Open_vSwitch DATABASE

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.
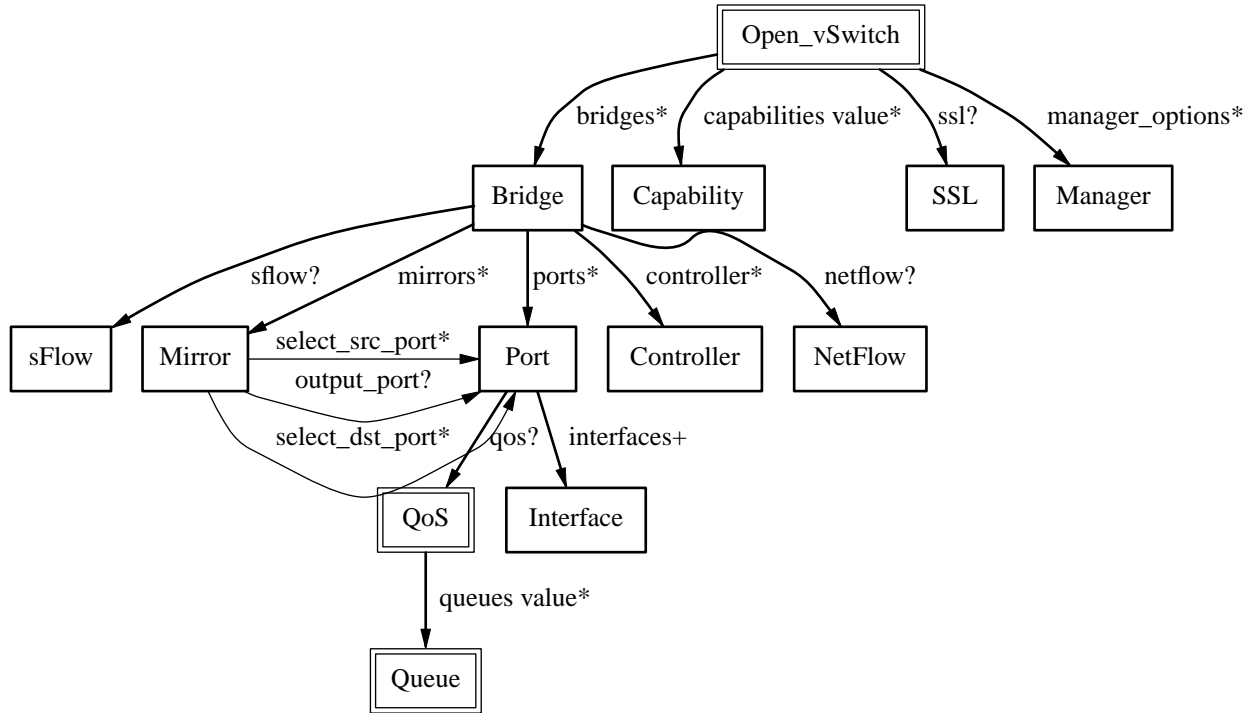
TABLE SUMMARY

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.
QoS	Quality of Service configuration
Queue	QoS output queue.
Mirror	Port mirroring (SPAN/RSPAN/ERSPAN).
Controller	OpenFlow controller configuration.
Manager	OVSDB management connection.
NetFlow	NetFlow configuration.
SSL	SSL configuration.
sFlow	sFlow configuration.
Capability	Capability 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

Status:

next_cfg	integer
cur_cfg	integer
capabilities	map of string-**Capability** pairs

    Statistics:

other_config : enable-statistics	optional string
statistics : cpu	optional string
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**.

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.

capabilities: map of string-**Capability** pairs

> Describes functionality supported by the hardware and software platform on which this Open vSwitch is based. Clients should not modify this column. See the **Capability** description for defined capability categories and the meaning of associated **Capability** records.

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

> 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

> 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**. If Open vSwitch was configured with a build number, then it is also included, e.g. **1.1.0+build6579**.

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	string (must be unique within table)
ports	set of **Port**s
mirrors	set of **Mirror**s
netflow	optional **NetFlow**
sflow	optional **sFlow**
flood_vlans	set of up to 4096 integers, in range 0 to 4,095

OpenFlow Configuration:

controller	set of **Controller**s
fail_mode	optional string, either **secure** or **standalone**
datapath_id	optional string
other_config : datapath-id	optional string
other_config : disable-in-band	optional string
other_config : in-band-queue	optional string

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 : flow-eviction-threshold	optional string
other_config : forward-bpdu	optional string

Common Columns:

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

Details:

Core Features:

name: 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 configuration.

flood_vlans: set of up to 4096 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.

