NAME

ovs-vswitchd.conf.db − 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.

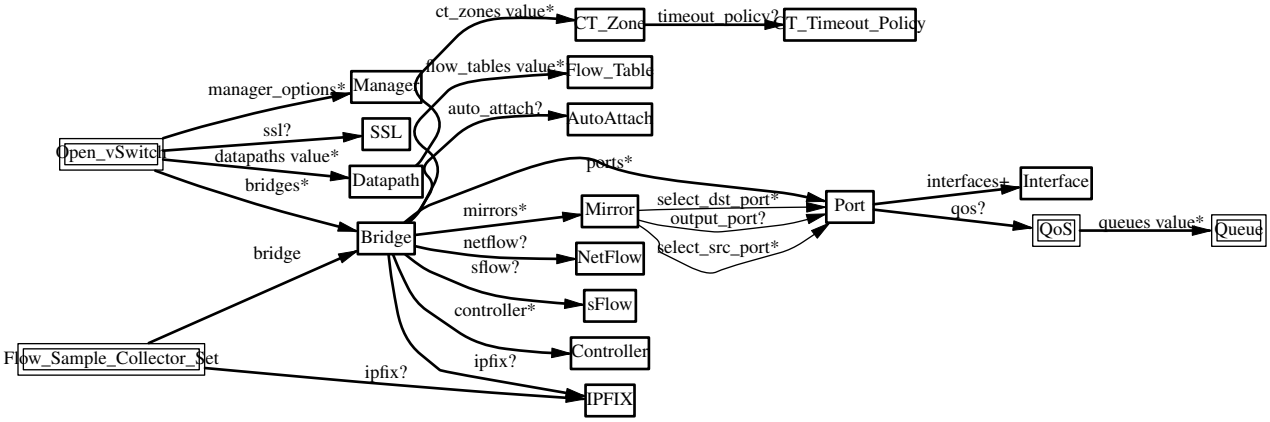
TABLE SUMMARY

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.
Datapath	Datapath configuration.
CT_Zone	CT_Zone configuration.
CT_Timeout_Policy	CT_Timeout_Policy configuration.
SSL	SSL configuration.
sFlow	sFlow configuration.
IPFIX	IPFIX configuration.
Flow_Sample_Collector_Set	Flow_Sample_Collector_Set configuration.
AutoAttach	AutoAttach 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:

datapaths	map of string-**Datapath** pairs
bridges	set of **Bridge**s
ssl	optional **SSL**
external_ids : system-id	optional string
external_ids : hostname	optional string
external_ids : rundir	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 : max-idle	optional string, containing an integer, at least 500
other_config : max-revalidator	optional string, containing an integer, at least 100
other_config : min-revalidate-pps	optional string, containing an integer, at least 0
other_config : offloaded-stats-delay	optional string, containing an integer, at least 0
other_config : hw-offload	optional string, either **true** or **false**
other_config : n-offload-threads	optional string, containing an integer, in range 1 to 10
other_config : tc-policy	optional string, one of **none**, **skip_hw**, or **skip_sw**
other_config : dpdk-init	optional string, one of **false**, **true**, or **try**
other_config : dpdk-lcore-mask	optional string, containing an integer, at least 1
other_config : pmd-cpu-mask	optional string
other_config : dpdk-alloc-mem	optional string, containing an integer, at least 0
other_config : dpdk-socket-mem	optional string
other_config : dpdk-socket-limit	optional string
other_config : dpdk-hugepage-dir	optional string
other_config : dpdk-extra	optional string
other_config : vhost-sock-dir	optional string
other_config : vhost-iommu-support	optional string, either **true** or **false**
other_config : vhost-postcopy-support	optional string, either **true** or **false**
other_config : per-port-memory	optional string, either **true** or **false**
other_config : shared-mempool-config	optional string
other_config : tx-flush-interval	optional string, containing an integer, in range 0 to 1,000,000
other_config : pmd-perf-metrics	optional string, either **true** or **false**
other_config : smc-enable	optional string, either **true** or **false**
other_config : pmd-rxq-assign	optional string, one of **cycles**, **group**, or **roundrobin**
other_config : pmd-rxq-isolate	optional string, either **true** or **false**
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
other_config : emc-insert-inv-prob	optional string, containing an integer, in range 0 to 4,294,967,295
other_config : vlan-limit	optional string, containing an integer, at least 0
other_config : bundle-idle-timeout	optional string, containing an integer, at least 1
other_config : offload-rebalance	optional string, either **true** or **false**
other_config : pmd-auto-lb	optional string, either **true** or **false**
other_config : pmd-auto-lb-rebal-interval	optional string, containing an integer, in range 0 to 20,000
other_config : pmd-auto-lb-load-threshold	optional string, containing an integer, in range 0 to 100
other_config : pmd-auto-lb-improvement-threshold	optional string, containing an integer, in range 0 to 100

other_config : pmd-sleep-max	optional string, containing an integer, in range 0 to 10,000
other_config : userspace-tso-enable	optional string, either **true** or **false**
Status:	
next_cfg	integer
cur_cfg	integer
dpdk_initialized	boolean
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
dpdk_version	optional string
Capabilities:	
datapath_types	set of strings
iface_types	set of strings
Database Configuration:	
manager_options	set of **Manager**s
IPsec:	
other_config : private_key	optional string
other_config : certificate	optional string
other_config : ca_cert	optional string
Plaintext Tunnel Policy:	
other_config : ipsec_skb_mark	optional string
Common Columns:	
other_config	map of string-string pairs
external_ids	map of string-string pairs

Details:

Configuration:

datapaths: map of string-**Datapath** pairs

Map of datapath types to datapaths. The **datapath_type** column of the **Bridge** table is used as a key for this map. The value points to a row in the **Datapath** table.

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.

external_ids : hostname: optional string

The hostname for the host running Open vSwitch. This is a fully qualified domain name since version 2.6.2.

external_ids : rundir: optional string

In Open vSwitch 2.8 and later, the run directory of the running Open vSwitch daemon. This directory is used for runtime state such as control and management sockets. The value of

other_config:vhost-sock-dir is relative to this directory.

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.

Additionally, **ovs−vswitchd** is prevented from connecting to controllers when this value is set to **true**. This prevents controllers from making changes to the flow table in the middle of flow restoration, which could result in undesirable intermediate states. Once this value has been set to **false** and the desired flow state has been restored, **ovs−vswitchd** will be able to reconnect to controllers and process any new flow table modifications.

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. Tweaking this value is discouraged unless you know exactly what you’re doing.

The default is 200000.

other_config : max-idle: optional string, containing an integer, at least 500
The maximum time (in ms) that idle flows will remain cached in the datapath. Internally OVS will check the validity and activity for datapath flows regularly and may expire flows quicker than this number, based on real time network conditions. Tweaking this value is discouraged unless you know exactly what you’re doing.

The default is 10000.

other_config : max-revalidator: optional string, containing an integer, at least 100
The maximum time (in ms) that revalidator threads will wait before executing flow revalidation. Note that this is maximum allowed value. Actual timeout used by OVS is minimum of max-idle and max-revalidator values. Tweaking this value is discouraged unless you know exactly what you’re doing.

The default is 500.

other_config : min-revalidate-pps: optional string, containing an integer, at least 0
Set minimum pps that flow must have in order to be revalidated when revalidation duration exceeds half of max-revalidator config variable. Setting to 0 means always revalidate flows regardless of pps.

The default is 5.

other_config : offloaded-stats-delay: optional string, containing an integer, at least 0
Set worst case delay (in ms) it might take before statistics of offloaded flows are updated. Offloaded flows younger than this delay will always be revalidated regardless of **other_config:min-revalidate-pps**.

The default is 2000.

other_config : hw-offload: optional string, either **true** or **false**
Set this value to **true** to enable netdev flow offload.

The default value is **false**. Changing this value requires restarting the daemon

Currently Open vSwitch supports hardware offloading on Linux systems. On other systems, this value is ignored. This functionality is considered 'experimental'. Depending on which OpenFlow matches and actions are configured, which kernel version is used, and what hardware is available, Open vSwitch may not be able to offload functionality to hardware.

In order to dump HW offloaded flows use **ovs−appctl dpctl/dump−flows**, **ovs−dpctl** doesn't support this functionality. See ovs-vswitchd(8) for details.

other_config : n-offload-threads: optional string, containing an integer, in range 1 to 10
Set this value to the number of threads created to manage hardware offloads.

The default value is **1**. Changing this value requires restarting the daemon.

This is only relevant for userspace datapath and only if **other_config:hw-offload** is enabled.

other_config : tc-policy: optional string, one of **none**, **skip_hw**, or **skip_sw**
Specified the policy used with HW offloading. Options:

none Add software rule and offload rule to HW.

skip_sw
Offload rule to HW only.

skip_hw
Add software rule without offloading rule to HW.

This is only relevant if **other_config:hw-offload** is enabled.

The default value is **none**.

other_config : dpdk-init: optional string, one of **false**, **true**, or **try**
Set this value to **true** or **try** to enable runtime support for DPDK ports. The vswitch must have compile-time support for DPDK as well.

A value of **true** will cause the ovs-vswitchd process to abort if DPDK cannot be initialized. A value of **try** will allow the ovs-vswitchd process to continue running even if DPDK cannot be initialized.

The default value is **false**. Changing this value requires restarting the daemon

If this value is **false** at startup, any dpdk ports which are configured in the bridge will fail due to memory errors.

other_config : dpdk-lcore-mask: optional string, containing an integer, at least 1
Specifies the CPU cores where dpdk lcore threads should be spawned. The DPDK lcore threads are used for DPDK library tasks, such as library internal message processing, logging, etc. Value should be in the form of a hex string (so '0x123') similar to the 'taskset' mask input.

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

For performance reasons, it is best to set this to a single core on the system, rather than allow lcore threads to float.

If not specified, the value will be determined by choosing the lowest CPU core from initial cpu affinity list. Otherwise, the value will be passed directly to the DPDK library.

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 : dpdk-alloc-mem: optional string, containing an integer, at least 0
Specifies the amount of memory to preallocate from the hugepage pool, regardless of socket. It is recommended that dpdk-socket-mem is used instead.

other_config : dpdk-socket-mem: optional string
Specifies the amount of memory to preallocate from the hugepage pool, on a per-socket basis.

The specifier is a comma-separated string, in ascending order of CPU socket. E.g. On a four socket system 1024,0,2048 would set socket 0 to preallocate 1024MB, socket 1 to preallocate 0MB, socket 2 to preallocate 2048MB and socket 3 (no value given) to preallocate 0MB.

If **other_config:dpdk-socket-mem** and **other_config:dpdk-alloc-mem** are not specified, neither will be used and there will be no default value for each numa node. DPDK defaults will be used instead. If **other_config:dpdk-socket-mem** and **other_config:dpdk-alloc-mem** are specified at the same time, **other_config:dpdk-socket-mem** will be used as default. Changing this value requires restarting the daemon.

other_config : dpdk-socket-limit: optional string
Limits the maximum amount of memory that can be used from the hugepage pool, on a per-socket basis.

The specifier is a comma-separated list of memory limits per socket. **0** will disable the limit for a particular socket.

If not specified, OVS will not configure limits by default. Changing this value requires restarting the daemon.

other_config : dpdk-hugepage-dir: optional string
Specifies the path to the hugetlbfs mount point.

If not specified, this will be guessed by the DPDK library (default is /dev/hugepages). Changing this value requires restarting the daemon.

other_config : dpdk-extra: optional string
Specifies additional eal command line arguments for DPDK.

The default is empty. Changing this value requires restarting the daemon

other_config : vhost-sock-dir: optional string
Specifies a relative path from **external_ids:rundir** to the vhost-user unix domain socket files. If this value is unset, the sockets are put directly in **external_ids:rundir**.

Changing this value requires restarting the daemon.

other_config : vhost-iommu-support: optional string, either **true** or **false**
vHost IOMMU is a security feature, which restricts the vhost memory that a virtio device may access. vHost IOMMU support is disabled by default, due to a bug in QEMU implementations of the vhost REPLY_ACK protocol, (on which vHost IOMMU relies) prior to v2.9.1. Setting this value to **true** enables vHost IOMMU support for vHost User Client ports in OvS-DPDK, starting from DPDK v17.11.

Changing this value requires restarting the daemon.

other_config : vhost-postcopy-support: optional string, either **true** or **false**
vHost post-copy is a feature which allows switching live migration of VM attached to dpdkvhostuserclient port to post-copy mode if default pre-copy migration can not be converged or takes too long to converge. Setting this value to **true** enables vHost post-copy support for all dpdkvhostuserclient ports. Available starting from DPDK v18.11 and QEMU 2.12.

Changing this value requires restarting the daemon.

other_config : per-port-memory: optional string, either **true** or **false**
By default OVS DPDK uses a shared memory model wherein devices that have the same MTU and socket values can share the same mempool. Setting this value to **true** changes this behaviour. Per port memory allow DPDK devices to use private memory per device. This can provide greater transparency as regards memory usage but potentially at the cost of greater memory requirements.

Changing this value requires restarting the daemon if dpdk-init has already been set to true.

other_config : shared-mempool-config: optional string
Specifies dpdk shared mempool config.

Value should be set in the following form:

other_config:shared−mempool−config=< user−shared−mempool−mtu−list>

where

- <user-shared-mempool-mtu-list> ::= NULL | <non-empty-list>
- <non-empty-list> ::= <user-mtus> | <user-mtus> , <non-empty-list>
- <user-mtus> ::= <mtu-all-socket> | <mtu-socket-pair>
- <mtu-all-socket> ::= <mtu>
- <mtu-socket-pair> ::= <mtu> : <socket-id>

Changing this value requires restarting the daemon if dpdk-init has already been set to true.

other_config : tx-flush-interval: optional string, containing an integer, in range 0 to 1,000,000
Specifies the time in microseconds that a packet can wait in output batch for sending i.e. amount of time that packet can spend in an intermediate output queue before sending to netdev. This option can be used to configure balance between throughput and latency. Lower values decreases latency while higher values may be useful to achieve higher performance.

