Oracle MySQL and Entrust KeyControl

with nShield® HSM Integration Guide
1. Introduction .................................................................... 4
   1.1. Requirements ....................................................... 4
   1.2. High-availability considerations ............................... 4
   1.3. Product configuration ............................................. 4
2. Procedures ...................................................................... 5
   2.1. Installation overview .............................................. 5
   2.2. Install the Entrust KeyControl Server ...................... 5
   2.3. Configure the nShield HSM in the KeyControl Server .. 6
   2.4. Configure the KeyControl Server ............................ 6
   2.5. Configure the KeyControl Server as a KMIP server .... 6
   2.6. Install the Oracle MySQL server ............................. 7
   2.7. Install the keyring_okv plugin ............................... 8
   2.8. Import the KeyControl KMIP Certificates to the keyring_okv plugin ........................................... 8
   2.9. Verify that the keyring_okv plugin is working .......... 10
   2.10. Use keyring_okv plugin to create encrypted tables ...... 10
   2.11. Test that encryption KeyControl is working ............ 11
   2.12. Secure the MySQL database ................................. 12
Contact Us ....................................................................... 15
1. Introduction

This document describes the configuration of Oracle MySQL Enterprise Server 8.0.25 for integration with the Entrust KeyControl (formerly HyTrust KeyControl) 5.3 key management solution. Oracle MySQL Enterprise Server is compatible with the Entrust KeyControl solution. Entrust KeyControl can serve as a key manager MySQL encryption by using the open standard Key Management Interoperability Protocol (KMIP).

1.1. Requirements

- Entrust KeyControl version 5.3 or later

  An Entrust KeyControl license is required for the installation. You can obtain this license from your Entrust KeyControl and Oracle MySQL account team or through Entrust KeyControl customer support.

- MySQL Enterprise Server 8.0.25 or later

1.2. High-availability considerations

The Entrust KeyControl solution uses an active-active deployment, which provides high-availability capability to manage encryption keys. We recommend this deployment configuration. In an active-active cluster, changes made to any KeyControl node in the cluster are automatically reflected on all nodes in the cluster. For information about the Entrust KeyControl solution, see the HyTrust KeyControl Product Overview.

1.3. Product configuration

The integration between the Oracle MySQL Enterprise Server, Entrust KeyControl, and nShield HSM has been successfully tested in the following configurations:

<table>
<thead>
<tr>
<th>Product</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle MySQL Enterprise Server</td>
<td>8.0.25</td>
</tr>
<tr>
<td>Entrust KeyControl</td>
<td>5.3</td>
</tr>
<tr>
<td>MySQL Keyring library</td>
<td>1.10</td>
</tr>
<tr>
<td>nShield client software</td>
<td>12.60.11</td>
</tr>
<tr>
<td>nShield Connect XC</td>
<td>12.50.11(12.60.10)</td>
</tr>
</tbody>
</table>
2. Procedures

2.1. Installation overview

1. Install the Entrust KeyControl server.
2. Configure the Entrust KeyControl server with high availability.
3. Generate a KMIP certificate for each controller/cluster.
4. Extract the signing certificates from the KeyControl server.
5. Install Oracle MySQL Enterprise Server.
6. Install the MySQL `keyring_okv` plugin.
7. Configure the KeyControl server as a KMIP server for MySQL.
8. Verify that the encryption is working and that Oracle MySQL is using KeyControl to manage the keys.

2.2. Install the Entrust KeyControl Server

The Entrust KeyControl server is a software solution deployed from an OVA or ISO image. We recommend that you read the HyTrust KeyControl Installation Overview to fully understand the KeyControl server deployment. To configure a KeyControl cluster (active-active configuration is recommended), as performed in this integration validation, we recommend the use of the OVA installation method for simplicity, as described in the HyTrust KeyControl OVA Installation instructions.

The KeyControl OVA must be deployed from the VCenter server, and not from an ESXi host.

After the KeyControl server is deployed, configure the first KeyControl node as described in the HyTrust Configuring the First KeyControl Node installation guide.

