

Integration Guide

USING DATA SECURITY MANAGER FOR PERCONA MSSQL ENCRYPTION AT REST

VERSION 1.0

TABLE OF CONTENTS

1.0	INTRODUCTION	2
1.1	Overview	2
2.0	INTEGRATION STEPS	3
2.1	Create an App in Fortanix DSM	3
2.2	Creating a Client Certificate with Custom OID Value	4
2.3	Creating a Client Certificate with App ID as CN	8
2.4	Setting App Authentication Method as Certificate	10
3.0	CONFIGURING ENCRYPTION IN PERCONA MYSQL	11
3.1	Keyring Component Installation	11
3.1.1	Creating Manifest for the Keyring Component	11
3.1.2	Creating a Configuration File	12
3.2	Creating Encrypted Tables	13
4.0	DOCUMENT INFORMATION	15
4.1	Document Location	15
4.2	Document Updates	15

1.0 INTRODUCTION

This article describes how to set up an app in Fortanix Data Security Manager (DSM) for Percona MySQL to integrate with Fortanix DSM. It also contains the information to:

- Create an app in Fortanix DSM.
- Create a client certificate with custom Object Identifier (OID) value.
- Create a client certificate with App ID as Common Name (CN).
- Configure Encryption in Percona MySQL.

1.1 OVERVIEW

Percona is a leading provider of open-source databases like MySQL, MongoDB, PostgreSQL, and so on. Percona Server for MYSQL DB is fully compatible with MySQL and all its capabilities are equated with those of the MYSQL Enterprise Version.

Percona Server for MySQL 8.0.27-18 has added support for the [OASIS Key Management Interoperability Protocol \(KMIP\)](#), making it possible for MYSQL data encryption at rest keys to be stored in external key management systems.

Unlike MYSQL data encryption at rest, here the keyring component can be installed using a manifest file directly. The server uses a manifest and the component consults its configuration file during initialization.


Cryptographically secure generation and secure management of encryption keys are required for the true security of data at rest encrypted by MySQL. Fortanix Data Security Manager (DSM) with its KMIP support provides a secure and flexible solution.

Percona Server for MySQL KMIP authenticates to a KMIP-enabled key management server using the client certificate. Fortanix DSM supports clients/apps to authenticate using an API Key, App ID, and Certificate, or only a Certificate.

2.0 INTEGRATION STEPS

2.1 CREATE AN APP IN FORTANIX DSM

Start by adding an app in Fortanix DSM in an appropriate group or a new group. For instructions on how to add a group or app please see the Getting Started Guide.

1. Log in to the Fortanix DSM UI.
2. Click the **Apps** tab. On the Apps page click the create a new app icon  to create a new app.