Defaults to 0 i.e. instant packet sending (latency optimized).

other_config : pmd-perf-metrics: optional string, either **true** or **false**
Enables recording of detailed PMD performance metrics for analysis and trouble-shooting. This can have a performance impact in the order of 1%.

Defaults to false but can be changed at any time.

other_config : smc-enable: optional string, either **true** or **false**
Signature match cache or SMC is a cache between EMC and megaflow cache. It does not store the full key of the flow, so it is more memory efficient comparing to EMC cache. SMC is especially useful when flow count is larger than EMC capacity.

Defaults to false but can be changed at any time.

other_config : pmd-rxq-assign: optional string, one of **cycles**, **group**, or **roundrobin**
Specifies how RX queues will be automatically assigned to CPU cores. Options:

cycles Rxqs will be sorted by order of measured processing cycles before being assigned to CPU cores.

roundrobin
Rxqs will be round-robined across CPU cores.

group Rxqs will be sorted by order of measured processing cycles before being assigned to CPU cores with lowest estimated load.

The default value is **cycles**.

Changing this value will affect an automatic re-assignment of Rxqs to CPUs. Note: Rxqs mapped to CPU cores with **pmd−rxq−affinity** are unaffected.

other_config : pmd-rxq-isolate: optional string, either **true** or **false**
Specifies if a CPU core will be isolated after being pinned with an Rx queue.

Set this value to **false** to non-isolate a CPU core after it is pinned with an Rxq using **pmd−rxq−affinity**. This will allow OVS to assign other Rxqs to that CPU core.

The default value is **true**.

This can only be **false** when **pmd−rxq−assign** is set to **group**.

other_config : n-handler-threads: optional string, containing an integer, at least 1
Attempts to specify the number of threads for software datapaths to use for handling new flows. Some datapaths may choose to ignore this and it will be set to a sensible option for the datapath type.

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
Attempts to specify the number of threads for software datapaths to use for revalidating flows in the datapath. Some datapaths may choose to ignore this and will set to a sensible option for the datapath type.

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.

other_config : emc-insert-inv-prob: optional string, containing an integer, in range 0 to 4,294,967,295
Specifies the inverse probability (1/emc-insert-inv-prob) of a flow being inserted into the Exact Match Cache (EMC). On average one in every **emc−insert−inv−prob** packets that generate a unique flow will cause an insertion into the EMC. A value of 1 will result in an insertion for every flow (1/1 = 100%) whereas a value of zero will result in no insertions and essentially disable the EMC.

Defaults to 100 ie. there is (1/100 =) 1% chance of EMC insertion.

other_config : vlan-limit: optional string, containing an integer, at least 0
Limits the number of VLAN headers that can be matched to the specified number. Further VLAN headers will be treated as payload, e.g. a packet with more 802.1q headers will match Ethernet type 0x8100.

Open vSwitch userspace currently supports at most 2 VLANs, and each datapath has its own limit. If **vlan−limit** is nonzero, it acts as a further limit.

If this value is absent, the default is currently 1. This maintains backward compatibility with controllers that were designed for use with Open vSwitch versions earlier than 2.8, which only supported one VLAN.

other_config : bundle-idle-timeout: optional string, containing an integer, at least 1
The maximum time (in seconds) that idle bundles will wait to be expired since it was either opened, modified or closed.

OpenFlow specification mandates the timeout to be at least one second. The default is 10 seconds.

other_config : offload-rebalance: optional string, either **true** or **false**
Configures HW offload rebalancing, that allows to dynamically offload and un-offload flows while an offload-device is out of resources (OOR). This policy allows flows to be selected for offloading based on the packets-per-second (pps) rate of flows.

Set this value to **true** to enable this option.

The default value is **false**. Changing this value requires restarting the daemon.

This is only relevant if HW offloading is enabled (hw-offload). When this policy is enabled, it also requires 'tc-policy' to be set to 'skip_sw'.

other_config : pmd-auto-lb: optional string, either **true** or **false**
Configures PMD Auto Load Balancing that allows automatic assignment of RX queues to PMDs if any of PMDs is overloaded (i.e. a processing cycles > **other_config:pmd-auto-lb-load-threshold**).

It uses current scheme of cycle based assignment of RX queues that are not statically pinned to PMDs.

The default value is **false**.

Set this value to **true** to enable this option. It is currently disabled by default and an experimental feature.

This only comes in effect if cycle based assignment is enabled and there are more than one non-isolated PMDs present and at least one of it polls more than one queue.

other_config : pmd-auto-lb-rebal-interval: optional string, containing an integer, in range 0 to 20,000
The minimum time (in minutes) 2 consecutive PMD Auto Load Balancing iterations.

The default value is 1 min. If configured to 0 then it would be converted to default value i.e. 1 min

This option can be configured to avoid frequent trigger of auto load balancing of PMDs. For e.g. set the value (in min) such that it occurs once in few hours or a day or a week.

other_config : pmd-auto-lb-load-threshold: optional string, containing an integer, in range 0 to 100
Specifies the minimum PMD thread load threshold (% of used cycles) of any non-isolated PMD threads when a PMD Auto Load Balance may be triggered.

The default value is **95%**.

other_config : pmd-auto-lb-improvement-threshold: optional string, containing an integer, in range 0 to 100

Specifies the minimum evaluated % improvement in load distribution across the non-isolated PMD threads that will allow a PMD Auto Load Balance to occur.

Note, setting this parameter to 0 will always allow an auto load balance to occur regardless of estimated improvement or not.

The default value is **25%**.

other_config : pmd-sleep-max: optional string, containing an integer, in range 0 to 10,000
Specifies the maximum sleep time that will be requested in microseconds per iteration for a PMD thread which has received zero or a small amount of packets from the Rx queues it is polling.

The actual sleep time requested is based on the load of the Rx queues that the PMD polls and may be less than the maximum value.

The default value is **0 microseconds**, which means that the PMD will not sleep regardless of the load from the Rx queues that it polls.

The maximum value is **10000 microseconds**.

other_config : userspace-tso-enable: optional string, either **true** or **false**
Set this value to **true** to enable userspace support for TCP Segmentation Offloading (TSO). When it is enabled, the interfaces can provide an oversized TCP segment to the datapath and the datapath will offload the TCP segmentation and checksum calculation to the interfaces when necessary.

The default value is **false**. Changing this value requires restarting the daemon.

The feature only works if Open vSwitch is built with DPDK support.

The feature is considered experimental.

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.

dpdk_initialized: boolean
True if **other_config:dpdk-init** is set to true and the DPDK library is successfully initialized.

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, e.g. **1.2.3**. See ovsdb-tool(1) for an explanation of the numbering scheme.

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. **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. **4.18.0–372.19.1.el8_6** on RHEL 8.6 with kernel 4.18.0–372.19.1.

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

dpdk_version: optional string
The version of the linked DPDK library.

Capabilities:

These columns report capabilities of the Open vSwitch instance.

datapath_types: set of strings
This column reports the different dpifs registered with the system. These are the values that this instance supports in the **datapath_type** column of the **Bridge** table.

iface_types: set of strings
This column reports the different netdevs registered with the system. These are the values that this instance supports in the **type** column of the **Interface** table.

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 connections should be configured. See the **Manager** table for more information.

For this column to serve its purpose, **ovsdb−server** must be configured to honor it. The easiest way to do this is to invoke **ovsdb−server** with the option **−−remote=db:Open_vSwitch,Open_vSwitch,manager_options** The startup scripts that accompany Open vSwitch do this by default.

IPsec:

These settings control the global configuration of IPsec tunnels. The **options** column of the **Interface** table configures IPsec for individual tunnels. The **options** column also allows for custom options prefixed with **ipsec_** to be passed to the individual connections.

OVS IPsec supports the following three forms of authentication. Currently, all IPsec tunnels must use the same form:

1. Pre-shared keys: Omit the global settings. On each tunnel, set **options:psk**.
2. Self-signed certificates: Set the **private_key** and **certificate** global settings. On each tunnel, set **options:remote_cert**. The remote certificate can be self-signed.
3. CA-signed certificates: Set all of the global settings. On each tunnel, set **options:remote_name** to the common name (CN) of the remote certificate. The remote certificate

must be signed by the CA.

other_config : private_key: optional string
Name of a PEM file containing the private key used as the switch's identity for IPsec tunnels.

other_config : certificate: optional string
Name of a PEM file containing a certificate that certifies the switch's private key, and identifies a trustworthy switch for IPsec tunnels. The certificate must be x.509 version 3 and with the string in common name (CN) also set in the subject alternative name (SAN).

other_config : ca_cert: optional string
Name of a PEM file containing the CA certificate used to verify that a remote switch of the IPsec tunnel is trustworthy.

Plaintext Tunnel Policy:

When an IPsec tunnel is configured in this database, multiple independent components take responsibility for implementing it. **ovs−vswitchd** and its datapath handle packet forwarding to the tunnel and a separate daemon pushes the tunnel's IPsec policy configuration to the kernel or other entity that implements it. There is a race: if the former configuration completes before the latter, then packets sent by the local host over the tunnel can be transmitted in plaintext. Using this setting, OVS users can avoid this undesirable situation.

other_config : ipsec_skb_mark: optional string
This setting takes the form *value*/*mask*. If it is specified, then the **skb_mark** field in every outgoing tunneled packet sent in plaintext is compared against it and, if it matches, the packet is dropped. This is a global setting that is applied to every tunneled packet, regardless of whether IPsec encryption is enabled for the tunnel, the type of tunnel, or whether OVS is involved.

Example policies:

1/1 Drop all unencrypted tunneled packets in which the least-significant bit of **skb_mark** is 1. This would be a useful policy given an OpenFlow flow table that sets **skb_mark** to 1 for traffic that should be encrypted. The default **skb_mark** is 0, so this would not affect other traffic.

0/1 Drop all unencrypted tunneled packets in which the least-significant bit of **skb_mark** is 0. This would be a useful policy if no unencrypted tunneled traffic should exit the system without being specially permitted by setting **skb_mark** to 1.

(empty)
If this setting is empty or unset, then all unencrypted tunneled packets are transmitted in the usual way.

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
auto_attach	optional **AutoAttach**

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
datapath_version	string
other_config : datapath-id	optional string
other_config : dp-desc	optional string
other_config : dp-sn	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
other_config : controller-queue-size	optional string, containing an integer, in range 1 to 512
protocols	set of strings, one of **OpenFlow10**, **OpenFlow11**, **OpenFlow12**, **OpenFlow13**, **OpenFlow14**, or **OpenFlow15**

Spanning Tree Configuration:

STP 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**

STP Status:

status : stp_bridge_id	optional string
status : stp_designated_root	optional string
status : stp_root_path_cost	optional string

Rapid Spanning Tree:

RSTP 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
RSTP Status:	
rstp_status : rstp_bridge_id	optional string
rstp_status : rstp_root_id	optional string
rstp_status : rstp_root_path_cost	optional string, containing an integer, at least 0
rstp_status : rstp_designated_id	optional string
rstp_status : rstp_designated_port_id	optional string
rstp_status : rstp_bridge_port_id	optional string
Multicast Snooping Configuration:	
mcast_snooping_enable	boolean
Other Features:	
datapath_type	string
external_ids : bridge-id	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
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. Must be unique among the names of ports, interfaces, and bridges on a host.

The name must be alphanumeric and must not contain forward or backward slashes. The name of a bridge is also the name of an **Interface** (and a **Port**) within the bridge, so the restrictions on the **name** column in the **Interface** table, particularly on length, also apply to bridge names. Refer to the documentation for **Interface** names for details.

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.

auto_attach: optional **AutoAttach**
Auto Attach configuration.

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 OpenFlow flow tables, group table, and meter table. If there are no primary controllers, adding one also clears these tables. 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 these tables.

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**).

The **fail_mode** setting applies only to primary controllers. When more than one primary 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 OpenFlow flow tables, group table, and meter 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.)

datapath_version: string
Reports the datapath version. This column is maintained for backwards compatibility. The preferred locatation is the **datapath_id** column of the **Datapath** table. The full documentation for this column is there.

other_config : datapath-id: optional string
Overrides the default OpenFlow datapath ID, setting it to the specified value specified in hex. The value must either have a **0x** prefix or be exactly 16 hex digits long. May not be all-zero.

other_config : dp-desc: optional string
Human readable description of datapath. It is a maximum 256 byte-long free-form string to describe the datapath for debugging purposes, e.g. **switch3 in room 3120**. The value is returned by the switch as a part of reply to OFPMP_DESC request (ofp_desc). The OpenFlow specification (e.g. 1.3.5) describes the ofp_desc structure to contaion "NULL terminated ASCII strings". For the compatibility reasons no more than 255 ASCII characters should be used.

other_config : dp-sn: optional string
Serial number. It is a maximum 32 byte-long free-form string to provide an additional switch identification. The value is returned by the switch as a part of reply to OFPMP_DESC request (ofp_desc). Same as mentioned in the description of **other-config:dp-desc**, the string should be no more than 31 ASCII characters for the compatibility.

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.

other_config : controller-queue-size: optional string, containing an integer, in range 1 to 512
This sets the maximum size of the queue of packets that need to be sent to the OpenFlow management controller. The value must be less than 512. If not specified the queue size is limited to 100 packets by default. Note: increasing the queue size might have a negative impact on latency.

protocols: set of strings, one of **OpenFlow10**, **OpenFlow11**, **OpenFlow12**, **OpenFlow13**, **OpenFlow14**, or **OpenFlow15**
List of OpenFlow protocols that may be used when negotiating a connection with a controller. OpenFlow 1.0, 1.1, 1.2, 1.3, 1.4, and 1.5 are enabled by default if this column is empty.