After completing this procedure, add the second node as described in the HyTrust Adding a New KeyControl Node to an Existing Cluster (OVA Installation) to create the recommended active-active cluster.

Although an active-active cluster is not a requirement, and a single KeyControl node can be deployed to perform the functions of KMIP, we highly recommends deploying the solution with a minimum of two nodes for an active-active cluster solution that instantiates a highly available and robust architecture.

Your KeyControl license determines how many KeyControl nodes you can have in a cluster. For full information about the KeyControl licensing, see the HyTrust Managing the
2.3. Configure the nShield HSM in the KeyControl Server

For instructions on how to integrate an nShield HSM with KeyControl, see the *Entrust KeyControl nShield HSM Integration Guide*.

2.4. Configure the KeyControl Server

After the Entrust KeyControl server is deployed and the initial installation is complete, you can configure the network settings, e-mail server preferences, and certificate configuration. For these procedures, see the *HyTrust KeyControl System Configuration admin guide*.

2.5. Configure the KeyControl Server as a KMIP server

To use external key management, MySQL requires an external key management server such as the Entrust KeyControl server. To configure the KeyControl server as a KMIP server, see the *HyTrust Configuring a KeyControl KMIP Server section of the admin guide*. When using external key management, as is the case in this solution, the KeyControl server is the KMIP server and Oracle MySQL is the KMIP client.

Certificates are required to facilitate the KMIP communications from the KeyControl server to Oracle MySQL and conversely.

To import certificates for use by KeyControl and MySQL, the simplest solution is to leverage the built-in capabilities in the KeyControl server to create and publish the certificates. To perform this operation, create the certificate bundle as described in the *Creating KMIP Client Certificate Bundles* section of the Entrust KeyControl admin guide.


After you create and download these certificates, you need to upload or import them into the MySQL server.

First we install the Oracle MySQL server.
2.6. Install the Oracle MySQL server

Installing the Oracle MySQL Enterprise Edition depends on the operating system on which you are installing it. See the Oracle documentation for details on how to install Oracle MySQL Enterprise Edition in your environment. The steps below were used to install and configure MySQL on a CentOS 7 Linux server.


2. Unzip the file and list the contents

   ```
   % unzip V1000260-01.zip
   % ls
   mysql-commercial-backup-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-client-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-client-plugins-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-common-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-devel-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-embedded-compat-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-libs-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-libs-compat-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-server-8.0.25-1.1.el7.x86_64.rpm
   mysql-commercial-test-8.0.25-1.1.el7.x86_64.rpm
   mysql-router-commercial-8.0.25-1.1.el7.x86_64.rpm
   README.txt
   ```

3. Update the server prior to the install

   ```
   % sudo yum update
   ```

4. Install MySQL using the following command:

   ```
   % sudo yum localinstall ./*.rpm
   ```

   During the installation, you might be prompted to enter the password for the root user for your MySQL installation.

5. Select **Use Strong password encryption**.

   If the password prompt does not appear, set the root password:

   1. Log in to the MySQL client

      ```
      % mysql -h 127.0.0.1 -uroot -p
      ```

   2. Enter the Temporary password created for the root user. You will find that in the mysqld.log file located under /var/log.

   3. Change the root password:
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY '<NEW_PASSWORD>';
mysql> FLUSH PRIVILEGES;
mysql> exit;

4. Test the new password

% mysql -u root -p<new_password>

2.7. Install the keyring_okv plugin

The `keyring_okv` plugin is a KMIP 1.1 plugin for KMIP-compatible back-end keyring storage products, such as Entrust KeyStore. It is available in MySQL Enterprise Edition distributions.