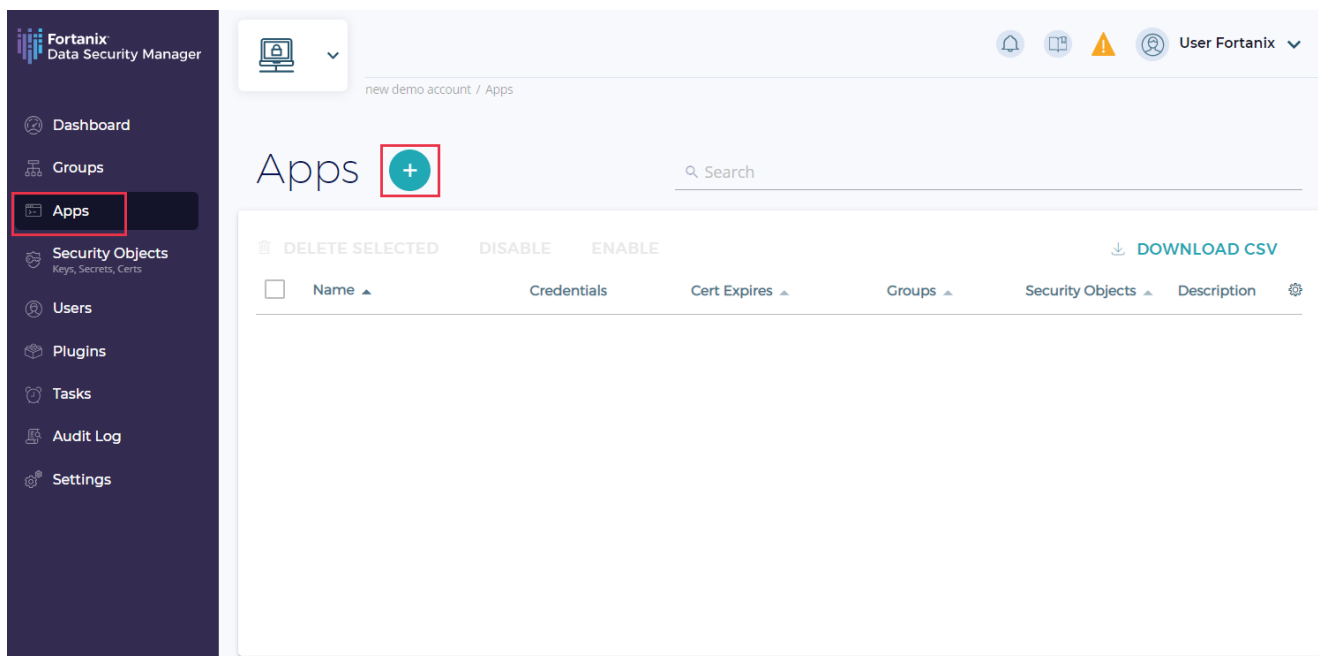


FIGURE 1: CREATE NEW APP

3. Enter the following information:
 - **App name:** This is the name to identify the Commvault app.
 - **Authentication method:** Select the default authentication method, that is **API Key**.
 - **Group:** Select a group for the app.
4. Click **Save** to complete creating the application.
5. Note down the application's UUID by clicking the icon for "**Copy UUID**". You will need this App-ID for the certificate.

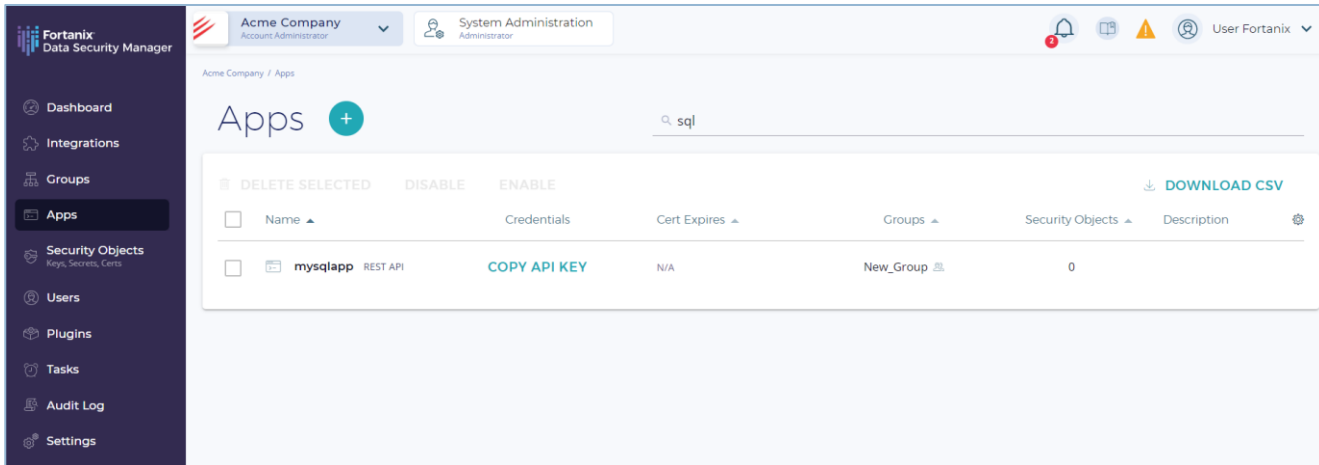


FIGURE 2: COPY APP API KEY

6. If an App / Client needs to authenticate to Fortanix DSM using only a certificate, then the App ID needs to be embedded in the certificate in one of the following ways:
 - a. Provided as the value of a custom OID 1.3.6.1.4.1.49690.1.2.1 in the certificate. The OID here is just an example.
 - b. Standard human-readable UUID encoding: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx provided as the value of CN. CN example: 070e0690-9230-4db3-9212-27a7bcf1c303.

