NAME

ovsdb−server − Open vSwitch database server

SYNOPSIS

ovsdb−server [*database*]... [*relay:schema_name:remote*]... [**−−remote=***remote*]... [**−−config−file=***file*] [**−−run=***command*]

Daemon options:
[**−−pidfile**[**=***pidfile*]] [**−−overwrite−pidfile**] [**−−detach**] [**−−no−chdir**] [**−−no−self−confinement**]

Service options:
[**−−service**] [**−−service−monitor**]

Logging options:
[**−v**[*module*[**:***destination*[**:***level*]]]]...
[**−−verbose[=***module*[**:***destination*[**:***level*]]]]...
[**−−log−file**[**=***file*]]

Active-backup options:
[**−−sync−from=***server*] [**−−sync−exclude-tables=***db***:***table*[**,***db***:***table*]...] [**−−active**]

Public key infrastructure options:
[**−−private−key=***privkey.pem*]
[**−−certificate=***cert.pem*]
[**−−ca−cert=***cacert.pem*]
[**−−bootstrap−ca−cert=***cacert.pem*]
[**−−peer−ca−cert=***peer-cacert.pem*]

SSL/TLS connection options:
[**−−ssl−protocols=***protocols*]
[**−−ssl−ciphers=***ciphers*]
[**−−ssl−ciphersuites=***ciphersuites*]

Runtime management options:
−−unixctl=*socket*

Replay options:
[**−−record**[**=***directory*]] [**−−replay**[**=***directory*]]

Common options:
[**−h** | **−−help**] [**−V** | **−−version**]

DESCRIPTION

The **ovsdb−server** program provides RPC interfaces to one or more Open vSwitch databases (OVSDBs). It supports JSON-RPC client connections over active or passive TCP/IP or Unix domain sockets. For an introduction to OVSDB and its implementation in Open vSwitch, see **ovsdb**(7).

Each OVSDB file may be specified on the command line as *database*. Relay databases may be specified on the command line as *relay:schema_name:remote*. For a detailed description of relay database argument, see **ovsdb**(7). If none of database files or relay databases is specified, the default is **/usr/local/etc/openvswitch/conf.db**. The database files must already have been created and initialized using, for example, **ovsdb−tool**'s **create**, **create−cluster**, or **join−cluster** command.

All types of databases can alternatively be added using a configuration file provided via **−−config−file** option. This option is mutually exclusive with specifying *database* on the command line. For a detailed description of the configuration file format see **ovsdb**(7).

This OVSDB implementation supports standalone, active-backup, relay and clustered database service models, as well as database replication. See the Service Models section of **ovsdb**(7) for more information.

For clustered databases, when the **−−detach** option is used, **ovsdb−server** detaches without waiting for the server to successfully join a cluster (if the database file is freshly created with **ovsdb−tool join−cluster**) or

connect to a cluster that it has already joined. Use **ovsdb−client wait** (see **ovsdb−client**(1)) to wait until the server has successfully joined and connected to a cluster. The same is true for relay databases. Same commands could be used to wait for a relay database to connect to the relay source (remote).

In addition to user-specified databases, **ovsdb−server** version 2.9 and later also always hosts a built-in database named **_Server**. Please see **ovsdb−server**(5) for documentation on this database's schema.

OPTIONS

−−remote=*remote*

Adds *remote* as a connection method used by **ovsdb−server**. The *remote* may be an OVSDB active or passive connection method, e.g. **pssl:6640**, as described in **ovsdb**(7). The following additional form is also supported:

db:*db***,***table***,***column*

Reads additional connection methods from *column* in all of the rows in *table* within *db*. As the contents of *column* changes, **ovsdb−server** also adds and drops connection methods accordingly.

If *column*'s type is string or set of strings, then the connection methods are taken directly from the column. The connection methods in the column must have one of the forms described above.

If *column*'s type is UUID or set of UUIDs and references a table, then each UUID is looked up in the referenced table to obtain a row. The following columns in the row, if present and of the correct type, configure a connection method. Any additional columns are ignored.

target (string)

Connection method, in one of the forms described above. This column is mandatory: if it is missing or empty then no connection method can be configured.

max_backoff (integer)

Maximum number of milliseconds to wait between connection attempts.

inactivity_probe (integer)

Maximum number of milliseconds of idle time on connection to client before sending an inactivity probe message.

read_only (boolean)

If true, only read-only transactions are allowed on this connection.

