NAME

Open_vSwitch – Open_vSwitch database schema

A database with this schema holds the configuration for one Open vSwitch daemon. The top-level configuration for the daemon is the **Open_vSwitch** table, which must have exactly one record. Records in other tables are significant only when they can be reached directly or indirectly from the **Open_vSwitch** table. Records that are not reachable from the **Open_vSwitch** table are automatically deleted from the database, except for records in a few distinguished "root set" tables.

Common Columns

Most tables contain two special columns, named **other_config** and **external_ids**. These columns have the same form and purpose each place that they appear, so we describe them here to save space later.

other_config: map of string-string pairs

Key-value pairs for configuring rarely used features. Supported keys, along with the forms taken by their values, are documented individually for each table.

A few tables do not have **other_config** columns because no key-value pairs have yet been defined for them.

external_ids: map of string-string pairs

Key-value pairs for use by external frameworks that integrate with Open vSwitch, rather than by Open vSwitch itself. System integrators should either use the Open vSwitch development mailing list to coordinate on common key-value definitions, or choose key names that are likely to be unique. In some cases, where key-value pairs have been defined that are likely to be widely useful, they are documented individually for each table.
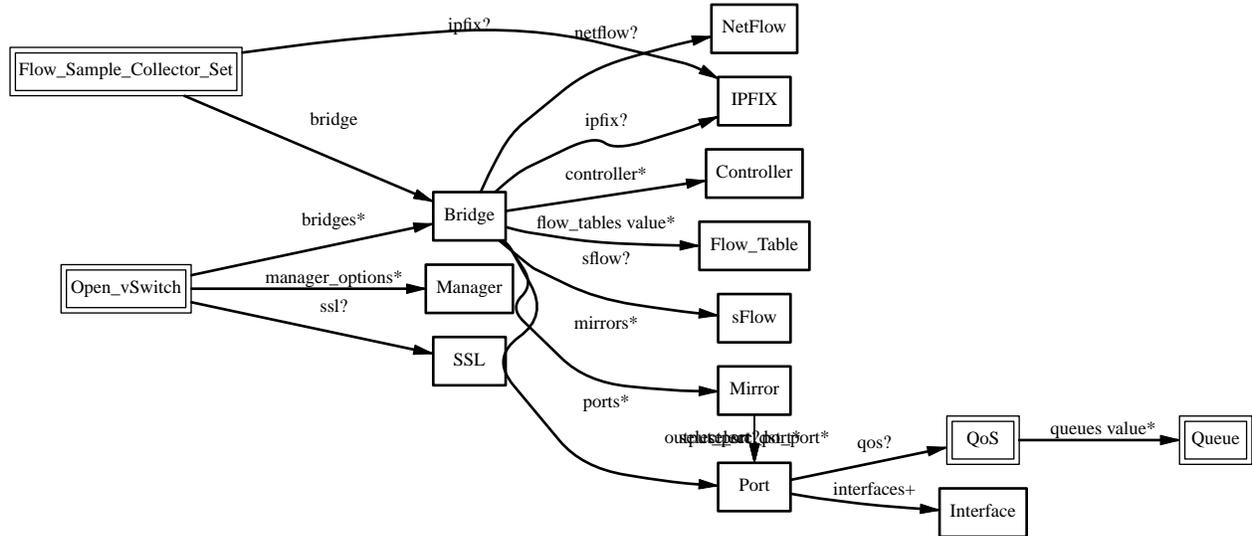
TABLE SUMMARY

The following list summarizes the purpose of each of the tables in the **Open_vSwitch** database. Each table is described in more detail on a later page.

Table	Purpose
Open_vSwitch	Open vSwitch configuration.
Bridge	Bridge configuration.
Port	Port configuration.
Interface	One physical network device in a Port.
Flow_Table	OpenFlow table configuration
QoS	Quality of Service configuration
Queue	QoS output queue.
Mirror	Port mirroring.
Controller	OpenFlow controller configuration.
Manager	OVSDB management connection.
NetFlow	NetFlow configuration.
SSL	SSL configuration.
sFlow	sFlow configuration.
IPFIX	IPFIX configuration.
Flow_Sample_Collector_Set	
	Flow_Sample_Collector_Set configuration.

TABLE RELATIONSHIPS

The following diagram shows the relationship among tables in the database. Each node represents a table. Tables that are part of the "root set" are shown with double borders. Each edge leads from the table that contains it and points to the table that its value represents. Edges are labeled with their column names, followed by a constraint on the number of allowed values: **?** for zero or one, ***** for zero or more, **+** for one or more. Thick lines represent strong references; thin lines represent weak references.

Open_vSwitch TABLE
Configuration for an Open vSwitch daemon. There must be exactly one record in the **Open_vSwitch** table.

Summary:
Configuration:

bridges	set of **Bridge**s
ssl	optional **SSL**
external_ids : system-id	optional string
external_ids : xs-system-uuid	optional string
other_config : stats-update-interval	optional string, containing an integer, at least 5,000
other_config : flow-restore-wait	optional string, either **true** or **false**
other_config : flow-limit	optional string, containing an integer, at least 0
other_config : n-dpdk-rxqs	optional string, containing an integer, at least 1
other_config : pmd-cpu-mask	optional string
other_config : n-handler-threads	optional string, containing an integer, at least 1
other_config : n-revalidator-threads	optional string, containing an integer, at least 1

Status:

next_cfg	integer
cur_cfg	integer

    Statistics:

other_config : enable-statistics	optional string, either **true** or **false**
statistics : cpu	optional string, containing an integer, at least 1
statistics : load_average	optional string
statistics : memory	optional string
statistics : process_NAME	optional string
statistics : file_systems	optional string

Version Reporting:

ovs_version	optional string
db_version	optional string
system_type	optional string
system_version	optional string

Database Configuration:

manager_options	set of **Manager**s

Common Columns:

other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:
Configuration:

bridges: set of **Bridge**s
> Set of bridges managed by the daemon.

ssl: optional **SSL**
> SSL used globally by the daemon.

external_ids : system-id: optional string
> A unique identifier for the Open vSwitch's physical host. The form of the identifier depends on the type of the host. On a Citrix XenServer, this will likely be the same as **external_ids:xs-system-uuid**.

external_ids : xs-system-uuid: optional string
> The Citrix XenServer universally unique identifier for the physical host as displayed by **xe host−list**.

other_config : stats-update-interval: optional string, containing an integer, at least 5,000
> Interval for updating statistics to the database, in milliseconds. This option will affect the update of the **statistics** column in the following tables: **Port**, **Interface** , **Mirror**.

Default value is 5000 ms.

Getting statistics more frequently can be achieved via OpenFlow.

other_config : flow-restore-wait: optional string, either **true** or **false**

When **ovs−vswitchd** starts up, it has an empty flow table and therefore it handles all arriving packets in its default fashion according to its configuration, by dropping them or sending them to an OpenFlow controller or switching them as a standalone switch. This behavior is ordinarily desirable. However, if **ovs−vswitchd** is restarting as part of a ''hot-upgrade,'' then this leads to a relatively long period during which packets are mishandled.

This option allows for improvement. When **ovs−vswitchd** starts with this value set as **true**, it will neither flush or expire previously set datapath flows nor will it send and receive any packets to or from the datapath. When this value is later set to **false**, **ovs−vswitchd** will start receiving packets from the datapath and re-setup the flows.

Thus, with this option, the procedure for a hot-upgrade of **ovs−vswitchd** becomes roughly the following:

1. Stop **ovs−vswitchd**.

2. Set **other_config:flow-restore-wait** to **true**.

3. Start **ovs−vswitchd**.

4. Use **ovs−ofctl** (or some other program, such as an OpenFlow controller) to restore the OpenFlow flow table to the desired state.

5. Set **other_config:flow-restore-wait** to **false** (or remove it entirely from the database).

The **ovs−ctl**'s ''restart'' and ''force-reload-kmod'' functions use the above config option during hot upgrades.

other_config : flow-limit: optional string, containing an integer, at least 0

The maximum number of flows allowed in the datapath flow table. Internally OVS will choose a flow limit which will likely be lower than this number, based on real time network conditions.

The default is 200000.

other_config : n-dpdk-rxqs: optional string, containing an integer, at least 1

Specifies the number of rx queues to be created for each dpdk interface. If not specified or specified to 0, one rx queue will be created for each dpdk interface by default.

other_config : pmd-cpu-mask: optional string

Specifies CPU mask for setting the cpu affinity of PMD (Poll Mode Driver) threads. Value should be in the form of hex string, similar to the dpdk EAL '-c COREMASK' option input or the 'taskset' mask input.

The lowest order bit corresponds to the first CPU core. A set bit means the corresponding core is available and a pmd thread will be created and pinned to it. If the input does not cover all cores, those uncovered cores are considered not set.

If not specified, one pmd thread will be created for each numa node and pinned to any available core on the numa node by default.

other_config : n-handler-threads: optional string, containing an integer, at least 1

Specifies the number of threads for software datapaths to use for handling new flows. The default the number of online CPU cores minus the number of revalidators.

This configuration is per datapath. If you have more than one software datapath (e.g. some **system** bridges and some **netdev** bridges), then the total number of threads is **n−handler−threads** times the number of software datapaths.

other_config : n-revalidator-threads: optional string, containing an integer, at least 1

Specifies the number of threads for software datapaths to use for revalidating flows in the datapath. Typically, there is a direct correlation between the number of revalidator threads, and the number

of flows allowed in the datapath. The default is the number of cpu cores divided by four plus one. If **n−handler−threads** is set, the default changes to the number of cpu cores minus the number of handler threads.

This configuration is per datapath. If you have more than one software datapath (e.g. some **system** bridges and some **netdev** bridges), then the total number of threads is **n−handler−threads** times the number of software datapaths.

Status:

next_cfg: integer
> Sequence number for client to increment. When a client modifies any part of the database configuration and wishes to wait for Open vSwitch to finish applying the changes, it may increment this sequence number.

cur_cfg: integer
> Sequence number that Open vSwitch sets to the current value of **next_cfg** after it finishes applying a set of configuration changes.

Statistics:

The **statistics** column contains key-value pairs that report statistics about a system running an Open vSwitch. These are updated periodically (currently, every 5 seconds). Key-value pairs that cannot be determined or that do not apply to a platform are omitted.

other_config : enable-statistics: optional string, either **true** or **false**
> Statistics are disabled by default to avoid overhead in the common case when statistics gathering is not useful. Set this value to **true** to enable populating the **statistics** column or to **false** to explicitly disable it.

statistics : cpu: optional string, containing an integer, at least 1
> Number of CPU processors, threads, or cores currently online and available to the operating system on which Open vSwitch is running, as an integer. This may be less than the number installed, if some are not online or if they are not available to the operating system.

> Open vSwitch userspace processes are not multithreaded, but the Linux kernel-based datapath is.

statistics : load_average: optional string
> A comma-separated list of three floating-point numbers, representing the system load average over the last 1, 5, and 15 minutes, respectively.

statistics : memory: optional string
> A comma-separated list of integers, each of which represents a quantity of memory in kilobytes that describes the operating system on which Open vSwitch is running. In respective order, these values are:

> 1. Total amount of RAM allocated to the OS.

> 2. RAM allocated to the OS that is in use.

> 3. RAM that can be flushed out to disk or otherwise discarded if that space is needed for another purpose. This number is necessarily less than or equal to the previous value.

> 4. Total disk space allocated for swap.

> 5. Swap space currently in use.

> On Linux, all five values can be determined and are included. On other operating systems, only the first two values can be determined, so the list will only have two values.

statistics : process_NAME: optional string
> One such key-value pair, with **NAME** replaced by a process name, will exist for each running Open vSwitch daemon process, with *name* replaced by the daemon's name (e.g. **process_ovs−vswitchd**). The value is a comma-separated list of integers. The integers represent the following, with memory measured in kilobytes and durations in milliseconds:

1. The process's virtual memory size.

2. The process's resident set size.

3. The amount of user and system CPU time consumed by the process.

4. The number of times that the process has crashed and been automatically restarted by the monitor.

5. The duration since the process was started.

6. The duration for which the process has been running.

The interpretation of some of these values depends on whether the process was started with the **−−monitor**. If it was not, then the crash count will always be 0 and the two durations will always be the same. If **−−monitor** was given, then the crash count may be positive; if it is, the latter duration is the amount of time since the most recent crash and restart.

There will be one key-value pair for each file in Open vSwitch's "run directory" (usually **/var/run/openvswitch**) whose name ends in **.pid**, whose contents are a process ID, and which is locked by a running process. The *name* is taken from the pidfile's name.

Currently Open vSwitch is only able to obtain all of the above detail on Linux systems. On other systems, the same key-value pairs will be present but the values will always be the empty string.

statistics : file_systems: optional string
A space-separated list of information on local, writable file systems. Each item in the list describes one file system and consists in turn of a comma-separated list of the following:

1. Mount point, e.g. **/** or **/var/log**. Any spaces or commas in the mount point are replaced by underscores.

2. Total size, in kilobytes, as an integer.

3. Amount of storage in use, in kilobytes, as an integer.

This key-value pair is omitted if there are no local, writable file systems or if Open vSwitch cannot obtain the needed information.

Version Reporting:
These columns report the types and versions of the hardware and software running Open vSwitch. We recommend in general that software should test whether specific features are supported instead of relying on version number checks. These values are primarily intended for reporting to human administrators.

ovs_version: optional string
The Open vSwitch version number, e.g. **1.1.0**.

db_version: optional string
The database schema version number in the form *major***.***minor***.***tweak*, e.g. **1.2.3**. Whenever the database schema is changed in a non-backward compatible way (e.g. deleting a column or a table), *major* is incremented. When the database schema is changed in a backward compatible way (e.g. adding a new column), *minor* is incremented. When the database schema is changed cosmetically (e.g. reindenting its syntax), *tweak* is incremented.

The schema version is part of the database schema, so it can also be retrieved by fetching the schema using the Open vSwitch database protocol.

system_type: optional string
An identifier for the type of system on top of which Open vSwitch runs, e.g. **XenServer** or **KVM**.

System integrators are responsible for choosing and setting an appropriate value for this column.

system_version: optional string
The version of the system identified by **system_type**, e.g. **5.6.100−39265p** on XenServer 5.6.100 build 39265.

System integrators are responsible for choosing and setting an appropriate value for this column.

Database Configuration:

These columns primarily configure the Open vSwitch database (**ovsdb−server**), not the Open vSwitch switch (**ovs−vswitchd**). The OVSDB database also uses the **ssl** settings.

The Open vSwitch switch does read the database configuration to determine remote IP addresses to which in-band control should apply.

manager_options: set of **Manager**s

Database clients to which the Open vSwitch database server should connect or to which it should listen, along with options for how these connection should be configured. See the **Manager** table for more information.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

other_config: map of string-string pairs

external_ids: map of string-string pairs

Bridge TABLE

Configuration for a bridge within an **Open_vSwitch**.

A **Bridge** record represents an Ethernet switch with one or more ''ports,'' which are the **Port** records pointed to by the **Bridge**'s **ports** column.