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.

> If this value is unset, the default is implementation-specific.

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

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 : disable-in-band: optional string

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

other_config : in-band-queue: optional string

> 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.

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 : flow-eviction-threshold: optional string

> A number of flows as a nonnegative integer. This sets number of flows at which eviction from the kernel flow table will be triggered. If there are a large number of flows then increasing this value to around the number of flows present can result in reduced CPU usage and packet loss.

> The default is 1000. Values below 100 will be rounded up to 100.

other_config : forward-bpdu: optional string

> Option to allow forwarding of BPDU frames when NORMAL action if invoked. Frames with reserved Ethernet addresses (e.g. STP BPDU) will be forwarded when this option is enabled. 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.

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 logically 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** members.

Summary:

name	string (must be unique within table)
interfaces	set of 1 or more **Interface**s
VLAN Configuration:	
tag	optional integer, in range 0 to 4,095
trunks	set of up to 4096 integers, in range 0 to 4,095
Bonding Configuration:	
bond_mode	optional string, one of **active−backup**, **balance−tcp**, **balance−slb**, or **stable**
Link Failure Detection:	
other_config : bond-detect-mode	optional string
other_config : bond-miimon-interval	optional string
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
other_config : lacp-time	optional string
other_config : lacp-heartbeat	optional string
other_config : bond-hash-basis	optional string
SLB Configuration:	
other_config : bond-rebalance-interval	optional string
bond_fake_iface	boolean
Other Features:	
qos	optional **QoS**
mac	optional string
fake_bridge	boolean
external_ids : fake-bridge-id-*	optional string
Common Columns:	
other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:

name: 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:

A bridge port must be configured for VLANs in one of two mutually exclusive ways:

- A "trunk port" has an empty value for **tag**. Its **trunks** value may be empty or non-empty.

- An ''implicitly tagged VLAN port'' or ''access port'' has an nonempty value for **tag**. Its **trunks** value must be empty.

If **trunks** and **tag** are both nonempty, the configuration is ill-formed.

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

If this is an access port (see above), the port's implicitly tagged VLAN. Must be empty if this is a trunk port.

Frames arriving on trunk ports will be forwarded to this port only if they are tagged with the given VLAN (or, if **tag** is 0, then if they lack a VLAN header). Frames arriving on other access ports will be forwarded to this port only if they have the same **tag** value. Frames forwarded to this port will not have an 802.1Q header.

When a frame with a 802.1Q header that indicates a nonzero VLAN is received on an access port, it is discarded.

trunks: set of up to 4096 integers, in range 0 to 4,095

If this is a trunk port (see above), the 802.1Q VLAN(s) that this port trunks; if it is empty, then the port trunks all VLANs. Must be empty if this is an access port.

Frames arriving on trunk ports are dropped if they are not in one of the specified VLANs. For this purpose, packets that have no VLAN header are treated as part of VLAN 0.

Bonding Configuration:

A port that has more than one interface is a ''bonded port.'' Bonding allows for load balancing and failover. Some kinds of bonding will work with any kind of upstream switch:

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.

The following modes require the upstream switch to support 802.3ad with successful LACP negotiation. If LACP negotiation fails then **balance−slb** style flow hashing is used as a fallback:

balance−tcp

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

stable   Attempts to always assign a given flow to the same slave consistently. In an effort to maintain stability, no load balancing is done. Uses a similar hashing strategy to **balance−tcp**, always taking into account L3 and L4 fields even if LACP negotiations are unsuccessful.

Slave selection decisions are made based on **other_config:bond-stable-id** if set. Otherwise, OpenFlow port number is used. Decisions are consistent across all **ovs−vswitchd** instances with equivalent **other_config:bond-stable-id** values.

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

bond_mode: optional string, one of **active−backup**, **balance−tcp**, **balance−slb**, or **stable**

The type of bonding used for a bonded port. Defaults to **balance−slb** if unset.

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

The means used to detect link failures. Options are **carrier** and **miimon**. 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
>   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 carrier 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 carrier 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. 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.

other_config : lacp-system-priority: optional string
>   The LACP system priority of this **Port**. In LACP negotiations, link status decisions are made by the system with the numerically lower priority. Must be a number between 1 and 65535.

other_config : lacp-time: optional string
>   The LACP timing which should be used on this **Port**. Possible values are **fast**, **slow** and a positive number of milliseconds. 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.

