```

Use this command to add a data plane 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 dataplane <interface> vif <vif-id> bridge-group

Adds a data plane vif to a bridge group.

Syntax:
`set interfaces dataplane` *interface* **vif** *vif-id* **bridge-group** **admin-edge** | **auto-edge** | **bpdu-guard** | **bridge** *brx* | **network-port** | **cost** *cost* | **point-to-point** *status* | **priority** *priority* | **restrict-tcn** | **root-block**]

Syntax:
`delete interfaces dataplane` *interface* **vif** *vif-id* **bridge-group** [**admin-edge** | **auto-edge** | **bpdu-guard** | **bridge** | **network-port** | **cost** | **priority** | **restrict-tcn** | **root-block**]

Syntax:
`show interfaces dataplane` *interface* **vif** *vif-id* **bridge-group**

interface
The name of a data plane interface. For more information about the supported name formats of a data plane interface, refer to Supported Interface Types *(page 39)*.

vif-id
A virtual interface ID. The ID ranges from 1 through 4094.

admin-edge
Enables the Spanning Tree admin edge mode.

auto-edge
Enables Spanning Tree automatic admin edge detection.

bpdu-guard
Enables Spanning Tree Protocol PortFast Bridge Protocol Data Unit (BPDU) guard.

bridge ***brx***
The bridge group ID.

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 200000000. The default is 19.

network-port
Enables Spanning Tree uni-directional link detection.

point-to-point ***status***
Sets the point-to-point operational status to one of the following values:

- auto: Determines the point-to-point operational status from the duplex setting.
- off: Disable the point-to-point operational status.
- on: Enable the point-to-point operational status.

priority
The path priority for the interface within its bridge group. The range is 0 to 15. The default is 8.

restrict-tcn
Restricts propagation of topology change notifications for Spanning Tree.

root-block
Restricts the ability of ports to assume the Spanning Tree root role.

Configuration mode

```
interfaces {
       dataplane interface {
              vif vif-id {
                     bridge-group {
                            admin-edge
                            auto-edge
                            bpdu-guard
                            bridge brx
                            cost cost
                            network-port
                            point-to-point status
                            priority priority
                            restrict-tcn
                            root-block
                     }
              }
       }
}
```

Use this command to add a data plane vif to a bridge group, and to set the supported values for the bridge on the interface.

Use the `set` form of this command to add a data plane vif to a bridge group, or to set the supported values for the bridge on the interface.

Use the `delete` form of this command to remove the interface from the bridge group, or to restore default values.

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

monitor interfaces bridge <brx>

Monitors traffic to the vRouter.

Syntax:
`monitor interfaces bridge` brx

brx

Bridge group ID.

Operational mode

```
monitor interfaces bridge
```

Use this command to monitor traffic to the vRouter and traffic forwarding.

show interfaces bridge

Shows bridge interface information.

Syntax:
`show interfaces bridge` [*bridge-group* [**`brief`**] | **`detail`**]

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.

Operational mode

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.

Supported Interface Types

The following table shows the syntax and parameters of supported interface types. Depending on the command, some of these types may not apply.

Interface Type	Syntax	Parameters
Bridge	`bridge` *brx*	*brx*: The name of a bridge group. The name ranges from br0 through br999.

Interface Type	Syntax	Parameters
Data plane	`dataplane` *interface-name*	*interface-name*: The name of a data plane interface. Following are the supported formats of the interface name: • `dp`*x*`p`*y*`p`*z*—The name of a data plane interface, where — `dp`*x* specifies the data plane identifier (ID). Currently, only dp0 is supported. — `p`*y* specifies a physical or virtual PCI slot index (for example, p129). — `p`*z* specifies a port index (for example, p1). For example, dp0p1p2, dp0p160p1, and dp0p192p1. • `dp`*x*`em`*y* —The name of a data plane interface on a LAN-on-motherboard (LOM) device that does not have a PCI slot, where `em`*y* specifies an embedded network interface number (typically, a small number). For example, dp0em3. • `dp`*x*`s`*y*—The name of a data plane interface in a system in which the BIOS identifies the network interface card to reside in a particular physical or virtual slot *y*, where *y* is typically a small number. For example, for the dp0s2 interface, the BIOS identifies slot 2 in the system to contain this interface. • `dp`*x*`P`*n*`p`*y*`p`*z* —The name of a data plane interface on a device that is installed on a secondary PCI bus, where `P`*n* specifies the bus number. You can use this format to name data plane interfaces on large physical devices with multiple PCI buses. For these devices, it is possible to have network interface cards installed on different buses with these cards having the same slot ID. The value of *n* must be an integer greater than 0. For example, dp0P1p162p1 and dp0P2p162p1.