Summary:

Core Features:

name	immutable string (must be unique within table)
ports	set of **Port**s
mirrors	set of **Mirror**s
netflow	optional **NetFlow**
sflow	optional **sFlow**
ipfix	optional **IPFIX**
flood_vlans	set of up to 4,096 integers, in range 0 to 4,095

OpenFlow Configuration:

controller	set of **Controller**s
flow_tables	map of integer-**Flow_Table** pairs, key in range 0 to 254
fail_mode	optional string, either **secure** or **standalone**
datapath_id	optional string
other_config : datapath-id	optional string
other_config : dp-desc	optional string
other_config : disable-in-band	optional string, either **true** or **false**
other_config : in-band-queue	optional string, containing an integer, in range 0 to 4,294,967,295
protocols	set of strings, one of **OpenFlow11**, **OpenFlow10**, **OpenFlow13**, **OpenFlow12**, **OpenFlow15**, or **Open-Flow14**

Spanning Tree Configuration:

stp_enable	boolean
other_config : stp-system-id	optional string
other_config : stp-priority	optional string, containing an integer, in range 0 to 65,535
other_config : stp-hello-time	optional string, containing an integer, in range 1 to 10
other_config : stp-max-age	optional string, containing an integer, in range 6 to 40
other_config : stp-forward-delay	optional string, containing an integer, in range 4 to 30
other_config : mcast-snooping-aging-time	optional string, containing an integer, at least 1
other_config : mcast-snooping-table-size	optional string, containing an integer, at least 1
other_config : mcast-snooping-disable-flood-unregistered	
	optional string, either **true** or **false**

Multicast Snooping Configuration:

mcast_snooping_enable	boolean

Rapid Spanning Tree Configuration:

rstp_enable	boolean
other_config : rstp-address	optional string
other_config : rstp-priority	optional string, containing an integer, in range 0 to 61,440
other_config : rstp-ageing-time	optional string, containing an integer, in range 10 to 1,000,000
other_config : rstp-force-protocol-version	optional string, containing an integer
other_config : rstp-max-age	optional string, containing an integer, in range 6 to 40
other_config : rstp-forward-delay	optional string, containing an integer, in range 4 to 30
other_config : rstp-transmit-hold-count	optional string, containing an integer, in range 1 to 10

Other Features:

datapath_type	string
external_ids : bridge-id	optional string
external_ids : xs-network-uuids	optional string
other_config : hwaddr	optional string
other_config : forward-bpdu	optional string, either **true** or **false**
other_config : mac-aging-time	optional string, containing an integer, at least 1
other_config : mac-table-size	optional string, containing an integer, at least 1

Bridge Status:

status	map of string-string pairs
status : stp_bridge_id	optional string
status : stp_designated_root	optional string
status : stp_root_path_cost	optional string

Common Columns:

other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:

Core Features:

name: immutable string (must be unique within table)
> Bridge identifier. Should be alphanumeric and no more than about 8 bytes long. Must be unique among the names of ports, interfaces, and bridges on a host.

ports: set of **Port**s
> Ports included in the bridge.

mirrors: set of **Mirror**s
> Port mirroring configuration.

netflow: optional **NetFlow**
> NetFlow configuration.

sflow: optional **sFlow**
> sFlow(R) configuration.

ipfix: optional **IPFIX**
> IPFIX configuration.

flood_vlans: set of up to 4,096 integers, in range 0 to 4,095
> VLAN IDs of VLANs on which MAC address learning should be disabled, so that packets are flooded instead of being sent to specific ports that are believed to contain packets' destination MACs. This should ordinarily be used to disable MAC learning on VLANs used for mirroring (RSPAN VLANs). It may also be useful for debugging.
>
> SLB bonding (see the **bond_mode** column in the **Port** table) is incompatible with **flood_vlans**. Consider using another bonding mode or a different type of mirror instead.

OpenFlow Configuration:

controller: set of **Controller**s
> OpenFlow controller set. If unset, then no OpenFlow controllers will be used.
>
> If there are primary controllers, removing all of them clears the flow table. If there are no primary controllers, adding one also clears the flow table. Other changes to the set of controllers, such as adding or removing a service controller, adding another primary controller to supplement an existing primary controller, or removing only one of two primary controllers, have no effect on the flow table.

flow_tables: map of integer-**Flow_Table** pairs, key in range 0 to 254
> Configuration for OpenFlow tables. Each pair maps from an OpenFlow table ID to configuration for that table.

fail_mode: optional string, either **secure** or **standalone**

>When a controller is configured, it is, ordinarily, responsible for setting up all flows on the switch. Thus, if the connection to the controller fails, no new network connections can be set up. If the connection to the controller stays down long enough, no packets can pass through the switch at all. This setting determines the switch's response to such a situation. It may be set to one of the following:

>**standalone**
>>If no message is received from the controller for three times the inactivity probe interval (see **inactivity_probe**), then Open vSwitch will take over responsibility for setting up flows. In this mode, Open vSwitch causes the bridge to act like an ordinary MAC-learning switch. Open vSwitch will continue to retry connecting to the controller in the background and, when the connection succeeds, it will discontinue its standalone behavior.

>**secure**   Open vSwitch will not set up flows on its own when the controller connection fails or when no controllers are defined. The bridge will continue to retry connecting to any defined controllers forever.

>The default is **standalone** if the value is unset, but future versions of Open vSwitch may change the default.

>The **standalone** mode can create forwarding loops on a bridge that has more than one uplink port unless STP is enabled. To avoid loops on such a bridge, configure **secure** mode or enable STP (see **stp_enable**).

>When more than one controller is configured, **fail_mode** is considered only when none of the configured controllers can be contacted.

>Changing **fail_mode** when no primary controllers are configured clears the flow table.

datapath_id: optional string

>Reports the OpenFlow datapath ID in use. Exactly 16 hex digits. (Setting this column has no useful effect. Set **other-config:datapath-id** instead.)

other_config : datapath-id: optional string

>Exactly 16 hex digits to set the OpenFlow datapath ID to a specific value. May not be all-zero.

other_config : dp-desc: optional string

>Human readable description of datapath. It it a maximum 256 byte-long free-form string to describe the datapath for debugging purposes, e.g. **switch3 in room 3120**.

other_config : disable-in-band: optional string, either **true** or **false**

>If set to **true**, disable in-band control on the bridge regardless of controller and manager settings.

other_config : in-band-queue: optional string, containing an integer, in range 0 to 4,294,967,295

>A queue ID as a nonnegative integer. This sets the OpenFlow queue ID that will be used by flows set up by in-band control on this bridge. If unset, or if the port used by an in-band control flow does not have QoS configured, or if the port does not have a queue with the specified ID, the default queue is used instead.

protocols: set of strings, one of **OpenFlow11**, **OpenFlow10**, **OpenFlow13**, **OpenFlow12**, **OpenFlow15**, or **OpenFlow14**

>List of OpenFlow protocols that may be used when negotiating a connection with a controller. OpenFlow 1.0, 1.1, 1.2, and 1.3 are enabled by default if this column is empty.

>OpenFlow 1.4 is not enabled by default because its implementation is missing features.

>OpenFlow 1.5 has the same risks as OpenFlow 1.4, but it is even more experimental because the OpenFlow 1.5 specification is still under development and thus subject to change. Pass **−−enable−of15** to **ovs−vswitchd** to allow OpenFlow 1.5 to be enabled.

Spanning Tree Configuration:

>The IEEE 802.1D Spanning Tree Protocol (STP) is a network protocol that ensures loop-free topologies. It

allows redundant links to be included in the network to provide automatic backup paths if the active links fails.

stp_enable: boolean

> Enable spanning tree on the bridge. By default, STP is disabled on bridges. Bond, internal, and mirror ports are not supported and will not participate in the spanning tree.

other_config : stp-system-id: optional string

> The bridge's STP identifier (the lower 48 bits of the bridge-id) in the form *xx:xx:xx:xx:xx:xx*. By default, the identifier is the MAC address of the bridge.

other_config : stp-priority: optional string, containing an integer, in range 0 to 65,535

> The bridge's relative priority value for determining the root bridge (the upper 16 bits of the bridge-id). A bridge with the lowest bridge-id is elected the root. By default, the priority is 0x8000.

other_config : stp-hello-time: optional string, containing an integer, in range 1 to 10

> The interval between transmissions of hello messages by designated ports, in seconds. By default the hello interval is 2 seconds.

other_config : stp-max-age: optional string, containing an integer, in range 6 to 40

> The maximum age of the information transmitted by the bridge when it is the root bridge, in seconds. By default, the maximum age is 20 seconds.

other_config : stp-forward-delay: optional string, containing an integer, in range 4 to 30

> The delay to wait between transitioning root and designated ports to **forwarding**, in seconds. By default, the forwarding delay is 15 seconds.

other_config : mcast-snooping-aging-time: optional string, containing an integer, at least 1

> The maximum number of seconds to retain a multicast snooping entry for which no packets have been seen. The default is currently 300 seconds (5 minutes). The value, if specified, is forced into a reasonable range, currently 15 to 3600 seconds.

other_config : mcast-snooping-table-size: optional string, containing an integer, at least 1

> The maximum number of multicast snooping addresses to learn. The default is currently 2048. The value, if specified, is forced into a reasonable range, currently 10 to 1,000,000.

other_config : mcast-snooping-disable-flood-unregistered: optional string, either **true** or **false**

> If set to **false**, unregistered multicast packets are forwarded to all ports. If set to **true**, unregistered multicast packets are forwarded to ports connected to multicast routers.

Multicast Snooping Configuration:

Multicast snooping (RFC 4541) monitors the Internet Group Management Protocol (IGMP) traffic between hosts and multicast routers. The switch uses what IGMP snooping learns to forward multicast traffic only to interfaces that are connected to interested receivers. Currently it supports IGMPv1 and IGMPv2 protocols.

mcast_snooping_enable: boolean

> Enable multicast snooping on the bridge. For now, the default is disabled.

Rapid Spanning Tree Configuration:

In IEEE Std 802.1D, 1998 Edition, and prior editions of this standard, Clause 8 specified the spanning tree algorithm and protocol (STP). STP has now been superseded by the Rapid Spanning Tree Protocol (RSTP) specified in Clause 17 of the IEEE Std 802.1D, 2004 Edition. The IEEE 802.1D–2004 Rapid Spanning Tree Algorithm Protocol configures full, simple, and symmetric connectivity throughout a Bridged Local Area Network that comprises individual LANs interconnected by Bridges. Like STP, RSTP is a network protocol that ensures loop-free topologies. It allows redundant links to be included in the network to provide automatic backup paths if the active links fails.

rstp_enable: boolean

> Enable Rapid Spanning Tree on the bridge. By default, RSTP is disabled on bridges. Bond, internal, and mirror ports are not supported and will not participate in the spanning tree.

other_config : rstp-address: optional string
>   The bridge's RSTP address (the lower 48 bits of the bridge-id) in the form *xx*:*xx*:*xx*:*xx*:*xx*:*xx*. By default, the address is the MAC address of the bridge.

other_config : rstp-priority: optional string, containing an integer, in range 0 to 61,440
>   The bridge's relative priority value for determining the root bridge (the upper 16 bits of the bridge-id). A bridge with the lowest bridge-id is elected the root. By default, the priority is 0x8000 (32768). This value needs to be a multiple of 4096, otherwise it's rounded to the nearest inferior one.

other_config : rstp-ageing-time: optional string, containing an integer, in range 10 to 1,000,000
>   The Ageing Time parameter for the Bridge. The default value is 300 seconds.

other_config : rstp-force-protocol-version: optional string, containing an integer
>   The Force Protocol Version parameter for the Bridge. This can take the value 0 (STP Compatibility mode) or 2 (the default, normal operation).

other_config : rstp-max-age: optional string, containing an integer, in range 6 to 40
>   The maximum age of the information transmitted by the Bridge when it is the Root Bridge. The default value is 20.

other_config : rstp-forward-delay: optional string, containing an integer, in range 4 to 30
>   The delay used by STP Bridges to transition Root and Designated Ports to Forwarding. The default value is 15.

other_config : rstp-transmit-hold-count: optional string, containing an integer, in range 1 to 10
>   The Transmit Hold Count used by the Port Transmit state machine to limit transmission rate. The default value is 6.

Other Features:

datapath_type: string
>   Name of datapath provider. The kernel datapath has type **system**. The userspace datapath has type **netdev**.

external_ids : bridge-id: optional string
>   A unique identifier of the bridge. On Citrix XenServer this will commonly be the same as **external_ids:xs-network-uuids**.

external_ids : xs-network-uuids: optional string
>   Semicolon-delimited set of universally unique identifier(s) for the network with which this bridge is associated on a Citrix XenServer host. The network identifiers are RFC 4122 UUIDs as displayed by, e.g., **xe network–list**.

other_config : hwaddr: optional string
>   An Ethernet address in the form *xx*:*xx*:*xx*:*xx*:*xx*:*xx* to set the hardware address of the local port and influence the datapath ID.

other_config : forward-bpdu: optional string, either **true** or **false**
>   Option to allow forwarding of BPDU frames when NORMAL action is invoked. Frames with reserved Ethernet addresses (e.g. STP BPDU) will be forwarded when this option is enabled and the switch is not providing that functionality. If STP is enabled on the port, STP BPDUs will never be forwarded. If the Open vSwitch bridge is used to connect different Ethernet networks, and if Open vSwitch node does not run STP, then this option should be enabled. Default is disabled, set to **true** to enable. The following destination MAC addresss will not be forwarded when this option is enabled.

>   **01:80:c2:00:00:00**
>   >   IEEE 802.1D Spanning Tree Protocol (STP).

>   **01:80:c2:00:00:01**
>   >   IEEE Pause frame.

01:80:c2:00:00:0x
> Other reserved protocols.

00:e0:2b:00:00:00
> Extreme Discovery Protocol (EDP).

00:e0:2b:00:00:04 and **00:e0:2b:00:00:06**
> Ethernet Automatic Protection Switching (EAPS).

01:00:0c:cc:cc:cc
> Cisco Discovery Protocol (CDP), VLAN Trunking Protocol (VTP), Dynamic Trunking Protocol (DTP), Port Aggregation Protocol (PAgP), and others.

01:00:0c:cc:cc:cd
> Cisco Shared Spanning Tree Protocol PVSTP+.

01:00:0c:cd:cd:cd
> Cisco STP Uplink Fast.

01:00:0c:00:00:00
> Cisco Inter Switch Link.

01:00:0c:cc:cc:cx
> Cisco CFM.

other_config : mac-aging-time: optional string, containing an integer, at least 1
> The maximum number of seconds to retain a MAC learning entry for which no packets have been seen. The default is currently 300 seconds (5 minutes). The value, if specified, is forced into a reasonable range, currently 15 to 3600 seconds.
>
> A short MAC aging time allows a network to more quickly detect that a host is no longer connected to a switch port. However, it also makes it more likely that packets will be flooded unnecessarily, when they are addressed to a connected host that rarely transmits packets. To reduce the incidence of unnecessary flooding, use a MAC aging time longer than the maximum interval at which a host will ordinarily transmit packets.

other_config : mac-table-size: optional string, containing an integer, at least 1
> The maximum number of MAC addresses to learn. The default is currently 2048. The value, if specified, is forced into a reasonable range, currently 10 to 1,000,000.

Bridge Status:
> Status information about bridges.

status: map of string-string pairs
> Key-value pairs that report bridge status.

status : stp_bridge_id: optional string
> The bridge-id (in hex) used in spanning tree advertisements. Configuring the bridge-id is described in the **stp−system−id** and **stp−priority** keys of the **other_config** section earlier.

status : stp_designated_root: optional string
> The designated root (in hex) for this spanning tree.

status : stp_root_path_cost: optional string
> The path cost of reaching the designated bridge. A lower number is better.

Common Columns:
> The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

other_config: map of string-string pairs

external_ids: map of string-string pairs

Port TABLE

A port within a **Bridge**.

Most commonly, a port has exactly one ''interface,'' pointed to by its **interfaces** column. Such a port logi-
cally corresponds to a port on a physical Ethernet switch. A port with more than one interface is a ''bonded
port'' (see **Bonding Configuration**).

Some properties that one might think as belonging to a port are actually part of the port's **Interface** mem-
bers.