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.

These settings configure the slower-to-converge but still widely supported version of Spanning Tree Protocol, sometimes known as 802.1D−1998. Open vSwitch also supports the newer Rapid Spanning Tree Protocol (RSTP), documented later in the section titled **Rapid Spanning Tree Configuration**.

STP Configuration:

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.

STP and RSTP are mutually exclusive. If both are enabled, RSTP will be used.

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.

STP Status:

These key-value pairs report the status of 802.1D−1998. They are present only if STP is enabled (via the **stp_enable** column).

status : stp_bridge_id: optional string
The bridge ID used in spanning tree advertisements, in the form *xxxx.yyyyyyyyyyyy* where the *x*s are the STP priority, the *y*s are the STP system ID, and each *x* and *y* is a hex digit.

status : stp_designated_root: optional string
The designated root for this spanning tree, in the same form as **status:stp_bridge_id**. If this bridge is the root, this will have the same value as **status:stp_bridge_id**, otherwise it will differ.

status : stp_root_path_cost: optional string
The path cost of reaching the designated bridge. A lower number is better. The value is 0 if this bridge is the root, otherwise it is higher.

Rapid Spanning Tree:

Rapid Spanning Tree Protocol (RSTP), like STP, is a network protocol that ensures loop-free topologies. RSTP superseded STP with the publication of 802.1D−2004. Compared to STP, RSTP converges more quickly and recovers more quickly from failures.

RSTP Configuration:

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.

STP and RSTP are mutually exclusive. If both are enabled, RSTP will be used.

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.

RSTP Status:

These key-value pairs report the status of 802.1D−2004. They are present only if RSTP is enabled (via the **rstp_enable** column).

rstp_status : rstp_bridge_id: optional string
The bridge ID used in rapid spanning tree advertisements, in the form *x.yyy.zzzzzzzzzzzz* where *x* is the RSTP priority, the *y*s are a locally assigned system ID extension, the *z*s are the STP system ID, and each *x*, *y*, or *z* is a hex digit.

rstp_status : rstp_root_id: optional string
The root of this spanning tree, in the same form as **rstp_status:rstp_bridge_id**. If this bridge is the root, this will have the same value as **rstp_status:rstp_bridge_id**, otherwise it will differ.

rstp_status : rstp_root_path_cost: optional string, containing an integer, at least 0
The path cost of reaching the root. A lower number is better. The value is 0 if this bridge is the root, otherwise it is higher.

rstp_status : rstp_designated_id: optional string
The RSTP designated ID, in the same form as **rstp_status:rstp_bridge_id**.

rstp_status : rstp_designated_port_id: optional string
The RSTP designated port ID, as a 4-digit hex number.

rstp_status : rstp_bridge_port_id: optional string
The RSTP bridge port ID, as a 4-digit hex number.

Multicast Snooping Configuration:

Multicast snooping (RFC 4541) monitors the Internet Group Management Protocol (IGMP) and Multicast Listener Discovery traffic between hosts and multicast routers. The switch uses what IGMP and MLD snooping learns to forward multicast traffic only to interfaces that are connected to interested receivers. Currently it supports IGMPv1, IGMPv2, IGMPv3, MLDv1 and MLDv2 protocols.

mcast_snooping_enable: boolean
Enable multicast snooping on the bridge. For now, the default is disabled.

Other Features:

datapath_type: string
Name of datapath provider. The kernel datapath has type **system**. The userspace datapath has type **netdev**. A manager may refer to the **datapath_types** column of the **Open_vSwitch** table for a list of the types accepted by this Open vSwitch instance.

external_ids : bridge-id: optional string
A unique identifier of the bridge.

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**
Controls forwarding of BPDUs and other network control frames when NORMAL action is invoked. When this option is **false** or unset, frames with reserved Ethernet addresses (see table below) will not be forwarded. When this option is **true**, such frames will not be treated specially.

The above general rule has the following exceptions:

- If STP is enabled on the bridge (see the **stp_enable** column in the **Bridge** table), the bridge processes all received STP packets and never passes them to OpenFlow or forwards them. This is true even if STP is disabled on an individual port.
- If LLDP is enabled on an interface (see the **lldp** column in the **Interface** table), the interface processes received LLDP packets and never passes them to OpenFlow or forwards them.

Set this option to **true** if the Open vSwitch bridge connects different Ethernet networks and is not configured to participate in STP.

This option affects packets with the following destination MAC addresses:

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

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

01:80:c2:00:00:0*x*
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:c*x*
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 8192. The value, if specified, is forced into a reasonable range, currently 10 to 1,000,000.

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	immutable string (must be unique within table)
interfaces	set of 1 or more **Interface**s
VLAN Configuration:	
vlan_mode	optional string, one of **access**, **dot1q−tunnel**, **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
cvlans	set of up to 4,096 integers, in range 0 to 4,095
other_config : qinq-ethtype	optional string, either **802.1ad** or **802.1q**
other_config : priority-tags	optional string, one of **always**, **if−nonzero**, or **never**
Bonding Configuration:	
bond_mode	optional string, one of **active−backup**, **balance−slb**, or **balance−tcp**
other_config : bond-hash-basis	optional string, containing an integer
other_config : lb-output-action	optional string, either **true** or **false**
other_config : bond-primary	optional string
other_config : all-members-active	optional string, either **true** or **false**
Link Failure Detection:	
other_config : bond-detect-mode	optional string, either **carrier** or **miimon**
other_config : bond-miimon-interval	optional string, containing an integer
bond_updelay	integer
bond_downdelay	integer
LACP Configuration:	
lacp	optional string, one of **active**, **off**, or **passive**
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 **fast** or **slow**
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 2,147,483,647
bond_fake_iface	boolean
Spanning Tree Protocol:	
STP 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
STP Status:	
status : stp_port_id	optional string

status : stp_state	optional string, one of **blocking**, **disabled**, **forwarding**, **learning**, or **listening**
status : stp_sec_in_state	optional string, containing an integer, at least 0
status : stp_role	optional string, one of **alternate**, **designated**, or **root**
Rapid Spanning Tree Protocol:	
RSTP 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-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**
RSTP Status:	
rstp_status : rstp_port_id	optional string
rstp_status : rstp_port_role	optional string, one of **Alternate**, **Backup**, **Designated**, **Disabled**, or **Root**
rstp_status : rstp_port_state	optional string, one of **Disabled**, **Discarding**, **Forwarding**, or **Learning**
rstp_status : rstp_designated_bridge_id	optional string
rstp_status : rstp_designated_port_id	optional string
rstp_status : rstp_designated_path_cost	optional string, containing an integer
RSTP Statistics:	
rstp_statistics : rstp_tx_count	optional integer
rstp_statistics : rstp_rx_count	optional integer
rstp_statistics : rstp_error_count	optional integer
rstp_statistics : rstp_uptime	optional integer
Multicast Snooping:	
other_config : mcast-snooping-flood	optional string, either **true** or **false**
other_config : mcast-snooping-flood-reports	optional string, either **true** or **false**
Other Features:	
qos	optional **QoS**
mac	optional string
fake_bridge	boolean
protected	boolean
external_ids : fake-bridge-*	optional string
other_config : transient	optional string, either **true** or **false**
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. For a non-bonded port, this should be the same as its interface's name. Port names must otherwise be unique among the names of ports, interfaces, and bridges on a host. Because port and interfaces names are usually the same, the restrictions on the **name** column in the **Interface** table, particularly on length, also apply to port names. Refer to the documentation for

Interface names for details.

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:

In short, a VLAN (short for ''virtual LAN'') is a way to partition a single switch into multiple switches. VLANs can be confusing, so for an introduction, please refer to the question ''What's a VLAN?'' in the Open vSwitch FAQ.

A VLAN is sometimes encoded into a packet using a 802.1Q or 802.1ad VLAN header, but every packet is part of some VLAN whether or not it is encoded in the packet. (A packet that appears to have no VLAN is part of VLAN 0, by default.) As a result, it's useful to think of a VLAN as a metadata property of a packet, separate from how the VLAN is encoded. For a given port, this column determines how the encoding of a packet that ingresses or egresses the port maps to the packet's VLAN. When a packet enters the switch, its VLAN is determined based on its setting in this column and its VLAN headers, if any, and then, conceptually, the VLAN headers are then stripped off. Conversely, when a packet exits the switch, its VLAN and the settings in this column determine what VLAN headers, if any, are pushed onto the packet before it egresses the port.

The VLAN configuration in this column affects Open vSwitch only when it is doing ''normal switching.'' It does not affect flows set up by an OpenFlow controller, outside of the OpenFlow ''normal action.''

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.

dot1q-tunnel
A dot1q-tunnel port is somewhat like an access port. Like an access port, it carries packets on the single VLAN specified in the **tag** column and this VLAN, called the service VLAN, does not appear in an 802.1Q header for packets that ingress or egress on the port. The main difference lies in the behavior when packets that include a 802.1Q header ingress on the port. Whereas an access port drops such packets, a dot1q-tunnel port treats these as double-tagged with the outer service VLAN **tag** and the inner customer VLAN taken from the 802.1Q header. Correspondingly, to egress on the port, a packet outer VLAN (or only VLAN) must be **tag**, which is removed before egress, which exposes the inner (customer) VLAN if one is present.

If **cvlans** is set, only allows packets in the specified customer VLANs.

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**, **dot1q−tunnel**, **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.

cvlans: set of up to 4,096 integers, in range 0 to 4,095
For a dot1q-tunnel port, the customer VLANs that this port includes. If this is empty, the port includes all customer VLANs.

For other kinds of ports, this setting is ignored.

other_config : qinq-ethtype: optional string, either **802.1ad** or **802.1q**
For a dot1q-tunnel port, this is the TPID for the service tag, that is, for the 802.1Q header that contains the service VLAN ID. Because packets that actually ingress and egress a dot1q-tunnel port do not include an 802.1Q header for the service VLAN, this does not affect packets on the dot1q-tunnel port itself. Rather, it determines the service VLAN for a packet that ingresses on a dot1q-tunnel port and egresses on a trunk port.

The value **802.1ad** specifies TPID 0x88a8, which is also the default if the setting is omitted. The value **802.1q** specifies TPID 0x8100.

For other kinds of ports, this setting is ignored.

other_config : priority-tags: optional string, one of **always**, **if−nonzero**, or **never**
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 **if−nonzero** to enable priority-tagged frames on a port.

For **if−nonzero** Open vSwitch omits the 802.1Q header on output if both the VLAN ID and priority would be zero. Set to **always** to retain the 802.1Q header in such frames as well.

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 members based on source MAC address and output VLAN, with periodic rebalancing as traffic patterns change.

active−backup
Assigns all flows to one member, failing over to a backup member when the active member 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 members based on L3 and L4 protocol information such as IP addresses and TCP/UDP ports.

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

bond_mode: optional string, one of **active−backup**, **balance−slb**, or **balance−tcp**
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 members in load balanced bonds. When changed, all flows will be assigned different hash values possibly causing member selection decisions to change. Does not affect bonding modes which do not employ load balancing such as **active−backup**.

other_config : lb-output-action: optional string, either **true** or **false**
Enable/disable usage of optimized **lb_output** action for balancing flows among output members in load balanced bonds in **balance−tcp**. When enabled, it uses optimized path for balance-tcp mode by using rss hash and avoids recirculation. This knob does not affect other balancing modes.

other_config : bond-primary: optional string
If a slave interface with this name exists in the bond and is up, it will be made active. Relevant only when **other_config:bond_mode** is **active−backup** or if **balance−tcp** falls back to **active−backup** (e.g., LACP negotiation fails and **other_config:lacp-fallback-ab** is **true**).

other_config : all-members-active: optional string, either **true** or **false**
Enable/Disable delivery of broadcast/multicast packets on secondary interface of a balance-slb bond. Relevant only when **lacp** is **off**.

This parameter is identical to **all_slaves_active** for Linux kernel bonds. Disabled by default as it is not a desirable configuration for most users.

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 **carrier** or **miimon**
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**, **off**, or **passive**
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 **fast** or **slow**
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 2,147,483,647
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 Protocol:

The configuration here is only meaningful, and the status is only populated, when 802.1D−1998 Spanning Tree Protocol is enabled on the port's **Bridge** with its **stp_enable** column.

STP Configuration:

other_config : stp-enable: optional string, either **true** or **false**
When STP is enabled on a bridge, it is enabled by default on all of the bridge's ports except bond, internal, and mirror ports (which do not work with STP). If this column's value is **false**, STP 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.

STP Status:

status : stp_port_id: optional string
The port ID used in spanning tree advertisements for this port, as 4 hex digits. 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 **blocking**, **disabled**, **forwarding**, **learning**, or **listening**
STP state of the port.

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

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

Rapid Spanning Tree Protocol:

The configuration here is only meaningful, and the status and statistics are only populated, when 802.1D−1998 Spanning Tree Protocol is enabled on the port's **Bridge** with its **stp_enable** column.

RSTP Configuration:

other_config : rstp-enable: optional string, either **true** or **false**
When RSTP is enabled on a bridge, it is enabled by default on all of the bridge's ports except bond, internal, and mirror ports (which do not work with RSTP). If this column's value is **false**, RSTP 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-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**.

RSTP Status:

rstp_status : rstp_port_id: optional string
The port ID used in spanning tree advertisements for this port, as 4 hex digits. Configuring the port ID is described in the **rstp−port−num** and **rstp−port−priority** keys of the **other_config** section earlier.

rstp_status : rstp_port_role: optional string, one of **Alternate**, **Backup**, **Designated**, **Disabled**, or **Root**
RSTP role of the port.

rstp_status : rstp_port_state: optional string, one of **Disabled**, **Discarding**, **Forwarding**, or **Learning**
RSTP state of the port.

rstp_status : rstp_designated_bridge_id: optional string
The port's RSTP designated bridge ID, in the same form as **rstp_status:rstp_bridge_id** in the **Bridge** table.

rstp_status : rstp_designated_port_id: optional string
The port's RSTP designated port ID, as 4 hex digits.

rstp_status : rstp_designated_path_cost: optional string, containing an integer
The port's RSTP designated path cost. Lower is better.