The following sections will explain how to generate a client certificate to use with MySQL for each of these methods.

2.2 CREATING A CLIENT CERTIFICATE WITH CUSTOM OID VALUE

You can generate a self-signed certificate such that the custom OID is part of the certificate. To achieve this:

1. Edit the file `/etc/ssl/openssl.cnf` and add the custom OID in the "new_oids" section. These sections in the file should look as follows:

```
oid_section          = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
```

```
# X.509v3 extensions to use:
# extensions =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]
my_app_id=1.3.6.1.4.1.49690.1.2.1
```

2. Now add a description in the “req_distinguished_name” section. In this section add the following line:

```
my_app_id = custom attribute for app id
```

3. Save the file and generate a self-signed certificate as shown below:
 - a. Change directory to SDKMS_Certs and run the following command.

```
openssl req -newkey rsa:2048 -nodes -keyout private.key -x509
-days 365 -out certificate.crt
```

```
[~]$ openssl req -newkey rsa:2048 -nodes -keyout private.key -x509 -days 365 -out certificate.crt
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'private.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:XX
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:da7f2800-4122-4681-aebf-90beb779b73f
Email Address []:test@fortanix.com
[~]$
```

FIGURE 3: CREATE SELF-SIGNED CERTIFICATE

```
[~]$ openssl x509 -in certificate.crt -noout -text
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      87:8d:06:6c:f8:b0:22:f2:52:32:b9:c9:be:3b:bf:6e:da:62:27
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C = XX, ST = Some-State, O = Internet Widgits Pty Ltd, CN = da7f2800-4122-4681-aebf-90beb779b73f, emailA
address = test@fortanix.com
  Validity
    Not Before: May 12 06:07:43 2021 GMT
    Not After : May 12 06:07:43 2022 GMT
  Subject: C = XX, ST = Some-State, O = Internet Widgits Pty Ltd, CN = da7f2800-4122-4681-aebf-90beb779b73f, email
Address = test@fortanix.com
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
    RSA Public-Key: (2048 bit)
    Modulus:
      00:b2:75:b3:25:e7:f1:10:27:c5:35:06:78:90:f3:
      05:fa:de:16:87:81:81:fb:00:59:c8:4e:3e:c2:b9:
      2d:ff:f5:03:4d:b9:9c:11:63:08:e4:2e:d5:96:66:
      30:48:c1:3b:32:d3:75:49:7f:c8:7c:82:b6:a9:4a:
      8e:d3:ca:ce:ba:0d:6d:34:55:0a:0f:eb:05:d0:0a:
      9e:27:58:5c:d4:5a:b4:d6:6a:ea:a4:bc:1e:11:59:
      d5:ac:20:60:1a:ea:dd:08:60:f7:ba:c8:1f:81:57:
      19:8e:a5:d1:72:c5:8b:e5:85:2f:ff:cb:3e:39:48:
      82:a3:62:6d:21:1d:2e:da:c2:6d:90:35:2f:db:5b:
      a2:5f:29:7a:2c:84:e3:14:16:f2:f9:46:22:1a:2f:
      d9:51:6f:40:5a:ab:e1:7f:7a:b8:55:6f:9c:2a:5d:
      78:77:84:17:71:3a:7f:4e:24:95:21:51:df:35:a1:
      e9:c9:f4:50:23:f4:af:c3:2b:ef:84:c0:1a:21:a8:
      c7:6d:6c:61:03:72:52:80:5e:be:11:55:22:1f:ff:
      9a:9d:af:f9:a0:6c:f4:b2:55:c7:24:74:48:a7:3f:
      3d:75:17:06:4a:69:bb:96:75:a7:fb:23:fb:55:a7:
      9b:10:a9:37:09:64:1c:a2:98:71:0c:5e:28:35:54:
      0a:15
    Exponent: 65537 (0x10001)
  X509v3 extensions:
    X509v3 Subject Key Identifier:
      30:0B:6D:D4:87:BD:D4:68:3C:AD:9F:86:D8:BE:8C:AE:4B:85:A8:AF
    X509v3 Authority Key Identifier:
      keyid:30:0B:6D:D4:87:BD:D4:68:3C:AD:9F:86:D8:BE:8C:AE:4B:85:A8:AF

    X509v3 Basic Constraints: critical
      CA:TRUE
  Signature Algorithm: sha256WithRSAEncryption
  79:3f:00:63:7e:02:1c:15:73:9e:48:d4:93:fe:14:7f:57:cb:
  d4:10:40:8b:de:d6:73:5d:1f:31:72:ee:44:2d:d5:6e:48:ed:
  77:70:5b:9a:87:03:0e:45:dd:5c:80:75:97:64:fb:40:83:f7:
  80:cd:66:71:38:21:da:a3:49:db:b5:49:fb:69:12:fa:d2:05:
  13:72:c8:b4:3c:6d:7d:1e:c7:e6:5b:fa:1d:bd:61:bd:71:7a:
  53:0c:67:b9:3f:14:ed:30:29:a4:5b:12:eb:08:ca:20:9f:6b:
  b1:82:21:83:69:ed:04:0b:7c:da:d7:aa:f5:40:d6:fc:3e:e9:
  12:61:26:10:c9:cf:87:12:26:5e:f9:0b:fb:cd:c9:74:15:27:
  47:c9:38:d5:8d:cf:fb:f7:ad:bb:63:d8:0c:13:c2:1b:28:15:
  91:5d:a0:d0:84:28:25:11:40:48:4f:8a:79:40:4f:cd:8c:c7:
  e7:a8:bf:b6:4c:05:05:21:6a:d2:6d:82:8e:37:f4:df:25:ba:
  35:0e:c5:ef:0e:60:83:65:28:49:e3:8a:d9:c4:2d:b4:f7:98:
  13:e5:95:a1:8a:9b:38:f3:db:d9:a5:b0:72:36:c7:0f:91:5a:
  65:1f:ae:ef:0c:de:12:47:d1:a6:97:4d:50:6c:ed:f5:d2:0b:
  71:7b:1d:a7
```