Summary:

name	immutable string (must be unique within table)
interfaces	set of 1 or more **Interface**s
VLAN Configuration:	
vlan_mode	optional string, one of **access**, **native–tagged**, **native–untagged**, or **trunk**
tag	optional integer, in range 0 to 4,095
trunks	set of up to 4,096 integers, in range 0 to 4,095
other_config : priority-tags	optional string, either **true** or **false**
Bonding Configuration:	
bond_mode	optional string, one of **active–backup**, **balance–tcp**, or **balance–slb**
other_config : bond-hash-basis	optional string, containing an integer
Link Failure Detection:	
other_config : bond-detect-mode	optional string, either **miimon** or **carrier**
other_config : bond-miimon-interval	optional string, containing an integer
bond_updelay	integer
bond_downdelay	integer
LACP Configuration:	
lacp	optional string, one of **active**, **passive**, or **off**
other_config : lacp-system-id	optional string
other_config : lacp-system-priority	optional string, containing an integer, in range 1 to 65,535
other_config : lacp-time	optional string, either **slow** or **fast**
other_config : lacp-fallback-ab	optional string, either **true** or **false**
Rebalancing Configuration:	
other_config : bond-rebalance-interval	optional string, containing an integer, in range 0 to 10,000
bond_fake_iface	boolean
Spanning Tree Configuration:	
other_config : stp-enable	optional string, either **true** or **false**
other_config : stp-port-num	optional string, containing an integer, in range 1 to 255
other_config : stp-port-priority	optional string, containing an integer, in range 0 to 255
other_config : stp-path-cost	optional string, containing an integer, in range 0 to 65,535
Rapid Spanning Tree Configuration:	
other_config : rstp-enable	optional string, either **true** or **false**
other_config : rstp-port-priority	optional string, containing an integer, in range 0 to 240
other_config : rstp-port-num	optional string, containing an integer, in range 1 to 4,095
other_config : rstp-port-path-cost	optional string, containing an integer
other_config : rstp-port-admin-edge	optional string, either **true** or **false**

other_config : rstp-port-auto-edge	optional string, either **true** or **false**
other_config : rstp-port-mcheck	optional string, either **true** or **false**

Multicast Snooping:

other_config : mcast-snooping-flood	optional string, either **true** or **false**

Other Features:

qos	optional **QoS**
mac	optional string
fake_bridge	boolean
external_ids : fake-bridge-id-*	optional string

Port Status:

status	map of string-string pairs
status : stp_port_id	optional string
status : stp_state	optional string, one of **disabled**, **forwarding**, **learning**, **listening**, or **blocking**
status : stp_sec_in_state	optional string, containing an integer, at least 0
status : stp_role	optional string, one of **designated**, **alternate**, or **root**
status : bond_active_slave	optional string

Port Statistics:

Statistics: STP transmit and receive counters:

statistics : stp_tx_count	optional integer
statistics : stp_rx_count	optional integer
statistics : stp_error_count	optional integer

Common Columns:

other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:

name: immutable string (must be unique within table)

> Port name. Should be alphanumeric and no more than about 8 bytes long. May be the same as the interface name, for non-bonded ports. Must otherwise be unique among the names of ports, interfaces, and bridges on a host.

interfaces: set of 1 or more **Interface**s

> The port's interfaces. If there is more than one, this is a bonded Port.

VLAN Configuration:

Bridge ports support the following types of VLAN configuration:

trunk A trunk port carries packets on one or more specified VLANs specified in the **trunks** column (often, on every VLAN). A packet that ingresses on a trunk port is in the VLAN specified in its 802.1Q header, or VLAN 0 if the packet has no 802.1Q header. A packet that egresses through a trunk port will have an 802.1Q header if it has a nonzero VLAN ID.

Any packet that ingresses on a trunk port tagged with a VLAN that the port does not trunk is dropped.

access An access port carries packets on exactly one VLAN specified in the **tag** column. Packets egressing on an access port have no 802.1Q header.

Any packet with an 802.1Q header with a nonzero VLAN ID that ingresses on an access port is dropped, regardless of whether the VLAN ID in the header is the access port's VLAN ID.

native-tagged

A native-tagged port resembles a trunk port, with the exception that a packet without an 802.1Q header that ingresses on a native-tagged port is in the "native VLAN" (specified in the **tag** column).

native-untagged

> A native-untagged port resembles a native-tagged port, with the exception that a packet that egresses on a native-untagged port in the native VLAN will not have an 802.1Q header.

A packet will only egress through bridge ports that carry the VLAN of the packet, as described by the rules above.

vlan_mode: optional string, one of **access**, **native−tagged**, **native−untagged**, or **trunk**

> The VLAN mode of the port, as described above. When this column is empty, a default mode is selected as follows:
>
> - If **tag** contains a value, the port is an access port. The **trunks** column should be empty.
>
> - Otherwise, the port is a trunk port. The **trunks** column value is honored if it is present.

tag: optional integer, in range 0 to 4,095

> For an access port, the port's implicitly tagged VLAN. For a native-tagged or native-untagged port, the port's native VLAN. Must be empty if this is a trunk port.

trunks: set of up to 4,096 integers, in range 0 to 4,095

> For a trunk, native-tagged, or native-untagged port, the 802.1Q VLAN or VLANs that this port trunks; if it is empty, then the port trunks all VLANs. Must be empty if this is an access port.
>
> A native-tagged or native-untagged port always trunks its native VLAN, regardless of whether **trunks** includes that VLAN.

other_config : priority-tags: optional string, either **true** or **false**

> An 802.1Q header contains two important pieces of information: a VLAN ID and a priority. A frame with a zero VLAN ID, called a "priority-tagged" frame, is supposed to be treated the same way as a frame without an 802.1Q header at all (except for the priority).
>
> However, some network elements ignore any frame that has 802.1Q header at all, even when the VLAN ID is zero. Therefore, by default Open vSwitch does not output priority-tagged frames, instead omitting the 802.1Q header entirely if the VLAN ID is zero. Set this key to **true** to enable priority-tagged frames on a port.
>
> Regardless of this setting, Open vSwitch omits the 802.1Q header on output if both the VLAN ID and priority would be zero.
>
> All frames output to native-tagged ports have a nonzero VLAN ID, so this setting is not meaningful on native-tagged ports.

Bonding Configuration:

A port that has more than one interface is a "bonded port." Bonding allows for load balancing and failover.

The following types of bonding will work with any kind of upstream switch. On the upstream switch, do not configure the interfaces as a bond:

balance−slb

> Balances flows among slaves based on source MAC address and output VLAN, with periodic rebalancing as traffic patterns change.

active−backup

> Assigns all flows to one slave, failing over to a backup slave when the active slave is disabled. This is the only bonding mode in which interfaces may be plugged into different upstream switches.

The following modes require the upstream switch to support 802.3ad with successful LACP negotiation. If LACP negotiation fails and other-config:lacp-fallback-ab is true, then **active−backup** mode is used:

balance−tcp

> Balances flows among slaves based on L2, L3, and L4 protocol information such as destination MAC address, IP address, and TCP port.

These columns apply only to bonded ports. Their values are otherwise ignored.

bond_mode: optional string, one of **active−backup**, **balance−tcp**, or **balance−slb**
> The type of bonding used for a bonded port. Defaults to **active−backup** if unset.

other_config : bond-hash-basis: optional string, containing an integer
> An integer hashed along with flows when choosing output slaves in load balanced bonds. When changed, all flows will be assigned different hash values possibly causing slave selection decisions to change. Does not affect bonding modes which do not employ load balancing such as **active−backup**.

Link Failure Detection:
> An important part of link bonding is detecting that links are down so that they may be disabled. These settings determine how Open vSwitch detects link failure.

other_config : bond-detect-mode: optional string, either **miimon** or **carrier**
> The means used to detect link failures. Defaults to **carrier** which uses each interface's carrier to detect failures. When set to **miimon**, will check for failures by polling each interface's MII.

other_config : bond-miimon-interval: optional string, containing an integer
> The interval, in milliseconds, between successive attempts to poll each interface's MII. Relevant only when **other_config:bond-detect-mode** is **miimon**.

bond_updelay: integer
> The number of milliseconds for which the link must stay up on an interface before the interface is considered to be up. Specify **0** to enable the interface immediately.
>
> This setting is honored only when at least one bonded interface is already enabled. When no interfaces are enabled, then the first bond interface to come up is enabled immediately.

bond_downdelay: integer
> The number of milliseconds for which the link must stay down on an interface before the interface is considered to be down. Specify **0** to disable the interface immediately.

LACP Configuration:
> LACP, the Link Aggregation Control Protocol, is an IEEE standard that allows switches to automatically detect that they are connected by multiple links and aggregate across those links. These settings control LACP behavior.

lacp: optional string, one of **active**, **passive**, or **off**
> Configures LACP on this port. LACP allows directly connected switches to negotiate which links may be bonded. LACP may be enabled on non-bonded ports for the benefit of any switches they may be connected to. **active** ports are allowed to initiate LACP negotiations. **passive** ports are allowed to participate in LACP negotiations initiated by a remote switch, but not allowed to initiate such negotiations themselves. If LACP is enabled on a port whose partner switch does not support LACP, the bond will be disabled, unless other-config:lacp-fallback-ab is set to true. Defaults to **off** if unset.

other_config : lacp-system-id: optional string
> The LACP system ID of this **Port**. The system ID of a LACP bond is used to identify itself to its partners. Must be a nonzero MAC address. Defaults to the bridge Ethernet address if unset.

other_config : lacp-system-priority: optional string, containing an integer, in range 1 to 65,535
> The LACP system priority of this **Port**. In LACP negotiations, link status decisions are made by the system with the numerically lower priority.

other_config : lacp-time: optional string, either **slow** or **fast**
> The LACP timing which should be used on this **Port**. By default **slow** is used. When configured to be **fast** LACP heartbeats are requested at a rate of once per second causing connectivity problems to be detected more quickly. In **slow** mode, heartbeats are requested at a rate of once every 30 seconds.

other_config : lacp-fallback-ab: optional string, either **true** or **false**
> Determines the behavior of openvswitch bond in LACP mode. If the partner switch does not support LACP, setting this option to **true** allows openvswitch to fallback to active-backup. If the option is set to **false**, the bond will be disabled. In both the cases, once the partner switch is configured to LACP mode, the bond will use LACP.

Rebalancing Configuration:
> These settings control behavior when a bond is in **balance−slb** or **balance−tcp** mode.

other_config : bond-rebalance-interval: optional string, containing an integer, in range 0 to 10,000
> For a load balanced bonded port, the number of milliseconds between successive attempts to rebalance the bond, that is, to move flows from one interface on the bond to another in an attempt to keep usage of each interface roughly equal. If zero, load balancing is disabled on the bond (link failure still cause flows to move). If less than 1000ms, the rebalance interval will be 1000ms.

bond_fake_iface: boolean
> For a bonded port, whether to create a fake internal interface with the name of the port. Use only for compatibility with legacy software that requires this.

Spanning Tree Configuration:

other_config : stp-enable: optional string, either **true** or **false**
> If spanning tree is enabled on the bridge, member ports are enabled by default (with the exception of bond, internal, and mirror ports which do not work with STP). If this column's value is **false** spanning tree is disabled on the port.

other_config : stp-port-num: optional string, containing an integer, in range 1 to 255
> The port number used for the lower 8 bits of the port-id. By default, the numbers will be assigned automatically. If any port's number is manually configured on a bridge, then they must all be.

other_config : stp-port-priority: optional string, containing an integer, in range 0 to 255
> The port's relative priority value for determining the root port (the upper 8 bits of the port-id). A port with a lower port-id will be chosen as the root port. By default, the priority is 0x80.

other_config : stp-path-cost: optional string, containing an integer, in range 0 to 65,535
> Spanning tree path cost for the port. A lower number indicates a faster link. By default, the cost is based on the maximum speed of the link.

Rapid Spanning Tree Configuration:

other_config : rstp-enable: optional string, either **true** or **false**
> If rapid spanning tree is enabled on the bridge, member ports are enabled by default (with the exception of bond, internal, and mirror ports which do not work with RSTP). If this column's value is **false** rapid spanning tree is disabled on the port.

other_config : rstp-port-priority: optional string, containing an integer, in range 0 to 240
> The port's relative priority value for determining the root port, in multiples of 16. By default, the port priority is 0x80 (128). Any value in the lower 4 bits is rounded off. The significant upper 4 bits become the upper 4 bits of the port-id. A port with the lowest port-id is elected as the root.

other_config : rstp-port-num: optional string, containing an integer, in range 1 to 4,095
> The local RSTP port number, used as the lower 12 bits of the port-id. By default the port numbers are assigned automatically, and typically may not correspond to the OpenFlow port numbers. A port with the lowest port-id is elected as the root.

other_config : rstp-port-path-cost: optional string, containing an integer
> The port path cost. The Port's contribution, when it is the Root Port, to the Root Path Cost for the Bridge. By default the cost is automatically calculated from the port's speed.

other_config : rstp-port-admin-edge: optional string, either **true** or **false**
> The admin edge port parameter for the Port. Default is **false**.

other_config : rstp-port-auto-edge: optional string, either **true** or **false**
> The auto edge port parameter for the Port. Default is **true**.

other_config : rstp-port-mcheck: optional string, either **true** or **false**
> The mcheck port parameter for the Port. Default is **false**. May be set to force the Port Protocol Migration state machine to transmit RST BPDUs for a MigrateTime period, to test whether all STP Bridges on the attached LAN have been removed and the Port can continue to transmit RSTP BPDUs. Setting mcheck has no effect if the Bridge is operating in STP Compatibility mode.

> Changing the value from **true** to **false** has no effect, but needs to be done if this behavior is to be triggered again by subsequently changing the value from **false** to **true**.

Multicast Snooping:

other_config : mcast-snooping-flood: optional string, either **true** or **false**
> If set to **true**, multicast packets are unconditionally forwarded to the specific port.

Other Features:

qos: optional **QoS**
> Quality of Service configuration for this port.

mac: optional string
> The MAC address to use for this port for the purpose of choosing the bridge's MAC address. This column does not necessarily reflect the port's actual MAC address, nor will setting it change the port's actual MAC address.

fake_bridge: boolean
> Does this port represent a sub-bridge for its tagged VLAN within the Bridge? See ovs-vsctl(8) for more information.

external_ids : fake-bridge-id-*: optional string
> External IDs for a fake bridge (see the **fake_bridge** column) are defined by prefixing a **Bridge external_ids** key with **fake−bridge−**, e.g. **fake−bridge−xs−network−uuids**.

Port Status:
> Status information about ports attached to bridges.

status: map of string-string pairs
> Key-value pairs that report port status.

status : stp_port_id: optional string
> The port-id (in hex) used in spanning tree advertisements for this port. Configuring the port-id is described in the **stp−port−num** and **stp−port−priority** keys of the **other_config** section earlier.

status : stp_state: optional string, one of **disabled**, **forwarding**, **learning**, **listening**, or **blocking**
> STP state of the port.

status : stp_sec_in_state: optional string, containing an integer, at least 0
> The amount of time (in seconds) port has been in the current STP state.

status : stp_role: optional string, one of **designated**, **alternate**, or **root**
> STP role of the port.

status : bond_active_slave: optional string
> For a bonded port, record the mac address of the current active slave.

Port Statistics:
> Key-value pairs that report port statistics. The update period is controlled by **other_config:stats-update-interval** in the **Open_vSwitch** table.

Statistics: STP transmit and receive counters:

statistics : stp_tx_count: optional integer
> Number of STP BPDUs sent on this port by the spanning tree library.

statistics : stp_rx_count: optional integer
> Number of STP BPDUs received on this port and accepted by the spanning tree library.

statistics : stp_error_count: optional integer
> Number of bad STP BPDUs received on this port. Bad BPDUs include runt packets and those with an unexpected protocol ID.

Common Columns:
> The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

other_config: map of string-string pairs

external_ids: map of string-string pairs

Interface TABLE

An interface within a **Port**.