RSTP Statistics:

rstp_statistics : rstp_tx_count: optional integer
Number of RSTP BPDUs transmitted through this port.

rstp_statistics : rstp_rx_count: optional integer
Number of valid RSTP BPDUs received by this port.

rstp_statistics : rstp_error_count: optional integer
Number of invalid RSTP BPDUs received by this port.

rstp_statistics : rstp_uptime: optional integer
The duration covered by the other RSTP statistics, in seconds.

Multicast Snooping:

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

other_config : mcast-snooping-flood-reports: optional string, either **true** or **false**
If set to **true**, multicast Reports 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.

protected: boolean
The protected ports feature allows certain ports to be designated as protected. Traffic between protected ports is blocked. Protected ports can send traffic to unprotected ports. Unprotected ports can send traffic to any port. Default is false.

external_ids : fake-bridge-*: 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−bridge−id**.

other_config : transient: optional string, either **true** or **false**
If set to **true**, the port will be removed when **ovs−ctl start −−delete−transient−ports** is used.

bond_active_slave: optional string
For a bonded port, record the MAC address of the current active member.

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 : dst_port	optional string
options : key	optional string
options : tos	optional string
options : ttl	optional string
options : df_default	optional string, either **true** or **false**
options : egress_pkt_mark	optional string
Tunnel Options: lisp only:	
options : packet_type	optional string, either **legacy_l3** or **ptap**
Tunnel Options: vxlan only:	
options : exts	optional string
options : packet_type	optional string, one of **legacy_l2**, **legacy_l3**, or **ptap**
Tunnel Options: gre only:	
options : packet_type	optional string, one of **legacy_l2**, **legacy_l3**, or **ptap**
options : seq	optional string, either **true** or **false**
Tunnel Options: gre, ip6gre, geneve, bareudp and vxlan:	
options : csum	optional string, either **true** or **false**
Tunnel Options: IPsec:	
options : psk	optional string
options : remote_cert	optional string
options : remote_name	optional string
Tunnel Options: erspan only:	
options : erspan_idx	optional string
options : erspan_ver	optional string
options : erspan_dir	optional string
options : erspan_hwid	optional string
Tunnel Options: Bareudp only:	
options : payload_type	optional string
Tunnel Options: srv6 only:	
options : srv6_segs	optional string
options : srv6_flowlabel	optional string, one of **compute**, **copy**, or **zero**
Patch Options:	
options : peer	optional string
PMD (Poll Mode Driver) Options:	
options : n_rxq	optional string, containing an integer, at least 1

options : dpdk-devargs	optional string
other_config : pmd-rxq-affinity	optional string
options : xdp-mode	optional string, one of **best−effort**, **generic**, **native−with−zerocopy**, or **native**
options : use-need-wakeup	optional string, either **true** or **false**
options : vhost-server-path	optional string
options : tx-retries-max	optional string, containing an integer, in range 0 to 32
options : n_rxq_desc	optional string, containing an integer, in range 1 to 4,096
options : n_txq_desc	optional string, containing an integer, in range 1 to 4,096
options : dpdk-vf-mac	optional string
options : rx-steering	optional string, either **rss+lacp** or **rss**
other_config : tx-steering	optional string, either **hash** or **thread**
EMC (Exact Match Cache) Configuration:	
other_config : emc-enable	optional string, either **true** or **false**
MTU:	
mtu	optional integer
mtu_request	optional integer, at least 1
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**
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**
dpdk:	
status : port_no	optional string
status : numa_id	optional string
status : min_rx_bufsize	optional string
status : max_rx_pktlen	optional string
status : max_rx_queues	optional string
status : max_tx_queues	optional string
status : max_mac_addrs	optional string
status : max_hash_mac_addrs	optional string
status : max_vfs	optional string
status : max_vmdq_pools	optional string
status : if_type	optional string
status : if_descr	optional string
status : bus_info	optional string
status : dpdk-vf-mac	optional string
status : rx_steering	optional string
status : rx_steering_queue	optional string
status : rss_queues	optional string
dpdkvhostuser:	
status : mode	optional string

status : features	optional string
status : num_of_vrings	optional string
status : numa	optional string
status : socket	optional string
status : status	optional string
status : vring_n_size	optional string
status : userspace-tso	optional string

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_kpkts_rate	integer, at least 0
ingress_policing_burst	integer, at least 0
ingress_policing_kpkts_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 : oam	optional string
bfd : mult	optional string, containing an integer, in range 1 to 255

BFD Status:

bfd_status : state	optional string, one of **admin_down**, **down**, **init**, or **up**
bfd_status : forwarding	optional string, either **true** or **false**
bfd_status : diagnostic	optional string
bfd_status : remote_state	optional string, one of **admin_down**, **down**, **init**, or **up**
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 : vm-id	optional string
Auto Attach Configuration:	
lldp : enable	optional string, either **true** or **false**
Flow control Configuration:	
options : rx-flow-ctrl	optional string, either **true** or **false**
options : tx-flow-ctrl	optional string, either **true** or **false**
options : flow-ctrl-autoneg	optional string, either **true** or **false**
Link State Change detection mode:	
options : dpdk-lsc-interrupt	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. For non-bonded port, this should be the same as the port name. It must otherwise be unique among the names of ports, interfaces, and bridges on a host.

The maximum length of an interface name depends on the underlying datapath:

- The names of interfaces implemented as Linux and BSD network devices, including interfaces with type **internal**, **tap**, or **system** plus the different types of tunnel ports, are

limited to 15 bytes. Windows limits these names to 255 bytes.

- The names of patch ports are not used in the underlying datapath, so operating system restrictions do not apply. Thus, they may have arbitrary length.

Regardless of other restrictions, OpenFlow only supports 15-byte names, which means that **ovs−ofctl** and OpenFlow controllers will show names truncated to 15 bytes.

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 member 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. This option only affects internal ports. For other type ports, you can change the MAC address outside Open vSwitch, using ip command.

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 OpenFlow 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. The types supported by a particular instance of Open vSwitch are listed in the **iface_types** column in the **Open_vSwitch** table. The following types are defined:

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.

Open vSwitch checks the interface state before send packets to the device. When it is **down**, the packets are dropped and the tx_dropped statistic is updated accordingly. Older versions of Open vSwitch did not check the interface state and then the tx_packets was incremented along with tx_dropped.

geneve An Ethernet over Geneve (**http://tools.ietf.org/html/draft−ietf−nvo3−geneve**) IPv4/IPv6 tunnel. A description of how to match and set Geneve options can be found in the **ovs−ofctl** manual page.

gre Generic Routing Encapsulation (GRE) over IPv4 tunnel, configurable to encapsulate layer 2 or layer 3 traffic.

ip6gre Generic Routing Encapsulation (GRE) over IPv6 tunnel, encapsulate layer 2 traffic.

vxlan An Ethernet tunnel over the UDP-based VXLAN protocol described in RFC 7348.

Open vSwitch uses IANA-assigned 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.

stt The Stateless TCP Tunnel (STT) is particularly useful when tunnel endpoints are in end-systems, as it utilizes the capabilities of standard network interface cards to improve performance. STT utilizes a TCP-like header inside the IP header. It is stateless, i.e., there is no TCP connection state of any kind associated with the tunnel. The TCP-like header is used to leverage the capabilities of existing network interface cards, but should not be interpreted as implying any sort of connection state between endpoints. Since the STT protocol does not engage in the usual TCP 3-way handshake, so it will have difficulty traversing stateful firewalls. The protocol is documented at https://tools.ietf.org/html/draft-davie-stt All traffic uses a default destination port of 7471.

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

gtpu GPRS Tunneling Protocol (GTP) is a group of IP-based communications protocols used to carry general packet radio service (GPRS) within GSM, UMTS and LTE networks. GTP-U is used for carrying user data within the GPRS core network and between the radio access network and the core network. The user data transported can be packets in any

of IPv4, IPv6, or PPP formats.

The protocol is documented at http://www.3gpp.org/DynaReport/29281.htm

Open vSwitch uses UDP destination port 2152. The source port used for GTP traffic varies on a per-flow basis and is in the ephemeral port range.

Bareudp
The Bareudp tunnel provides a generic L3 encapsulation support for tunnelling different L3 protocols like MPLS, IP, NSH etc. inside a UDP tunnel.

srv6 Segment Routing IPv6 (SRv6) tunnel encapsulates L3 traffic as "IPv6 in IPv6" or "IPv4 in IPv6" with Segment Routing Header (SRH) defined in RFC 8754. The segment list in SRH can be set using a SRv6 specific option.

Tunnel Options:

These options apply to interfaces with **type** of **geneve**, **bareudp**, **gre**, **ip6gre**, **vxlan**, **lisp**, **stt** and **srv6**.

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:in_key** is not applicable for bareudp and srv6 tunnels. Hence it is not considered while identifying bareudp or srv6 tunnels.

options : remote_ip: optional string
Required. The remote tunnel endpoint, one of:

- An IPv4 or IPv6 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** or **tun_ipv6_src**field. When sending packets to a **remote_ip=flow** tunnel, the flow actions must explicitly set the **tun_dst** or **tun_ipv6_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/IPv6 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** or **tun_ipv6_dst** field. When sending packets to a **local_ip=flow** tunnel, the flow actions may explicitly set the **tun_src** or **tun_ipv6_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** or **tun_ipv6_dst** field for matching in the flow table.

options : in_key: optional string
Optional, not applicable for **bareudp** and **srv6**. 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 STT) 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−fields**(7) manual page contains additional information about matching fields in OpenFlow flows.

options : out_key: optional string
Optional, not applicable for **bareudp** and **srv6**. 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 STT) 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−fields**(7) manual page contains additional information about the Nicira OpenFlow vendor extensions.

options : dst_port: optional string
Optional. The tunnel transport layer destination port, for UDP and TCP based tunnel protocols (Geneve, VXLAN, LISP, and STT).

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.

options : egress_pkt_mark: optional string
Optional. The pkt_mark to be set on the encapsulating packet. This option sets packet mark for the tunnel endpoint for all tunnel packets including tunnel monitoring.

Tunnel Options: lisp only:

options : packet_type: optional string, either **legacy_l3** or **ptap**
A LISP tunnel sends and receives only IPv4 and IPv6 packets. This option controls what how the tunnel represents the packets that it sends and receives:

- By default, or if this option is **legacy_l3**, the tunnel represents packets as Ethernet frames for compatibility with legacy OpenFlow controllers that expect this behavior.
- If this option is **ptap**, the tunnel represents packets using the **packet_type** mechanism introduced in OpenFlow 1.5.

Tunnel Options: vxlan only:

options : exts: optional string
Optional. Comma separated list of optional VXLAN extensions to enable. The following extensions are supported:

- **gbp**: VXLAN-GBP allows to transport the group policy context of a packet across the VXLAN tunnel to other network peers. See the description of **tun_gbp_id** and **tun_gbp_flags** in **ovs−fields**(7) for additional information. (**https://tools.ietf.org/html/draft−smith−vxlan−group−policy**)
- **gpe**: Support for Generic Protocol Encapsulation in accordance with IETF draft **https://tools.ietf.org/html/draft−ietf−nvo3−vxlan−gpe**. Without this option, a VXLAN packet always encapsulates an Ethernet frame. With this option, an VXLAN packet may also encapsulate an IPv4, IPv6, NSH, or MPLS packet.

options : packet_type: optional string, one of **legacy_l2**, **legacy_l3**, or **ptap**
This option controls what types of packets the tunnel sends and receives and how it represents them:

- By default, or if this option is **legacy_l2**, the tunnel sends and receives only Ethernet frames.
- If this option is **legacy_l3**, the tunnel sends and receives only non-Ethernet (L3) packet, but the packets are represented as Ethernet frames for compatibility with legacy OpenFlow controllers that expect this behavior. This requires enabling **gpe** in **options:exts**.
- If this option is **ptap**, Open vSwitch represents packets in the tunnel using the **packet_type** mechanism introduced in OpenFlow 1.5. This mechanism supports any kind of packet, but actually sending and receiving non-Ethernet packets requires additionally enabling **gpe** in **options:exts**.

Tunnel Options: gre only:

gre interfaces support these options.

options : packet_type: optional string, one of **legacy_l2**, **legacy_l3**, or **ptap**
This option controls what types of packets the tunnel sends and receives and how it represents them:

- By default, or if this option is **legacy_l2**, the tunnel sends and receives only Ethernet frames.
- If this option is **legacy_l3**, the tunnel sends and receives only non-Ethernet (L3) packet, but the packets are represented as Ethernet frames for compatibility with legacy OpenFlow controllers that expect this behavior.
- The **legacy_l3** option is only available via the user space datapath. The OVS kernel datapath does not support devices of type ARPHRD_IPGRE which is the requirement for **legacy_l3** type packets.
- If this option is **ptap**, the tunnel sends and receives any kind of packet. Open vSwitch represents packets in the tunnel using the **packet_type** mechanism introduced in OpenFlow 1.5.

options : seq: optional string, either **true** or **false**
Optional. A 4-byte sequence number field for GRE tunnel only. Default is disabled, set to **true** to enable. Sequence number is incremented by one on each outgoing packet.

Tunnel Options: gre, ip6gre, geneve, bareudp and vxlan:

gre, **ip6gre**, **geneve**, **bareudp** and **vxlan** interfaces support these options.

options : csum: optional string, either **true** or **false**
Optional. Compute encapsulation header (either GRE or UDP) checksums on outgoing packets. Default is disabled, set to **true** to enable. Checksums present on incoming packets will be validated regardless of this setting.

