NAME

bridge - show / manipulate bridge addresses and devices

SYNOPSIS

bridge [*OPTIONS*] *OBJECT* { *COMMAND* | **help** }

OBJECT := { **link** | **fdb** | **mdb** | **vlan** | **monitor** }

OPTIONS := { **-V**[*ersion*] | **-s**[*tatistics*] }

bridge link set dev *DEV [* **cost** *COST*] [**priority** *PRIO*] [**state** *STATE*] [**guard** { **on** | **off** }] [**hairpin** { **on** | **off** }] [**fastleave** { **on** | **off** }] [**root_block** { **on** | **off** }] [**learning** { **on** | **off** }] [**flood** { **on** | **off** }] [**hwmode** { **vepa** | **veb** }]

bridge link [**show**] [**dev** *DEV*]

bridge fdb { **add** | **append** | **del** } *LLADDR* **dev** *DEV* { **local** | **temp** } { **self** } { **embedded** } { **router** } [**dst** *IPADDR*] [**vni** *VNI*] [**port** *PORT*] [**via** *DEVICE*]

bridge fdb [**show**] [**dev** *DEV*]

bridge mdb { **add** | **del** } **dev** *DEV* **port** *PORT* **grp** *GROUP* [**permanent** | **temp**]

bridge mdb show [**dev** *DEV*]

bridge vlan { **add** | **del** } **dev** *DEV* **vid** *VID* [**pvid**] [**untagged**] [**self**] [**master**]

bridge vlan [**show**] [**dev** *DEV*]

bridge monitor [**all** | **neigh** | **link** | **mdb**]

OPTIONS

-V, **-Version**
: print the version of the **bridge** utility and exit.

-s, **-stats**, **-statistics**
: output more information. If this option is given multiple times, the amount of information increases. As a rule, the information is statistics or some time values.

BRIDGE - COMMAND SYNTAX

OBJECT

link - Bridge port.

fdb - Forwarding Database entry.

mdb - Multicast group database entry.

vlan - VLAN filter list.

COMMAND

Specifies the action to perform on the object. The set of possible actions depends on the object type. As a rule, it is possible to **add**, **delete** and **show** (or **list**) objects, but some objects do not allow all of these operations or have some additional commands. The **help** command is available for all objects. It prints out a list of available commands and argument syntax conventions.

If no command is given, some default command is assumed. Usually it is **list** or, if the objects of this class cannot be listed, **help**.

bridge link - bridge port

link objects correspond to the port devices of the bridge.

The corresponding commands set and display port status and bridge specific attributes.

bridge link set - set bridge specific attributes on a port

dev *NAME*
interface name of the bridge port

cost *COST*
the STP path cost of the specified port.

priority *PRIO*
the STP port priority. The priority value is an unsigned 8-bit quantity (number between 0 and 255). This metric is used in the designated port an droot port selectio algorithms.

state *STATE*
the operation state of the port. This is primarily used by user space STP/RSTP implementation. The following is a list of valid values:

0 - port is DISABLED. Make this port completely inactive.

1 - STP LISTENING state. Only valid if STP is enabled on the brige. In this state the port for list for STP BPDUs and drop all other traffic.

2 - STP LEARNING state. Only valid if STP is enabled on the bridge. In this state the port will accept traffic only for the purpose of updating MAC adress tables.

3 - STP FORWARDING state. Port is fully active.

4 - STP BLOCKING state. Only valid if STP is enabled on the bridge. This state is used during the STP election process. In this state, port will only process STP BPDUs.

guard on or **guard off**
Controls whether STP BPUDs will be processed by the bridge port. By default, the flag is turned off allowed BPDU processing. Turning this flag on will cause the port to stop processing STP BPDUs.

hairpin on or **hairpin off**
Controls whether traffic may be send back out of the port on which it was received. By default, this flag is turned off and the bridge will not forward traffic back out of the receiving port.

fastleave on or **fastleave off**
This flag allows the bridge to immediately stop multicast traffic on a port that receives IGMP Leave message. It is only used with IGMP snooping is enabled on the bridge. By default the flag is off.

root_block on or **root_block off**
Controls whether a given port is allowed to become root port or not. Only used when STP is enabled on the bridge. By default the flag is off.

learning on or **learning off**
Controls whether a given port will learn MAC addresses from received traffic or not. If learning if off, the bridge will end up flooding any traffic for which it has no FDB entry. By default this flag is on.

flooding on or **flooding off**
Controls whether a given port will flood unicast traffic for which there is no FDB entry. By default this flag is on.

hwmode
Some network interface cards support HW bridge functionality and they may be configured in different modes. Currently support modes are:

vepa - Data sent between HW ports is sent on the wire to the external switch.

veb - bridging happens in hardware.

bridge link show - list bridge port configuration.
This command displays the current bridge port configuration and flags.

bridge fdb - forwarding database management

fdb objects contain known Ethernet addresses on a link.