It is an error for *column* to have another type.

To connect or listen on multiple connection methods, use multiple **−−remote** options.

Alternatively, remotes can be specified in a "remotes" section of the configuration file, if provided using **−−config−file** option. **−−config−file** and **−−remote** options are mutually exclusive.

−−config-file=*file*

Specifies a configuration file for **ovsdb−server**. This *file* can contain connection methods and databases used by the server. The *file* contains a JSON object with two main elements:

remotes

JSON object that contains a set of connection methods in the following format: "*target*": { "*option*": *value*, ... }. Where *target* is in the same format as *remote* in **−−remote** option. *option* can be **max-backoff** (integer), **inactivity-probe** (integer), **read-only** (boolean), **role** (string) or **dscp** (integer) with their allowed *value*s respectively. The meaning of these *option*s is the same as in configuration of *remote* via a database row with **−−remote** option.

databases

JSON object that describes databases that should be added to the **ovsdb−server** in the following format: "*name*":{ "*option*": *value*, ... }. Where *name* is either a file name of a previously created and initialized database or a schema name in case of relay databases. Available *option*s are:

service-model (string)

Describes the service model of this database. One of: **standalone**, **clustered**, **active-backup** or **relay**. This option is required for all types, except for standalone and clustered. For these databases the service model will be inferred from the file, if not specified explicitly. **ovsdb-server** will refuse to add a database if the specified **service-model** doesn't match with the provided file.

source (JSON object; active-backup or relay)

Describes the connection method to the active database or to the relay source. It is a JSON object with exactly one element in the same format as elements of "**remotes**", except that **read-only** and **role** options are not applicable. E.g. **"source": { "unix:db.sock": { "inactivity-probe": 10000, "max-backoff": 8000 } }**

backup (boolean; active-backup only)

If set to **true**, **ovsdb-server** will use this database as a backup for the specified **source**. Will be served as an active database otherwise.

exclude-tables (JSON array of strings; active-backup only)

List of table names that should be excluded from replication in backup mode, e.g. **"exclude-tables": ["Table_One", "Table_Two"]**.

Content of the most basic configuration file may look like this: **{ "remotes": { "pssl:6640": {} }, "databases": { "conf.db": {} } }**

Examples of configuration files for different service models can be found in in **ovsdb**(7).

−−config-file option is mutually exclusive with the **−−remote** as well as with specifying *database* on a command line. It is also mutually exclusive with all the **Active-Backup Options** and all the **RUNTIME MANAGEMENT COMMANDS** that can change the configuration of the server in conflict with the content of the file, i.e. all the commands that manipulate with remotes and databases. Read-only commands can still be used.

In case of changes in the *file*, users should run the **ovsdb-server/reload** command with **ovs-appctl**(8) in order for changes to take effect.

−−run=*command*]

Ordinarily **ovsdb−server** runs forever, or until it is told to exit (see **RUNTIME MANAGEMENT COMMANDS** below). With this option, **ovsdb−server** instead starts a shell subprocess running *command*. When the subprocess terminates, **ovsdb−server** also exits gracefully. If the subprocess exits normally with exit code 0, then **ovsdb−server** exits with exit code 0 also; otherwise, it exits with exit code 1.

This option can be useful where a database server is needed only to run a single command, e.g.: **ovsdb−server −−remote=punix:socket −−run='ovsdb−client dump unix:socket Open_vSwitch'**

This option is not supported on Windows platform.

Daemon Options

The following options are valid on POSIX based platforms.

−−pidfile[**=***pidfile*]

Causes a file (by default, **ovsdb−server.pid**) to be created indicating the PID of the running process. If the *pidfile* argument is not specified, or if it does not begin with **/**, then it is created in **/usr/local/var/run/openvswitch**.

If **−−pidfile** is not specified, no pidfile is created.

−−overwrite−pidfile
By default, when **−−pidfile** is specified and the specified pidfile already exists and is locked by a running process, **ovsdb−server** refuses to start. Specify **−−overwrite−pidfile** to cause it to instead overwrite the pidfile.

When **−−pidfile** is not specified, this option has no effect.

−−detach
Runs **ovsdb−server** as a background process. The process forks, and in the child it starts a new session, closes the standard file descriptors (which has the side effect of disabling logging to the console), and changes its current directory to the root (unless **−−no−chdir** is specified). After the child completes its initialization, the parent exits. **ovsdb−server** detaches only after it starts listening on all configured remotes. At this point, all standalone and active-backup databases are ready for use. Clustered databases only become ready for use after they finish joining their clusters (which could have already happened in previous runs of **ovsdb−server**).