When using the upstream Linux kernel module, computation of checksums for **geneve** and **vxlan** requires Linux kernel version 4.0 or higher. **gre** and **ip6gre** support checksums for all versions of Open vSwitch that support GRE. The out of tree kernel module distributed as part of OVS can

compute all tunnel checksums on any kernel version that it is compatible with.

Tunnel Options: IPsec:

Setting any of these options enables IPsec support for a given tunnel. **gre**, **geneve**, **vxlan** and **stt** interfaces support these options. See the **IPsec** section in the **Open_vSwitch** table for a description of each mode.

options : psk: optional string
In PSK mode only, the preshared secret to negotiate tunnel. This value must match on both tunnel ends.

options : remote_cert: optional string
In self-signed certificate mode only, name of a PEM file containing a certificate of the remote switch. The certificate must be x.509 version 3 and with the string in common name (CN) also set in the subject alternative name (SAN).

options : remote_name: optional string
In CA-signed certificate mode only, common name (CN) of the remote certificate.

Tunnel Options: erspan only:

Only **erspan** interfaces support these options.

options : erspan_idx: optional string
20 bit index/port number associated with the ERSPAN traffic's source port and direction (ingress/egress). This field is platform dependent.

options : erspan_ver: optional string
ERSPAN version: 1 for version 1 (type II) or 2 for version 2 (type III).

options : erspan_dir: optional string
Specifies the ERSPAN v2 mirrored traffic's direction. 1 for egress traffic, and 0 for ingress traffic.

options : erspan_hwid: optional string
ERSPAN hardware ID is a 6-bit unique identifier of an ERSPAN v2 engine within a system.

Tunnel Options: Bareudp only:

options : payload_type: optional string
Specifies the ethertype of the l3 protocol the bareudp device is tunnelling. For the tunnels which supports multiple ethertypes of a l3 protocol (IP, MPLS) this field specifies the protocol name as a string.

Tunnel Options: srv6 only:

options : srv6_segs: optional string
Specifies the segment list in Segment Routing Header (SRH). It consists of a comma-separated list of segments represented in IPv6 format, e.g. "fc00:100::1,fc00:200::1,fc00:300::1". Note that the first segment must be the same as **options:remote_ip**.

options : srv6_flowlabel: optional string, one of **compute**, **copy**, or **zero**
Optional. This option controls how flowlabel in outer IPv6 header is configured. It gives the benefit of IPv6 flow label based load balancing, which is supported by some popular vendor appliances. Like net.ipv6.seg6_flowlabel sysconfig, it is one of the three values below:

- By default, or if this option is **copy**, copy the flowlabel of inner IPv6 header to the flowlabel of outer IPv6 header. If inner header is not IPv6, it is set to 0.
- If this option is **zero**, simply set flowlabel to 0.
- If this option is **compute**, set flowlabel to a hash over the L3/L4 fields of the inner packet.

Patch Options:

These options apply only to *patch ports*, that is, interfaces whose **type** column is **patch**. Patch ports are mainly a way to connect otherwise independent bridges to one another, similar to how one might plug an Ethernet cable (a ‘‘patch cable’’) into two physical switches to connect those switches. The effect of

plugging a patch port into two switches is conceptually similar to that of plugging the two ends of a Linux **veth** device into those switches, but the implementation of patch ports makes them much more efficient.

Patch ports may connect two different bridges (the usual case) or the same bridge. In the latter case, take special care to avoid loops, e.g. by programming appropriate flows with OpenFlow. Patch ports do not work if its ends are attached to bridges on different datapaths, e.g. to connect bridges in **system** and **netdev** datapaths.

The following command creates and connects patch ports **p0** and **p1** and adds them to bridges **br0** and **br1**, respectively:

**ovs−vsctl add−port br0 p0 −− set Interface p0 type=patch options:peer=p1 **
−− add−port br1 p1 −− set Interface p1 type=patch options:peer=p0

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.

PMD (Poll Mode Driver) Options:

Only PMD netdevs support these options.

options : n_rxq: optional string, containing an integer, at least 1
Specifies the maximum number of rx queues to be created for PMD netdev. If not specified or specified to 0, one rx queue will be created by default. Not supported by DPDK vHost interfaces.

options : dpdk-devargs: optional string
Specifies the PCI address associated with the port for physical devices, or the virtual driver to be used for the port when a virtual PMD is intended to be used. For the latter, the argument string typically takes the form of **eth_***driver_namex*, where *driver_name* is a valid virtual DPDK PMD driver name and *x* is a unique identifier of your choice for the given port. Only supported by the dpdk port type.

other_config : pmd-rxq-affinity: optional string
Specifies mapping of RX queues of this interface to CPU cores.

Value should be set in the following form:

other_config:pmd−rxq−affinity=<rxq−affinity−list>

where

- <rxq-affinity-list> ::= NULL | <non-empty-list>
- <non-empty-list> ::= <affinity-pair> | <affinity-pair> , <non-empty-list>
- <affinity-pair> ::= <queue-id> : <core-id>

options : xdp-mode: optional string, one of **best−effort**, **generic**, **native−with−zerocopy**, or **native**
Specifies the operational mode of the XDP program.

In **native−with−zerocopy** mode the XDP program is loaded into the device driver with zero-copy RX and TX enabled. This mode requires device driver support and has the best performance because there should be no copying of packets.

native is the same as **native−with−zerocopy**, but without zero-copy capability. This requires at least one copy between kernel and the userspace. This mode also requires support from device driver.

In **generic** case the XDP program in kernel works after skb allocation on early stages of packet processing inside the network stack. This mode doesn't require driver support, but has much lower performance.

best−effort tries to detect and choose the best (fastest) from the available modes for current interface.

Note that this option is specific to netdev-afxdp. Defaults to **best−effort** mode.

options : use-need-wakeup: optional string, either **true** or **false**
Specifies whether to use need_wakeup feature in afxdp netdev. If enabled, OVS explicitly wakes up the kernel RX, using poll() syscall and wakes up TX, using sendto() syscall. For physical devices, this feature improves the performance by avoiding unnecessary sendto syscalls. Defaults to true if supported by libbpf.

options : vhost-server-path: optional string
The value specifies the path to the socket associated with a vHost User client mode device that has been or will be created by QEMU. Only supported by dpdkvhostuserclient interfaces.

options : tx-retries-max: optional string, containing an integer, in range 0 to 32
The value specifies the maximum amount of vhost tx retries that can be made while trying to send a batch of packets to an interface. Only supported by dpdkvhostuserclient interfaces.

Default value is 8.

options : n_rxq_desc: optional string, containing an integer, in range 1 to 4,096
Specifies the rx queue size (number rx descriptors) for dpdk ports. The value must be a power of 2, less than 4096 and supported by the hardware of the device being configured. If not specified or an incorrect value is specified, 2048 rx descriptors will be used by default.

options : n_txq_desc: optional string, containing an integer, in range 1 to 4,096
Specifies the tx queue size (number tx descriptors) for dpdk ports. The value must be a power of 2, less than 4096 and supported by the hardware of the device being configured. If not specified or an incorrect value is specified, 2048 tx descriptors will be used by default.

options : dpdk-vf-mac: optional string
Ethernet address to set for this VF interface. If unset then the default MAC address is used:

- For most drivers, the default MAC address assigned by their hardware.
- For bifurcated drivers, the MAC currently used by the kernel netdevice.

This option may only be used with dpdk VF representors.

options : rx-steering: optional string, either **rss+lacp** or **rss**
Configure hardware Rx queue steering policy.

This option takes one of the following values:

rss Distribution of ingress packets in all Rx queues according to the RSS algorithm. This is the default behaviour.

rss+lacp
Distribution of ingress packets according to the RSS algorithm on all but the last Rx queue. An extra Rx queue is allocated for LACP packets.

If the user has already configured multiple **options:n_rxq** on the port, an additional one will be allocated for the specified protocols. Even if the hardware cannot satisfy the requested number of requested Rx queues, the last Rx queue will be used. If only one Rx queue is available or if the hardware does not support the rte_flow matchers/actions required to redirect the selected protocols, custom **rx−steering** will fall back to default **rss** mode.

This feature is mutually exclusive with **other_config:hw-offload** as it may conflict with the offloaded flows. If both are enabled, **rx−steering** will fall back to default **rss** mode.

This option is only applicable to interfaces with type **dpdk**.

other_config : tx-steering: optional string, either **hash** or **thread**
Specifies the Tx steering mode for the interface.

thread enables static (1:1) thread-to-txq mapping when the number of Tx queues is greater than number of PMD threads, and dynamic (N:1) mapping if equal or lower. In this mode a single thread can not use more than 1 transmit queue of a given port.

hash enables hash-based Tx steering, which distributes the packets on all the transmit queues based on their 5-tuples hashes.

Defaults to **thread**.

EMC (Exact Match Cache) Configuration:

These settings controls behaviour of EMC lookups/insertions for packets received from the interface.

other_config : emc-enable: optional string, either **true** or **false**
Specifies if Exact Match Cache (EMC) should be used while processing packets received from this interface. If true, **other_config:emc-insert-inv-prob** will have effect on this interface.

Defaults to true.

MTU:

The MTU (maximum transmission unit) is 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.

A client may change an interface MTU by filling in **mtu_request**. Open vSwitch then reports in **mtu** the currently configured value.

mtu: optional integer
The currently configured MTU for the interface.

This column will be empty for an interface that does not have an MTU as, for example, some kinds of tunnels do not.

Open vSwitch sets this column's value, so other clients should treat it as read-only.

mtu_request: optional integer, at least 1
Requested MTU (Maximum Transmission Unit) for the interface. A client can fill this column to change the MTU of an interface.

RFC 791 requires every internet module to be able to forward a datagram of 68 octets without further fragmentation. The maximum size of an IP packet is 65535 bytes.

If this is not set and if the interface has **internal** type, Open vSwitch will change the MTU to match the minimum of the other interfaces in the bridge.

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.

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/IPv6 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**.

dpdk:

DPDK specific interface status options.

status : port_no: optional string
DPDK port ID.

status : numa_id: optional string
NUMA socket ID to which an Ethernet device is connected.

status : min_rx_bufsize: optional string
Minimum size of RX buffer.

status : max_rx_pktlen: optional string
Maximum configurable length of RX pkt.

status : max_rx_queues: optional string
Maximum number of RX queues.

status : max_tx_queues: optional string
Maximum number of TX queues.

status : max_mac_addrs: optional string
Maximum number of MAC addresses.

status : max_hash_mac_addrs: optional string
Maximum number of hash MAC addresses for MTA and UTA.

status : max_vfs: optional string
Maximum number of hash MAC addresses for MTA and UTA. Maximum number of VFs.

status : max_vmdq_pools: optional string
Maximum number of VMDq pools.

status : if_type: optional string
Interface type ID according to IANA ifTYPE MIB definitions.

status : if_descr: optional string
Interface description string.

status : bus_info: optional string
Bus name and bus info such as Vendor ID and Device ID of PCI device.

status : dpdk-vf-mac: optional string
Ethernet address set for this VF interface. Only reported for dpdk VF representors.

status : rx_steering: optional string
Hardware Rx queue steering policy in use.

status : rx_steering_queue: optional string
ID of rx steering queue. Only reported if **rx−steering** is supported by hardware.

status : rss_queues: optional string
IDs of rss queues. Only reported if **rx−steering** is supported by hardware.

dpdkvhostuser:

dpdkvhostuser and dpdkvhostuserclient netdev specific interface status information.

status : mode: optional string
client (connecting) or server (listening) in the socket communication.

status : features: optional string
virtio features bitmap as per virtio specification.

status : num_of_vrings: optional string
The number of available virtqueues.

status : numa: optional string
The numa id of the device and guest memory.

status : socket: optional string
The path to the socket used for communication.

status : status: optional string
Status of connection to the device.

status : vring_n_size: optional string
Each virtqueue will have it's size reported, where n is the virtqueue number from 0..(num_of_vrings−1).

status : userspace-tso: optional string
Whether userspace-tso is enabled or disabled.

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 settings can be set with byte rate or packet rate, and they can be configured together, in which case they take effect together, that means the smaller speed limit of them is in effect.

Currently, byte rate policing is implemented on Linux and OVS with DPDK, while packet rate policing is only implemented on Linux. Both Linux and OVS DPDK implementations use a simple ‘‘token bucket’’ approach.

Byte rate policing:

- 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**.

Packet rate policing:

- The size of the bucket corresponds to **ingress_policing_kpkts_burst**. Initially the bucket is full.
- Whenever a packet is received, it will consume one token from the current bucket. If the token is available in the bucket, it's 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_kpkts_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_kpkts_rate: integer, at least 0
: Maximum rate for data received on this interface, in kpps (1 kpps is 1000 pps). 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 8000 kbit. 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 80% of **ingress_policing_rate** helps TCP come closer to achieving the full rate.

ingress_policing_kpkts_burst: integer, at least 0
: Maximum burst size for data received on this interface, in kpkts (1 kpkts is 1000 packets). The default burst size if set to **0** is 16 kpkts. This value has no effect if **ingress_policing_kpkts_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. Specifying a value that is numerically at least as large as 80% of **ingress_policing_kpkts_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. By default, 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.

OVS 2.13 and earlier intercepted and processed all BFD packets. OVS 2.14 and later only intercept and

process BFD packets destined to a configured BFD instance, and other BFD packets are made available to the OVS flow table for forwarding.