Summary:

Core Features:

name	immutable string (must be unique within table)
ifindex	optional integer, in range 0 to 4,294,967,295
mac_in_use	optional string
mac	optional string
error	optional string

OpenFlow Port Number:

ofport	optional integer
ofport_request	optional integer, in range 1 to 65,279

System-Specific Details:

type	string

Tunnel Options:

options : remote_ip	optional string
options : local_ip	optional string
options : in_key	optional string
options : out_key	optional string
options : key	optional string
options : tos	optional string
options : ttl	optional string
options : df_default	optional string, either **true** or **false**

Tunnel Options: gre and ipsec_gre only:

options : csum	optional string, either **true** or **false**

Tunnel Options: ipsec_gre only:

options : peer_cert	optional string
options : certificate	optional string
options : private_key	optional string
options : psk	optional string

Patch Options:

options : peer	optional string

Interface Status:

admin_state	optional string, either **down** or **up**
link_state	optional string, either **down** or **up**
link_resets	optional integer
link_speed	optional integer
duplex	optional string, either **full** or **half**
mtu	optional integer
lacp_current	optional boolean
status	map of string-string pairs
status : driver_name	optional string
status : driver_version	optional string
status : firmware_version	optional string
status : source_ip	optional string
status : tunnel_egress_iface	optional string
status : tunnel_egress_iface_carrier	optional string, either **down** or **up**

Statistics:

Statistics: Successful transmit and receive counters:

statistics : rx_packets	optional integer
statistics : rx_bytes	optional integer
statistics : tx_packets	optional integer
statistics : tx_bytes	optional integer

Statistics: Receive errors:

statistics : rx_dropped	optional integer
statistics : rx_frame_err	optional integer
statistics : rx_over_err	optional integer
statistics : rx_crc_err	optional integer
statistics : rx_errors	optional integer

Statistics: Transmit errors:

statistics : tx_dropped	optional integer
statistics : collisions	optional integer
statistics : tx_errors	optional integer

Ingress Policing:

ingress_policing_rate	integer, at least 0
ingress_policing_burst	integer, at least 0

Bidirectional Forwarding Detection (BFD):

    BFD Configuration:

bfd : enable	optional string, either **true** or **false**
bfd : min_rx	optional string, containing an integer, at least 1
bfd : min_tx	optional string, containing an integer, at least 1
bfd : decay_min_rx	optional string, containing an integer
bfd : forwarding_if_rx	optional string, either **true** or **false**
bfd : cpath_down	optional string, either **true** or **false**
bfd : check_tnl_key	optional string, either **true** or **false**
bfd : bfd_local_src_mac	optional string
bfd : bfd_local_dst_mac	optional string
bfd : bfd_remote_dst_mac	optional string
bfd : bfd_src_ip	optional string
bfd : bfd_dst_ip	optional string

    BFD Status:

bfd_status : state	optional string, one of **down**, **init**, **up**, or **admin_down**
bfd_status : forwarding	optional string, either **true** or **false**
bfd_status : diagnostic	optional string
bfd_status : remote_state	optional string, one of **down**, **init**, **up**, or **admin_down**
bfd_status : remote_diagnostic	optional string
bfd_status : flap_count	optional string, containing an integer, at least 0

Connectivity Fault Management:

cfm_mpid	optional integer
cfm_flap_count	optional integer
cfm_fault	optional boolean
cfm_fault_status : recv	none
cfm_fault_status : rdi	none
cfm_fault_status : maid	none
cfm_fault_status : loopback	none
cfm_fault_status : overflow	none
cfm_fault_status : override	none
cfm_fault_status : interval	none
cfm_remote_opstate	optional string, either **down** or **up**
cfm_health	optional integer, in range 0 to 100
cfm_remote_mpids	set of integers
other_config : cfm_interval	optional string, containing an integer
other_config : cfm_extended	optional string, either **true** or **false**
other_config : cfm_demand	optional string, either **true** or **false**

other_config : cfm_opstate	optional string, either **down** or **up**
other_config : cfm_ccm_vlan	optional string, containing an integer, in range 1 to 4,095
other_config : cfm_ccm_pcp	optional string, containing an integer, in range 1 to 7

Bonding Configuration:

other_config : lacp-port-id	optional string, containing an integer, in range 1 to 65,535
other_config : lacp-port-priority	optional string, containing an integer, in range 1 to 65,535
other_config : lacp-aggregation-key	optional string, containing an integer, in range 1 to 65,535

Virtual Machine Identifiers:

external_ids : attached-mac	optional string
external_ids : iface-id	optional string
external_ids : iface-status	optional string, either **active** or **inactive**
external_ids : xs-vif-uuid	optional string
external_ids : xs-network-uuid	optional string
external_ids : vm-id	optional string
external_ids : xs-vm-uuid	optional string

VLAN Splinters:

other_config : enable-vlan-splinters	optional string, either **true** or **false**

Common Columns:

other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:

Core Features:

name: immutable string (must be unique within table)
> Interface name. Should be alphanumeric and no more than about 8 bytes long. May be the same as the port name, for non-bonded ports. Must otherwise be unique among the names of ports, interfaces, and bridges on a host.

ifindex: optional integer, in range 0 to 4,294,967,295
> A positive interface index as defined for SNMP MIB-II in RFCs 1213 and 2863, if the interface has one, otherwise 0. The ifindex is useful for seamless integration with protocols such as SNMP and sFlow.

mac_in_use: optional string
> The MAC address in use by this interface.

mac: optional string
> Ethernet address to set for this interface. If unset then the default MAC address is used:

> • For the local interface, the default is the lowest-numbered MAC address among the other bridge ports, either the value of the **mac** in its **Port** record, if set, or its actual MAC (for bonded ports, the MAC of its slave whose name is first in alphabetical order). Internal ports and bridge ports that are used as port mirroring destinations (see the **Mirror** table) are ignored.

> • For other internal interfaces, the default MAC is randomly generated.

> • External interfaces typically have a MAC address associated with their hardware.

> Some interfaces may not have a software-controllable MAC address.

error: optional string
> If the configuration of the port failed, as indicated by −1 in **ofport**, Open vSwitch sets this column to an error description in human readable form. Otherwise, Open vSwitch clears this column.

OpenFlow Port Number:

When a client adds a new interface, Open vSwitch chooses an OpenFlow port number for the new port. If the client that adds the port fills in **ofport_request**, then Open vSwitch tries to use its value as the Open-Flow port number. Otherwise, or if the requested port number is already in use or cannot be used for another reason, Open vSwitch automatically assigns a free port number. Regardless of how the port number was obtained, Open vSwitch then reports in **ofport** the port number actually assigned.

Open vSwitch limits the port numbers that it automatically assigns to the range 1 through 32,767, inclusive. Controllers therefore have free use of ports 32,768 and up.

ofport: optional integer

> OpenFlow port number for this interface. Open vSwitch sets this column's value, so other clients should treat it as read-only.
>
> The OpenFlow "local" port (**OFPP_LOCAL**) is 65,534. The other valid port numbers are in the range 1 to 65,279, inclusive. Value −1 indicates an error adding the interface.

ofport_request: optional integer, in range 1 to 65,279

> Requested OpenFlow port number for this interface.
>
> A client should ideally set this column's value in the same database transaction that it uses to create the interface. Open vSwitch version 2.1 and later will honor a later request for a specific port number, althuogh it might confuse some controllers: OpenFlow does not have a way to announce a port number change, so Open vSwitch represents it over OpenFlow as a port deletion followed immediately by a port addition.
>
> If **ofport_request** is set or changed to some other port's automatically assigned port number, Open vSwitch chooses a new port number for the latter port.

System-Specific Details:

type: string

> The interface type, one of:
>
> **system**   An ordinary network device, e.g. **eth0** on Linux. Sometimes referred to as "external interfaces" since they are generally connected to hardware external to that on which the Open vSwitch is running. The empty string is a synonym for **system**.
>
> **internal**
>
> > A simulated network device that sends and receives traffic. An internal interface whose **name** is the same as its bridge's **name** is called the "local interface." It does not make sense to bond an internal interface, so the terms "port" and "interface" are often used imprecisely for internal interfaces.
>
> **tap**   A TUN/TAP device managed by Open vSwitch.
>
> **geneve**   An Ethernet over Geneve (**http://tools.ietf.org/html/draft−gross−geneve−00**) IPv4 tunnel. Geneve supports options as a means to transport additional metadata, however, currently only the 24-bit VNI is supported. This is planned to be extended in the future.
>
> **gre**   An Ethernet over RFC 2890 Generic Routing Encapsulation over IPv4 tunnel.
>
> **ipsec_gre**
>
> > An Ethernet over RFC 2890 Generic Routing Encapsulation over IPv4 IPsec tunnel.
>
> **gre64**   It is same as GRE, but it allows 64 bit key. To store higher 32-bits of key, it uses GRE protocol sequence number field. This is non standard use of GRE protocol since OVS does not increment sequence number for every packet at time of encap as expected by standard GRE implementation. See **Tunnel Options** for information on configuring GRE tunnels.
>
> **ipsec_gre64**
>
> > Same as IPSEC_GRE except 64 bit key.
>
> **vxlan**   An Ethernet tunnel over the experimental, UDP-based VXLAN protocol described at **http://tools.ietf.org/html/draft−mahalingam−dutt−dcops−vxlan−03**.

Open vSwitch uses UDP destination port 4789. The source port used for VXLAN traffic varies on a per-flow basis and is in the ephemeral port range.

lisp　　A layer 3 tunnel over the experimental, UDP-based Locator/ID Separation Protocol (RFC 6830).

Only IPv4 and IPv6 packets are supported by the protocol, and they are sent and received without an Ethernet header. Traffic to/from LISP ports is expected to be configured explicitly, and the ports are not intended to participate in learning based switching. As such, they are always excluded from packet flooding.

patch　　A pair of virtual devices that act as a patch cable.

null　　An ignored interface. Deprecated and slated for removal in February 2013.

Tunnel Options:

These options apply to interfaces with **type** of **geneve**, **gre**, **ipsec_gre**, **gre64**, **ipsec_gre64**, **vxlan**, and **lisp**.

Each tunnel must be uniquely identified by the combination of **type**, **options:remote_ip**, **options:local_ip**, and **options:in_key**. If two ports are defined that are the same except one has an optional identifier and the other does not, the more specific one is matched first. **options:in_key** is considered more specific than **options:local_ip** if a port defines one and another port defines the other.

options : remote_ip: optional string

Required. The remote tunnel endpoint, one of:

- An IPv4 address (not a DNS name), e.g. **192.168.0.123**. Only unicast endpoints are supported.

- The word **flow**. The tunnel accepts packets from any remote tunnel endpoint. To process only packets from a specific remote tunnel endpoint, the flow entries may match on the **tun_src** field. When sending packets to a **remote_ip=flow** tunnel, the flow actions must explicitly set the **tun_dst** field to the IP address of the desired remote tunnel endpoint, e.g. with a **set_field** action.

The remote tunnel endpoint for any packet received from a tunnel is available in the **tun_src** field for matching in the flow table.

options : local_ip: optional string

Optional. The tunnel destination IP that received packets must match. Default is to match all addresses. If specified, may be one of:

- An IPv4 address (not a DNS name), e.g. **192.168.12.3**.

- The word **flow**. The tunnel accepts packets sent to any of the local IP addresses of the system running OVS. To process only packets sent to a specific IP address, the flow entries may match on the **tun_dst** field. When sending packets to a **local_ip=flow** tunnel, the flow actions may explicitly set the **tun_src** field to the desired IP address, e.g. with a **set_field** action. However, while routing the tunneled packet out, the local system may override the specified address with the local IP address configured for the outgoing system interface.

  This option is valid only for tunnels also configured with the **remote_ip=flow** option.

The tunnel destination IP address for any packet received from a tunnel is available in the **tun_dst** field for matching in the flow table.

options : in_key: optional string

Optional. The key that received packets must contain, one of:

- **0**. The tunnel receives packets with no key or with a key of 0. This is equivalent to specifying no **options:in_key** at all.

- A positive 24-bit (for Geneve, VXLAN, and LISP), 32-bit (for GRE) or 64-bit (for GRE64) number. The tunnel receives only packets with the specified key.

    •   The word **flow**. The tunnel accepts packets with any key. The key will be placed in the **tun_id** field for matching in the flow table. The **ovs−ofctl** manual page contains additional information about matching fields in OpenFlow flows.

options : out_key: optional string

    Optional. The key to be set on outgoing packets, one of:

    •   **0**. Packets sent through the tunnel will have no key. This is equivalent to specifying no **options:out_key** at all.

    •   A positive 24-bit (for Geneve, VXLAN and LISP), 32-bit (for GRE) or 64-bit (for GRE64) number. Packets sent through the tunnel will have the specified key.

    •   The word **flow**. Packets sent through the tunnel will have the key set using the **set_tunnel** Nicira OpenFlow vendor extension (0 is used in the absence of an action). The **ovs−ofctl** manual page contains additional information about the Nicira OpenFlow vendor extensions.

options : key: optional string

    Optional. Shorthand to set **in_key** and **out_key** at the same time.

options : tos: optional string

    Optional. The value of the ToS bits to be set on the encapsulating packet. ToS is interpreted as DSCP and ECN bits, ECN part must be zero. It may also be the word **inherit**, in which case the ToS will be copied from the inner packet if it is IPv4 or IPv6 (otherwise it will be 0). The ECN fields are always inherited. Default is 0.

options : ttl: optional string

    Optional. The TTL to be set on the encapsulating packet. It may also be the word **inherit**, in which case the TTL will be copied from the inner packet if it is IPv4 or IPv6 (otherwise it will be the system default, typically 64). Default is the system default TTL.

options : df_default: optional string, either **true** or **false**

    Optional. If enabled, the Don't Fragment bit will be set on tunnel outer headers to allow path MTU discovery. Default is enabled; set to **false** to disable.

Tunnel Options: gre and ipsec_gre only:

    Only **gre** and **ipsec_gre** interfaces support these options.

options : csum: optional string, either **true** or **false**

    Optional. Compute GRE checksums on outgoing packets. Default is disabled, set to **true** to enable. Checksums present on incoming packets will be validated regardless of this setting.

    GRE checksums impose a significant performance penalty because they cover the entire packet. The encapsulated L3, L4, and L7 packet contents typically have their own checksums, so this additional checksum only adds value for the GRE and encapsulated L2 headers.

    This option is supported for **ipsec_gre**, but not useful because GRE checksums are weaker than, and redundant with, IPsec payload authentication.

Tunnel Options: ipsec_gre only:

    Only **ipsec_gre** interfaces support these options.

options : peer_cert: optional string

    Required for certificate authentication. A string containing the peer's certificate in PEM format. Additionally the host's certificate must be specified with the **certificate** option.

options : certificate: optional string

    Required for certificate authentication. The name of a PEM file containing a certificate that will be presented to the peer during authentication.

options : private_key: optional string

    Optional for certificate authentication. The name of a PEM file containing the private key associated with **certificate**. If **certificate** contains the private key, this option may be omitted.

options : psk: optional string
>   Required for pre-shared key authentication. Specifies a pre-shared key for authentication that must be identical on both sides of the tunnel.

Patch Options:
>   Only **patch** interfaces support these options.

options : peer: optional string
>   The **name** of the **Interface** for the other side of the patch. The named **Interface**'s own **peer** option must specify this **Interface**'s name. That is, the two patch interfaces must have reversed **name** and **peer** values.

Interface Status:
>   Status information about interfaces attached to bridges, updated every 5 seconds. Not all interfaces have all of these properties; virtual interfaces don't have a link speed, for example. Non-applicable columns will have empty values.

admin_state: optional string, either **down** or **up**
>   The administrative state of the physical network link.

link_state: optional string, either **down** or **up**
>   The observed state of the physical network link. This is ordinarily the link's carrier status. If the interface's **Port** is a bond configured for miimon monitoring, it is instead the network link's miimon status.

link_resets: optional integer
>   The number of times Open vSwitch has observed the **link_state** of this **Interface** change.

link_speed: optional integer
>   The negotiated speed of the physical network link. Valid values are positive integers greater than 0.

duplex: optional string, either **full** or **half**
>   The duplex mode of the physical network link.

mtu: optional integer
>   The MTU (maximum transmission unit); i.e. the largest amount of data that can fit into a single Ethernet frame. The standard Ethernet MTU is 1500 bytes. Some physical media and many kinds of virtual interfaces can be configured with higher MTUs.
>
>   This column will be empty for an interface that does not have an MTU as, for example, some kinds of tunnels do not.

lacp_current: optional boolean
>   Boolean value indicating LACP status for this interface. If true, this interface has current LACP information about its LACP partner. This information may be used to monitor the health of interfaces in a LACP enabled port. This column will be empty if LACP is not enabled.

status: map of string-string pairs
>   Key-value pairs that report port status. Supported status values are **type**-dependent; some interfaces may not have a valid **status:driver_name**, for example.

status : driver_name: optional string
>   The name of the device driver controlling the network adapter.

status : driver_version: optional string
>   The version string of the device driver controlling the network adapter.

status : firmware_version: optional string
>   The version string of the network adapter's firmware, if available.

status : source_ip: optional string
>   The source IP address used for an IPv4 tunnel end-point, such as **gre**.

status : tunnel_egress_iface: optional string
> Egress interface for tunnels. Currently only relevant for tunnels on Linux systems, this column will show the name of the interface which is responsible for routing traffic destined for the configured **options:remote_ip**. This could be an internal interface such as a bridge port.

status : tunnel_egress_iface_carrier: optional string, either **down** or **up**
> Whether carrier is detected on **status:tunnel_egress_iface**.