FIGURE 4: CERTIFICATE GENERATED

- b. This will prompt for the value of the custom attribute where you should enter the App ID you noted earlier. The generated certificate will have the value of the custom OID populated.
- c. Examine the subject in the certificate to verify it contains the custom OID. A correctly generated certificate should look as follows (note the value of custom OID in the subject).

Certificate:

Data:

Version: 3 (0x2)

Serial Number: 18122652583846371291 (0xfb809881cffa5fdb)

Signature Algorithm: sha256WithRSAEncryption

Issuer: C=US, ST=California, L=Mountain View, O=Fortanix Inc,

OU=Engineering,

CN=test.kmip.fortanix.com/emailAddress=test@fortanix.com/1.3.6.1.4.1.4969

0.1.2.1=acc15bf3-e626-47aa-9373-7b08b3f26ee8

Validity

Not Before: Aug 8 23:19:45 2018 GMT

Not After : Aug 8 23:19:45 2019 GMT

Subject: C=US, ST=California, L=Mountain View, O=Fortanix Inc,

OU=Engineering,

CN=test.kmip.fortanix.com/emailAddress=test@fortanix.com/1.3.6.1.4.1.4969

0.1.2.1=acc15bf3-e626-47aa-9373-7b08b3f26ee8

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

Public-Key: (2048 bit)