−−monitor
Creates an additional process to monitor the **ovsdb−server** daemon. If the daemon dies due to a signal that indicates a programming error (**SIGABRT**, **SIGALRM**, **SIGBUS**, **SIGFPE**, **SIGILL**, **SIGPIPE**, **SIGSEGV**, **SIGXCPU**, or **SIGXFSZ**) then the monitor process starts a new copy of it. If the daemon dies or exits for another reason, the monitor process exits.

This option is normally used with **−−detach**, but it also functions without it.

−−no−chdir
By default, when **−−detach** is specified, **ovsdb−server** changes its current working directory to the root directory after it detaches. Otherwise, invoking **ovsdb−server** from a carelessly chosen directory would prevent the administrator from unmounting the file system that holds that directory.

Specifying **−−no−chdir** suppresses this behavior, preventing **ovsdb−server** from changing its current working directory. This may be useful for collecting core files, since it is common behavior to write core dumps into the current working directory and the root directory is not a good directory to use.

This option has no effect when **−−detach** is not specified.

−−no−self−confinement
By default daemon will try to self-confine itself to work with files under well-known directories determined during build. It is better to stick with this default behavior and not to use this flag unless some other Access Control is used to confine daemon. Note that in contrast to other access control implementations that are typically enforced from kernel-space (e.g. DAC or MAC), self-confinement is imposed from the user-space daemon itself and hence should not be considered as a full confinement strategy, but instead should be viewed as an additional layer of security.

−−user Causes **ovsdb−server** to run as a different user specified in "user:group", thus dropping most of the root privileges. Short forms "user" and ":group" are also allowed, with current user or group are assumed respectively. Only daemons started by the root user accepts this argument.

On Linux, daemons will be granted CAP_IPC_LOCK and CAP_NET_BIND_SERVICES before dropping root privileges. Daemons that interact with a datapath, such as **ovs−vswitchd**, will be granted three additional capabilities, namely CAP_NET_ADMIN, CAP_NET_BROADCAST and CAP_NET_RAW. The capability change will apply even if the new user is root.

On Windows, this option is not currently supported. For security reasons, specifying this option will cause the daemon process not to start.

Service Options
The following options are valid only on Windows platform.

−−service
Causes **ovsdb−server** to run as a service in the background. The service should already have been created through external tools like **SC.exe**.

−−service−monitor
Causes the **ovsdb−server** service to be automatically restarted by the Windows services manager if the service dies or exits for unexpected reasons.

When **−−service** is not specified, this option has no effect.

Logging Options

−v[*spec*]
−−verbose=[*spec*]
Sets logging levels. Without any *spec*, sets the log level for every module and destination to **dbg**. Otherwise, *spec* is a list of words separated by spaces or commas or colons, up to one from each category below:

- A valid module name, as displayed by the **vlog/list** command on **ovs−appctl**(8), limits the log level change to the specified module.
- **syslog**, **console**, or **file**, to limit the log level change to only to the system log, to the console, or to a file, respectively. (If **−−detach** is specified, **ovsdb−server** closes its standard file descriptors, so logging to the console will have no effect.)

  On Windows platform, **syslog** is accepted as a word and is only useful along with the **−−syslog−target** option (the word has no effect otherwise).
- **off**, **emer**, **err**, **warn**, **info**, or **dbg**, to control the log level. Messages of the given severity or higher will be logged, and messages of lower severity will be filtered out. **off** filters out all messages. See **ovs−appctl**(8) for a definition of each log level.

Case is not significant within *spec*.

Regardless of the log levels set for **file**, logging to a file will not take place unless **−−log−file** is also specified (see below).

For compatibility with older versions of OVS, **any** is accepted as a word but has no effect.

−v
−−verbose
Sets the maximum logging verbosity level, equivalent to **−−verbose=dbg**.

−vPATTERN:*destination***:***pattern*
−−verbose=PATTERN:*destination***:***pattern*
Sets the log pattern for *destination* to *pattern*. Refer to **ovs−appctl**(8) for a description of the valid syntax for *pattern*.