>   Users may manually set a heartbeat transmission rate to increase the fault detection speed further. When manually set, OVS expects the partner switch to be configured with the same transmission rate. Manually setting **lacp−time** to something other than **fast** or **slow** is not supported by the LACP specification.

other_config : lacp-heartbeat: optional string
>   Treats LACP like a simple heartbeat protocol for link state monitoring. Most features of the LACP protocol are disabled when this mode is in use.

other_config : bond-hash-basis: optional string
>   An integer hashed along with flows when choosing output slaves. When changed, all flows will be assigned different hash values possibly causing slave selection decisions to change.

SLB Configuration:
These settings control behavior when a bond is in **balance−slb** mode, regardless of whether the bond was intentionally configured in SLB mode or it fell back to SLB mode because LACP negotiation failed.

other_config : bond-rebalance-interval: optional string
>   For an SLB bonded port, the number of milliseconds between successive attempts to rebalance the bond, that is, to move source MACs and their flows from one interface on the bond to another in an attempt to keep usage of each interface roughly equal. The default is 10000 (10 seconds), and

the minimum is 1000 (1 second).

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.

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**.

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	string (must be unique within table)
mac	optional string
ofport	optional integer

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 : csum	optional string
options : df_inherit	optional string
options : df_default	optional string
options : pmtud	optional string
options : header_cache	optional string

    Tunnel Options: ipsec_gre only:

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

Patch Port Options:

options : peer	optional string

Interface Status:

admin_state	optional string, either **down** or **up**
link_state	optional string, either **down** or **up**
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
statistics	map of string-integer pairs

Ingress Policing:

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

Connectivity Fault Management:

cfm_mpid	optional integer
cfm_fault	optional boolean
cfm_remote_mpids	set of integers
other_config : cfm_interval	optional string
other_config : cfm_extended	optional string

Bonding Configuration:

other_config : bond-stable-id	optional string
other_config : lacp-port-id	optional string
other_config : lacp-port-priority	optional string
other_config : lacp-aggregation-key	optional string

Virtual Machine Identifiers:

external_ids : attached-mac	optional string
external_ids : iface-id	optional string
external_ids : xs-vif-uuid	optional string
external_ids : xs-network-uuid	optional string
external_ids : xs-vm-uuid	optional string

Common Columns:

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

Details:

Core Features:

name: 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.

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.

ofport: optional integer

OpenFlow port number for this interface. Unlike most columns, this column's value should be set only by Open vSwitch itself. Other clients should set this column to an empty set (the default) when creating an **Interface**.

Open vSwitch populates this column when the port number becomes known. If the interface is successfully added, **ofport** will be set to a number between 1 and 65535 (generally either in the range 1 to 65279, inclusive, or 65534, the port number for the OpenFlow "local port"). If the interface cannot be added then Open vSwitch sets this column to -1.

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.

gre   An Ethernet over RFC 2890 Generic Routing Encapsulation over IPv4 tunnel. See **Tunnel Options** for information on configuring GRE tunnels.

ipsec_gre
              An Ethernet over RFC 2890 Generic Routing Encapsulation over IPv4 IPsec tunnel.

capwap
              An Ethernet tunnel over the UDP transport portion of CAPWAP (RFC 5415). This allows interoperability with certain switches that do not support GRE. Only the tunneling component of the protocol is implemented. UDP ports 58881 and 58882 are used as the source and destination ports respectively. CAPWAP is currently supported only with the Linux kernel datapath with kernel version 2.6.25 or later.

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

null   An ignored interface.

See **Tunnel Options** for information on configuring GRE, GRE-over-IPsec, and CAPWAP tunnels. See **Patch Options** for information on configured patch ports.

Tunnel Options:

These options apply to interfaces with **type** of **gre**, **ipsec_gre**, and **capwap**.

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 tunnel endpoint.

options : local_ip: optional string
        Optional. The destination IP that received packets must match. Default is to match all addresses.

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 32-bit (for GRE) or 64-bit (for CAPWAP) 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 32-bit (for GRE) or 64-bit (for CAPWAP) 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. 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 : csum: optional string

> Optional. Compute GRE checksums on outgoing packets. Checksums present on incoming packets will be validated regardless of this setting. Note that GRE checksums impose a significant performance penalty as they cover the entire packet. As the contents of the packet is typically covered by L3 and L4 checksums, this additional checksum only adds value for the GRE and encapsulated Ethernet headers. Default is disabled, set to **true** to enable.

options : df_inherit: optional string

> Optional. If enabled, the Don't Fragment bit will be copied from the inner IP headers (those of the encapsulated traffic) to the outer (tunnel) headers. Default is disabled; set to **true** to enable.