Statistics:
> Key-value pairs that report interface statistics. The current implementation updates these counters periodically. The update period is controlled by **other_config:stats-update-interval** in the **Open_vSwitch** table. Future implementations may update them when an interface is created, when they are queried (e.g. using an OVSDB **select** operation), and just before an interface is deleted due to virtual interface hot-unplug or VM shutdown, and perhaps at other times, but not on any regular periodic basis.

> These are the same statistics reported by OpenFlow in its **struct ofp_port_stats** structure. If an interface does not support a given statistic, then that pair is omitted.

Statistics: Successful transmit and receive counters:

statistics : rx_packets: optional integer
> Number of received packets.

statistics : rx_bytes: optional integer
> Number of received bytes.

statistics : tx_packets: optional integer
> Number of transmitted packets.

statistics : tx_bytes: optional integer
> Number of transmitted bytes.

Statistics: Receive errors:

statistics : rx_dropped: optional integer
> Number of packets dropped by RX.

statistics : rx_frame_err: optional integer
> Number of frame alignment errors.

statistics : rx_over_err: optional integer
> Number of packets with RX overrun.

statistics : rx_crc_err: optional integer
> Number of CRC errors.

statistics : rx_errors: optional integer
> Total number of receive errors, greater than or equal to the sum of the above.

Statistics: Transmit errors:

statistics : tx_dropped: optional integer
> Number of packets dropped by TX.

statistics : collisions: optional integer
> Number of collisions.

statistics : tx_errors: optional integer
> Total number of transmit errors, greater than or equal to the sum of the above.

Ingress Policing:
> These settings control ingress policing for packets received on this interface. On a physical interface, this limits the rate at which traffic is allowed into the system from the outside; on a virtual interface (one connected to a virtual machine), this limits the rate at which the VM is able to transmit.

> Policing is a simple form of quality-of-service that simply drops packets received in excess of the

configured rate. Due to its simplicity, policing is usually less accurate and less effective than egress QoS (which is configured using the **QoS** and **Queue** tables).

Policing is currently implemented only on Linux. The Linux implementation uses a simple ''token bucket'' approach:

- The size of the bucket corresponds to **ingress_policing_burst**. Initially the bucket is full.

- Whenever a packet is received, its size (converted to tokens) is compared to the number of tokens currently in the bucket. If the required number of tokens are available, they are removed and the packet is forwarded. Otherwise, the packet is dropped.

- Whenever it is not full, the bucket is refilled with tokens at the rate specified by **ingress_policing_rate**.

Policing interacts badly with some network protocols, and especially with fragmented IP packets. Suppose that there is enough network activity to keep the bucket nearly empty all the time. Then this token bucket algorithm will forward a single packet every so often, with the period depending on packet size and on the configured rate. All of the fragments of an IP packets are normally transmitted back-to-back, as a group. In such a situation, therefore, only one of these fragments will be forwarded and the rest will be dropped. IP does not provide any way for the intended recipient to ask for only the remaining fragments. In such a case there are two likely possibilities for what will happen next: either all of the fragments will eventually be retransmitted (as TCP will do), in which case the same problem will recur, or the sender will not realize that its packet has been dropped and data will simply be lost (as some UDP-based protocols will do). Either way, it is possible that no forward progress will ever occur.

ingress_policing_rate: integer, at least 0
> Maximum rate for data received on this interface, in kbps. Data received faster than this rate is dropped. Set to **0** (the default) to disable policing.

ingress_policing_burst: integer, at least 0
> Maximum burst size for data received on this interface, in kb. The default burst size if set to **0** is 1000 kb. This value has no effect if **ingress_policing_rate** is **0**.

> Specifying a larger burst size lets the algorithm be more forgiving, which is important for protocols like TCP that react severely to dropped packets. The burst size should be at least the size of the interface's MTU. Specifying a value that is numerically at least as large as 10% of **ingress_policing_rate** helps TCP come closer to achieving the full rate.

Bidirectional Forwarding Detection (BFD):
BFD, defined in RFC 5880 and RFC 5881, allows point-to-point detection of connectivity failures by occasional transmission of BFD control messages. Open vSwitch implements BFD to serve as a more popular and standards compliant alternative to CFM.

BFD operates by regularly transmitting BFD control messages at a rate negotiated independently in each direction. Each endpoint specifies the rate at which it expects to receive control messages, and the rate at which it is willing to transmit them. Open vSwitch uses a detection multiplier of three, meaning that an endpoint signals a connectivity fault if three consecutive BFD control messages fail to arrive. In the case of a unidirectional connectivity issue, the system not receiving BFD control messages signals the problem to its peer in the messages it transmits.

The Open vSwitch implementation of BFD aims to comply faithfully with RFC 5880 requirements. Open vSwitch does not implement the optional Authentication or ''Echo Mode'' features.

BFD Configuration:
A controller sets up key-value pairs in the **bfd** column to enable and configure BFD.

bfd : enable: optional string, either **true** or **false**
> True to enable BFD on this **Interface**. If not specified, BFD will not be enabled by default.

bfd : min_rx: optional string, containing an integer, at least 1
> The shortest interval, in milliseconds, at which this BFD session offers to receive BFD control messages. The remote endpoint may choose to send messages at a slower rate. Defaults to **1000**.

bfd : min_tx: optional string, containing an integer, at least 1
> The shortest interval, in milliseconds, at which this BFD session is willing to transmit BFD control messages. Messages will actually be transmitted at a slower rate if the remote endpoint is not willing to receive as quickly as specified. Defaults to **100**.

bfd : decay_min_rx: optional string, containing an integer
> An alternate receive interval, in milliseconds, that must be greater than or equal to **bfd:min_rx**. The implementation switches from **bfd:min_rx** to **bfd:decay_min_rx** when there is no obvious incoming data traffic at the interface, to reduce the CPU and bandwidth cost of monitoring an idle interface. This feature may be disabled by setting a value of 0. This feature is reset whenever **bfd:decay_min_rx** or **bfd:min_rx** changes.

bfd : forwarding_if_rx: optional string, either **true** or **false**
> When **true**, traffic received on the **Interface** is used to indicate the capability of packet I/O. BFD control packets are still transmitted and received. At least one BFD control packet must be received every 100 * **bfd:min_rx** amount of time. Otherwise, even if traffic are received, the **bfd:forwarding** will be **false**.

bfd : cpath_down: optional string, either **true** or **false**
> Set to true to notify the remote endpoint that traffic should not be forwarded to this system for some reason other than a connectivty failure on the interface being monitored. The typical underlying reason is "concatenated path down," that is, that connectivity beyond the local system is down. Defaults to false.

bfd : check_tnl_key: optional string, either **true** or **false**
> Set to true to make BFD accept only control messages with a tunnel key of zero. By default, BFD accepts control messages with any tunnel key.

bfd : bfd_local_src_mac: optional string
> Set to an Ethernet address in the form *xx:xx:xx:xx:xx:xx* to set the MAC used as source for transmitted BFD packets. The default is the mac address of the BFD enabled interface.

bfd : bfd_local_dst_mac: optional string
> Set to an Ethernet address in the form *xx:xx:xx:xx:xx:xx* to set the MAC used as destination for transmitted BFD packets. The default is **00:23:20:00:00:01**.

bfd : bfd_remote_dst_mac: optional string
> Set to an Ethernet address in the form *xx:xx:xx:xx:xx:xx* to set the MAC used for checking the destination of received BFD packets. Packets with different destination MAC will not be considered as BFD packets. If not specified the destination MAC address of received BFD packets are not checked.

bfd : bfd_src_ip: optional string
> Set to an IPv4 address to set the IP address used as source for transmitted BFD packets. The default is **169.254.1.1**.

bfd : bfd_dst_ip: optional string
> Set to an IPv4 address to set the IP address used as destination for transmitted BFD packets. The default is **169.254.1.0**.

BFD Status:
The switch sets key-value pairs in the **bfd_status** column to report the status of BFD on this interface. When BFD is not enabled, with **bfd:enable**, the switch clears all key-value pairs from **bfd_status**.

bfd_status : state: optional string, one of **down**, **init**, **up**, or **admin_down**
> Reports the state of the BFD session. The BFD session is fully healthy and negotiated if **UP**.

bfd_status : forwarding: optional string, either **true** or **false**
> Reports whether the BFD session believes this **Interface** may be used to forward traffic. Typically this means the local session is signaling **UP**, and the remote system isn't signaling a problem such as concatenated path down.

bfd_status : diagnostic: optional string

> In case of a problem, set to an error message that reports what the local BFD session thinks is wrong. The error messages are defined in section 4.1 of [RFC 5880].

bfd_status : remote_state: optional string, one of **down**, **init**, **up**, or **admin_down**

> Reports the state of the remote endpoint's BFD session.

bfd_status : remote_diagnostic: optional string

> In case of a problem, set to an error message that reports what the remote endpoint's BFD session thinks is wrong. The error messages are defined in section 4.1 of [RFC 5880].

bfd_status : flap_count: optional string, containing an integer, at least 0

> Counts the number of **bfd_status:forwarding** flaps since start. A flap is considered as a change of the **bfd_status:forwarding** value.

Connectivity Fault Management:

802.1ag Connectivity Fault Management (CFM) allows a group of Maintenance Points (MPs) called a Maintenance Association (MA) to detect connectivity problems with each other. MPs within a MA should have complete and exclusive interconnectivity. This is verified by occasionally broadcasting Continuity Check Messages (CCMs) at a configurable transmission interval.

According to the 802.1ag specification, each Maintenance Point should be configured out-of-band with a list of Remote Maintenance Points it should have connectivity to. Open vSwitch differs from the specification in this area. It simply assumes the link is faulted if no Remote Maintenance Points are reachable, and considers it not faulted otherwise.

When operating over tunnels which have no **in_key**, or an **in_key** of **flow**. CFM will only accept CCMs with a tunnel key of zero.

cfm_mpid: optional integer

> A Maintenance Point ID (MPID) uniquely identifies each endpoint within a Maintenance Association. The MPID is used to identify this endpoint to other Maintenance Points in the MA. Each end of a link being monitored should have a different MPID. Must be configured to enable CFM on this **Interface**.
>
> According to the 802.1ag specification, MPIDs can only range between [1, 8191]. However, extended mode (see **other_config:cfm_extended**) supports eight byte MPIDs.

cfm_flap_count: optional integer

> Counts the number of cfm fault flapps since boot. A flap is considered to be a change of the **cfm_fault** value.

cfm_fault: optional boolean

> Indicates a connectivity fault triggered by an inability to receive heartbeats from any remote endpoint. When a fault is triggered on **Interface**s participating in bonds, they will be disabled.
>
> Faults can be triggered for several reasons. Most importantly they are triggered when no CCMs are received for a period of 3.5 times the transmission interval. Faults are also triggered when any CCMs indicate that a Remote Maintenance Point is not receiving CCMs but able to send them. Finally, a fault is triggered if a CCM is received which indicates unexpected configuration. Notably, this case arises when a CCM is received which advertises the local MPID.

cfm_fault_status : recv: none

> Indicates a CFM fault was triggered due to a lack of CCMs received on the **Interface**.

cfm_fault_status : rdi: none

> Indicates a CFM fault was triggered due to the reception of a CCM with the RDI bit flagged. Endpoints set the RDI bit in their CCMs when they are not receiving CCMs themselves. This typically indicates a unidirectional connectivity failure.

cfm_fault_status : maid: none

> Indicates a CFM fault was triggered due to the reception of a CCM with a MAID other than the one Open vSwitch uses. CFM broadcasts are tagged with an identification number in addition to

the MPID called the MAID. Open vSwitch only supports receiving CCM broadcasts tagged with the MAID it uses internally.

cfm_fault_status : loopback: none
> Indicates a CFM fault was triggered due to the reception of a CCM advertising the same MPID configured in the **cfm_mpid** column of this **Interface**. This may indicate a loop in the network.

cfm_fault_status : overflow: none
> Indicates a CFM fault was triggered because the CFM module received CCMs from more remote endpoints than it can keep track of.

cfm_fault_status : override: none
> Indicates a CFM fault was manually triggered by an administrator using an **ovs−appctl** command.

cfm_fault_status : interval: none
> Indicates a CFM fault was triggered due to the reception of a CCM frame having an invalid interval.

cfm_remote_opstate: optional string, either **down** or **up**
> When in extended mode, indicates the operational state of the remote endpoint as either **up** or **down**. See **other_config:cfm_opstate**.

cfm_health: optional integer, in range 0 to 100
> Indicates the health of the interface as a percentage of CCM frames received over 21 **other_config:cfm_interval**s. The health of an interface is undefined if it is communicating with more than one **cfm_remote_mpids**. It reduces if healthy heartbeats are not received at the expected rate, and gradually improves as healthy heartbeats are received at the desired rate. Every 21 **other_config:cfm_interval**s, the health of the interface is refreshed.

> As mentioned above, the faults can be triggered for several reasons. The link health will deteriorate even if heartbeats are received but they are reported to be unhealthy. An unhealthy heartbeat in this context is a heartbeat for which either some fault is set or is out of sequence. The interface health can be 100 only on receiving healthy heartbeats at the desired rate.

cfm_remote_mpids: set of integers
> When CFM is properly configured, Open vSwitch will occasionally receive CCM broadcasts. These broadcasts contain the MPID of the sending Maintenance Point. The list of MPIDs from which this **Interface** is receiving broadcasts from is regularly collected and written to this column.

other_config : cfm_interval: optional string, containing an integer
> The interval, in milliseconds, between transmissions of CFM heartbeats. Three missed heartbeat receptions indicate a connectivity fault.

> In standard operation only intervals of 3, 10, 100, 1,000, 10,000, 60,000, or 600,000 ms are supported. Other values will be rounded down to the nearest value on the list. Extended mode (see **other_config:cfm_extended**) supports any interval up to 65,535 ms. In either mode, the default is 1000 ms.

> We do not recommend using intervals less than 100 ms.

other_config : cfm_extended: optional string, either **true** or **false**
> When **true**, the CFM module operates in extended mode. This causes it to use a nonstandard destination address to avoid conflicting with compliant implementations which may be running concurrently on the network. Furthermore, extended mode increases the accuracy of the **cfm_interval** configuration parameter by breaking wire compatibility with 802.1ag compliant implementations. And extended mode allows eight byte MPIDs. Defaults to **false**.

other_config : cfm_demand: optional string, either **true** or **false**
> When **true**, and **other_config:cfm_extended** is true, the CFM module operates in demand mode. When in demand mode, traffic received on the **Interface** is used to indicate liveness. CCMs are still transmitted and received. At least one CCM must be received every 100 * **other_config:cfm_interval** amount of time. Otherwise, even if traffic are received, the CFM module will

raise the connectivity fault.

Demand mode has a couple of caveats:

- To ensure that ovs-vswitchd has enough time to pull statistics from the datapath, the fault detection interval is set to 3.5 * MAX(**other_config:cfm_interval**, 500) ms.

- To avoid ambiguity, demand mode disables itself when there are multiple remote maintenance points.

- If the **Interface** is heavily congested, CCMs containing the **other_config:cfm_opstate** status may be dropped causing changes in the operational state to be delayed. Similarly, if CCMs containing the RDI bit are not received, unidirectional link failures may not be detected.

other_config : cfm_opstate: optional string, either **down** or **up**
> When **down**, the CFM module marks all CCMs it generates as operationally down without triggering a fault. This allows remote maintenance points to choose not to forward traffic to the **Interface** on which this CFM module is running. Currently, in Open vSwitch, the opdown bit of CCMs affects **Interface**s participating in bonds, and the bundle OpenFlow action. This setting is ignored when CFM is not in extended mode. Defaults to **up**.

other_config : cfm_ccm_vlan: optional string, containing an integer, in range 1 to 4,095
> When set, the CFM module will apply a VLAN tag to all CCMs it generates with the given value. May be the string **random** in which case each CCM will be tagged with a different randomly generated VLAN.

other_config : cfm_ccm_pcp: optional string, containing an integer, in range 1 to 7
> When set, the CFM module will apply a VLAN tag to all CCMs it generates with the given PCP value, the VLAN ID of the tag is governed by the value of **other_config:cfm_ccm_vlan**. If **other_config:cfm_ccm_vlan** is unset, a VLAN ID of zero is used.