−vFACILITY:*facility*
−−verbose=FACILITY:*facility*
Sets the RFC5424 facility of the log message. *facility* can be one of **kern**, **user**, **mail**, **daemon**, **auth**, **syslog**, **lpr**, **news**, **uucp**, **clock**, **ftp**, **ntp**, **audit**, **alert**, **clock2**, **local0**, **local1**, **local2**, **local3**, **local4**, **local5**, **local6** or **local7**. If this option is not specified, **daemon** is used as the default for the local system syslog and **local0** is used while sending a message to the target provided via the **−−syslog−target** option.

−−log−file[**=***file*]
Enables logging to a file. If *file* is specified, then it is used as the exact name for the log file. The default log file name used if *file* is omitted is **/usr/local/var/log/openvswitch/ovsdb−server.log**.

−−syslog−target=*host***:***port*
Send syslog messages to UDP *port* on *host*, in addition to the system syslog. The *host* must be a numerical IP address, not a hostname.

−−syslog−method=*method*

Specify *method* how syslog messages should be sent to syslog daemon. Following forms are supported:

- **libc**, use libc **syslog()** function. Downside of using this options is that libc adds fixed prefix to every message before it is actually sent to the syslog daemon over **/dev/log** UNIX domain socket.
- **unix:***file*, use UNIX domain socket directly. It is possible to specify arbitrary message format with this option. However, **rsyslogd 8.9** and older versions use hard coded parser function anyway that limits UNIX domain socket use. If you want to use arbitrary message format with older **rsyslogd** versions, then use UDP socket to localhost IP address instead.
- **udp:***ip*:*port*, use UDP socket. With this method it is possible to use arbitrary message format also with older **rsyslogd**. When sending syslog messages over UDP socket extra precaution needs to be taken into account, for example, syslog daemon needs to be configured to listen on the specified UDP port, accidental iptables rules could be interfering with local syslog traffic and there are some security considerations that apply to UDP sockets, but do not apply to UNIX domain sockets.
- **null**, discards all messages logged to syslog.

The default is taken from the **OVS_SYSLOG_METHOD** environment variable; if it is unset, the default is **libc**.

Active-Backup Options

These options support the **ovsdb−server** active-backup service model and database replication. These options apply only to databases in the format used for standalone and active-backup databases, which is the database format created by **ovsdb−tool create**. By default, when it serves a database in this format, **ovsdb−server** runs as a standalone server. These options can configure it for active-backup use:

- Use **−−sync−from=***server* to start the server in the backup role, replicating data from *server*. When **ovsdb−server** is running as a backup server, it rejects all transactions that can modify the database content, including lock commands. The same form can be used to configure the local database as a replica of *server*.
- Use **−−sync−from=***server* **−−active** to start the server in the active role, but prepared to switch to the backup role in which it would replicate data from *server*. When **ovsdb−server** runs in active mode, it allows all transactions, including those that modify the database.

At runtime, management commands can change a server's role and otherwise manage active-backup features. See **Active-Backup Commands**, below, for more information.

−−sync−from=*server*

Sets up **ovsdb−server** to synchronize its databases with the databases in *server*, which must be an active connection method in one of the forms documented in **ovsdb−client**(1). Every transaction committed by *server* will be replicated to **ovsdb−server**. This option makes **ovsdb−server** start as a backup server; add **−−active** to make it start as an active server.

−−sync−exclude−tables=*db***:***table*[**,***db***:***table*]...

Causes the specified tables to be excluded from replication.

−−active

By default, **−−sync−from** makes **ovsdb−server** start up as a backup for *server*. With **−−active**, however, **ovsdb−server** starts as an active server. Use this option to allow the syncing options to be specified using command line options, yet start the server, as the default, active server. To switch the running server to backup mode, use **ovs-appctl(1)** to execute the **ovsdb−server/connect−active−ovsdb−server** command.

These options are mutually exclusive with the **−−config−file**.

Public Key Infrastructure Options

The options described below for configuring the SSL public key infrastructure accept a special syntax for obtaining their configuration from the database. If any of these options is given **db:***db*,*table*,*column* as its argument, then the actual file name is read from the specified *column* in *table* within the *db* database. The *column* must have type string or set of strings. The first nonempty string in the table is taken as the file name. (This means that ordinarily there should be at most one row in *table*.)

−p *privkey.pem*
−−private−key=*privkey.pem*

Specifies a PEM file containing the private key used as **ovsdb−server**'s identity for outgoing SSL/TLS connections.