The configuration directory used by `keyring_okv` as the location for its support files should have a restrictive mode and be accessible only to the account used to run the MySQL server. For example, on Unix and Unix-like systems, to use the `/usr/local/mysql/mysql-keyring-okv` directory, the following commands, executed as root, create the directory and set its mode and ownership:

```
cd /usr/local
sudo mkdir -p mysql/mysql-keyring-okv/ssl
sudo chmod -R 750 mysql
sudo chown -R mysql mysql
sudo chgrp -R mysql mysql
```

To be usable during the server startup process, `keyring_okv` must be loaded using the `--early-plugin-load` option. Also, set the `keyring_okv_conf_dir` system variable to tell `keyring_okv` where to find its configuration directory. Edit the `/etc/my.cnf` file and add the plugin into the `mysqld` section:

```
[mysqld]
early-plugin-load=keyring_okv.so
keyring_okv_conf_dir=/usr/local/mysql/mysql-keyring-okv
```

2.8. Import the KeyControl KMIP Certificates to the keyring_okv plugin

The certificates must be installed before running the plugin, so that the plugin can be initialized.

The following files need to be imported:

- A `<cert_name>.pem` file that includes both the client certificate and private key. The
The administrator needs to open this single file and paste the two sections of the file into the `cert.pem` and `key.pem` files in the `/usr/local/mysql/mysql-keyring-okv/ssl` directory.

- The client certificate section of the `<cert_name>.pem` file includes the lines "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" and all text between them.

Open or create `/usr/local/mysql/mysql-keyring-okv/ssl/cert.pem` and paste "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" and all text between them into this file. Make sure it has a carriage return at the end of the file.

- The private key section of the `<cert_name>.pem` file includes the lines "-----BEGIN PRIVATE KEY-----" and "-----END PRIVATE KEY-----" and all text in between them.

Open or create `/usr/local/mysql/mysql-keyring-okv/ssl/key.pem` and paste "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" and all text between them into this file. Make sure it has a carriage return at the end of the file.

- A `cacert.pem` file, which is the root certificate for the KMS cluster. It is always named `cacert.pem`.

This file needs to be copied to `/usr/local/mysql/mysql-keyring-okv/ssl/CA.pem`.

1. In the configuration directory, create a file named `okvclient.ora`. It should have the following format:

```
SERVER=xxx.xxx.xxx.xxx:5696
STANDBY_SERVER=xxx.xxx.xxx.xxx:5696
```

STANDBY_SERVER is optional.

Example:

```
SERVER=198.51.100.20:5696
STANDBY_SERVER=198.51.100.21:5696
```

2. Set the permissions on these files

```
cd /usr/local/mysql/mysql-keyring-okv
sudo chmod -R 750 mysql
sudo chown -R mysql
sudo chgrp -R mysql
```

3. After completing the preceding procedure, restart the MySQL server. It loads the `keyring_okv` plugin and `keyring_okv` uses the files in its configuration directory to communicate with KeyControl.
2.9. Verify that the keyring_okv plugin is working

After configuration is complete and you restarted MySQL to load `keyring_okv`, look in the logs to make sure there are no errors in connecting to KeyControl. To verify plugin installation, with the MySQL server running, examine the `INFORMATION_SCHEMA.PLUGINS` table or use the `SHOW PLUGINS` statement. For example:

```
mysql> SELECT PLUGIN_NAME, PLUGIN_STATUS FROM INFORMATION_SCHEMA.PLUGINS WHERE PLUGIN_NAME LIKE 'keyring%';
+-------------+---------------+
| PLUGIN_NAME | PLUGIN_STATUS |
|-------------+---------------|
| keyring_okv | ACTIVE        |
+-------------+---------------+
1 row in set (0.00 sec)
```

2.10. Use keyring_okv plugin to create encrypted tables

When you create the first encrypted table, InnoDB will ask `keyring_okv` to generate primary key (AES-256) in KeyControl. You can check this in the KeyControl Web UI in `KMIP settings > Objects`.

His primary key is used to encrypt tablespace keys. InnoDB also asks KeyControl to generate a key (AES-256) for encrypting table. The tablespace key is wrapped using the primary key and stored alongside the encrypted table. For subsequent encrypted tables, only the tablespace key is generated and the same primary key is used to wrap the tablespace key.

