NAME

ovn-controller-vtep – Open Virtual Network local controller for vtep enabled physical switches.

SYNOPSIS

ovn−controller−vtep [*options*] [*−−vtep-db=vtep-database*] [*−−ovnsb-db=ovnsb-database*]

DESCRIPTION

ovn−controller−vtep is the local controller daemon in OVN, the Open Virtual Network, for VTEP enabled physical switches. It connects up to the OVN Southbound database (see **ovn−sb**(5)) over the OVSDB protocol, and down to the VTEP database (see **vtep**(5)) over the OVSDB protocol.

PKI Options

PKI configuration is required in order to use SSL for the connections to the VTEP and Southbound databases.

−p *privkey.pem*
−−private−key=*privkey.pem*
Specifies a PEM file containing the private key used as identity for outgoing SSL 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 connections will use to verify it.

−C *cacert.pem*
−−ca−cert=*cacert.pem*
Specifies a PEM file containing the CA certificate for verifying certificates presented to this program by SSL peers. (This may be the same certificate that SSL 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 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 the executable will attempt to obtain the CA certificate from the SSL peer 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, but it may be useful for bootstrapping.

This option is only useful if the SSL peer sends its CA certificate as part of the SSL certificate chain. The SSL protocol does 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 peers. *peer-cacert.pem* should be the CA certificate used to sign the program's own certificate, that is, the certificate specified on **−c** or **−−certificate**. If the program'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 peer must already have the CA certificate for the peer to have any confidence in the program's identity. However, this

offers a way for a new installation to bootstrap the CA certificate on its first SSL connection.

CONFIGURATION

ovn−controller−vtep retrieves its configuration information from both the ovnsb and the vtep database. If the database locations are not given from command line, the default is the **db.sock** in local OVSDB's 'run' directory. The datapath location must take one of the following forms:

- **ssl:***host***:***port*

  The specified SSL *port* on the give *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, then wrap *host* with square brackets, e.g.: **ssl:[::1]:6640**. The **−−private−key**, **−−certificate** and either of **−−ca−cert** or **−−bootstrap−ca−cert** options are mandatory when this form is used.

- **tcp:***host***:***port*

  Connect to the given TCP *port* on *host*, where *host* can be a DNS name (if built with unbound library) or IP address (IPv4 or IPv6). If *host* is an IPv6 address, then wrap *host* with square brackets, e.g.: **tcp:[::1]:6640**.

- **unix:***file*

  On POSIX, connect to the Unix domain server socket named *file*.

  On Windows, connect to a localhost TCP port whose value is written in *file*.