Bonding Configuration:

other_config : lacp-port-id: optional string, containing an integer, in range 1 to 65,535
> The LACP port ID of this **Interface**. Port IDs are used in LACP negotiations to identify individual ports participating in a bond.

other_config : lacp-port-priority: optional string, containing an integer, in range 1 to 65,535
> The LACP port priority of this **Interface**. In LACP negotiations **Interface**s with numerically lower priorities are preferred for aggregation.

other_config : lacp-aggregation-key: optional string, containing an integer, in range 1 to 65,535
> The LACP aggregation key of this **Interface**. **Interface**s with different aggregation keys may not be active within a given **Port** at the same time.

Virtual Machine Identifiers:

These key-value pairs specifically apply to an interface that represents a virtual Ethernet interface connected to a virtual machine. These key-value pairs should not be present for other types of interfaces. Keys whose names end in **–uuid** have values that uniquely identify the entity in question. For a Citrix XenServer hypervisor, these values are UUIDs in RFC 4122 format. Other hypervisors may use other formats.

external_ids : attached-mac: optional string
> The MAC address programmed into the ''virtual hardware'' for this interface, in the form *xx:xx:xx:xx:xx:xx*. For Citrix XenServer, this is the value of the **MAC** field in the VIF record for this interface.

external_ids : iface-id: optional string
> A system-unique identifier for the interface. On XenServer, this will commonly be the same as **external_ids:xs-vif-uuid**.

external_ids : iface-status: optional string, either **active** or **inactive**

> Hypervisors may sometimes have more than one interface associated with a given **external_ids:iface-id**, only one of which is actually in use at a given time. For example, in some circumstances XenServer has both a "tap" and a "vif" interface for a single **external_ids:iface-id**, but only uses one of them at a time. A hypervisor that behaves this way must mark the currently in use interface **active** and the others **inactive**. A hypervisor that never has more than one interface for a given **external_ids:iface-id** may mark that interface **active** or omit **external_ids:iface-status** entirely.

> During VM migration, a given **external_ids:iface-id** might transiently be marked **active** on two different hypervisors. That is, **active** means that this **external_ids:iface-id** is the active instance within a single hypervisor, not in a broader scope. There is one exception: some hypervisors support "migration" from a given hypervisor to itself (most often for test purposes). During such a "migration," two instances of a single **external_ids:iface-id** might both be briefly marked **active** on a single hypervisor.

external_ids : xs-vif-uuid: optional string

> The virtual interface associated with this interface.

external_ids : xs-network-uuid: optional string

> The virtual network to which this interface is attached.

external_ids : vm-id: optional string

> The VM to which this interface belongs. On XenServer, this will be the same as **external_ids:xs-vm-uuid**.

external_ids : xs-vm-uuid: optional string

> The VM to which this interface belongs.

VLAN Splinters:

> The "VLAN splinters" feature increases Open vSwitch compatibility with buggy network drivers in old versions of Linux that do not properly support VLANs when VLAN devices are not used, at some cost in memory and performance.

> When VLAN splinters are enabled on a particular interface, Open vSwitch creates a VLAN device for each in-use VLAN. For sending traffic tagged with a VLAN on the interface, it substitutes the VLAN device. Traffic received on the VLAN device is treated as if it had been received on the interface on the particular VLAN.

> VLAN splinters consider a VLAN to be in use if:

> - The VLAN is the **tag** value in any **Port** record.

> - The VLAN is listed within the **trunks** column of the **Port** record of an interface on which VLAN splinters are enabled. An empty **trunks** does not influence the in-use VLANs: creating 4,096 VLAN devices is impractical because it will exceed the current 1,024 port per datapath limit.

> - An OpenFlow flow within any bridge matches the VLAN.

> The same set of in-use VLANs applies to every interface on which VLAN splinters are enabled. That is, the set is not chosen separately for each interface but selected once as the union of all in-use VLANs based on the rules above.

> It does not make sense to enable VLAN splinters on an interface for an access port, or on an interface that is not a physical port.

> VLAN splinters are deprecated. When broken device drivers are no longer in widespread use, we will delete this feature.

other_config : enable-vlan-splinters: optional string, either **true** or **false**

> Set to **true** to enable VLAN splinters on this interface. Defaults to **false**.

VLAN splinters increase kernel and userspace memory overhead, so do not use them unless they are needed.

VLAN splinters do not support 802.1p priority tags. Received priorities will appear to be 0, regardless of their actual values, and priorities on transmitted packets will also be cleared to 0.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

other_config: map of string-string pairs

external_ids: map of string-string pairs

Flow_Table TABLE
> Configuration for a particular OpenFlow table.

Summary:

name	optional string
flow_limit	optional integer, at least 0
overflow_policy	optional string, either **refuse** or **evict**
groups	set of strings
prefixes	set of up to 3 strings
Common Columns:	
external_ids	map of string-string pairs

Details:

name: optional string
> The table's name. Set this column to change the name that controllers will receive when they request table statistics, e.g. **ovs−ofctl dump−tables**. The name does not affect switch behavior.

flow_limit: optional integer, at least 0
> If set, limits the number of flows that may be added to the table. Open vSwitch may limit the number of flows in a table for other reasons, e.g. due to hardware limitations or for resource availability or performance reasons.

overflow_policy: optional string, either **refuse** or **evict**
> Controls the switch's behavior when an OpenFlow flow table modification request would add flows in excess of **flow_limit**. The supported values are:

> **refuse** Refuse to add the flow or flows. This is also the default policy when **overflow_policy** is unset.

> **evict** Delete the flow that will expire soonest. See **groups** for details.

groups: set of strings
> When **overflow_policy** is **evict**, this controls how flows are chosen for eviction when the flow table would otherwise exceed **flow_limit** flows. Its value is a set of NXM fields or sub-fields, each of which takes one of the forms *field*[] or *field*[*start*..*end*], e.g. **NXM_OF_IN_PORT[]**. Please see **nicira−ext.h** for a complete list of NXM field names.

> When a flow must be evicted due to overflow, the flow to evict is chosen through an approximation of the following algorithm:

> 1. Divide the flows in the table into groups based on the values of the specified fields or subfields, so that all of the flows in a given group have the same values for those fields. If a flow does not specify a given field, that field's value is treated as 0.

> 2. Consider the flows in the largest group, that is, the group that contains the greatest number of flows. If two or more groups all have the same largest number of flows, consider the flows in all of those groups.

> 3. Among the flows under consideration, choose the flow that expires soonest for eviction.

> The eviction process only considers flows that have an idle timeout or a hard timeout. That is, eviction never deletes permanent flows. (Permanent flows do count against **flow_limit**.)

> Open vSwitch ignores any invalid or unknown field specifications.

> When **overflow_policy** is not **evict**, this column has no effect.

prefixes: set of up to 3 strings
> This string set specifies which fields should be used for address prefix tracking. Prefix tracking allows the classifier to skip rules with longer than necessary prefixes, resulting in better wildcarding for datapath flows.

> Prefix tracking may be beneficial when a flow table contains matches on IP address fields with different prefix lengths. For example, when a flow table contains IP address matches on both full

addresses and proper prefixes, the full address matches will typically cause the datapath flow to un-wildcard the whole address field (depending on flow entry priorities). In this case each packet with a different address gets handed to the userspace for flow processing and generates its own datapath flow. With prefix tracking enabled for the address field in question packets with addresses matching shorter prefixes would generate datapath flows where the irrelevant address bits are wildcarded, allowing the same datapath flow to handle all the packets within the prefix in question. In this case many userspace upcalls can be avoided and the overall performance can be better.

This is a performance optimization only, so packets will receive the same treatment with or without prefix tracking.

The supported fields are: **tun_id**, **tun_src**, **tun_dst**, **nw_src**, **nw_dst** (or aliases **ip_src** and **ip_dst**), **ipv6_src**, and **ipv6_dst**. (Using this feature for **tun_id** would only make sense if the tunnel IDs have prefix structure similar to IP addresses.)

By default, the **prefixes=ip_dst,ip_src** are used on each flow table. This instructs the flow classifier to track the IP destination and source addresses used by the rules in this specific flow table.

The keyword **none** is recognized as an explicit override of the default values, causing no prefix fields to be tracked.

To set the prefix fields, the flow table record needs to exist:

ovs−vsctl set Bridge br0 flow_tables:0=@N1 −− −−id=@N1 create Flow_Table name=table0
> Creates a flow table record for the OpenFlow table number 0.

ovs−vsctl set Flow_Table table0 prefixes=ip_dst,ip_src
> Enables prefix tracking for IP source and destination address fields.

There is a maximum number of fields that can be enabled for any one flow table. Currently this limit is 3.

Common Columns:
> The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

> **external_ids**: map of string-string pairs

QoS TABLE

Quality of Service (QoS) configuration for each Port that references it.

Summary:

type	string
queues	map of integer-**Queue** pairs, key in range 0 to 4,294,967,295
Configuration for linux-htb and linux-hfsc:	
other_config : max-rate	optional string, containing an integer
Common Columns:	
other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:

type: string

The type of QoS to implement. The currently defined types are listed below:

linux−htb

Linux "hierarchy token bucket" classifier. See tc-htb(8) (also at **http://linux.die.net/man/8/tc−htb**) and the HTB manual (**http://luxik.cdi.cz/˜devik/qos/htb/manual/userg.htm**) for information on how this classifier works and how to configure it.

linux−hfsc

Linux "Hierarchical Fair Service Curve" classifier. See **http://linux−ip.net/articles/hfsc.en/** for information on how this classifier works.

queues: map of integer-**Queue** pairs, key in range 0 to 4,294,967,295

A map from queue numbers to **Queue** records. The supported range of queue numbers depend on **type**. The queue numbers are the same as the **queue_id** used in OpenFlow in **struct ofp_action_enqueue** and other structures.

Queue 0 is the "default queue." It is used by OpenFlow output actions when no specific queue has been set. When no configuration for queue 0 is present, it is automatically configured as if a **Queue** record with empty **dscp** and **other_config** columns had been specified. (Before version 1.6, Open vSwitch would leave queue 0 unconfigured in this case. With some queuing disciplines, this dropped all packets destined for the default queue.)

Configuration for linux-htb and linux-hfsc:

The **linux−htb** and **linux−hfsc** classes support the following key-value pair:

other_config : max-rate: optional string, containing an integer

Maximum rate shared by all queued traffic, in bit/s. Optional. If not specified, for physical interfaces, the default is the link rate. For other interfaces or if the link rate cannot be determined, the default is currently 100 Mbps.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

other_config: map of string-string pairs

external_ids: map of string-string pairs

Queue TABLE

A configuration for a port output queue, used in configuring Quality of Service (QoS) features. May be referenced by **queues** column in **QoS** table.

Summary:

dscp	optional integer, in range 0 to 63
Configuration for linux-htb QoS:	
other_config : min-rate	optional string, containing an integer, at least 1
other_config : max-rate	optional string, containing an integer, at least 1
other_config : burst	optional string, containing an integer, at least 1
other_config : priority	optional string, containing an integer, in range 0 to 4,294,967,295
Configuration for linux-hfsc QoS:	
other_config : min-rate	optional string, containing an integer, at least 1
other_config : max-rate	optional string, containing an integer, at least 1
Common Columns:	
other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:

dscp: optional integer, in range 0 to 63

> If set, Open vSwitch will mark all traffic egressing this **Queue** with the given DSCP bits. Traffic egressing the default **Queue** is only marked if it was explicitly selected as the **Queue** at the time the packet was output. If unset, the DSCP bits of traffic egressing this **Queue** will remain unchanged.

Configuration for linux-htb QoS:

QoS type linux–htb may use **queue_id**s less than 61440. It has the following key-value pairs defined.

other_config : min-rate: optional string, containing an integer, at least 1

> Minimum guaranteed bandwidth, in bit/s.

other_config : max-rate: optional string, containing an integer, at least 1

> Maximum allowed bandwidth, in bit/s. Optional. If specified, the queue's rate will not be allowed to exceed the specified value, even if excess bandwidth is available. If unspecified, defaults to no limit.

other_config : burst: optional string, containing an integer, at least 1

> Burst size, in bits. This is the maximum amount of ''credits'' that a queue can accumulate while it is idle. Optional. Details of the **linux–htb** implementation require a minimum burst size, so a too-small **burst** will be silently ignored.

other_config : priority: optional string, containing an integer, in range 0 to 4,294,967,295

> A queue with a smaller **priority** will receive all the excess bandwidth that it can use before a queue with a larger value receives any. Specific priority values are unimportant; only relative ordering matters. Defaults to 0 if unspecified.

Configuration for linux-hfsc QoS:

QoS type linux–hfsc may use **queue_id**s less than 61440. It has the following key-value pairs defined.

other_config : min-rate: optional string, containing an integer, at least 1

> Minimum guaranteed bandwidth, in bit/s.

other_config : max-rate: optional string, containing an integer, at least 1

> Maximum allowed bandwidth, in bit/s. Optional. If specified, the queue's rate will not be allowed to exceed the specified value, even if excess bandwidth is available. If unspecified, defaults to no limit.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

other_config: map of string-string pairs

external_ids: map of string-string pairs

Mirror TABLE

A port mirror within a **Bridge**.

A port mirror configures a bridge to send selected frames to special ''mirrored'' ports, in addition to their normal destinations. Mirroring traffic may also be referred to as SPAN or RSPAN, depending on how the mirrored traffic is sent.

Summary:

name	string
Selecting Packets for Mirroring:	
select_all	boolean
select_dst_port	set of weak reference to **Port**s
select_src_port	set of weak reference to **Port**s
select_vlan	set of up to 4,096 integers, in range 0 to 4,095
Mirroring Destination Configuration:	
output_port	optional weak reference to **Port**
output_vlan	optional integer, in range 1 to 4,095
Statistics: Mirror counters:	
statistics : tx_packets	optional integer
statistics : tx_bytes	optional integer
Common Columns:	
external_ids	map of string-string pairs

Details:

name: string

Arbitrary identifier for the **Mirror**.

Selecting Packets for Mirroring:

To be selected for mirroring, a given packet must enter or leave the bridge through a selected port and it must also be in one of the selected VLANs.

select_all: boolean

If true, every packet arriving or departing on any port is selected for mirroring.

select_dst_port: set of weak reference to **Port**s

Ports on which departing packets are selected for mirroring.

select_src_port: set of weak reference to **Port**s

Ports on which arriving packets are selected for mirroring.

select_vlan: set of up to 4,096 integers, in range 0 to 4,095

VLANs on which packets are selected for mirroring. An empty set selects packets on all VLANs.

Mirroring Destination Configuration:

These columns are mutually exclusive. Exactly one of them must be nonempty.

output_port: optional weak reference to **Port**

Output port for selected packets, if nonempty.

Specifying a port for mirror output reserves that port exclusively for mirroring. No frames other than those selected for mirroring via this column will be forwarded to the port, and any frames received on the port will be discarded.

The output port may be any kind of port supported by Open vSwitch. It may be, for example, a physical port (sometimes called SPAN) or a GRE tunnel.

output_vlan: optional integer, in range 1 to 4,095

Output VLAN for selected packets, if nonempty.

The frames will be sent out all ports that trunk **output_vlan**, as well as any ports with implicit VLAN **output_vlan**. When a mirrored frame is sent out a trunk port, the frame's VLAN tag will be set to **output_vlan**, replacing any existing tag; when it is sent out an implicit VLAN port, the frame will not be tagged. This type of mirroring is sometimes called RSPAN.

See the documentation for **other_config:forward-bpdu** in the **Interface** table for a list of destination MAC addresses which will not be mirrored to a VLAN to avoid confusing switches that interpret the protocols that they represent.