−c *cert.pem*
−−certificate=*cert.pem*

Specifies a PEM file containing a certificate that certifies the private key specified on **−p** or **−−private−key** to be trustworthy. The certificate must be signed by the certificate authority (CA) that the peer in SSL/TLS connections will use to verify it.

−C *cacert.pem*
−−ca−cert=*cacert.pem*

Specifies a PEM file containing the CA certificate that **ovsdb−server** should use to verify certificates presented to it by SSL/TLS peers. (This may be the same certificate that SSL/TLS peers use to verify the certificate specified on **−c** or **−−certificate**, or it may be a different one, depending on the PKI design in use.)

−C none
−−ca−cert=none

Disables verification of certificates presented by SSL/TLS peers. This introduces a security risk, because it means that certificates cannot be verified to be those of known trusted hosts.

−−bootstrap−ca−cert=*cacert.pem*

When *cacert.pem* exists, this option has the same effect as **−C** or **−−ca−cert**. If it does not exist, then **ovsdb−server** will attempt to obtain the CA certificate from the SSL/TLS peer on its first SSL/TLS 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/TLS connections must be authenticated by a certificate signed by the CA certificate thus obtained.

This option exposes the SSL/TLS connection to a man-in-the-middle attack obtaining the initial CA certificate, but it may be useful for bootstrapping.

This option is only useful if the SSL/TLS peer sends its CA certificate as part of the SSL/TLS certificate chain. SSL/TLS protocols do not require the server to send the CA certificate.

This option is mutually exclusive with **−C** and **−−ca−cert**.

−−peer−ca−cert=*peer-cacert.pem*

Specifies a PEM file that contains one or more additional certificates to send to SSL/TLS peers. *peer-cacert.pem* should be the CA certificate used to sign **ovsdb−server**'s own certificate, that is, the certificate specified on **−c** or **−−certificate**. If **ovsdb−server**'s certificate is self-signed, then **−−certificate** and **−−peer−ca−cert** should specify the same file.

This option is not useful in normal operation, because the SSL/TLS peer must already have the CA certificate for the peer to have any confidence in **ovsdb−server**'s identity. However, this offers a way for a new installation to bootstrap the CA certificate on its first SSL/TLS connection.

SSL Connection Options

−−ssl−protocols=*protocols*

Specifies a range or a comma- or space-delimited list of the SSL/TLS protocols **ovsdb−server** will enable for SSL/TLS connections. Supported *protocols* include **TLSv1** (deprecated), **TLSv1.1** (deprecated), **TLSv1.2** and **TLSv1.3**. Ranges can be provided in a form of two protocol names separated with a dash, or as a single protocol name with a plus sign. For example, use

TLSv1.1-TLSv1.3 to allow **TLSv1.1**, **TLSv1.2** and **TLSv1.3**. Use **TLSv1.2+** to allow **TLSv1.2** and any later protocol. The option accepts a list of protocols or exactly one range. The range is a preferred way of specifying protocols and the option always behaves as if the range between the minimum and the maximum specified version is provided, i.e., if the option is set to **TLSv1.1,TLSv1.3**, the **TLSv1.2** will also be enabled as if it was a range. Regardless of order, the highest protocol supported by both sides will be chosen when making the connection. The default when this option is omitted is **TLSv1.2** or later.

−−ssl−ciphers=*ciphers*
Specifies, in OpenSSL cipher string format, the ciphers **ovsdb−server** will support for SSL/TLS connections with TLSv1.2 and earlier. The default when this option is omitted is **DEFAULT:@SECLEVEL=2**.

−−ssl−ciphersuites=*ciphersuites*
Specifies, in OpenSSL ciphersuite string format, the ciphersuites **ovsdb−server** will support for SSL/TLS connections with TLSv1.3 and later. Default value from OpenSSL will be used when this option is omitted.

Other Options

−−unixctl=*socket*
Sets the name of the control socket on which **ovsdb−server** listens for runtime management commands (see **RUNTIME MANAGEMENT COMMANDS**, below). If *socket* does not begin with **/**, it is interpreted as relative to **/usr/local/var/run/openvswitch**. If **−−unixctl** is not used at all, the default socket is **/usr/local/var/run/openvswitch/ovsdb−server.***pid***.ctl**, where *pid* is **ovsdb−server**'s process ID.

On Windows a local named pipe is used to listen for runtime management commands. A file is created in the absolute path as pointed by *socket* or if **−−unixctl** is not used at all, a file is created as **ovsdb−server.ctl** in the configured *OVS_RUNDIR* directory. The file exists just to mimic the behavior of a Unix domain socket.