With KeyControl you will see a complete audit trail if every time the primary key or tablespace key is retrieved. You will also have complete control on these keys and you can revoke access to a key or disable it, in case you want to lock down your data at rest.
Here is an example of how you create an encrypted table

```sql
CREATE DATABASE MySQL_TDE_Test;
USE MySQL_TDE_Test;
CREATE TABLE `test_encryption` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(15) NOT NULL,
  PRIMARY KEY (`id`) ENCRYPTION = 'Y';
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=latin1;
```

The Objects tab in the KeyControl UI as described in the Managing KMIP Objects section of the **HyTrust KeyControl admin guide**.

### 2.11. Test that encryption KeyControl is working

1. Insert a record to the table we created earlier.

   ```sql
   mysql> INSERT INTO test_encryption VALUES (1, 'cleandro');
   Query OK, 1 row affected (0.00 sec)
   mysql> select * from test_encryption;
   +----+----------+
   | id | name     |
   +----+----------+
   |  1 | cleandro |
   +----+----------+
   1 row in set (0.00 sec)
   ```

2. Edit the MySQL configuration file and disable the `keyring_okv` plugin.

   ```bash
   [root@oraclemysql-centos-7 etc]# vi my.cnf
   #early-plugin-load=keyring_okv.so
   #keyring_okv_conf_dir=/usr/local/mysql/mysql-keyring-okv
   ```

3. Restart MySQL.

   ```bash
   [root@oraclemysql-centos-7 etc]# sudo service mysqld restart
   Redirecting to /bin/systemctl restart mysqld.service
   ```

4. Check if you can read the encrypted table.

   ```sql
   mysql> use MySQL_TDE_Test;
   Reading table information for completion of table and column names
   You can turn off this feature to get a quicker startup with -A
   Database changed
   mysql> select * from test_encryption;
   ERROR 3185 (HY000): Can't find master key from keyring, please check in the server log if a keyring is loaded and initialized successfully.
   ```

   The table is not accessible because MySQL cannot get to the master key from the keyring.
5. Re-enable the keyring in the MySQL configuration file.

```
[root@oraclemysql-centos-7 etc]# vi my.cnf
early-plugin-load=keyring_okv.so
keyring_okv_conf_dir=/usr/local/mysql/mysql-keyring-okv
```

6. Restart MySQL.

```
[root@oraclemysql-centos-7 etc]# sudo service mysqld restart
Redirecting to /bin/systemctl restart mysqld.service
```

7. Check you can view the encrypted table.

```
mysql> use MySQL_TDE_Test;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```
```
Database changed
mysql> select * from test_encryption;
+----+----------+
| id | name     |
+----+----------+
|  1 | cleandro |
+----+----------+
1 row in set (0.00 sec)
```

2.12. Secure the MySQL database

The information below was taken from the following STIG page and can be used as guideline to address confidentiality and integrity of all information at rest in a MySQL database. For more information, see [InnoDB Data-at-Rest Encryption](#).

- **Group Title:** SRG-APP-000231-DB-000154
- **Rule Title:** The MySQL Database Server 8.0 must protect the confidentiality and integrity of all information at rest.
- **Discussion:**

  This control is intended to address the confidentiality and integrity of information at rest in non-mobile devices and covers user information and system information. Information at rest refers to the state of information when it is located on a secondary storage device (e.g., disk drive, tape drive) within an organizational information system. Applications and application users generate information throughout the course of their application use.

  User data generated, as well as application-specific configuration data, needs to be protected. Organizations may choose to employ different mechanisms to achieve confidentiality and integrity protections, as appropriate.
If the confidentiality and integrity of application data is not protected, the data will be open to compromise and unauthorized modification.

Apply appropriate controls to protect the confidentiality and integrity of data at rest in the database.

Using SQL determine if all data-at-rest is encrypted.

```sql
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE variable_name = 'audit_log_encryption';
```

If `audit_log_encryption` is not set to AES, this is a finding.

```sql
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE variable_name = 'binlog_encryption';
```

If `binlog_encrypt` is not set to ON, this is a finding.

```sql
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE variable_name = 'innodb_redo_log_encrypt';
```

If `innodb_redo_log_encrypt` is not set to ON, this is a finding.