options : df_default: optional string

> Optional. If enabled, the Don't Fragment bit will be set by default on tunnel headers if the **df_inherit** option is not set, or if the encapsulated packet is not IP. Default is enabled; set to **false** to disable.

options : pmtud: optional string

> Optional. Enable tunnel path MTU discovery. If enabled "ICMP Destination Unreachable - Fragmentation Needed" messages will be generated for IPv4 packets with the DF bit set and IPv6 packets above the minimum MTU if the packet size exceeds the path MTU minus the size of the tunnel headers. Note that this option causes behavior that is typically reserved for routers and therefore is not entirely in compliance with the IEEE 802.1D specification for bridges. Default is enabled; set to **false** to disable.

options : header_cache: optional string

> Optional. Enable caching of tunnel headers and the output path. This can lead to a significant performance increase without changing behavior. In general it should not be necessary to adjust this setting. However, the caching can bypass certain components of the IP stack (such as IP tables) and it may be useful to disable it if these features are required or as a debugging measure. Default is enabled, set to **false** to disable.
>
> Header caching is not supported for **ipsec_gre** tunnels.

Tunnel Options: ipsec_gre only:

> These options are supported only for **ipsec_gre** tunnels.

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 Port Options:
These options apply to interfaces with **type** of **patch**.

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_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** or **capwap**.

status : tunnel_egress_iface: optional string
> Egress interface for tunnels. Currently only relevant for GRE and CAPWAP 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
> Whether a carrier is detected on **status:tunnel_egress_iface**. Valid values are **down** and **up**.

statistics: map of string-integer pairs
> Key-value pairs that report interface statistics. The current implementation updates these counters periodically. In the future, we plan to, instead, 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.
>
> The currently defined key-value pairs are listed below. 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.
>
> - Successful transmit and receive counters:
>
>> **rx_packets**
>>> Number of received packets.
>>
>> **rx_bytes**
>>> Number of received bytes.
>>
>> **tx_packets**
>>> Number of transmitted packets.
>>
>> **tx_bytes**
>>> Number of transmitted bytes.
>
> - Receive errors:
>
>> **rx_dropped**
>>> Number of packets dropped by RX.
>>
>> **rx_frame_err**
>>> Number of frame alignment errors.
>>
>> **rx_over_err**
>>> Number of packets with RX overrun.
>>
>> **rx_crc_err**
>>> Number of CRC errors.
>>
>> **rx_errors**
>>> Total number of receive errors, greater than or equal to the sum of the above.
>
> - Transmit errors:
>
>> **tx_dropped**
>>> Number of packets dropped by TX.
>>
>> **collisions**
>>> Number of collisions.
>>
>> **tx_errors**
>>> 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.

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.

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**.

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_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

> The interval, in milliseconds, between transmissions of CFM heartbeats. Three missed heartbeat receptions indicate a connectivity fault. Defaults to 1000.

other_config : cfm_extended: optional string

> 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. Defaults to false.

Bonding Configuration:

other_config : bond-stable-id: optional string

> A positive integer using in **stable** bond mode to make slave selection decisions. Allocating **other_config:bond-stable-id** values consistently across interfaces participating in a bond will guarantee consistent slave selection decisions across **ovs−vswitchd** instances when using **stable** bonding mode.

other_config : lacp-port-id: optional string

> The LACP port ID of this **Interface**. Port IDs are used in LACP negotiations to identify individual ports participating in a bond. Must be a number between 1 and 65535.

other_config : lacp-port-priority: optional string

> The LACP port priority of this **Interface**. In LACP negotiations **Interface**s with numerically lower priorities are preferred for aggregation. Must be a number between 1 and 65535.

other_config : lacp-aggregation-key: optional string

> 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. Must be a number between 1 and 65535.

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 : 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 : xs-vm-uuid: optional string

> The VM to which this interface belongs.

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

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
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 **capabilities** column in the **Open_vSwitch** table identifies the types that a switch actually supports. 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 used by OpenFlow output actions that do not specify a specific 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

>>   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:

Configuration for min-rate QoS:

    other_config : min-rate   optional string

Configuration for linux-htb QoS:

    other_config : min-rate   optional string

    other_config : max-rate   optional string

    other_config : burst   optional string

    other_config : priority   optional string

Configuration for linux-hfsc QoS:

    other_config : min-rate   optional string

    other_config : max-rate   optional string

Common Columns:

    other_config   map of string-string pairs

    external_ids   map of string-string pairs

Details:

Configuration for min-rate QoS:

These key-value pairs are defined for **QoS type** of **min−rate**.

other_config : min-rate: optional string

    Minimum guaranteed bandwidth, in bit/s. Required. The floor value is 1500 bytes/s (12,000 bit/s).