Specifying **none** for *socket* disables the control socket feature.

−−record[=*directory*]
Sets the process in "recording" mode, in which it will record all the connections, data from streams (Unix domain and network sockets) and some other important necessary bits, so they could be replayed later. Recorded data is stored in replay files in specified *directory*. If *directory* does not begin with **/**, it is interpreted as relative to **/usr/local/var/run/openvswitch**. If *directory* is not specified, **/usr/local/var/run/openvswitch** will be used.

−−replay[=*directory*]
Sets the process in "replay" mode, in which it will read information about connections, data from streams (Unix domain and network sockets) and some other necessary bits directly from replay files instead of using real sockets. Replay files from the *directory* will be used. If *directory* does not begin with **/**, it is interpreted as relative to **/usr/local/var/run/openvswitch**. If *directory* is not specified, **/usr/local/var/run/openvswitch** will be used.

−h
−−help Prints a brief help message to the console.

−V
−−version
Prints version information to the console.

RUNTIME MANAGEMENT COMMANDS

ovs−appctl(8) can send commands to a running **ovsdb−server** process. The currently supported commands are described below.

ovsdb−server Commands
These commands are specific to **ovsdb−server**.

exit Causes **ovsdb−server** to gracefully terminate.

ovsdb−server/compact [*db*]
Compacts database *db* in-place. If *db* is not specified, compacts every database in-place. A database is also compacted automatically when a transaction is logged if it is over 2 times as large as its previous compacted size (and at least 10 MB), but not before 100 commits have been added or 10 minutes have elapsed since the last compaction. It will also be compacted automatically after 24 hours since the last compaction if 100 commits were added regardless of its size.

ovsdb−server/memory-trim-on-compaction *on*|*off*
If this option is *on*, ovsdb-server will try to reclaim all unused heap memory back to the system after each successful database compaction to reduce the memory consumption of the process. *off* by default.

ovsdb−server/reconnect
Makes **ovsdb−server** drop all of the JSON−RPC connections to database clients and reconnect.

This command might be useful for debugging issues with database clients.

ovsdb−server/add−remote *remote*
Adds a remote, as if **−−remote=***remote* had been specified on the **ovsdb−server** command line. (If *remote* is already a remote, this command succeeds without changing the configuration.)

Mutually exclusive with the **−−config−file** option.

ovsdb−server/remove−remote *remote*
Removes the specified *remote* from the configuration, failing with an error if *remote* is not configured as a remote. This command only works with remotes that were named on **−−remote** or **ovsdb−server/add−remote**, that is, it will not remove remotes added indirectly because they were read from the database by configuring a **db:***db***,***table***,***column* remote. (You can remove a database source with **ovsdb−server/remove−remote db:***db***,***table***,***column*, but not individual remotes found indirectly through the database.)

Mutually exclusive with the **−−config−file** option.

ovsdb−server/list−remotes
Outputs a list of the currently configured remotes named on **−−remote** or **ovsdb−server/add−remote**, that is, it does not list remotes added indirectly because they were read from the database by configuring a **db:***db***,***table***,***column* remote.

ovsdb−server/add−db *database*
Adds the *database* to the running **ovsdb−server**. *database* could be a database file or a relay description in the following format: *relay:schema_name:remote*. The database file must already have been created and initialized using, for example, **ovsdb−tool create**.

Mutually exclusive with the **−−config−file** option.

ovsdb−server/remove−db *database*
Removes *database* from the running **ovsdb−server**. *database* must be a database name as listed by **ovsdb-server/list−dbs**.

If a remote has been configured that points to the specified *database* (e.g. **−−remote=db:***database***,...** on the command line), then it will be disabled until another database with the same name is added again (with **ovsdb−server/add−db**).

Any public key infrastructure options specified through this database (e.g. **−−private−key=db:***database,...* on the command line) will be disabled until another database with the same name is added again (with **ovsdb−server/add−db**).

Mutually exclusive with the **−−config−file** option.

ovsdb−server/list−dbs
Outputs a list of the currently configured databases added either through the command line or through the **ovsdb−server/add−db** command.

ovsdb−server/tlog−set *database*:*table on*|*off*
Enables or disables logging of all operations executed on the specified database and table. Logs are generated at INFO level and are rate limtied.

ovsdb−server/tlog−list
Displays the logging state for all currently configured databases and tables.