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 : oam: optional string
Some tunnel protocols (such as Geneve) include a bit in the header to indicate that the encapsulated packet is an OAM frame. By setting this to true, BFD packets will be marked as OAM if encapsulated in one of these tunnels.

bfd : mult: optional string, containing an integer, in range 1 to 255
The BFD detection multiplier, which defaults to 3. An endpoint signals a connectivity fault if the given number of consecutive BFD control messages fail to arrive.

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 **admin_down**, **down**, **init**, or **up**
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
A diagnostic code specifying the local system's reason for the last change in session state. The error messages are defined in section 4.1 of [RFC 5880].

bfd_status : remote_state: optional string, one of **admin_down**, **down**, **init**, or **up**
Reports the state of the remote endpoint's BFD session.

bfd_status : remote_diagnostic: optional string
A diagnostic code specifying the remote system's reason for the last change in session state. 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.

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*.

external_ids : iface-id: optional string
A system-unique identifier for the interface.

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 hypervisor may have 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 : vm-id: optional string
The VM to which this interface belongs.

Auto Attach Configuration:

Auto Attach configuration for a particular interface.

lldp : enable: optional string, either **true** or **false**
True to enable LLDP on this **Interface**. If not specified, LLDP will be disabled by default.

Flow control Configuration:

Ethernet flow control defined in IEEE 802.1Qbb provides link level flow control using MAC pause frames. Implemented only for interfaces with type **dpdk**.

options : rx-flow-ctrl: optional string, either **true** or **false**
Set to **true** to enable Rx flow control on physical ports. By default, Rx flow control is disabled.

options : tx-flow-ctrl: optional string, either **true** or **false**
Set to **true** to enable Tx flow control on physical ports. By default, Tx flow control is disabled.

options : flow-ctrl-autoneg: optional string, either **true** or **false**
Set to **true** to enable flow control auto negotiation on physical ports. By default, auto-neg is disabled.

Link State Change detection mode:

options : dpdk-lsc-interrupt: optional string, either **true** or **false**
Set this value to **true** to configure interrupt mode for Link State Change (LSC) detection instead of poll mode for the DPDK interface.

If this value is not set, poll mode is configured.

This parameter has an effect only on netdev dpdk interfaces.

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
Eviction Policy:	
flow_limit	optional integer, at least 0
overflow_policy	optional string, either **evict** or **refuse**
groups	set of strings
Classifier Optimization:	
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.

Eviction Policy:

Open vSwitch supports limiting the number of flows that may be installed in a flow table, via the **flow_limit** column. When adding a flow would exceed this limit, by default Open vSwitch reports an error, but there are two ways to configure Open vSwitch to instead delete (''evict'') a flow to make room for the new one:

- Set the **overflow_policy** column to **evict**.
- Send an OpenFlow 1.4+ ''table mod request'' to enable eviction for the flow table (e.g. **ovs−ofctl −O OpenFlow14 mod−table br0 0 evict** to enable eviction on flow table 0 of bridge **br0**).

When a flow must be evicted due to overflow, the flow to evict is chosen through an approximation of the following algorithm. This algorithm is used regardless of how eviction was enabled:

1. Divide the flows in the table into groups based on the values of the fields or subfields specified in the **groups** column, 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. If **groups** is empty, then all of the flows in the flow table are treated as a single group.
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. If the flows under consideration have different importance values, eliminate from consideration any flows except those with the lowest importance. (''Importance,'' a 16-bit integer value attached to each flow, was introduced in OpenFlow 1.4. Flows inserted with older versions of OpenFlow always have an importance of 0.)
4. 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**.)

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 **evict** or **refuse**

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 a flow chosen according to the algorithm described above.

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 **meta−flow.h** for a complete list of NXM field names.

Open vSwitch ignores any invalid or unknown field specifications.

When eviction is not enabled, via **overflow_policy** or an OpenFlow 1.4+ ‘‘table mod,’’ this column has no effect.

Classifier Optimization:

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 unwildcard 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**, **tun_ipv6_src**, **tun_ipv6_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
Configuration for egress-policer QoS:	
other_config : cir	optional string, containing an integer
other_config : cbs	optional string, containing an integer
other_config : eir	optional string, containing an integer
other_config : ebs	optional string, containing an integer
Configuration for linux-sfq:	
other_config : perturb	optional string, containing an integer
other_config : quantum	optional string, containing an integer
Configuration for linux-netem:	
other_config : latency	optional string, containing an integer
other_config : limit	optional string, containing an integer
other_config : loss	optional string, containing an integer
other_config : jitter	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.

linux−sfq

Linux ‘‘Stochastic Fairness Queueing’’ classifier. See **tc−sfq**(8) (also at **http://linux.die.net/man/8/tc−sfq**) for information on how this classifier works.

linux−codel

Linux ‘‘Controlled Delay’’ classifier. See **tc−codel**(8) (also at **http://man7.org/linux/man−pages/man8/tc−codel.8.html**) for information on how this classifier works.

linux−fq_codel

Linux ‘‘Fair Queuing with Controlled Delay’’ classifier. See **tc−fq_codel**(8) (also at **http://man7.org/linux/man−pages/man8/tc−fq_codel.8.html**) for information on how this classifier works.

linux−netem

Linux ‘‘Network Emulator’’ classifier. See **tc−netem**(8) (also at **http://man7.org/linux/man−pages/man8/tc−netem.8.html**) for information on how this classifier works.

linux−noop
Linux ‘‘No operation.’’ By default, Open vSwitch manages quality of service on all of its configured ports. This can be helpful, but sometimes administrators prefer to use other software to manage QoS. This **type** prevents Open vSwitch from changing the QoS configuration for a port.

egress−policer
A DPDK egress policer algorithm using the DPDK rte_meter library. The rte_meter library provides an implementation which allows the metering and policing of traffic. The implementation in OVS essentially creates a single token bucket used to police traffic. It should be noted that when the rte_meter is configured as part of QoS there will be a performance overhead as the rte_meter itself will consume CPU cycles in order to police traffic. These CPU cycles ordinarily are used for packet proccessing. As such the drop in performance will be noticed in terms of overall aggregate traffic throughput.

trtcm−policer
A DPDK egress policer algorithm using RFC 4115’s Two-Rate, Three-Color marker. It’s a two-level hierarchical policer which first does a color-blind marking of the traffic at the queue level, followed by a color-aware marking at the port level. At the end traffic marked as Green or Yellow is forwarded, Red is dropped. For details on how traffic is marked, see RFC 4115. If the ‘‘default queue’’, 0, is not configured it’s automatically created with the same **other_config** values as the physical port.

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 10 Gbps.

Configuration for egress-policer QoS:

QoS type egress−policer provides egress policing for userspace port types with DPDK. It has the following key-value pairs defined.

other_config : cir: optional string, containing an integer
The Committed Information Rate (CIR) is measured in bytes of IP packets per second, i.e. it includes the IP header, but not link specific (e.g. Ethernet) headers. This represents the bytes per second rate at which the token bucket will be updated. The cir value is calculated by (pps x packet data size). For example assuming a user wishes to limit a stream consisting of 64 byte packets to 1 million packets per second the CIR would be set to to to 46000000. This value can be broken into ’1,000,000 x 46’. Where 1,000,000 is the policing rate for the number of packets per second and 46 represents the size of the packet data for a 64 bytes IP packet without 14 bytes Ethernet and 4 bytes FCS header.

other_config : cbs: optional string, containing an integer
The Committed Burst Size (CBS) is measured in bytes and represents a token bucket. At a minimum this value should be be set to the expected largest size packet in the traffic stream. In practice

larger values may be used to increase the size of the token bucket. If a packet can be transmitted then the cbs will be decremented by the number of bytes/tokens of the packet. If there are not enough tokens in the cbs bucket the packet will be dropped.

other_config : eir: optional string, containing an integer
The Excess Information Rate (EIR) is measured in bytes of IP packets per second, i.e. it includes the IP header, but not link specific (e.g. Ethernet) headers. This represents the bytes per second rate at which the token bucket will be updated. The eir value is calculated by (pps x packet data size). For example assuming a user wishes to limit a stream consisting of 64 byte packets to 1 million packets per second the EIR would be set to to to 46000000. This value can be broken into '1,000,000 x 46'. Where 1,000,000 is the policing rate for the number of packets per second and 46 represents the size of the packet data for a 64 bytes IP packet without 14 bytes Ethernet and 4 bytes FCS header.

other_config : ebs: optional string, containing an integer
The Excess Burst Size (EBS) is measured in bytes and represents a token bucket. At a minimum this value should be be set to the expected largest size packet in the traffic stream. In practice larger values may be used to increase the size of the token bucket. If a packet can be transmitted then the ebs will be decremented by the number of bytes/tokens of the packet. If there are not enough tokens in the cbs bucket the packet might be dropped.

Configuration for linux-sfq:

The **linux−sfq** QoS supports the following key-value pairs:

other_config : perturb: optional string, containing an integer
Number of seconds between consecutive perturbations in hashing algorithm. Different flows can end up in the same hash bucket causing unfairness. Perturbation's goal is to remove possible unfairness. The default and recommended value is 10. Too low a value is discouraged because each perturbation can cause packet reordering.

other_config : quantum: optional string, containing an integer
Number of bytes **linux−sfq** QoS can dequeue in one turn in round-robin from one flow. The default and recommended value is equal to interface's MTU.

Configuration for linux-netem:

The **linux−netem** QoS supports the following key-value pairs:

other_config : latency: optional string, containing an integer
Adds the chosen delay to the packets outgoing to chosen network interface. The latency value expressed in us.

other_config : limit: optional string, containing an integer
Maximum number of packets the qdisc may hold queued at a time. The default value is 1000.

other_config : loss: optional string, containing an integer
Adds an independent loss probability to the packets outgoing from the chosen network interface.

other_config : jitter: optional string, containing an integer
Adds the provided jitter to the latency outgoing to the chosen network interface. The jitter value expressed in us.

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.

When a packet enters an Open vSwitch bridge, it becomes eligible for mirroring based on its ingress port and VLAN. As the packet travels through the flow tables, each time it is output to a port, it becomes eligible for mirroring based on the egress port and VLAN. In Open vSwitch 2.5 and later, mirroring occurs just after a packet first becomes eligible, using the packet as it exists at that point; in Open vSwitch 2.4 and earlier, mirroring occurs only after a packet has traversed all the flow tables, using the original packet as it entered the bridge. This makes a difference only when the flow table modifies the packet: in Open vSwitch 2.4, the modifications are never visible to mirrors, whereas in Open vSwitch 2.5 and later modifications made before the first output that makes it eligible for mirroring to a particular destination are visible.

A packet that enters an Open vSwitch bridge is mirrored to a particular destination only once, even if it is eligible for multiple reasons. For example, a packet would be mirrored to a particular **output_port** only once, even if it is selected for mirroring to that port by **select_dst_port** and **select_src_port** in the same or different **Mirror** records.

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
snaplen	optional integer, in range 14 to 65,535
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.

snaplen: optional integer, in range 14 to 65,535
Maximum per-packet number of bytes to mirror.

A mirrored packet with size larger than **snaplen** will be truncated in datapath to **snaplen** bytes before sending to the mirror output port. If omitted, packets are not truncated.

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.

Summary:

Core Features:	
type	optional string, either **primary** or **service**
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_queue_size	optional integer, in range 1 to 512
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 **master**, **other**, or **slave**
status : last_error	optional string
status : state	optional string, one of **ACTIVE**, **BACKOFF**, **CONNECTING**, **IDLE**, or **VOID**
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:

type: optional string, either **primary** or **service**

Open vSwitch supports two kinds of OpenFlow controllers. A bridge may have any number of each kind:

Primary controllers

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

The **fail_mode** column in the **Bridge** table applies to primary controllers.

When multiple primary controllers are configured, Open vSwitch connects to all of them simultaneously. OpenFlow provides few facilities to allow multiple controllers to coordinate in interacting with a single switch, so more than one primary controller should be specified only if the controllers are themselves designed to coordinate with each other.

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.

The **fail_mode** column in the **Bridge** table does not apply to service controllers.

By default, Open vSwitch treats controllers with active connection methods as primary controllers and those with passive connection methods as service controllers. Set this column to the desired type to override this default.

target: string

Connection method for controller.

The following active connection methods are currently supported:

ssl:*host*[**:***port*]

The specified SSL *port* on the host at the given *host*, which can either be a DNS name (if built with unbound library) or an IP address. 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 defaults to 6653.

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

tcp:*host*[**:***port*]

The specified TCP *port* on the host at the given *host*, which can either be a DNS name (if built with unbound library) or an IP address (IPv4 or IPv6). If *host* is an IPv6 address, wrap it in square brackets, e.g. **tcp:[::1]:6653**.

If *port* is not specified, it defaults to 6653.

The following passive connection methods are currently supported:

pssl:[*port*][**:***host*]

Listens for SSL connections on the specified TCP *port*. If *host*, which can either be a DNS name (if built with unbound library) or an IP address, is specified, then connections are restricted to the resolved or specified local IP address (either IPv4 or IPv6). If *host* is an IPv6 address, wrap it in square brackets, e.g. **pssl:6653:[::1]**.

If *port* is not specified, it defaults to 6653. If *host* 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 to 6653.

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

ptcp:[*port*][**:***host*]

Listens for connections on the specified TCP *port*. If *host*, which can either be a DNS name (if built with unbound library) or an IP address, is specified, then connections are restricted to the resolved or specified local IP address (either IPv4 or IPv6). If *host* is an IPv6 address, wrap it in square brackets, e.g. **ptcp:6653:[::1]**. If *host* is not specified then it listens only on IPv4 addresses.

If *port* is not specified, it defaults to 6653.

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_queue_size: optional integer, in range 1 to 512
This sets the maximum size of the queue of packets that need to be sent to this OpenFlow controller. The value must be less than 512. If not specified the queue size is limited to the value set for the management controller in **other_config:controller-queue-size** if present or 100 packets by default. Note: increasing the queue size might have a negative impact on latency.