Modulus:

```
00:97:a4:5b:d4:11:ee:c6:89:e1:f8:44:39:f9:69:
43:be:ee:69:78:5b:32:26:53:9d:a7:46:f4:17:0e:
5a:dc:b4:58:23:af:69:a1:86:de:2e:c5:46:14:98:
b6:6a:fc:f5:26:73:f7:56:6f:60:d8:2c:52:69:c9:
58:2a:d6:fd:4e:6e:22:0d:8c:e5:99:01:10:70:59:
6c:68:a2:a8:ee:e6:37:f7:08:8a:8a:75:bb:91:2b:
db:ad:1c:03:56:5f:01:ae:55:ff:3a:8b:40:91:e7:
04:4d:49:31:76:dc:ec:9e:d5:cb:d5:73:00:4f:13:
f2:12:f3:45:9f:df:fc:aa:2d:5f:d4:95:b2:e9:fa:
ad:38:d8:36:a5:f3:99:92:e5:b4:0a:39:99:85:ee:
13:39:fb:8d:1c:7a:52:03:e3:86:8a:d8:24:e9:28:
70:18:72:e0:b5:e6:f2:66:6f:1c:1a:be:f7:23:2c:
e0:9f:79:2b:2e:6e:be:c6:b1:31:65:00:cb:9c:8b:
bd:c0:56:dc:bd:0c:24:6a:d2:20:91:5f:14:84:63:
ef:18:b2:de:33:a8:ec:dd:4e:a5:3f:11:7b:7d:eb:
a1:e1:49:fc:d7:9e:26:98:6f:cb:3b:7e:5d:7e:2d:
```



```

1e:34:ca:3a:f9:12:95:b2:aa:ff:40:95:e1:5e:b9:
a5:a3
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Subject Key Identifier:
9C:74:2E:5B:16:76:F9:59:9F:E0:B5:53:C9:26:45:45:F7:4C:8D:99
X509v3 Authority Key Identifier:
keyid:9C:74:2E:5B:16:76:F9:59:9F:E0:B5:53:C9:26:45:45:F7:4C:8D:99

X509v3 Basic Constraints:
CA:TRUE
Signature Algorithm: sha256WithRSAEncryption
72:95:6a:8a:4c:18:53:e9:f6:3d:87:e9:97:d2:48:fe:2b:60:
ea:e2:ca:81:cb:9b:15:48:38:30:62:16:6b:b0:54:f6:91:2d:
b0:72:af:36:36:39:8e:78:1f:8c:17:19:df:5c:e5:ae:4d:f4:
ae:41:39:04:f2:95:d1:0a:99:ef:ef:63:72:5e:83:96:c1:c7:
f1:d7:f6:45:58:23:76:3d:1a:ba:a3:08:e4:4a:a0:6a:33:8f:
e5:50:04:b1:08:74:b3:37:9c:fd:f9:9c:5d:27:7d:63:a8:7d:
40:3e:d5:aa:7d:a7:9e:70:79:38:91:45:68:29:0d:a8:80:42:
f8:9b:e0:17:bb:93:9f:71:89:04:0f:39:d0:2e:3c:10:62:44:
6b:41:5d:e5:78:42:50:c5:f7:ee:bc:a8:5e:90:01:ad:3c:f2:
27:f2:81:16:ba:1e:79:d8:c4:09:cb:01:fd:71:11:9f:91:14:
72:71:0f:f1:d3:b0:4d:91:78:dd:12:fb:fd:d6:22:93:15:67:
df:4e:da:df:74:de:68:95:d7:d8:70:48:e2:5f:bc:ec:b2:0f:
bb:14:83:ad:c9:f9:a0:81:0d:a8:68:64:77:db:5a:71:4a:8b:
8f:91:d6:ce:e1:33:42:ba:98:76:a1:cd:89:8e:3a:cb:aa:b1:
8e:ca:42:af

```

2.3 CREATING A CLIENT CERTIFICATE WITH APP ID AS CN

You can generate a self-signed certificate such that the CN contains the App ID.