The corresponding commands display fdb entries, add new entries, append entries, and delete old ones.

bridge fdb add - add a new fdb entry
This command creates a new fdb entry.

LLADDR
the Ethernet MAC address.

dev *DEV*
the interface to which this address is associated.

self - the address is associated with a software fdb (default)

embedded - the address is associated with an offloaded fdb

router - the destination address is associated with a router. Valid if the referenced device is a VXLAN type device and has route shortcircuit enabled.

The next command line parameters apply only when the specified device *DEV* is of type VXLAN.

dst *IPADDR*
the IP address of the destination VXLAN tunnel endpoint where the Ethernet MAC ADDRESS resides.

vni *VNI*
the VXLAN VNI Network Identifier (or VXLAN Segment ID) to use to connect to the remote VXLAN tunnel endpoint. If omitted the value specified at vxlan device creation will be used.

port *PORT*
the UDP destination PORT number to use to connect to the remote VXLAN tunnel endpoint. If omitted the default value is used.

via *DEVICE*
device name of the outgoing interface for the VXLAN device driver to reach the remote VXLAN tunnel endpoint.

bridge fdb append - append a forwarding database entry
This command adds a new fdb entry with an already known *LLADDR*. Valid only for multicast link layer addresses. The command adds support for broadcast and multicast Ethernet MAC addresses. The Ethernet MAC address is added multiple times into the forwarding database and the vxlan device driver sends a copy of the data packet to each entry found.

The arguments are the same as with **bridge fdb add**,

bridge fdb delete - delete a forwarding database entry
This command removes an existing fdb entry.

The arguments are the same as with **bridge fdb add**,

bridge fdb show - list forwarding entries.
This command displays the current forwarding table.

With the **-statistics** option, the command becomes verbose. It prints out the last updated and last used time for each entry.

bridge mdb - multicast group database management

mdb objects contain known IP multicast group addresses on a link.

The corresponding commands display mdb entries, add new entries, and delete old ones.

bridge mdb add - add a new multicast group database entry
This command creates a new mdb entry.

dev *DEV*
the interface where this group address is associated.

port *PORT*
the port whose link is known to have members of this multicast group.

grp *GROUP*
the IP multicast group address whose members reside on the link connected to the port.

permanent - the mdb entry is permanent

temp - the mdb entry is temporary (default)

bridge mdb delete - delete a multicast group database entry
This command removes an existing mdb entry.

The arguments are the same as with **bridge mdb add**.

bridge mdb show - list multicast group database entries
This command displays the current multicast group membership table. The table is populated by IGMP and MLD snooping in the bridge driver automatically. It can be altered by **bridge mdb add** and **bridge mdb del** commands manually too.

dev *DEV*
the interface only whose entries should be listed. Default is to list all bridge interfaces.

With the **-details** option, the command becomes verbose. It prints out the ports known to have a connected router.

bridge vlan - VLAN filter list

vlan objects contain known VLAN IDs for a link.

The corresponding commands display vlan filter entries, add new entries, and delete old ones.

bridge vlan add - add a new vlan filter entry
This command creates a new vlan filter entry.

dev *NAME*
the interface with which this vlan is associated.

vid *VID*
the VLAN ID that identifies the vlan.

pvid the vlan specified is to be considered a PVID at ingress. Any untagged frames will be assigned to this VLAN.

untagged
the vlan specified is to be treated as untagged on egress.

self the vlan is configured on the specified physical device. Required if the device is the bridge device.

master the vlan is configured on the software bridge (default).

bridge vlan delete - delete a forwarding database entry
This command removes an existing fdb entry.

The arguments are the same as with **bridge vlan add**. The**pvid** and **untagged** flags are ignored.

bridge vlan show - list vlan configuration.
This command displays the current VLAN filter table.

bridge monitor - state monitoring

The **bridge** utility can monitor the state of devices and addresses continuously. This option has a slightly different format. Namely, the **monitor** command is the first in the command line and then the object list follows:

bridge monitor [**all** | *OBJECT-LIST*]

OBJECT-LIST is the list of object types that we want to monitor. It may contain**link**, **fdb**, and **mdb**. If no **file** argument is given, **bridge** opens RTNETLINK, listens on it and dumps state changes in the format described in previous sections.

If a file name is given, it does not listen on RTNETLINK, but opens the file containing RTNETLINK messages saved in binary format and dumps them. Such a history file can be generated with the

NOTES

This command uses facilities added in Linux 3.0.

Although the forwarding table is maintained on a per-bridge device basis the bridge device is not part of the syntax. This is a limitation of the underlying netlink neighbour message protocol. When displaying the forwarding table, entries for all bridges are displayed. Add/delete/modify commands determine the underlying bridge device based on the bridge to which the corresponding ethernet device is attached.

SEE ALSO

ip(8)

BUGS

Please direct bugreports and patches to: **<netdev@vger.kernel.org>**

AUTHOR

Original Manpage by Stephen Hemminger