Interface Type	Syntax	Parameters
Data plane vif	`dataplane` *interface-name* `vif` *vif-id* [`vlan` *vlan-id*]	*interface-name*: Refer to the preceding description. *vif-id*: A virtual interface ID. The ID ranges from 1 through 4094. *vlan-id*: The VLAN ID of a virtual interface. The ID ranges from 1 through 4094.
Loopback	`loopback lo` or `loopback lo`*n*	*n*: The name of a loopback interface, where *n* ranges from 1 through 99999.
OpenVPN	`openvpn` *vtunx*	*vtunx*: The identifier of an OpenVPN interface. The identifier ranges from vtun0 through vtun*x*, where *x* is a nonnegative integer.
Tunnel	`tunnel` *tunx* or `tunnel` *tunx* `parameters`	*tunx*: The identifier of a tunnel interface you are defining. The identifier ranges from tun0 through tun*x*, where *x* is a nonnegative integer.
Virtual tunnel	`vti` *vtix*	*vtix*: The identifier of a virtual tunnel interface you are defining. The identifier ranges from vti0 through vti*x*, where *x* is a nonnegative integer. **Note:** Before you can configure a vti interface, you must configure a corresponding vpn. **Note:** This interface does not support IPv6.
VRRP	*parent-interface* `vrrp vrrp-group` *group*	*parent-interface*: The type and identifier of a parent interface; for example, data plane dp0p1p2 or bridge br999. *group*: A VRRP group identifier. The name of a VRRP interface is not specified. The system internally constructs the interface name from the parent interface identifier plus the VRRP group number; for example, dp0p1p2v99. Note that VRRP interfaces support the same feature set as does the parent interface.

List of Acronyms

Acronym	Description
ACL	access control list
ADSL	Asymmetric Digital Subscriber Line
AH	Authentication Header
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
DMVPN	dynamic multipoint VPN
DMZ	demilitarized zone
DN	distinguished name
DNS	Domain Name System

Acronym	Description
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/Output
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IGMP	Internet Group Management Protocol
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
ISAKMP	Internet Security Association and Key Management Protocol
ISM	Internet Standard Multicast

Acronym	Description
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
mGRE	multipoint GRE
MIB	Management Information Base
MLD	Multicast Listener Discovery
MLPPP	multilink PPP
MRRU	maximum received reconstructed unit
MTU	maximum transmission unit
NAT	Network Address Translation
NBMA	Non-Broadcast Multi-Access
ND	Neighbor Discovery
NHRP	Next Hop Resolution Protocol
NIC	network interface card
NTP	Network Time Protocol
OSPF	Open Shortest Path First
OSPFv2	OSPF Version 2
OSPFv3	OSPF Version 3
PAM	Pluggable Authentication Module
PAP	Password Authentication Protocol
PAT	Port Address Translation
PCI	peripheral component interconnect

Acronym	Description
PIM	Protocol Independent Multicast
PIM-DM	PIM Dense Mode
PIM-SM	PIM Sparse Mode
PKI	Public Key Infrastructure
PPP	Point-to-Point Protocol
PPPoA	PPP over ATM
PPPoE	PPP over Ethernet
PPTP	Point-to-Point Tunneling Protocol
PTMU	Path Maximum Transfer Unit
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
RP	Rendezvous Point
RPF	Reverse Path Forwarding
RSA	Rivest, Shamir, and Adleman
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
SPT	Shortest Path Tree
SSH	Secure Shell

Acronym	Description
SSID	Service Set Identifier
SSM	Source-Specific Multicast
STP	Spanning Tree Protocol
TACACS+	Terminal Access Controller Access Control System Plus
TBF	Token Bucket Filter
TCP	Transmission Control Protocol
TKIP	Temporal Key Integrity Protocol
ToS	Type of Service
TSS	TCP Maximum Segment Size
Tx	transmit
UDP	User Datagram Protocol
VHD	virtual hard disk
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