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 **master**, **other**, or **slave**
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 such controller at a time. If a given controller promotes itself to this role, **ovs−vswitchd** demotes any existing controller with the role to **slave**.

slave Allows the controller read-only access to OpenFlow features. Attempts to modify the flow table will be rejected with an error. Such 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**, **BACKOFF**, **CONNECTING**, **IDLE**, or **VOID**
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**, **BACKOFF**, **CONNECTING**, **IDLE**, or **VOID**
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:*host*[**:***port*]

The specified SSL *port* on the host at the given *host*, which can either be a DNS name (if built with unbound library) or an IP address. 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 defaults to 6640.

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

tcp:*host*[**:***port*]

The specified TCP *port* on the host at the given *host*, which can either be a DNS name (if built with unbound library) or an IP address (IPv4 or IPv6). If *host* is an IPv6 address, wrap it in square brackets, e.g. **tcp:[::1]:6640**.

If *port* is not specified, it defaults to 6640.

pssl:[*port*][**:***host*]

Listens for SSL connections on the specified TCP *port*. Specify 0 for *port* to have the kernel automatically choose an available port. If *host*, which can either be a DNS name (if

built with unbound library) or an IP address, is specified, then connections are restricted to the resolved or specified local IP address (either IPv4 or IPv6 address). If *host* is an IPv6 address, wrap in square brackets, e.g. **pssl:6640:[::1]**. If *host* 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 defaults to 6640.

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

ptcp:[*port*][**:***host*]

Listens for connections on the specified TCP *port*. Specify 0 for *port* to have the kernel automatically choose an available port. If *host*, which can either be a DNS name (if built with unbound library) or an IP address, is specified, then connections are restricted to the resolved or specified local IP address (either IPv4 or IPv6 address). If *host* is an IPv6 address, wrap it in square brackets, e.g. **ptcp:6640:[::1]**. If *host* is not specified then it listens only on IPv4 addresses.

If *port* is not specified, it defaults to 6640.

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:

Key-value pair of **is_connected** is always updated. Other key-value pairs in the status columns may be updated depends on the **target** type.

When **target** specifies a connection method that listens for inbound connections (e.g. **ptcp:** or **punix:**), both **n_connections** and **is_connected** may also be updated while the remaining key-value pairs are omitted.

On the other hand, when **target** specifies an outbound connection, all key-value pairs may be updated, except the above-mentioned two key-value pairs associated with inbound connection targets. They are omitted.

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, one of **ACTIVE**, **BACKOFF**, **CONNECTING**, **IDLE**, or **VOID**
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.

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 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.

The NetFlow passive timeout, for flows that become inactive, is not configurable. It will vary depending on the Open vSwitch version, the forms and contents of the OpenFlow flow tables, CPU and memory usage, and network activity. A typical passive timeout is about a second.

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 NetFlow 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

Datapath TABLE

Configuration for a datapath within **Open_vSwitch**.

A datapath is responsible for providing the packet handling in Open vSwitch. There are two primary datapath implementations used by Open vSwitch: kernel and userspace. Kernel datapath implementations are available for Linux and Hyper-V, and selected as **system** in the **datapath_type** column of the **Bridge** table. The userspace datapath is used by DPDK and AF-XDP, and is selected as **netdev** in the **datapath_type** column of the **Bridge** table.

A datapath of a particular type is shared by all the bridges that use that datapath. Thus, configurations applied to this table affect all bridges that use this datapath.

Summary:

datapath_version	string
ct_zones	map of integer-**CT_Zone** pairs, key in range 0 to 65,535
Capabilities:	
capabilities : max_vlan_headers	optional string, containing an integer, at least 0
capabilities : recirc	optional string, either **true** or **false**
capabilities : lb_output_action	optional string, either **true** or **false**
Connection-Tracking Capabilities:	
capabilities : ct_state	optional string, either **true** or **false**
capabilities : ct_state_nat	optional string, either **true** or **false**
capabilities : ct_zone	optional string, either **true** or **false**
capabilities : ct_mark	optional string, either **true** or **false**
capabilities : ct_label	optional string, either **true** or **false**
capabilities : ct_orig_tuple	optional string, either **true** or **false**
capabilities : ct_orig_tuple6	optional string, either **true** or **false**
capabilities : masked_set_action	optional string, either **true** or **false**
capabilities : tnl_push_pop	optional string, either **true** or **false**
capabilities : ufid	optional string, either **true** or **false**
capabilities : trunc	optional string, either **true** or **false**
capabilities : nd_ext	optional string, either **true** or **false**
Clone Actions:	
capabilities : clone	optional string, either **true** or **false**
capabilities : sample_nesting	optional string, containing an integer, at least 0
capabilities : ct_eventmask	optional string, either **true** or **false**
capabilities : ct_clear	optional string, either **true** or **false**
capabilities : max_hash_alg	optional string, containing an integer, at least 0
capabilities : check_pkt_len	optional string, either **true** or **false**
capabilities : ct_timeout	optional string, either **true** or **false**
capabilities : explicit_drop_action	optional string, either **true** or **false**
capabilities : ct_zero_snat	optional string, either **true** or **false**
capabilities : ct_flush	optional string, either **true** or **false**
Common Columns:	
external_ids	map of string-string pairs

Details:

datapath_version: string

Reports the version number of the Open vSwitch datapath in use. This allows management software to detect and report discrepancies between Open vSwitch userspace and datapath versions. (The **ovs_version** column in the **Open_vSwitch** reports the Open vSwitch userspace version.) The version reported depends on the datapath in use:

- When the kernel module included in the Open vSwitch source tree is used, this column reports the Open vSwitch version from which the module was taken.

- When the kernel module that is part of the upstream Linux kernel is used, this column reports **<unknown>**.
- When the datapath is built into the **ovs−vswitchd** binary, this column reports **<built−in>**. A built-in datapath is by definition the same version as the rest of the Open vSwitch userspace.
- Other datapaths (such as the Hyper-V kernel datapath) currently report **<unknown>**.

A version discrepancy between **ovs−vswitchd** and the datapath in use is not normally cause for alarm. The Open vSwitch kernel datapaths for Linux and Hyper-V, in particular, are designed for maximum inter-version compatibility: any userspace version works with with any kernel version. Some reasons do exist to insist on particular user/kernel pairings. First, newer kernel versions add new features, that can only be used by new-enough userspace, e.g. VXLAN tunneling requires certain minimal userspace and kernel versions. Second, as an extension to the first reason, some newer kernel versions add new features for enhancing performance that only new-enough userspace versions can take advantage of.

ct_zones: map of integer-**CT_Zone** pairs, key in range 0 to 65,535
Configuration for connection tracking zones. Each pair maps from a zone id to a configuration for that zone. Zone **0** applies to the default zone (ie, the one used if a zone is not specified in connection tracking-related OpenFlow matches and actions).

Capabilities:

The **capabilities** column reports a datapath's features. For the **netdev** datapath, the capabilities are fixed for a given version of Open vSwitch because this datapath is built into the **ovs−vswitchd** binary. The Linux kernel and Windows and other datapaths, which are external to OVS userspace, can vary in version and capabilities independently from **ovs−vswitchd**.

Some of these features indicate whether higher-level Open vSwitch features are available. For example, OpenFlow features for connection-tracking are available only when **capabilities:ct_state** is **true**. A controller that wishes to determine whether a feature is supported could, therefore, consult the relevant capabilities in this table. However, as a general rule, it is better for a controller to try to use the higher-level feature and use the result as an indication of support, since the low-level capabilities are more likely to shift over time than the high-level features that rely on them.

capabilities : max_vlan_headers: optional string, containing an integer, at least 0
Number of 802.1q VLAN headers supported by the datapath, as probed by the **ovs−vswitchd** slow path. If the datapath supports more VLAN headers than the slow path, this reports the slow path's limit. The value of **other-config:vlan-limit** in the **Open_vSwitch** table does not influence the number reported here.

capabilities : recirc: optional string, either **true** or **false**
If this is true, then the datapath supports recirculation, specifically OVS_KEY_ATTR_RECIRC_ID. Recirculation enables higher performance for MPLS and active-active load balancing bonding modes.

capabilities : lb_output_action: optional string, either **true** or **false**
If this is true, then the datapath supports optimized balance-tcp bond mode. This capability replaces existing **hash** and **recirc** actions with new action **lb_output** and avoids recirculation of packet in datapath. It is supported only for balance-tcp bond mode in netdev datapath. The new action gives higher performance by using bond buckets instead of post recirculation flows for selection of slave port from bond. By default this new action is disabled, however it can be enabled by setting **other-config:lb-output-action** in **Port** table.

Connection-Tracking Capabilities:

These capabilities are granular because Open vSwitch and its datapaths added support for connection tracking over several releases, with features added individually over that time.

capabilities : ct_state: optional string, either **true** or **false**
If true, datapath supports OVS_KEY_ATTR_CT_STATE, which indicates support for the bits in the OpenFlow **ct_state** field (see **ovs−fields**(7)) other than **snat** and **dnat**, which have a separate capability.

If this is false, the datapath does not support connection-tracking at all and the remaining connection-tracking capabilities should all be false. In this case, Open vSwitch will reject flows that match on the **ct_state** field or use the **ct** action.

capabilities : ct_state_nat: optional string, either **true** or **false**
If true, it means that the datapath supports the **snat** and **dnat** flags in the OpenFlow **ct_state** field. The **ct_state** capability must be true for this to make sense.

If false, Open vSwitch will reject flows that match on the **snat** or **dnat** bits in **ct_state** or use **nat** in the **ct** action.

capabilities : ct_zone: optional string, either **true** or **false**
If true, datapath supports OVS_KEY_ATTR_CT_ZONE. If false, Open vSwitch rejects flows that match on the **ct_zone** field or that specify a nonzero zone or a zone field on the **ct** action.

capabilities : ct_mark: optional string, either **true** or **false**
If true, datapath supports OVS_KEY_ATTR_CT_MARK. If false, Open vSwitch rejects flows that match on the **ct_mark** field or that set **ct_mark** in the **ct** action.

capabilities : ct_label: optional string, either **true** or **false**
If true, datapath supports OVS_KEY_ATTR_CT_LABEL. If false, Open vSwitch rejects flows that match on the **ct_label** field or that set **ct_label** in the **ct** action.

capabilities : ct_orig_tuple: optional string, either **true** or **false**
If true, the datapath supports matching the 5-tuple from the connection's original direction for IPv4 traffic. If false, Open vSwitch rejects flows that match on **ct_nw_src** or **ct_nw_dst**, that use the **ct** feature of the **resubmit** action, or the **force** keyword in the **ct** action. (The latter isn't tied to connection tracking support of original tuples in any technical way. They are conflated because all current datapaths implemented the two features at the same time.)

If this and **capabilities:ct_orig_tuple6** are both false, Open vSwitch rejects flows that match on **ct_nw_proto**, **ct_tp_src**, or **ct_tp_dst**.

capabilities : ct_orig_tuple6: optional string, either **true** or **false**
If true, the datapath supports matching the 5-tuple from the connection's original direction for IPv6 traffic. If false, Open vSwitch rejects flows that match on **ct_ipv6_src** or **ct_ipv6_dst**.

capabilities : masked_set_action: optional string, either **true** or **false**
True if the datapath supports masked data in OVS_ACTION_ATTR_SET actions. Masked data can improve performance by allowing megaflows to match on fewer fields.

capabilities : tnl_push_pop: optional string, either **true** or **false**
True if the datapath supports tnl_push and pop actions. This is a prerequisite for a datapath to support native tunneling.

capabilities : ufid: optional string, either **true** or **false**
True if the datapath supports OVS_FLOW_ATTR_UFID. UFID support improves revalidation performance by transferring less data between the slow path and the datapath.

capabilities : trunc: optional string, either **true** or **false**
True if the datapath supports OVS_ACTION_ATTR_TRUNC action. If false, the **output** action with packet truncation requires every packet to be sent to the Open vSwitch slow path, which is likely to make it too slow for mirroring traffic in bulk.

capabilities : nd_ext: optional string, either **true** or **false**
True if the datapath supports OVS_KEY_ATTR_ND_EXTENSIONS to match on ICMPv6 "ND reserved" and "ND option type" header fields. If false, the datapath reports error if the feature is used.

Clone Actions:

When Open vSwitch translates actions from OpenFlow into the datapath representation, some of the datapath actions may modify the packet or have other side effects that later datapath actions can't undo. The OpenFlow **ct**, **meter**, **output** with truncation, **encap**, **decap**, and **dec_nsh_ttl** actions fall into this category. Often, this is not a problem because nothing later on needs the original packet.

Such actions can, however, occur in circumstances where the translation does require the original packet. For example, an OpenFlow **output** action might direct a packet to a patch port, which might in turn lead to a **ct** action that NATs the packet (which cannot be undone), and then afterward when control flow pops back across the patch port some other action might need to act on the original packet.

Open vSwitch has two different ways to implement this ''save and restore'' via datapath actions. These capabilities indicate which one Open vSwitch will choose. When neither is available, Open vSwitch simply fails in situations that require this feature.

capabilities : clone: optional string, either **true** or **false**
: True if the datapath supports OVS_ACTION_ATTR_CLONE action. This is the preferred option for saving and restoring packets, since it is intended for the purpose, but old datapaths do not support it. Open vSwitch will use it whenever it is available.

