YOUR CUSTOMERS.
YOUR SOLUTIONS.

Tritium® Initial Setup Guide

Version 1.7
March 2022

Table Of Contents

1 How to use this document

The purpose of this document is to familiarize the reader with the requirements, steps, and other important information to assist with the initial setup of Tritium®.
Throughout this document, you will see references to commands, output, and settings that enable Tritium to run in a server environment. Here are a few ways this information is highlighted for easier reading:

1.1 Variables

To ensure the document maintains its validity across versions, rebranding, etc. we have utilized variables throughout the documentation. These variables appear in code blocks and inline snippets as capitalized words describing the content that should replace them.
For example, if you see a reference to a file with a name like `LICENSEE-VERSION.tgz`, the words LICENSEE and VERSION represent variables. In this case, the LICENSEE refers to the company name of the licensee and VERSION refers to the version number of the file.
The same process applies to code. If you come across terms in all-caps such as these, they should be replaced by the information that applies to the situation.

1.2 Inline code snippets

Commands and other information intended to be typed or entered into a file or command line appear in magenta when included within a text paragraph. For example, if the document mentions typing Hello World into the command line, it will appear as such.

1.3 Code blocks

Some information appears in grey boxes. This includes commands, output, and other information that would be entered into or read from either the command line or an individual file. This type of block is referred to as a code block.

```
Code blocks look like this. They contain input, output, and file data that is mentioned in the
document.
```

1.5 Optional sections

Not every deployment of Tritium is the same. Some features and components of Tritium aren't necessary for some situations. These optional chapters in the documentation are pointed out with the phrase **(if applicable)** in the chapter's main title.
For example, not all Tritium deployments require a hardware security module (HSM). It's for this reason that the Configuring Payment HSMs chapter includes this note in the title.

2 Introduction

The initial bootstrapping process involves multiple steps. These steps include:

- Loading the database schema
- Initializing the key server subsystem and generating application keys
- Loading program data
- Loading payment network keys
- Initial application deployment

This document provides details for each of these steps.
Once initially configured, the process for updating Tritium releases is quite straightforward.

2.1 Preparation

When you receive a Tritium software release, it comes preconfigured to run in each environment where you have provided details to Episode Six (E6). This means that for each environment, the software package is aware of the endpoints, such as database, hardware security modules (HSM), and key server IP address and ports which are required to operate in that environment.
Prior to running a Tritium release in a new environment, the following information should be provided to E6 so we can configure the software appropriately.

Description	IP address	Service port (defaults shown)
MySQL / MariaDB Database Server		3306
Tritium Key Server VIP		10000
Atalla HSM1		7000
Atalla HSM2		7000
Cassandra DB Servers		9042
C4 Reporting Servers		4550

It's also important to provide connection information for specific services (e.g., payment network endpoints) to be used in this environment as discussed with E6.

2.2 Runtime user

We recommend that Tritium runs under a dedicated user account, which must have the ability to write to the Tritium deployment directory for logging purposes. By convention, we call this runtime user e6tech, and it has its own group, also called e6tech.
Runtime users are defined as e6tech in this document.

3 Tritium infrastructure requirements

Tritium is Java software and is released as a software package with all necessary libraries for it to run. To run Tritium software, individual servers require an up-to-date Java 8 JRE.
On x86_64 systems, E6 tests with and recommends RedHat's build of OpenJDK 8.

To form a fully functioning system, a deployment of Tritium depends on 4 separate infrastructure components. These are:

- MySQL-compatible database
- Cassandra database
- Load balancer
- Static web server(s)

Here we provide configuration guidelines for each of these in turn.

3.1 MySQL-compatible database

Tritium stores all customer data used for online transaction processing and account servicing in a MySQL- compatible database. We say "MySQL-compatible" database because an ecosystem of compatible databases has sprung up around the original MySQL.
E6 develops and tests Tritium with recent versions of Oracle MySQL, and MariaDB. In addition, there are various cloud services that are MySQL compatible, and which may also be used with Tritium subject to compatibility testing.
Minimum versions of these database required to operate Tritium are MySQL 5.7, or MariaDB 10.2. Previous versions don't work correctly due to constraints they place on the schema.
For new implementations on RedHat/CentOS Linux servers, E6 currently recommends the latest release version of MariaDB 10.4 for the MySQL-compatible database. The rationale for this is that MariaDB pairs well with MariaDB MaxScale, which is a combined database query routing proxy, and database failover orchestration solution.
Users can start out with single instances of MariaDB database, then add in MaxScale for environments which have high availability requirements where the ability to detect and recover from failure is important.
Current versions of MariaDB are available for download from https://downloads.mariadb.org/.