1. Generate a self-signed certificate as shown in the *Section: "Creating a client certificate with custom OID value"*.

2. When prompted for a Common Name, you should enter the App ID you noted earlier. The generated certificate will have the App ID as CN.
3. Examine the subject in the certificate to verify it contains the App ID as CN. A correctly generated certificate should look as follows (note the value of CN):

```
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 11285796284824083476 (0x9c9f33ed245cdc14)
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=US, ST=CA, L=Mountain View, O=Fortanix, OU=Test,
CN=da7f2800-4122-4681-aebf-90beb779b73f/emailAddress=test@fortanix.com
    Validity
      Not Before: Aug  8 23:31:20 2018 GMT
      Not After  : Aug  8 23:31:20 2019 GMT
    Subject: C=US, ST=CA, L=Mountain View, O=Fortanix, OU=Test,
CN=da7f2800-4122-4681-aebf-90beb779b73f/emailAddress=test@fortanix.com
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      Public-Key: (2048 bit)
      Modulus:
        00:d2:ae:15:66:bf:78:d4:98:f4:4d:a5:57:bf:04:
        08:76:83:1f:40:e8:8b:c4:da:8a:a0:71:22:43:84:
        6d:c9:05:f2:81:91:83:04:75:bd:c9:83:86:92:bf:
        ff:a0:e4:b4:e4:ee:56:09:10:2a:dc:e2:f4:0c:65:
        43:96:a1:31:0d:15:92:49:87:ee:46:91:5d:f1:8c:
        61:b3:ca:4a:9f:be:01:00:d5:30:5f:ee:56:35:75:
        3c:e1:0d:a6:34:66:7f:3b:26:69:97:33:6d:2e:c7:
        fd:c9:42:7d:14:f7:12:18:4a:5b:a6:90:52:7a:4b:
        1b:45:b3:79:33:31:99:03:1d:a4:ed:51:dc:7b:43:
        20:02:bb:08:22:27:27:8c:51:6a:5f:59:87:45:95:
        d7:f3:ca:fa:30:3d:d5:a6:50:77:03:e3:de:eb:30:
        17:45:48:fe:5b:76:d4:c1:03:3f:b8:99:73:ae:ad:
        ae:e2:69:95:e2:14:1e:42:b1:ac:72:cd:0b:c6:01:
        e3:20:8d:5a:6a:5d:19:79:17:f0:80:5f:75:fc:d5:
        da:9c:af:07:d8:c7:96:02:a5:94:19:64:d7:9a:e4:
```

```
56:f1:cf:54:b9:a7:29:28:22:52:f2:c4:8a:97:04:
45:b1:9b:b5:4f:c0:18:53:ff:08:3f:3b:81:bd:f1:
d1:e9
Exponent: 65537 (0x10001)
X509v3 extensions:
    X509v3 Subject Key Identifier:
87:65:C6:B6:B6:3A:0A:A6:30:BA:CB:D2:27:9E:C4:E6:2E:7F:2F:6D
    X509v3 Authority Key Identifier:
keyid:87:65:C6:B6:B6:3A:0A:A6:30:BA:CB:D2:27:9E:C4:E6:2E:7F:2F:6D

    X509v3 Basic Constraints:
        CA:TRUE
Signature Algorithm: sha256WithRSAEncryption
71:da:8c:da:ab:9d:6d:8a:f1:9c:56:a9:7d:e2:e2:1b:fd:90:
b7:5e:45:db:d4:69:47:ca:98:2f:b0:3b:2c:1f:49:3a:75:dd:
1d:96:b3:bd:11:a6:d7:06:60:4f:18:11:e1:cf:db:5c:52:03:
29:78:47:6e:36:c0:64:d8:4d:34:00:d9:94:55:48:a9:d4:b2:
b2:ed:b8:13:fc:3d:c6:b4:61:a3:56:aa:9d:73:80:62:38:da:
0c:94:b0:4a:e6:86:da:6a:f9:aa:f3:a4:3c:48:32:93:f7:d3:
27:f9:2c:77:b4:91:9c:84:62:96:86:7d:d2:c8:20:79:d1:12:
ef:f0:cc:15:31:ea:86:e9:b4:02:00:55:83:0f:6a:c6:5b:d2:
19:67:9b:b2:44:f8:3b:36:f9:b0:02:b2:98:7d:1e:fa:95:58:
92:92:57:68:f8:56:bb:43:db:01:08:bb:d6:ab:52:e6:c7:88:
7a:1c:8d:f4:31:90:70:0a:dd:d2:96:7c:8b:93:8f:1f:4a:80:
fe:3a:f8:df:82:a7:99:ac:2f:e8:02:e5:8b:fe:ec:3b:3b:0a:
a3:c0:82:4d:f7:93:66:a1:76:6f:fa:c2:19:8e:d8:b6:b4:27:
8c:57:22:a4:f7:e6:45:61:27:af:fc:5f:51:88:eb:32:
```

2.4 SETTING APP AUTHENTICATION METHOD AS CERTIFICATE

After you have the certificate, you will need to change the authentication method for your app in Fortanix DSM to use a certificate instead of an API key. To change the authentication method, go to the detailed view of the application, navigate to the **INFO** tab, and open the **Change authentication method** drop down menu. Select the method as **Certificate** and click **Save**. You will

be prompted to upload a certificate. Upload your certificate and click **Update**. Now your app is set to authenticate using the certificate you created.