```sql
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE variable_name = 'innodb_undo_log_encrypt';
```

If `innodb_undo_log_encrypt` is not set to ON, this is a finding.

```sql
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE variable_name = 'table_encryption_privilege_check';
```

If `innodb_redo_log_encrypt` is not set to ON, this is a finding.

Find encryption status for all MySQL table and tablespaces.

```sql
SELECT `INNODB_TABLESPACES`. `NAME`,
`INNODB_TABLESPACES`. `ENCRYPTION`
FROM `information_schema`. `INNODB_TABLESPACES`;
```

If any tablespace is not ENCRYPTION set to Y (yes), this is a finding.

```sql
SELECT VARIABLE_NAME, VARIABLE_VALUE
FROM performance_schema.global_variables
WHERE variable_name = 'general_log';
```

If `general_log` is not OFF, this is a finding.
Apply appropriate MySQL Database 8.0 controls to protect the confidentiality and integrity of data at rest in the database.

```sql
sudo vi /etc/my.cnf
[mysqld]
audit-log=FORCE_PLUS_PERMANENT
audit-log-format=JSON
audit-log-encryption=AES

#Turn on binlog encryption
set persist binlog_encryption=ON;

#Turn on undo and redo log encryption
set persist innodb_redo_log_encrypt=ON;
set persist innodb_undo_log_encrypt=ON;
```

Enable encryption for a new file-per-table tablespace, specify the ENCRYPTION option in a CREATE TABLE statement. The following example assumes that `innodb_file_per_table` is enabled.

```sql
mysql> CREATE TABLE t1 (c1 INT) ENCRYPTION='Y';
```

To enable encryption for an existing file-per-table tablespace, specify the ENCRYPTION option in an ALTER TABLE statement.

```sql
mysql> ALTER TABLE t1 ENCRYPTION='Y';
```

To disable encryption for file-per-table tablespace, set ENCRYPTION='N' using ALTER TABLE.

```sql
mysql> ALTER TABLE t1 ENCRYPTION='N';
```

Disable `general_log`.

```sql
SET PERSIST general_log = 'OFF';
```
## Contact Us

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Web site</strong></td>
<td><a href="https://www.entrust.com">https://www.entrust.com</a></td>
</tr>
<tr>
<td><strong>Support</strong></td>
<td><a href="https://nshieldsupport.entrust.com">https://nshieldsupport.entrust.com</a></td>
</tr>
<tr>
<td><strong>Email Support</strong></td>
<td><a href="mailto:nShield.support@entrust.com">nShield.support@entrust.com</a></td>
</tr>
<tr>
<td><strong>Online documentation:</strong></td>
<td>Available from the Support site listed above.</td>
</tr>
</tbody>
</table>

You can also contact our Support teams by telephone, using the following numbers:

### Europe, Middle East, and Africa

**United Kingdom:**

+44 1223 622444  
One Station Square  
Cambridge, UK CB1 2GA

### Americas

**Toll Free:**

+1 833 425 1990

**Fort Lauderdale:**

+1 954 953 5229  
Sawgrass Commerce Center – A Suite 130  
13800 NW 14 Street  
Sunrise, FL 33323 USA

### Asia Pacific

**Australia:**

+61 8 9126 9070  
World Trade Centre Northbank Wharf  
Siddeley St  
Melbourne VIC 3005 Australia

**Japan:**

+81 50 3196 4994

**Hong Kong:**

+852 3008 3188  
31/F, Hysan Place,  
500 Hennessy Road,  
Causeway Bay
ABOUT ENTRUST CORPORATION

Entrust keeps the world moving safely by enabling trusted identities, payments, and data protection. Today more than ever, people demand seamless, secure experiences, whether they’re crossing borders, making a purchase, accessing e-government services, or logging into corporate networks. Entrust offers an unmatched breadth of digital security and credential issuance solutions at the very heart of all these interactions. With more than 2,500 colleagues, a network of global partners, and customers in over 150 countries, it’s no wonder the world’s most entrusted organizations trust us.