Please note: Mirroring to a VLAN can disrupt a network that contains unmanaged switches. Consider an unmanaged physical switch with two ports: port 1, connected to an end host, and port 2, connected to an Open vSwitch configured to mirror received packets into VLAN 123 on port 2. Suppose that the end host sends a packet on port 1 that the physical switch forwards to port 2. The Open vSwitch forwards this packet to its destination and then reflects it back on port 2 in VLAN 123. This reflected packet causes the unmanaged physical switch to replace the MAC learning table entry, which correctly pointed to port 1, with one that incorrectly points to port 2. Afterward, the physical switch will direct packets destined for the end host to the Open vSwitch on port 2, instead of to the end host on port 1, disrupting connectivity. If mirroring to a VLAN is desired in this scenario, then the physical switch must be replaced by one that learns Ethernet addresses on a per-VLAN basis. In addition, learning should be disabled on the VLAN containing mirrored traffic. If this is not done then intermediate switches will learn the MAC address of each end host from the mirrored traffic. If packets being sent to that end host are also mirrored, then they will be dropped since the switch will attempt to send them out the input port. Disabling learning for the VLAN will cause the switch to correctly send the packet out all ports configured for that VLAN. If Open vSwitch is being used as an intermediate switch, learning can be disabled by adding the mirrored VLAN to **flood_vlans** in the appropriate **Bridge** table or tables.

Mirroring to a GRE tunnel has fewer caveats than mirroring to a VLAN and should generally be preferred.

Statistics: Mirror counters:

Key-value pairs that report mirror statistics. The update period is controlled by **other_config:stats-update-interval** in the **Open_vSwitch** table.

statistics : tx_packets: optional integer
   Number of packets transmitted through this mirror.

statistics : tx_bytes: optional integer
   Number of bytes transmitted through this mirror.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

external_ids: map of string-string pairs

Controller TABLE

An OpenFlow controller.

Open vSwitch supports two kinds of OpenFlow controllers:

Primary controllers

This is the kind of controller envisioned by the OpenFlow 1.0 specification. Usually, a primary controller implements a network policy by taking charge of the switch's flow table.

Open vSwitch initiates and maintains persistent connections to primary controllers, retrying the connection each time it fails or drops. The **fail_mode** column in the **Bridge** table applies to primary controllers.

Open vSwitch permits a bridge to have any number of primary controllers. When multiple controllers are configured, Open vSwitch connects to all of them simultaneously. Because OpenFlow 1.0 does not specify how multiple controllers coordinate in interacting with a single switch, more than one primary controller should be specified only if the controllers are themselves designed to coordinate with each other. (The Nicira-defined **NXT_ROLE** OpenFlow vendor extension may be useful for this.)

Service controllers

These kinds of OpenFlow controller connections are intended for occasional support and maintenance use, e.g. with **ovs−ofctl**. Usually a service controller connects only briefly to inspect or modify some of a switch's state.

Open vSwitch listens for incoming connections from service controllers. The service controllers initiate and, if necessary, maintain the connections from their end. The **fail_mode** column in the **Bridge** table does not apply to service controllers.

Open vSwitch supports configuring any number of service controllers.

The **target** determines the type of controller.

Summary:

Core Features:

target	string
connection_mode	optional string, either **in−band** or **out−of−band**

Controller Failure Detection and Handling:

max_backoff	optional integer, at least 1,000
inactivity_probe	optional integer

Asynchronous Messages:

enable_async_messages	optional boolean

Controller Rate Limiting:

controller_rate_limit	optional integer, at least 100
controller_burst_limit	optional integer, at least 25

Controller Rate Limiting Statistics:

status : packet-in-TYPE-bypassed	optional string, containing an integer, at least 0
status : packet-in-TYPE-queued	optional string, containing an integer, at least 0
status : packet-in-TYPE-dropped	optional string, containing an integer, at least 0
status : packet-in-TYPE-backlog	optional string, containing an integer, at least 0

Additional In-Band Configuration:

local_ip	optional string
local_netmask	optional string
local_gateway	optional string

Controller Status:

is_connected	boolean
role	optional string, one of **slave**, **other**, or **master**
status : last_error	optional string

status : state   optional string, one of **ACTIVE**, **VOID**, **CON-
                                                                  NECTING**, **IDLE**, or **BACKOFF**
status : sec_since_connect   optional string, containing an integer, at least 0
status : sec_since_disconnect   optional string, containing an integer, at least 1
Connection Parameters:
other_config : dscp   optional string, containing an integer
Common Columns:
external_ids   map of string-string pairs
other_config   map of string-string pairs

Details:

> *Core Features:*

> > **target**: string

> > > Connection method for controller.

> > > The following connection methods are currently supported for primary controllers:

> > > **ssl:***ip*[**:***port*]

> > > > The specified SSL *port* on the host at the given *ip*, which must be expressed as an IP address (not a DNS name). The **ssl** column in the **Open_vSwitch** table must point to a valid SSL configuration when this form is used.

> > > > If *port* is not specified, it currently defaults to 6633. In the future, the default will change to 6653, which is the IANA-defined value.

> > > > SSL support is an optional feature that is not always built as part of Open vSwitch.

> > > **tcp:***ip*[**:***port*]

> > > > The specified TCP *port* on the host at the given *ip*, which must be expressed as an IP address (not a DNS name), where *ip* can be IPv4 or IPv6 address. If *ip* is an IPv6 address, wrap it in square brackets, e.g. **tcp:[::1]:6632**.

> > > > If *port* is not specified, it currently defaults to 6633. In the future, the default will change to 6653, which is the IANA-defined value.

> > > The following connection methods are currently supported for service controllers:

> > > **pssl:**[*port*][**:***ip*]

> > > > Listens for SSL connections on the specified TCP *port*. If *ip*, which must be expressed as an IP address (not a DNS name), is specified, then connections are restricted to the specified local IP address (either IPv4 or IPv6). If *ip* is an IPv6 address, wrap it in square brackets, e.g. **pssl:6632:[::1]**.

> > > > If *port* is not specified, it currently defaults to 6633. If *ip* is not specified then it listens only on IPv4 (but not IPv6) addresses. The **ssl** column in the **Open_vSwitch** table must point to a valid SSL configuration when this form is used.

> > > > If *port* is not specified, it currently defaults to 6633. In the future, the default will change to 6653, which is the IANA-defined value.

> > > > SSL support is an optional feature that is not always built as part of Open vSwitch.

> > > **ptcp:**[*port*][**:***ip*]

> > > > Listens for connections on the specified TCP *port*. If *ip*, which must be expressed as an IP address (not a DNS name), is specified, then connections are restricted to the specified local IP address (either IPv4 or IPv6). If *ip* is an IPv6 address, wrap it in square brackets, e.g. **ptcp:6632:[::1]**. If *ip* is not specified then it listens only on IPv4 addresses.

> > > > If *port* is not specified, it currently defaults to 6633. In the future, the default will change to 6653, which is the IANA-defined value.

> > > When multiple controllers are configured for a single bridge, the **target** values must be unique. Duplicate **target** values yield unspecified results.

connection_mode: optional string, either **in−band** or **out−of−band**

> If it is specified, this setting must be one of the following strings that describes how Open vSwitch contacts this OpenFlow controller over the network:

> > **in−band**
> >
> > > In this mode, this controller's OpenFlow traffic travels over the bridge associated with the controller. With this setting, Open vSwitch allows traffic to and from the controller regardless of the contents of the OpenFlow flow table. (Otherwise, Open vSwitch would never be able to connect to the controller, because it did not have a flow to enable it.) This is the most common connection mode because it is not necessary to maintain two independent networks.

> > **out−of−band**
> >
> > > In this mode, OpenFlow traffic uses a control network separate from the bridge associated with this controller, that is, the bridge does not use any of its own network devices to communicate with the controller. The control network must be configured separately, before or after **ovs−vswitchd** is started.

> If not specified, the default is implementation-specific.

Controller Failure Detection and Handling:

max_backoff: optional integer, at least 1,000

> Maximum number of milliseconds to wait between connection attempts. Default is implementation-specific.

inactivity_probe: optional integer

> Maximum number of milliseconds of idle time on connection to controller before sending an inactivity probe message. If Open vSwitch does not communicate with the controller for the specified number of seconds, it will send a probe. If a response is not received for the same additional amount of time, Open vSwitch assumes the connection has been broken and attempts to reconnect. Default is implementation-specific. A value of 0 disables inactivity probes.

Asynchronous Messages:

OpenFlow switches send certain messages to controllers spontanenously, that is, not in response to any request from the controller. These messages are called ''asynchronous messages.'' These columns allow asynchronous messages to be limited or disabled to ensure the best use of network resources.

enable_async_messages: optional boolean

> The OpenFlow protocol enables asynchronous messages at time of connection establishment, which means that a controller can receive asynchronous messages, potentially many of them, even if it turns them off immediately after connecting. Set this column to **false** to change Open vSwitch behavior to disable, by default, all asynchronous messages. The controller can use the **NXT_SET_ASYNC_CONFIG** Nicira extension to OpenFlow to turn on any messages that it does want to receive, if any.

Controller Rate Limiting:

A switch can forward packets to a controller over the OpenFlow protocol. Forwarding packets this way at too high a rate can overwhelm a controller, frustrate use of the OpenFlow connection for other purposes, increase the latency of flow setup, and use an unreasonable amount of bandwidth. Therefore, Open vSwitch supports limiting the rate of packet forwarding to a controller.

There are two main reasons in OpenFlow for a packet to be sent to a controller: either the packet ''misses'' in the flow table, that is, there is no matching flow, or a flow table action says to send the packet to the controller. Open vSwitch limits the rate of each kind of packet separately at the configured rate. Therefore, the actual rate that packets are sent to the controller can be up to twice the configured rate, when packets are sent for both reasons.

This feature is specific to forwarding packets over an OpenFlow connection. It is not general-purpose QoS. See the **QoS** table for quality of service configuration, and **ingress_policing_rate** in the **Interface** table for ingress policing configuration.

controller_rate_limit: optional integer, at least 100
> The maximum rate at which the switch will forward packets to the OpenFlow controller, in packets per second. If no value is specified, rate limiting is disabled.

controller_burst_limit: optional integer, at least 25
> When a high rate triggers rate-limiting, Open vSwitch queues packets to the controller for each port and transmits them to the controller at the configured rate. This value limits the number of queued packets. Ports on a bridge share the packet queue fairly.

> This value has no effect unless **controller_rate_limit** is configured. The current default when this value is not specified is one-quarter of **controller_rate_limit**, meaning that queuing can delay forwarding a packet to the controller by up to 250 ms.

Controller Rate Limiting Statistics:
> These values report the effects of rate limiting. Their values are relative to establishment of the most recent OpenFlow connection, or since rate limiting was enabled, whichever happened more recently. Each consists of two values, one with **TYPE** replaced by **miss** for rate limiting flow table misses, and the other with **TYPE** replaced by **action** for rate limiting packets sent by OpenFlow actions.

> These statistics are reported only when controller rate limiting is enabled.

status : packet-in-TYPE-bypassed: optional string, containing an integer, at least 0
> Number of packets sent directly to the controller, without queuing, because the rate did not exceed the configured maximum.

status : packet-in-TYPE-queued: optional string, containing an integer, at least 0
> Number of packets added to the queue to send later.

status : packet-in-TYPE-dropped: optional string, containing an integer, at least 0
> Number of packets added to the queue that were later dropped due to overflow. This value is less than or equal to **status:packet-in-TYPE-queued**.

status : packet-in-TYPE-backlog: optional string, containing an integer, at least 0
> Number of packets currently queued. The other statistics increase monotonically, but this one fluctuates between 0 and the **controller_burst_limit** as conditions change.

Additional In-Band Configuration:
> These values are considered only in in-band control mode (see **connection_mode**).

> When multiple controllers are configured on a single bridge, there should be only one set of unique values in these columns. If different values are set for these columns in different controllers, the effect is unspecified.

local_ip: optional string
> The IP address to configure on the local port, e.g. **192.168.0.123**. If this value is unset, then **local_netmask** and **local_gateway** are ignored.

local_netmask: optional string
> The IP netmask to configure on the local port, e.g. **255.255.255.0**. If **local_ip** is set but this value is unset, then the default is chosen based on whether the IP address is class A, B, or C.

local_gateway: optional string
> The IP address of the gateway to configure on the local port, as a string, e.g. **192.168.0.1**. Leave this column unset if this network has no gateway.

Controller Status:

is_connected: boolean
> **true** if currently connected to this controller, **false** otherwise.

role: optional string, one of **slave**, **other**, or **master**
> The level of authority this controller has on the associated bridge. Possible values are:

> **other**   Allows the controller access to all OpenFlow features.

master   Equivalent to **other**, except that there may be at most one master controller at a time. When a controller configures itself as **master**, any existing master is demoted to the **slave** role.

slave   Allows the controller read-only access to OpenFlow features. Attempts to modify the flow table will be rejected with an error. Slave controllers do not receive OFPT_PACKET_IN or OFPT_FLOW_REMOVED messages, but they do receive OFPT_PORT_STATUS messages.

status : last_error: optional string
A human-readable description of the last error on the connection to the controller; i.e. **strerror(errno)**. This key will exist only if an error has occurred.

status : state: optional string, one of **ACTIVE**, **VOID**, **CONNECTING**, **IDLE**, or **BACKOFF**
The state of the connection to the controller:

VOID   Connection is disabled.

BACKOFF
Attempting to reconnect at an increasing period.

CONNECTING
Attempting to connect.

ACTIVE
Connected, remote host responsive.

IDLE   Connection is idle. Waiting for response to keep-alive.

These values may change in the future. They are provided only for human consumption.

status : sec_since_connect: optional string, containing an integer, at least 0
The amount of time since this controller last successfully connected to the switch (in seconds). Value is empty if controller has never successfully connected.

status : sec_since_disconnect: optional string, containing an integer, at least 1
The amount of time since this controller last disconnected from the switch (in seconds). Value is empty if controller has never disconnected.

Connection Parameters:
Additional configuration for a connection between the controller and the Open vSwitch.

other_config : dscp: optional string, containing an integer
The Differentiated Service Code Point (DSCP) is specified using 6 bits in the Type of Service (TOS) field in the IP header. DSCP provides a mechanism to classify the network traffic and provide Quality of Service (QoS) on IP networks. The DSCP value specified here is used when establishing the connection between the controller and the Open vSwitch. If no value is specified, a default value of 48 is chosen. Valid DSCP values must be in the range 0 to 63.

Common Columns:
The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

external_ids: map of string-string pairs

other_config: map of string-string pairs

Manager TABLE

Configuration for a database connection to an Open vSwitch database (OVSDB) client.

This table primarily configures the Open vSwitch database (**ovsdb−server**), not the Open vSwitch switch (**ovs−vswitchd**). The switch does read the table to determine what connections should be treated as in-band.

The Open vSwitch database server can initiate and maintain active connections to remote clients. It can also listen for database connections.

Summary:

Core Features:

target	string (must be unique within table)
connection_mode	optional string, either **in−band** or **out−of−band**

Client Failure Detection and Handling:

max_backoff	optional integer, at least 1,000
inactivity_probe	optional integer

Status:

is_connected	boolean
status : last_error	optional string
status : state	optional string, one of **ACTIVE**, **VOID**, **CON-NECTING**, **IDLE**, or **BACKOFF**
status : sec_since_connect	optional string, containing an integer, at least 0
status : sec_since_disconnect	optional string, containing an integer, at least 0
status : locks_held	optional string
status : locks_waiting	optional string
status : locks_lost	optional string
status : n_connections	optional string, containing an integer, at least 2
status : bound_port	optional string, containing an integer

Connection Parameters:

other_config : dscp	optional string, containing an integer

Common Columns:

external_ids	map of string-string pairs
other_config	map of string-string pairs

Details:

Core Features:

target: string (must be unique within table)

Connection method for managers.

The following connection methods are currently supported:

ssl:*ip*[**:***port*]

The specified SSL *port* on the host at the given *ip*, which must be expressed as an IP address (not a DNS name). The **ssl** column in the **Open_vSwitch** table must point to a valid SSL configuration when this form is used.

If *port* is not specified, it currently defaults to 6632. In the future, the default will change to 6640, which is the IANA-defined value.

SSL support is an optional feature that is not always built as part of Open vSwitch.

tcp:*ip*[**:***port*]

The specified TCP *port* on the host at the given *ip*, which must be expressed as an IP address (not a DNS name), where *ip* can be IPv4 or IPv6 address. If *ip* is an IPv6 address, wrap it in square brackets, e.g. **tcp:[::1]:6632**.

