MySQL (/sql/docs/mysql/configure-ssl-instance) | **PostgreSQL** | SQL Server (/sql/docs/sqlserver/configure-ssl-instance)

This page describes how to configure an instance to use SSL/TLS, and how to manage your server and client certificates.

For more information about using SSL/TLS with PostgreSQL, see SSL Support (https://www.postgresql.org/docs/9.6/static/libpq-ssl.html) in the PostgreSQL documentation.

Cloud SQL supports connecting to an instance using the Transport Layer Security (SSL/TLS) protocol. If you are connecting to an instance using its public IP address, you should use SSL/TLS, so that the data you send to and receive from Cloud SQL is secure.

If you are connecting using private IP, configuring SSL/TLS is optional; private services access (/vpc/docs/private-access-options#service-networking) traffic stays within Google's network at all times.

Cloud SQL uses a self-signed, per-instance server certificate and a certificate (public/private key pair) on the client (for example, an external application accessing the Cloud SQL instance). These certificates work together to enable the server (instance) and client (application) to encrypt their communication. You must have both a valid server certificate and a valid client certificate (key pair) to support encrypted communication.

SSL/TLS is needed to provide security when you connect to Cloud SQL using IP addresses. If you are connecting to y ce only by using the Cloud SQL Proxy (/sql/docs/postgres/sql-proxy) or the Java Socket Library docs/postgres/connect-external-app#java), you do not need to configure your instance to use SSL/TLS. Connection App Engine applications are encrypted by default whether you configure SSL/TLS for the instance or not.

Setting up your Cloud SQL instance to accept SSL/TLS connections enables SSL/TLS connections for the instance, but unsecure connections are still accepted. If you do not require SSL/TLS for all connections, clients without a valid certificate are allowed to connect. For this reason, if you are

accessing your instance using IP, it is strongly recommended that you enforce SSL for all connections.

Connections to your instance through the Cloud SQL Proxy are encrypted whether you configure or enforce SSL/TLS or not. SSL/TLS configuration affects only connections made using IP addresses.

To enforce SSL/TLS for all connections to your instance:

Cloud SQL creates a server certificate automatically when you create your instance. As long as the server certificate is valid, you do not need to actively manage your server certificate. However, the certificate has an expiration date; after that date, it is no longer valid, and clients are not able to establish a secure connection to your instance using that certificate.

Cloud SQL provides a way to rotate your server certificate, so that a new certificate can be seamlessly swapped in before the old certificate expires.

About three months before the server certificate expires for a Cloud SQL instance, the project owners will receive an email from Cloud SQL, telling them that the certificate rotation process has been started for that instance. The email provides the name of the instance, and informs the project owners that a new server certificate has been added to the project. The existing server certificate, as well as any client certificates, continue to function normally. In effect, the instance has two server certificates during this time.

The project administrator, or someone with the proper permissions, downloads a new server certificate PEM file, which contains the certificate information for both the current and the new server certificates. Someone must then update all clients that access the Cloud SQL instance using SSL/TLS to use the new PEM file.

After all clients have been updated, the project administrator tells Cloud SQL to rotate to the new server certificate. This means that the old server certificate is no longer recognized, and only the new server certificate can be used.

At this point, the rotation is complete, and no further action is required. The client certificates are unaffected by server certificate rotation.

If you've received a notice about your certificates expiring, or you have initiated a rotation, then you must take the following steps to complete the rotation:

1. Download the new server certificate information.

2. Update your clients to use the new server certificate information.
3. Complete the rotation, which moves the currently active certificate into the "previous" slot and updates the newly added certificate to be the active certificate.

After you complete a certificate rotation, your clients must all use the new certificate to connect to your Cloud SQL instance. If the clients were not updated properly to use the new certificate information, they will not be able to connect using SSL/TLS to your instance. If this happens, you can roll back to the previous certificate configuration.

A rollback operation moves the currently active certificate into the "upcoming" slot (replacing any current "upcoming" certificate). The "previous" certificate becomes the currently active certificate, returning your certificate configuration to the state it was in before you completed the rotation.

Certificate rollback is available only until the old certificate expires.

To roll back to the previous certificate configuration:

You do not need to wait for the email from Cloud SQL to start a rotation. You can start one at any time. When you start a rotation, a new certificate is created and placed into the "upcoming" slot. If a certificate was already in the "upcoming" slot, it is deleted; there can be only one upcoming certificate.

To initiate a rotation:

You can get information about your server certificate, such as when it expires or what level of encryption it provides.

You can completely reset your SSL/TLS configuration.

**n:** Performing this action removes the ability to connect to your instance using SSL/TLS until you recreate your clien
cates.

You can create up to 10 client certificates for each instance. If you lose the private key for a certificate, you must create a new one; the private key cannot be recovered.

At this point, you have:

- A server certificate saved as `server-ca.pem`.
- A client public key certificate saved as `client-cert.pem`.
- A client private key saved as `client-key.pem`.

Depending on which tool you use to connect, these three items are specified in different ways. For example, when connecting using psql command-line client, these three files are the values for the `sslrootcert`, `sslcert`, and `sslkey` parameters in the psql connection string. For an example

connection using psql client and SSL/TLS, see Connecting with psql Client (/sql/docs/postgres/connect-admin-ip#connect).

You can retrieve the public key portion of a client certificate. You cannot retrieve the private key, however. If you have lost your private key, you must create a new certificate.

- Connect to your instance (/sql/docs/postgres/connect-admin-ip#connect-ssl) with the psql client using SSL/TLS.
- Learn more about how PostgreSQL uses SSL/TLS (https://www.postgresql.org/docs/9.6/static/libpq-ssl.html).