3.0 CONFIGURING ENCRYPTION IN PERCONA MYSQL

In order to configure encryption in MySQL, you will need to install and configure the keyring component.

3.1 KEYRING COMPONENT INSTALLATION

To install a keyring component, you must do the following:

1. Create a manifest in a valid JSON format.
2. Write a configuration file.

3.1.1 CREATING MANIFEST FOR THE KEYRING COMPONENT

A manifest file indicates which component to load. During start-up, the server reads the global manifest file from the installation directory. The global manifest file can contain the required information or point to a local manifest file located in the data directory. If you have multiple server instances that use different keyring components use a local manifest file in each data directory to load the correct keyring component for that instance.

1. Go to the directory where `mysqld` executable exists (mostly `/usr/sbin`) and create a global manifest file named as `mysqld.my` file with the following JSON:

```
{
  "read_local_manifest": false,
  "components": "file:///component_keyring_kmip"
}
```

2. If you will be using a local manifest file, your global and local manifest file should be as shown below.

Global manifest:

```
{
```

```
"read_local_manifest": true
}
```

Local Manifest:

```
{
  "components": "file:///component_keyring_kmip"
}
```

3.1.2 CREATING A CONFIGURATION FILE

The configuration file for your KMS should be located in the plugin folder where your `component_keyring_kmip.so` resides (mostly `/usr/lib/mysql/plugin`) and should have the following JSON:

```
cat /usr/lib/mysql/plugin/component_keyring_kmip.cnf

{
  "server_addr": "sdkms.fortanix.com",
  "server_port": "5696",
  "client_ca": "certificate.crt",
  "client_key": "client_key.pem",
  "server_ca": "dsm_ca.crt"
}
```

Where,

- `server_addr` is the Fortanix DSM endpoint.
- `server_port` is the KMIP port of 5696.
- `client_ca` is the CA certificate of the client certificate in the case of CA-signed client certificate or the client certificate itself in the case of a self-signed client certificate.
- `client_key` is the client's private key.
- `server_ca` is the Fortanix DSM CA certificate.

1. After performing any component-specific configuration, start the server.
2. Verify component installation by examining the Performance Schema `keyring_component_status` table.

```
mysql> SELECT * FROM performance_schema.keyring_component_status;
+-----+-----+
| STATUS_KEY | STATUS_VALUE |
+-----+-----+
| Component_name | component_keyring_kmip |
| Author | Percona Corporation |
| License | GPL |
| Implementation_name | component_keyring_kmip |
| Version | 1.0 |
| Component_status | Active |
| Server_addr | sdkms.fortanix.com |
| Server_port | 5696 |
| Client_ca | /home/adminguy/certs/certificate.crt |
| Client_key | /home/adminguy/certs/private.pem |
| Server_ca | /home/adminguy/certs/dsm_ca.crt |
| Object_group | <NONE> |
+-----+-----+
12 rows in set (0.00 sec)
```



NOTE: If the component cannot be loaded, the server start-up fails. Check the server error log for diagnostic messages. If the component loads but fails to initialize due to configuration problems, the server starts but the `Component_status` value is Disabled. Check the server error log, correct the configuration issues, and use the `ALTER INSTANCE RELOAD KEYRING` statement to reload the configuration.

3.2 CREATING ENCRYPTED TABLES

When you create the first encrypted table, InnoDB will generate the master key (AES-256) in Fortanix DSM. You can check this in the Fortanix DSM Web UI on the Security Objects page. This master key is used to encrypt tablespace keys. Tablespace keys are generated locally by InnoDB. The tablespace key is wrapped using the master key and stored alongside the encrypted table. For subsequent encrypted tables, only the tablespace key is generated, and the same master key is used to wrap the tablespace key.