Configuration for linux-htb QoS:

These key-value pairs are defined for **QoS type** of **linux−htb**.

other_config : min-rate: optional string

    Minimum guaranteed bandwidth, in bit/s.

other_config : max-rate: optional string

    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

    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

    A nonnegative 32-bit integer. Defaults to 0 if unspecified. 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.

Configuration for linux-hfsc QoS:

These key-value pairs are defined for **QoS type** of **linux−hfsc**.

other_config : min-rate: optional string

    Minimum guaranteed bandwidth, in bit/s.

other_config : max-rate: optional string

    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, RSPAN, or ERSPAN, 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 4096 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
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 4096 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 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 (sometimes called ERSPAN).

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.

The following destination MAC addresses will not be mirrored to a VLAN to avoid confusing switches that interpret the protocols that they represent:

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.

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.

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.

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

OpenFlow Rate Limiting:

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

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
status : sec_since_connect	optional string
status : sec_since_disconnect	optional string

Common Columns:

external_ids	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* (default: 6633) 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.

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

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

The specified TCP *port* (default: 6633) on the host at the given *ip*, which must be expressed as an IP address (not a DNS name).

The following connection methods are currently supported for service controllers:

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

Listens for SSL connections on the specified TCP *port* (default: 6633). 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.

The **ssl** column in the **Open_vSwitch** table must point to a valid SSL configuration when this form is used.

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* (default: 6633). 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.

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.

OpenFlow Rate Limiting:

controller_rate_limit: optional integer, at least 100

> The maximum rate at which packets in unknown flows will be forwarded to the OpenFlow controller, in packets per second. This feature prevents a single bridge from overwhelming the controller. If not specified, the default is implementation-specific.
>
> In addition, when a high rate triggers rate-limiting, Open vSwitch queues controller packets for each port and transmits them to the controller at the configured rate. The number of queued packets is limited by the **controller_burst_limit** value. The packet queue is shared fairly among the ports on a bridge.
>
> Open vSwitch maintains two such packet rate-limiters per bridge. One of these applies to packets sent up to the controller because they do not correspond to any flow. The other applies to packets sent up to the controller by request through flow actions. When both rate-limiters are filled with packets, the actual rate that packets are sent to the controller is up to twice the specified rate.

controller_burst_limit: optional integer, at least 25

> In conjunction with **controller_rate_limit**, the maximum number of unused packet credits that the bridge will allow to accumulate, in packets. If not specified, the default is implementation-specific.

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
The state of the connection to the controller. Possible values are:

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

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

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
status : sec_since_connect	optional string
status : sec_since_disconnect	optional string
status : locks_held	optional string
status : locks_waiting	optional string
status : locks_lost	optional string
status : n_connections	optional string

Common Columns:

external_ids	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* (default: 6632) 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.

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

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

The specified TCP *port* (default: 6632) on the host at the given *ip*, which must be expressed as an IP address (not a DNS name).

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

Listens for SSL connections on the specified TCP *port* (default: 6632). 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.

The **ssl** column in the **Open_vSwitch** table must point to a valid SSL configuration when this form is used.

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* (default: 6632). 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.

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. **strerror(errno)**. This key will exist only if an error has occurred.

status : state: optional string
> The state of the connection to the manager. Possible values are:
>
> **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

> 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

> 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

> 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.

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

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

An sFlow(R) target. 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 collectors. If not specified, the IP address defaults to the **local_ip** in the collector's **Controller**. If an agent IP address cannot be determined either way, 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 seconds.

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 document.

external_ids: map of string-string pairs

Capability TABLE

Records in this table describe functionality supported by the hardware and software platform on which this Open vSwitch is based. Clients should not modify this table.

A record in this table is meaningful only if it is referenced by the **capabilities** column in the **Open_vSwitch** table. The key used to reference it, called the record's "category," determines the meanings of the **details** column. The following general forms of categories are currently defined:

qos−*type*
> *type* is supported as the value for **type** in the **QoS** table.

Summary:

details   map of string-string pairs

Details:

details: map of string-string pairs
> Key-value pairs that describe capabilities. The meaning of the pairs depends on the category key that the **capabilities** column in the **Open_vSwitch** table uses to reference this record, as described above.
>
> The presence of a record for category **qos−***type* indicates that the switch supports *type* as the value of the **type** column in the **QoS** table. The following key-value pairs are defined to further describe QoS capabilities:

n−queues
> Number of supported queues, as a positive integer. Keys in the **queues** column for **QoS** records whose **type** value equals *type* must range between 0 and this value minus one, inclusive.