Active-Backup Commands
These commands query and update the role of **ovsdb−server** within an active-backup pair of servers. See **Active-Backup Options**, above, and **Active-Backup Database Service Model** in **ovsdb**(7) for more information.

All **Active-Backup Commands** that change the state of **ovsdb−server** are mutually exclusive with the **−−config−file** option.

ovsdb−server/set−active−ovsdb−server *server*
Sets the active *server* from which **ovsdb−server** connects through **ovsdb−server/connect−active−ovsdb−server**. This overrides the **−−sync−from** command-line option.

ovsdb−server/get−active−ovsdb−server
Gets the active server from which **ovsdb−server** is currently synchronizing its databases.

ovsdb−server/connect−active−ovsdb−server
Switches the server to a backup role. The server starts synchronizing its databases with the active server specified by **ovsdb−server/set−active−ovsdb−server** (or the **−−sync−from** command-line option) and closes all existing client connections, which requires clients to reconnect.

ovsdb−server/disconnect−active−ovsdb−server
Switches the server to an active role. The server stops synchronizing its databases with an active server and closes all existing client connections, which requires clients to reconnect.

ovsdb−server/set−active−ovsdb−server−probe−interval *probe interval*
Sets the probe interval (in milli seconds) for the connection to active *server*.

ovsdb−server/set−sync−exclude−tables *db*:*table*[,*db*:*table*]...
Sets the *table* within *db* that will be excluded from synchronization. This overrides the **−−sync−exclude-tables** command-line option.

ovsdb−server/get−sync−exclude−tables
Gets the tables that are currently excluded from synchronization.

ovsdb−server/sync−status
Prints a summary of replication run time information. The **state** information is always provided, indicating whether the server is running in the *active* or the *backup* mode. For all databases with active-backup service model, replication connection status, which can be either *connecting*, *replicating* or *error*, are shown. When the connection is in *replicating* state, further output shows the tables that are currently excluded from replication.

Cluster Commands
These commands support the **ovsdb−server** clustered service model. They apply only to databases in the format used for clustered databases, which is the database format created by **ovsdb−tool create−cluster** and **ovsdb−tool join−cluster**.

cluster/cid *db*
Prints the cluster ID for *db*, which is a UUID that identifies the cluster. If *db* is a database newly created by **ovsdb−tool cluster−join** that has not yet successfully joined its cluster, and **−−cid** was not specified on the **cluster−join** command line, then this command will report an error because the cluster ID is not yet known.

cluster/sid *db*
Prints the server ID for *db*, which is a UUID that identifies this server within the cluster.

cluster/status *db*
Prints this server's status within the cluster and the status of its connections to other servers in the cluster.

cluster/leave *db*
This command starts the server gracefully removing itself from its cluster. At least one server must remain, and the cluster must be healthy, that is, over half of the cluster's servers must be up.

When the server successfully leaves the cluster, it stops serving *db*, as if **ovsdb−server/remove−db** *db* had been executed.

Use **ovsdb−client wait** (see **ovsdb−client**(1)) to wait until the server has left the cluster.

Once a server leaves a cluster, it may never rejoin it. Instead, create a new server and join it to the cluster.

Note that removing the server from the cluster alters the total size of the cluster. For example, if you remove two servers from a three server cluster, then the "cluster" becomes a single functioning server. This does not result in a three server cluster that lacks quorum.

cluster/kick *db server*
Start graceful removal of *server* from *db*'s cluster, like **cluster/leave**, except that it can remove any server, not just this one.

server may be a server ID, as printed by **cluster/sid**, or the server's local network address as passed to **ovsdb-tool**'s **create−cluster** or **join−cluster** command. Use **cluster/status** to see a list of cluster members.

cluster/change-election-timer *db time*
Change the leader election timeout base value of the cluster, in milliseconds.

Leader election will be initiated by a follower if there is no heartbeat received from the leader within this time plus a random time within 1 second.

The default value is 1000, if not changed with this command. This command can be used to adjust the value when necessary, according to the expected load and response time of the servers.

This command must be executed on the leader. It initiates the change to the cluster. To see if the change takes effect (committed), use **cluster/status** to show the current setting. Once a change is committed, it persists at server restarts.

cluster/set−backlog−threshold *db n_msgs n_bytes*
Sets the backlog limits for *db*'s RAFT connections to a maximum of *n_msgs* messages or *n_bytes* bytes. If the backlog on one of the connections reaches the limit, it will be disconnected (and re-established). Values are checked only if the backlog contains more than 50 messages.