  (The OpenFlow **clone** action does not always yield a OVS_ACTION_ATTR_CLONE action. It only does so when the datapath supports it and the **clone** brackets actions that otherwise cannot be undone.)

capabilities : sample_nesting: optional string, containing an integer, at least 0
: Maximum level of nesting allowed by OVS_ACTION_ATTR_SAMPLE action. Open vSwitch misuses this action for saving and restoring packets when the datapath supports more than 3 levels of nesting and OVS_ACTION_ATTR_CLONE is not available.

capabilities : ct_eventmask: optional string, either **true** or **false**
: True if the datapath's OVS_ACTION_ATTR_CT action implements the OVS_CT_ATTR_EVENTMASK attribute. When this is true, Open vSwitch uses the event mask feature to limit the kinds of events reported to conntrack update listeners. When Open vSwitch doesn't limit the event mask, listeners receive reports of numerous usually unimportant events, such as TCP state machine changes, which can waste CPU time.

capabilities : ct_clear: optional string, either **true** or **false**
: True if the datapath supports OVS_ACTION_ATTR_CT_CLEAR action. If false, the OpenFlow **ct_clear** action has no effect on the datapath.

capabilities : max_hash_alg: optional string, containing an integer, at least 0
: Highest supported dp_hash algorithm. This allows Open vSwitch to avoid requesting a packet hash that the datapath does not support.

capabilities : check_pkt_len: optional string, either **true** or **false**
: True if the datapath supports OVS_ACTION_ATTR_CHECK_PKT_LEN. If false, Open vSwitch implements the **check_pkt_larger** action by sending every packet through the Open vSwitch slow path, which is likely to make it too slow for handling traffic in bulk.

capabilities : ct_timeout: optional string, either **true** or **false**
: True if the datapath supports OVS_CT_ATTR_TIMEOUT in the OVS_ACTION_ATTR_CT action. If false, Open vswitch cannot implement timeout policies based on connection tracking zones, as configured through the **CT_Timeout_Policy** table.

capabilities : explicit_drop_action: optional string, either **true** or **false**
: True if the datapath supports OVS_ACTION_ATTR_DROP. If false, explicit drop action will not be sent to the datapath.

capabilities : ct_zero_snat: optional string, either **true** or **false**
: True if the datapath supports all-zero SNAT. This is a special case if the **src** IP address is configured as all 0's, i.e., **nat(src=0.0.0.0)**. In this case, when a source port collision is detected during

the commit, the source port will be translated to an ephemeral port. If there is no collision, no SNAT is performed.

capabilities : ct_flush: optional string, either **true** or **false**
True if the datapath supports CT flush OpenFlow Nicira extension called **NXT_CT_FLUSH**. The **NXT_CT_FLUSH** extensions allows to flush CT entries based on specified parameters.

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

CT_Zone TABLE

Connection tracking zone configuration

Summary:

timeout_policy	optional **CT_Timeout_Policy**
Common Columns:	
external_ids	map of string-string pairs

Details:

timeout_policy: optional **CT_Timeout_Policy**

Connection tracking timeout policy for this zone. If a timeout policy is not specified, it defaults to the timeout policy in the system.

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

CT_Timeout_Policy TABLE

Connection tracking timeout policy configuration

Summary:

Timeouts:

timeouts	map of string-integer pairs, key one of **icmp_first**, **icmp_reply**, **tcp_close**, **tcp_close_wait**, **tcp_established**, **tcp_fin_wait**, **tcp_last_ack**, **tcp_retransmit**, **tcp_syn_recv**, **tcp_syn_sent2**, **tcp_syn_sent**, **tcp_time_wait**, **tcp_unack**, **udp_first**, **udp_multiple**, or **udp_single**, value in range 0 to 4,294,967,295

TCP Timeouts:

timeouts : tcp_syn_sent	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_syn_recv	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_established	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_fin_wait	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_close_wait	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_last_ack	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_time_wait	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_close	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_syn_sent2	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_retransmit	optional integer, in range 0 to 4,294,967,295
timeouts : tcp_unack	optional integer, in range 0 to 4,294,967,295

UDP Timeouts:

timeouts : udp_first	optional integer, in range 0 to 4,294,967,295
timeouts : udp_single	optional integer, in range 0 to 4,294,967,295
timeouts : udp_multiple	optional integer, in range 0 to 4,294,967,295

ICMP Timeouts:

timeouts : icmp_first	optional integer, in range 0 to 4,294,967,295
timeouts : icmp_reply	optional integer, in range 0 to 4,294,967,295

Common Columns:

external_ids	map of string-string pairs

Details:

Timeouts:

timeouts: map of string-integer pairs, key one of **icmp_first**, **icmp_reply**, **tcp_close**, **tcp_close_wait**, **tcp_established**, **tcp_fin_wait**, **tcp_last_ack**, **tcp_retransmit**, **tcp_syn_recv**, **tcp_syn_sent2**, **tcp_syn_sent**, **tcp_time_wait**, **tcp_unack**, **udp_first**, **udp_multiple**, or **udp_single**, value in range 0 to 4,294,967,295

The **timeouts** column contains key-value pairs used to configure connection tracking timeouts in a datapath. Key-value pairs that are not supported by a datapath are ignored. The timeout value is in seconds.

TCP Timeouts:

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

The timeout for the connection after the first TCP SYN packet has been seen by conntrack.

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

The timeout of the connection after the first TCP SYN-ACK packet has been seen by conntrack.

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

The timeout of the connection after the connection has been fully established.

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

The timeout of the connection after the first TCP FIN packet has been seen by conntrack.

timeouts : tcp_close_wait: optional integer, in range 0 to 4,294,967,295
The timeout of the connection after the first TCP ACK packet has been seen after it receives TCP FIN packet. This timeout is only supported by the Linux kernel datapath.

timeouts : tcp_last_ack: optional integer, in range 0 to 4,294,967,295
The timeout of the connection after TCP FIN packets have been seen by conntrack from both directions. This timeout is only supported by the Linux kernel datapath.

timeouts : tcp_time_wait: optional integer, in range 0 to 4,294,967,295
The timeout of the connection after conntrack has seen the TCP ACK packet for the second TCP FIN packet.

timeouts : tcp_close: optional integer, in range 0 to 4,294,967,295
The timeout of the connection after the first TCP RST packet has been seen by conntrack.

timeouts : tcp_syn_sent2: optional integer, in range 0 to 4,294,967,295
The timeout of the connection when only a TCP SYN packet has been seen by conntrack from both directions (simultaneous open). This timeout is only supported by the Linux kernel datapath.

timeouts : tcp_retransmit: optional integer, in range 0 to 4,294,967,295
The timeout of the connection when it exceeds the maximum number of retransmissions. This timeout is only supported by the Linux kernel datapath.

timeouts : tcp_unack: optional integer, in range 0 to 4,294,967,295
The timeout of the connection when non-SYN packets create an established connection in TCP loose tracking mode. This timeout is only supported by the Linux kernel datapath.

UDP Timeouts:

timeouts : udp_first: optional integer, in range 0 to 4,294,967,295
The timeout of the connection after the first UDP packet has been seen by conntrack. This timeout is only supported by the userspace datapath.

timeouts : udp_single: optional integer, in range 0 to 4,294,967,295
The timeout of the connection when conntrack only seen UDP packet from the source host, but the destination host has never sent one back.

timeouts : udp_multiple: optional integer, in range 0 to 4,294,967,295
The timeout of the connection when UDP packets have been seen in both directions.

ICMP Timeouts:

timeouts : icmp_first: optional integer, in range 0 to 4,294,967,295
The timeout of the connection after the first ICMP packet has been seen by conntrack.

timeouts : icmp_reply: optional integer, in range 0 to 4,294,967,295
The timeout of the connection when ICMP packets have been seen in both direction. This timeout is only supported by the userspace datapath.

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

Determines the agent address, that is, the IP address reported to collectors as the source of the sFlow data. It may be an IP address or the name of a network device. In the latter case, the network device's IP address is used,

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, 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

IPFIX TABLE

Configuration for sending packets to IPFIX collectors.

IPFIX is a protocol that exports a number of details about flows. The IPFIX implementation in Open vSwitch samples packets at a configurable rate, extracts flow information from those packets, optionally caches and aggregates the flow information, and sends the result to one or more collectors.

IPFIX in Open vSwitch can be configured two different ways:

- With **per−bridge sampling**, Open vSwitch performs IPFIX sampling automatically on all packets that pass through a bridge. To configure per-bridge sampling, create an **IPFIX** record and point a **Bridge** table's **ipfix** column to it. The **Flow_Sample_Collector_Set** table is not used for per-bridge sampling.
- With **flow−based sampling**, **sample** actions in the OpenFlow flow table drive IPFIX sampling. See **ovs−actions**(7) for a description of the **sample** action.

  Flow-based sampling also requires database configuration: create a **IPFIX** record that describes the IPFIX configuration and a **Flow_Sample_Collector_Set** record that points to the **Bridge** whose flow table holds the **sample** actions and to **IPFIX** record. The **ipfix** in the **Bridge** table is not used for flow-based sampling.

Summary:

targets	set of strings
cache_active_timeout	optional integer, in range 0 to 4,200
cache_max_flows	optional integer, in range 0 to 4,294,967,295
stats_interval	optional integer, in range 1 to 3,600
template_interval	optional integer, in range 1 to 3,600
other_config : enable-tunnel-sampling	optional string, either **true** or **false**
other_config : virtual_obs_id	optional string
Per-Bridge Sampling:	
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
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*.

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.

stats_interval: optional integer, in range 1 to 3,600
: Interval (in seconds) for sending IPFIX exporting process statistics according to IETF RFC 5101 Section 4.3.

  Default value is 600

template_interval: optional integer, in range 1 to 3,600
: Interval (in seconds) for sending IPFIX Template information for each Observation Domain ID.

  Default value is 600

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

Set to **true** to enable sampling and reporting tunnel header 7-tuples in IPFIX flow records. Tunnel sampling is enabled by default.

The following enterprise entities report the sampled tunnel info:

tunnelType:

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

type: unsigned 8-bit integer.

data type semantics: identifier.

description: Identifier of the layer 2 network overlay network encapsulation type: 0x01 VxLAN, 0x02 GRE, 0x03 LISP, 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-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 integer.

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.

Before Open vSwitch 2.5.90, **other_config:enable-tunnel-sampling** was only supported with per-bridge sampling, and ignored otherwise. Open vSwitch 2.5.90 and later support **other_config:enable-tunnel-sampling** for per-bridge and per-flow sampling.

other_config : virtual_obs_id: optional string
A string that accompanies each IPFIX flow record. Its intended use is for the ‘‘virtual observation ID,’’ an identifier of a virtual observation point that is locally unique in a virtual network. It describes a location in the virtual network where IP packets can be observed. The maximum length is 254 bytes. If not specified, the field is omitted from the IPFIX flow record.

The following enterprise entity reports the specified virtual observation ID:

virtualObsID:
ID: 898, and enterprise ID 6876 (VMware).

type: variable-length string.

data type semantics: identifier.

description: A virtual observation domain ID that is locally unique in a virtual network.

This feature was introduced in Open vSwitch 2.5.90.

Per-Bridge Sampling:

These values affect only per-bridge sampling. See above for a description of the differences between per-bridge and flow-based sampling.

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

obs_domain_id: optional integer, in range 0 to 4,294,967,295
The IPFIX Observation Domain ID sent in each IPFIX packet. If not specified, defaults to 0.

obs_point_id: optional integer, in range 0 to 4,294,967,295
The IPFIX Observation Point ID sent in each IPFIX flow record. If not specified, defaults to 0.

other_config : enable-input-sampling: optional string, either **true** or **false**
By default, Open vSwitch samples and reports flows at bridge port input in IPFIX flow records. Set this column to **false** to disable input sampling.

other_config : enable-output-sampling: optional string, either **true** or **false**
By default, Open vSwitch samples and reports flows at bridge port output in IPFIX flow records. Set this column to **false** to disable output sampling.

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. This table is used only for IPFIX flow-based sampling, not for per-bridge sampling (see the **IPFIX** table for a description of the two forms).

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

AutoAttach TABLE

Auto Attach configuration within a bridge. The IETF Auto-Attach SPBM draft standard describes a compact method of using IEEE 802.1AB Link Layer Discovery Protocol (LLDP) together with a IEEE 802.1aq Shortest Path Bridging (SPB) network to automatically attach network devices to individual services in a SPB network. The intent here is to allow network applications and devices using OVS to be able to easily take advantage of features offered by industry standard SPB networks.

Auto Attach (AA) uses LLDP to communicate between a directly connected Auto Attach Client (AAC) and Auto Attach Server (AAS). The LLDP protocol is extended to add two new Type-Length-Value tuples (TLVs). The first new TLV supports the ongoing discovery of directly connected AA correspondents. Auto Attach operates by regularly transmitting AA discovery TLVs between the AA client and AA server. By exchanging these discovery messages, both the AAC and AAS learn the system name and system description of their peer. In the OVS context, OVS operates as the AA client and the AA server resides on a switch at the edge of the SPB network.

Once AA discovery has been completed the AAC then uses the second new TLV to deliver identifier mappings from the AAC to the AAS. A primary feature of Auto Attach is to facilitate the mapping of VLANs defined outside the SPB network onto service ids (ISIDs) defined within the SPM network. By doing so individual external VLANs can be mapped onto specific SPB network services. These VLAN id to ISID mappings can be configured and managed locally using new options added to the ovs-vsctl command.

The Auto Attach OVS feature does not provide a full implementation of the LLDP protocol. Support for the mandatory TLVs as defined by the LLDP standard and support for the AA TLV extensions is provided. LLDP protocol support in OVS can be enabled or disabled on a port by port basis. LLDP support is disabled by default.

Summary:

system_name	string
system_description	string
mappings	map of integer-integer pairs, key in range 0 to 16,777,215, value in range 0 to 4,095

Details:

system_name: string

The system_name string is exported in LLDP messages. It should uniquely identify the bridge in the network.

system_description: string

The system_description string is exported in LLDP messages. It should describe the type of software and hardware.

mappings: map of integer-integer pairs, key in range 0 to 16,777,215, value in range 0 to 4,095

A mapping from SPB network Individual Service Identifier (ISID) to VLAN id.