If *port* is not specified, it currently defaults to 6632. In the future, the default will change to 6640, which is the IANA-defined value.

pssl:[*port*][**:***ip*]

> Listens for SSL connections on the specified TCP *port*. Specify 0 for *port* to have the kernel automatically choose an available port. If *ip*, which must be expressed as an IP address (not a DNS name), is specified, then connections are restricted to the specified local IP address (either IPv4 or IPv6 address). If *ip* is an IPv6 address, wrap in square brackets, e.g. **pssl:6632:[::1]**. If *ip* is not specified then it listens only on IPv4 (but not IPv6) addresses. The **ssl** column in the **Open_vSwitch** table must point to a valid SSL configuration when this form is used.

> If *port* is not specified, it currently defaults to 6632. In the future, the default will change to 6640, which is the IANA-defined value.

> SSL support is an optional feature that is not always built as part of Open vSwitch.

ptcp:[*port*][**:***ip*]

> Listens for connections on the specified TCP *port*. Specify 0 for *port* to have the kernel automatically choose an available port. If *ip*, which must be expressed as an IP address (not a DNS name), is specified, then connections are restricted to the specified local IP address (either IPv4 or IPv6 address). If *ip* is an IPv6 address, wrap it in square brackets, e.g. **ptcp:6632:[::1]**. If *ip* is not specified then it listens only on IPv4 addresses.

> If *port* is not specified, it currently defaults to 6632. In the future, the default will change to 6640, which is the IANA-defined value.

When multiple managers are configured, the **target** values must be unique. Duplicate **target** values yield unspecified results.

connection_mode: optional string, either **in−band** or **out−of−band**

> If it is specified, this setting must be one of the following strings that describes how Open vSwitch contacts this OVSDB client over the network:

> **in−band**

> > In this mode, this connection's traffic travels over a bridge managed by Open vSwitch. With this setting, Open vSwitch allows traffic to and from the client regardless of the contents of the OpenFlow flow table. (Otherwise, Open vSwitch would never be able to connect to the client, because it did not have a flow to enable it.) This is the most common connection mode because it is not necessary to maintain two independent networks.

> **out−of−band**

> > In this mode, the client's traffic uses a control network separate from that managed by Open vSwitch, that is, Open vSwitch does not use any of its own network devices to communicate with the client. The control network must be configured separately, before or after **ovs−vswitchd** is started.

> If not specified, the default is implementation-specific.

Client Failure Detection and Handling:

max_backoff: optional integer, at least 1,000

> Maximum number of milliseconds to wait between connection attempts. Default is implementation-specific.

inactivity_probe: optional integer

> Maximum number of milliseconds of idle time on connection to the client before sending an inactivity probe message. If Open vSwitch does not communicate with the client for the specified number of seconds, it will send a probe. If a response is not received for the same additional amount of time, Open vSwitch assumes the connection has been broken and attempts to reconnect. Default is implementation-specific. A value of 0 disables inactivity probes.

Status:

is_connected: boolean

> **true** if currently connected to this manager, **false** otherwise.

status : last_error: optional string

> A human-readable description of the last error on the connection to the manager; i.e. **str-error(errno)**. This key will exist only if an error has occurred.

status : state: optional string, one of **ACTIVE**, **VOID**, **CONNECTING**, **IDLE**, or **BACKOFF**

> The state of the connection to the manager:

> **VOID**   Connection is disabled.

> **BACKOFF**
>> Attempting to reconnect at an increasing period.

> **CONNECTING**
>> Attempting to connect.

> **ACTIVE**
>> Connected, remote host responsive.

> **IDLE**   Connection is idle. Waiting for response to keep-alive.

> These values may change in the future. They are provided only for human consumption.

status : sec_since_connect: optional string, containing an integer, at least 0

> The amount of time since this manager last successfully connected to the database (in seconds). Value is empty if manager has never successfully connected.

status : sec_since_disconnect: optional string, containing an integer, at least 0

> The amount of time since this manager last disconnected from the database (in seconds). Value is empty if manager has never disconnected.

status : locks_held: optional string

> Space-separated list of the names of OVSDB locks that the connection holds. Omitted if the connection does not hold any locks.

status : locks_waiting: optional string

> Space-separated list of the names of OVSDB locks that the connection is currently waiting to acquire. Omitted if the connection is not waiting for any locks.

status : locks_lost: optional string

> Space-separated list of the names of OVSDB locks that the connection has had stolen by another OVSDB client. Omitted if no locks have been stolen from this connection.

status : n_connections: optional string, containing an integer, at least 2

> When **target** specifies a connection method that listens for inbound connections (e.g. **ptcp:** or **pssl:**) and more than one connection is actually active, the value is the number of active connections. Otherwise, this key-value pair is omitted.

> When multiple connections are active, status columns and key-value pairs (other than this one) report the status of one arbitrarily chosen connection.

status : bound_port: optional string, containing an integer

> When **target** is **ptcp:** or **pssl:**, this is the TCP port on which the OVSDB server is listening. (This is is particularly useful when **target** specifies a port of 0, allowing the kernel to choose any available port.)

Connection Parameters:

> Additional configuration for a connection between the manager and the Open vSwitch Database.

other_config : dscp: optional string, containing an integer

> The Differentiated Service Code Point (DSCP) is specified using 6 bits in the Type of Service (TOS) field in the IP header. DSCP provides a mechanism to classify the network traffic and provide Quality of Service (QoS) on IP networks. The DSCP value specified here is used when

establishing the connection between the manager and the Open vSwitch. If no value is specified, a default value of 48 is chosen. Valid DSCP values must be in the range 0 to 63.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

external_ids: map of string-string pairs

other_config: map of string-string pairs

NetFlow TABLE

A NetFlow target. NetFlow is a protocol that exports a number of details about terminating IP flows, such as the principals involved and duration.

Summary:

targets	set of 1 or more strings
engine_id	optional integer, in range 0 to 255
engine_type	optional integer, in range 0 to 255
active_timeout	integer, at least −1
add_id_to_interface	boolean
Common Columns:	
external_ids	map of string-string pairs

Details:

targets: set of 1 or more strings

NetFlow targets in the form *ip***:***port*. The *ip* must be specified numerically, not as a DNS name.

engine_id: optional integer, in range 0 to 255

Engine ID to use in NetFlow messages. Defaults to datapath index if not specified.

engine_type: optional integer, in range 0 to 255

Engine type to use in NetFlow messages. Defaults to datapath index if not specified.

active_timeout: integer, at least −1

The interval at which NetFlow records are sent for flows that are still active, in seconds. A value of **0** requests the default timeout (currently 600 seconds); a value of **−1** disables active timeouts.

add_id_to_interface: boolean

If this column's value is **false**, the ingress and egress interface fields of NetFlow flow records are derived from OpenFlow port numbers. When it is **true**, the 7 most significant bits of these fields will be replaced by the least significant 7 bits of the engine id. This is useful because many Net-Flow collectors do not expect multiple switches to be sending messages from the same host, so they do not store the engine information which could be used to disambiguate the traffic.

When this option is enabled, a maximum of 508 ports are supported.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

external_ids: map of string-string pairs

SSL TABLE

   SSL configuration for an Open_vSwitch.

Summary:

private_key	string
certificate	string
ca_cert	string
bootstrap_ca_cert	boolean
Common Columns:	
external_ids	map of string-string pairs

Details:

   private_key: string

   Name of a PEM file containing the private key used as the switch's identity for SSL connections to the controller.

   certificate: string

   Name of a PEM file containing a certificate, signed by the certificate authority (CA) used by the controller and manager, that certifies the switch's private key, identifying a trustworthy switch.

   ca_cert: string

   Name of a PEM file containing the CA certificate used to verify that the switch is connected to a trustworthy controller.

   bootstrap_ca_cert: boolean

   If set to **true**, then Open vSwitch will attempt to obtain the CA certificate from the controller on its first SSL connection and save it to the named PEM file. If it is successful, it will immediately drop the connection and reconnect, and from then on all SSL connections must be authenticated by a certificate signed by the CA certificate thus obtained. **This option exposes the SSL connection to a man−in−the−middle attack obtaining the initial CA certificate.** It may still be useful for bootstrapping.

   Common Columns:

   The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

   external_ids: map of string-string pairs

sFlow TABLE
> A set of sFlow(R) targets. sFlow is a protocol for remote monitoring of switches.

Summary:
> | | |
> |---|---|
> | **agent** | optional string |
> | **header** | optional integer |
> | **polling** | optional integer |
> | **sampling** | optional integer |
> | **targets** | set of 1 or more strings |
> | *Common Columns:* | |
> | **external_ids** | map of string-string pairs |

Details:
> **agent**: optional string
> > Name of the network device whose IP address should be reported as the ''agent address'' to collec-
> > tors. If not specified, the agent device is figured from the first target address and the routing table.
> > If the routing table does not contain a route to the target, the IP address defaults to the **local_ip** in
> > the collector's **Controller**. If an agent IP address cannot be determined any of these ways, sFlow
> > is disabled.
>
> **header**: optional integer
> > Number of bytes of a sampled packet to send to the collector. If not specified, the default is 128
> > bytes.
>
> **polling**: optional integer
> > Polling rate in seconds to send port statistics to the collector. If not specified, defaults to 30 sec-
> > onds.
>
> **sampling**: optional integer
> > Rate at which packets should be sampled and sent to the collector. If not specified, defaults to
> > 400, which means one out of 400 packets, on average, will be sent to the collector.
>
> **targets**: set of 1 or more strings
> > sFlow targets in the form *ip*:*port*.

> *Common Columns:*
> The overall purpose of these columns is described under **Common Columns** at the beginning of this docu-
> ment.
>
> **external_ids**: map of string-string pairs

IPFIX TABLE

A set of IPFIX collectors. IPFIX is a protocol that exports a number of details about flows.

Summary:

targets	set of strings
sampling	optional integer, in range 1 to 4,294,967,295
obs_domain_id	optional integer, in range 0 to 4,294,967,295
obs_point_id	optional integer, in range 0 to 4,294,967,295
cache_active_timeout	optional integer, in range 0 to 4,200
cache_max_flows	optional integer, in range 0 to 4,294,967,295
other_config : enable-tunnel-sampling	optional string, either **true** or **false**
other_config : enable-input-sampling	optional string, either **true** or **false**
other_config : enable-output-sampling	optional string, either **true** or **false**

Common Columns:

external_ids	map of string-string pairs

Details:

targets: set of strings

> IPFIX target collectors in the form *ip***:***port*.

sampling: optional integer, in range 1 to 4,294,967,295

> For per-bridge packet sampling, i.e. when this row is referenced from a **Bridge**, the rate at which packets should be sampled and sent to each target collector. If not specified, defaults to 400, which means one out of 400 packets, on average, will be sent to each target collector. Ignored for per-flow sampling, i.e. when this row is referenced from a **Flow_Sample_Collector_Set**.

obs_domain_id: optional integer, in range 0 to 4,294,967,295

> For per-bridge packet sampling, i.e. when this row is referenced from a **Bridge**, the IPFIX Observation Domain ID sent in each IPFIX packet. If not specified, defaults to 0. Ignored for per-flow sampling, i.e. when this row is referenced from a **Flow_Sample_Collector_Set**.

obs_point_id: optional integer, in range 0 to 4,294,967,295

> For per-bridge packet sampling, i.e. when this row is referenced from a **Bridge**, the IPFIX Observation Point ID sent in each IPFIX flow record. If not specified, defaults to 0. Ignored for per-flow sampling, i.e. when this row is referenced from a **Flow_Sample_Collector_Set**.

cache_active_timeout: optional integer, in range 0 to 4,200

> The maximum period in seconds for which an IPFIX flow record is cached and aggregated before being sent. If not specified, defaults to 0. If 0, caching is disabled.

cache_max_flows: optional integer, in range 0 to 4,294,967,295

> The maximum number of IPFIX flow records that can be cached at a time. If not specified, defaults to 0. If 0, caching is disabled.

other_config : enable-tunnel-sampling: optional string, either **true** or **false**

> For per-bridge packet sampling, i.e. when this row is referenced from a **Bridge**, enable sampling and reporting tunnel header 7-tuples in IPFIX flow records. Disabled by default. Ignored for per-flow sampling, i.e. when this row is referenced from a **Flow_Sample_Collector_Set**.

> **Please note:** The following enterprise entities are currently used when exporting the sampled tunnel info.

> tunnelType:

>> ID: 891, and enterprise ID 6876 (VMware).

>> type: unsigned 8-bit interger.

>> data type semantics: identifier.

>> description: Identifier of the layer 2 network overlay network encapsulation type: 0x01 VxLAN, 0x02 GRE, 0x03 LISP, 0x05 IPsec+GRE, 0x07 GENEVE.

tunnelKey:
> ID: 892, and enterprise ID 6876 (VMware).
>
> type: variable-length octetarray.
>
> data type semantics: identifier.
>
> description: Key which is used for identifying an individual traffic flow within a VxLAN (24-bit VNI), GENEVE(24-bit VNI), GRE (32- or 64-bit key), or LISP (24-bit instance ID) tunnel. The key is encoded in this octetarray as a 3-, 4-, or 8-byte integer ID in network byte order.

tunnelSourceIPv4Address:
> ID: 893, and enterprise ID 6876 (VMware).
>
> type: unsigned 32-bit interger.
>
> data type semantics: identifier.
>
> description: The IPv4 source address in the tunnel IP packet header.

tunnelDestinationIPv4Address:
> ID: 894, and enterprise ID 6876 (VMware).
>
> type: unsigned 32-bit integer.
>
> data type semantics: identifier.
>
> description: The IPv4 destination address in the tunnel IP packet header.

tunnelProtocolIdentifier:
> ID: 895, and enterprise ID 6876 (VMware).
>
> type: unsigned 8-bit integer.
>
> data type semantics: identifier.
>
> description: The value of the protocol number in the tunnel IP packet header. The protocol number identifies the tunnel IP packet payload type.

tunnelSourceTransportPort:
> ID: 896, and enterprise ID 6876 (VMware).
>
> type: unsigned 16-bit integer.
>
> data type semantics: identifier.
>
> description: The source port identifier in the tunnel transport header. For the transport protocols UDP, TCP, and SCTP, this is the source port number given in the respective header.

tunnelDestinationTransportPort:
> ID: 897, and enterprise ID 6876 (VMware).
>
> type: unsigned 16-bit integer.
>
> data type semantics: identifier.
>
> description: The destination port identifier in the tunnel transport header. For the transport protocols UDP, TCP, and SCTP, this is the destination port number given in the respective header.

other_config : enable-input-sampling: optional string, either **true** or **false**
> For per-bridge packet sampling, i.e. when this row is referenced from a **Bridge**, enable sampling and reporting flows at bridge port input in IPFIX flow records. Enabled by default. Ignored for per-flow sampling, i.e. when this row is referenced from a **Flow_Sample_Collector_Set**.

other_config : enable-output-sampling: optional string, either **true** or **false**
> For per-bridge packet sampling, i.e. when this row is referenced from a **Bridge**, enable sampling and reporting flows at bridge port output in IPFIX flow records. Enabled by default. Ignored for per-flow sampling, i.e. when this row is referenced from a **Flow_Sample_Collector_Set**.

Common Columns:
> The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

external_ids: map of string-string pairs

Flow_Sample_Collector_Set TABLE

A set of IPFIX collectors of packet samples generated by OpenFlow **sample** actions.

Summary:

id	integer, in range 0 to 4,294,967,295
bridge	**Bridge**
ipfix	optional **IPFIX**
Common Columns:	
external_ids	map of string-string pairs

Details:

id: integer, in range 0 to 4,294,967,295

The ID of this collector set, unique among the bridge's collector sets, to be used as the **collector_set_id** in OpenFlow **sample** actions.

bridge: **Bridge**

The bridge into which OpenFlow **sample** actions can be added to send packet samples to this set of IPFIX collectors.

ipfix: optional **IPFIX**

Configuration of the set of IPFIX collectors to send one flow record per sampled packet to.

Common Columns:

The overall purpose of these columns is described under **Common Columns** at the beginning of this document.

external_ids: map of string-string pairs