VLOG COMMANDS

These commands manage **ovsdb−server**'s logging settings.

vlog/set [*spec*]
Sets logging levels. Without any *spec*, sets the log level for every module and destination to **dbg**. Otherwise, *spec* is a list of words separated by spaces or commas or colons, up to one from each category below:

- A valid module name, as displayed by the **vlog/list** command on **ovs−appctl**(8), limits the log level change to the specified module.
- **syslog**, **console**, or **file**, to limit the log level change to only to the system log, to the console, or to a file, respectively.

  On Windows platform, **syslog** is accepted as a word and is only useful along with the **−−syslog−target** option (the word has no effect otherwise).

- **off**, **emer**, **err**, **warn**, **info**, or **dbg**, to control the log level. Messages of the given severity or higher will be logged, and messages of lower severity will be filtered out. **off** filters out all messages. See **ovs−appctl**(8) for a definition of each log level.

Case is not significant within *spec*.

Regardless of the log levels set for **file**, logging to a file will not take place unless **ovsdb−server** was invoked with the **−−log−file** option.

For compatibility with older versions of OVS, **any** is accepted as a word but has no effect.

vlog/set PATTERN:*destination***:***pattern*
Sets the log pattern for *destination* to *pattern*. Refer to **ovs−appctl**(8) for a description of the valid syntax for *pattern*.

vlog/list
Lists the supported logging modules and their current levels.

vlog/list-pattern
Lists logging patterns used for each destination.

vlog/close
Causes **ovsdb−server** to close its log file, if it is open. (Use **vlog/reopen** to reopen it later.)

vlog/reopen
Causes **ovsdb−server** to close its log file, if it is open, and then reopen it. (This is useful after rotating log files, to cause a new log file to be used.)

This has no effect unless **ovsdb−server** was invoked with the **−−log−file** option.

vlog/disable−rate−limit [*module*]...
vlog/enable−rate−limit [*module*]...
By default, **ovsdb−server** limits the rate at which certain messages can be logged. When a message would appear more frequently than the limit, it is suppressed. This saves disk space, makes logs easier to read, and speeds up execution, but occasionally troubleshooting requires more detail. Therefore, **vlog/disable−rate−limit** allows rate limits to be disabled at the level of an individual log module. Specify one or more module names, as displayed by the **vlog/list** command. Specifying either no module names at all or the keyword **any** disables rate limits for every log module.

The **vlog/enable−rate−limit** command, whose syntax is the same as **vlog/disable−rate−limit**, can be used to re-enable a rate limit that was previously disabled.

MEMORY COMMANDS

These commands report memory usage.

memory/show
Displays some basic statistics about **ovsdb−server**'s memory usage. **ovsdb−server** also logs this information soon after startup and periodically as its memory consumption grows.

COVERAGE COMMANDS

These commands manage **ovsdb−server**'s ''coverage counters,'' which count the number of times particular events occur during a daemon's runtime. In addition to these commands, **ovsdb−server** automatically logs coverage counter values, at **INFO** level, when it detects that the daemon's main loop takes unusually long to run.

Coverage counters are useful mainly for performance analysis and debugging.

coverage/show
Displays the averaged per-second rates for the last few seconds, the last minute and the last hour, and the total counts of all of the coverage counters.

coverage/read-counter *counter*
Displays the total count for the given coverage *counter*.

BUGS

In Open vSwitch before version 2.4, when **ovsdb−server** sent JSON-RPC error responses to some requests, it incorrectly formulated them with the **result** and **error** swapped, so that the response appeared to indicate success (with a nonsensical result) rather than an error. The requests that suffered from this problem were:

transact
get_schema
: Only if the request names a nonexistent database.

monitor
lock
unlock In all error cases.

Of these cases, the only error that a well-written application is likely to encounter in practice is **monitor** of tables or columns that do not exist, in an situation where the application has been upgraded but the old database schema is still temporarily in use. To handle this situation gracefully, we recommend that clients should treat a **monitor** response with a **result** that contains an **error** key-value pair as an error (assuming that the database being monitored does not contain a table named **error**).

SEE ALSO

ovsdb(7), **ovsdb−tool**(1), **ovsdb−server**(5), **ovsdb−server**(7).