3.1.1 MySQL server configuration

MariaDB should be installed and configured per the vendor guidance. Tritium does not require that the MySQL- compatible database be configured to use SSL/TLS for its connections, however we are able to configure Tritium to support this if it is a customer requirement.
E6 recommends the following changes be added to the server configuration on dedicated database servers:

```
{mysqld}

max_connections=2048
default_storage_engine=InnoDB

# Setting this to 0 results in a RPO of 1 second
# Setting to 1 is required for full ACID compliance innodb_flush_log_at_trx_commit=1

# Allow server to accept connections on all interfaces bind-address=0.0.0.0

# Disable the query cache
# Query cache support is being removed in MySQL 8 because it causes issues
query_cache_type=OFF
query_cache_size=0

# The following enables slow query logging by default, with a slow query # defined as one taking longer than 5 seconds.
slow-query-log=1
long_query_time=5

# Log all deadlocks
innodb_print_all_deadlocks=ON

# Don't use OS block device cache, let InnoDB handle caching instead
innodb_flush_method=O_DIRECT_NO_FSYNC

# Rules for configuring innodb_buffer_pool_size # These follow the current MySQL guidelines
# < 1GB: 128M
# <= 4GB: Physical RAM * 0.5 #
> 4GB: Physical RAM * 0.75
innodb_buffer_pool_size=128M
```

3.1.2 MySQL user configuration

The MySQL database should be configured with both a user account Tritium will use to connect and authenticate, and a database schema which will hold the Tritium data. By convention, the database user account is called ‘h3’, and the database schema is also called ‘h3’. For test instances, setting a password of ‘h3’ will allow Tritium to connect using E6’s standard development key vault.

The database user may be configured by a DBA user as follows, where PASSWORD is replaced by the password being set for the h3 user account:

```
CREATE USER h3;
ALTER USER h3 IDENTIFIED BY
'PASSWORD'; CREATE DATABASE h3;
GRANT ALL PRIVILEGES ON h3.* TO h3;
```

In the case where the development password is being set to h3, this becomes:

```
CREATE USER h3;
ALTER USER h3 IDENTIFIED BY 'h3';
CREATE DATABASE h3;
GRANT ALL PRIVILEGES ON h3.* TO h3;
```

3.2 Apache Cassandra database

Tritium uses an Apache Cassandra database for reporting purposes. This database may contain customer personal information but does not contain any PCI DSS CHD.
For new implementations on RedHat / CentOS Linux servers, E6 currently recommends the latest release version of Apache Cassandra 3.11 for the Cassandra database. It may also be possible to use cloud-based managed instances of Cassandra, subject to compatibility testing.
The Cassandra database should be configured per the vendor guidance. Of note, Cassandra uses a very different storage model to MySQL, where it has multiple storage nodes across which data is automatically split while maintaining a configured “replication factor” number of copies of the data. E6 recommends that at least 3 servers are used for the Cassandra instance, to allow data to be stored with a replication factor of 3.
Tritium does not require that the Cassandra database be configured to use SSL/TLS for its connections, however we are able to configure Tritium to support this if it is a customer requirement.
When a Cassandra cluster starts, the nodes learn the identity and IP address of the others. A client connecting to one of these nodes can discover the details of all nodes in the Cassandra cluster, and the client will then connect to each of these as needed to retrieve data. The IP address of one or more Cassandra nodes in the cluster should be provided to E6, to configure Tritium to use these for bootstrapping purposes.
Analogous to the user schema in a SQL database, all data within Apache Cassandra is stored within a keyspace. Tritium can create its own keyspace when it starts up, so unlike for the MySQL database, no keyspace or schema preconfiguration is required.


episodesix