

[Home](#)[MariaDB Platform](#)[Server](#)[MaxScale](#)[Analytics](#)[Galera Cluster](#)[Cor](#)[SECURITY](#) > [SECURING MARIADB](#) > [ENCRYPTION](#) >[Ask](#)[DATA-IN-TRANSIT ENCRYPTION](#)

# Enabling TLS on MariaDB Server

Instructions for configuring MariaDB Server to use TLS by setting system variables like ``ssl_cert``, ``ssl_key``, and ``ssl_ca`` in the configuration file.

## Overview

[MariaDB Server](#) and MariaDB Community Server support data-in-transit encryption, which secures data transmitted over the network. The server and the clients encrypt data using the Transport Layer Security (TLS) protocol, which is a newer version of the Secure Socket Layer (SSL) protocol.

TLS must be manually enabled on the server.

## Enabling TLS

1. Acquire an X509 certificate and a private key for the server.  
If it is a test or development server, then self-signed certificates and keys might be sufficient.
2. Determine which [system variables](#) and [options](#) you need to configure.

Mandatory system variables and options for TLS include:

System Variable/Option	Description
<a href="#">ssl_cert</a>	X509 cert in PEM format
<a href="#">ssl_key</a>	X509 key in PEM format
<a href="#">ssl_ca</a>	JCA file in PEM format

Useful system variables and options for TLS include:

System Variable/Option	Description
<a href="#">require_secure_transport</a>	When this option is enabled, connections attempted using insecure transport will be rejected. Secure transports are SSL/TLS, Unix sockets, or named pipes.
<a href="#">ssl_caphath</a>	CA directory
<a href="#">ssl_cipher</a>	SSL cipher to use
<a href="#">ssl_crl</a>	CRL file in PEM format
<a href="#">ssl_crlpath</a>	CRL directory
<a href="#">tls_version</a>	TLS protocol version for secure connections.

3. Choose a configuration file in which to configure your system variables and options.

It is not recommended to make custom changes to one of the bundled configuration files. Instead, it is recommended to create a custom configuration file in one of the included directories. Configuration files in included directories are read in alphabetical order. If you want your custom configuration file to override the bundled configuration files, then it is a good idea to prefix the custom configuration file's name with a string that will be sorted last, such as z-.

- On RHEL, CentOS, Rocky Linux, and SLES, a good custom configuration file would be: `/etc/my.cnf.d/z-custom-my.cnf`
- On Debian and Ubuntu, a good custom configuration file would be: `/etc/mysql/mariadb.conf.d/z-custom-my.cnf`

4. Set your system variables and options in the configuration file.  
They need to be set in a group that will be read by [MariaDB Server](#), such as [mariadb] or [server].  
For example:

```
[mariadb]
...
ssl_cert = /certs/server-cert.pem
ssl_key = /certs/server-key.pem
ssl_ca = /certs/ca-cert.pem
```

5. Restart the server.

```
$ sudo systemctl restart mariadb
```

6. Connect to the server using [MariaDB Client](#):

```
$ sudo mariadb
```

7. Confirm that TLS is enabled by confirming that the have\_ssl system variable is YES with the SHOW GLOBAL VARIABLES statement:

```
SHOW GLOBAL VARIABLES LIKE 'have_ssl';

+-----+-----+
| Variable_name | Value |
+-----+-----+
| have_ssl      | YES   |
+-----+-----+
```

*This page is: Copyright © 2025 MariaDB. All rights reserved.*

Previous

Certificate Creation with OpenSSL

Next

Requiring TLS on MariaDB Server

Last updated 28 days ago

Was this helpful?

😊

😐

😞



Products

- Enterprise Platform
- Community Server
- Download MariaDB
- Pricing

Support

- Customer Login
- Technical Support
- Remote DBA
- Professional Services

Resources

- MariaDB Blog
- Webinars
- Customer Stories
- MariaDB Events
- Documentation
- Developer Hub

Company

- About MariaDB
- Newsroom
- Leadership
- MariaDB Careers
- Legal
- Privacy Policy

© 2026 MariaDB. All rights reserved.