With Fortanix DSM, you will see a complete audit trail every time the master key is retrieved. You will also have complete control over 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 to create an encrypted table.

```
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`)
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=latin1 ENCRYPTION = 'Y';
```

Rotate key from the database using the following command:

```
ALTER INSTANCE ROTATE INNODB MASTER KEY;
```

The following screenshot shows the activity logs for the MySQL application and an audit trail of the master key usage.

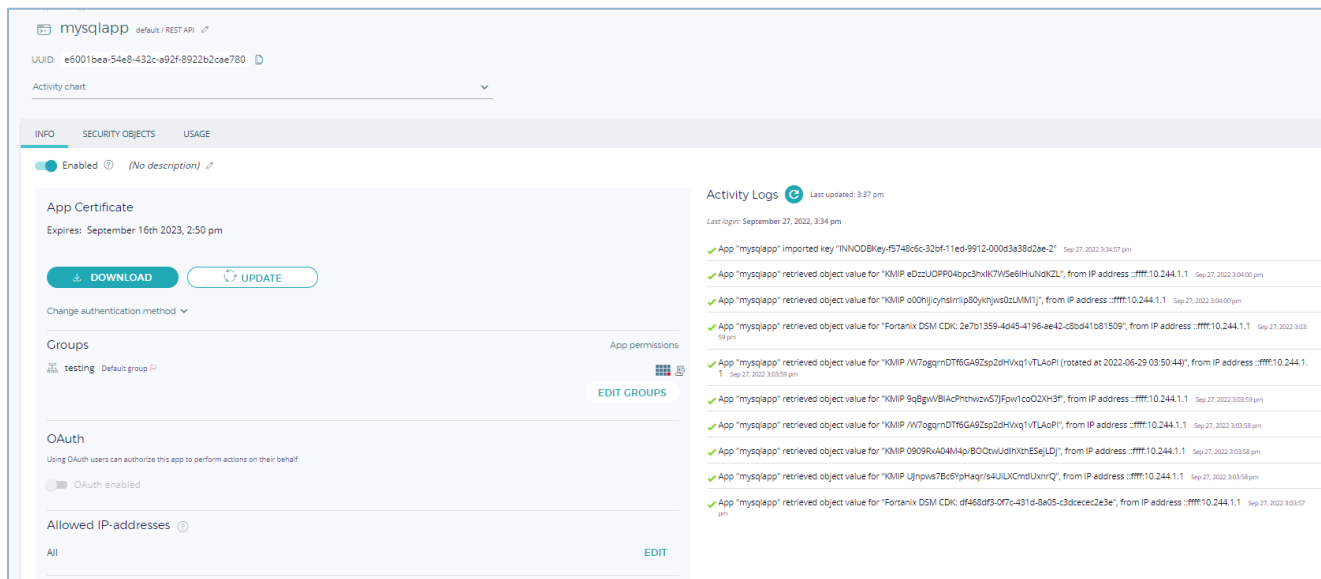


FIGURE 5: MYSQL ACTIVITY LOGS

4.0 DOCUMENT INFORMATION

4.1 DOCUMENT LOCATION

The latest published version of this document is located at the URL:

<https://support.fortanix.com/hc/en-us/articles/10145085542676-Using-Fortanix-Data-Security-Manager-for-Percona-MySQL-Encryption-at-Rest->

4.2 DOCUMENT UPDATES

This document will typically be updated on a periodic review and update cycle.

For any urgent document updates, please send an email to: support@fortanix.com

© 2016 – 2022 Fortanix, Inc. All Rights Reserved.

Fortanix[®] and DSM Applications are trademarks of Fortanix, Inc. All other trademarks are trademarked by their respective owners.

NOTICE: This document was produced by Fortanix, Inc. (Fortanix) and contains information which is proprietary and confidential to Fortanix. The document contains information that may be protected by patents, copyrights, and/or other IP laws. If you are not the intended recipient of this material, please destroy this document and inform info@fortanix.com immediately.