



# Red Hat OpenStack Platform 8 Database-as-a-Service ガイド

---

Red Hat OpenStack Platform の Database-as-a-Service

OpenStack Team



## Red Hat OpenStack Platform の Database-as-a-Service

OpenStack Team  
rhos-docs@redhat.com

## 法律上の通知

Copyright © 2017 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## 概要

本ガイドでは、Database-as-a-Service の Trove を Red Hat OpenStack Platform でデプロイ、設定、使用方法について説明します。

---

## 目次

前書き .....	3
第1章 TROVE の管理 .....	4
1.1. TROVE コントローラーのインストールと設定	4
1.2. TROVE のセキュリティー保護	4
第2章 TROVE の使用 .....	7
2.1. TROVE ゲストデータベースイメージの作成および設定の自動化	7



## 前書き

Database as a Service (DBaaS) により、ユーザーは、デプロイ、設定、バックアップなどの管理タスクについて懸念することなく、OpenStack 環境でリレーショナルデータベースまたは非リレーショナルデータベースを便利に活用することができます。この機能は、ゲストのデータベースイメージのプロビジョニングと管理を行う **Trove** コンポーネントによって提供されます。

Trove は、複数のサービスで構成されます。

- ※ ゲストエージェントサービス。ゲストインスタンス上で実行され、データストアを管理します。これには、新規データストアをオンラインにしたり、データベース操作を実行したりする作業が含まれます。
- ※ インスタンスのプロビジョニング、管理、操作を実行するタスクマネージャーサービス
- ※ RESTful API を提供し、API 要求をゲストエージェントまたはタスクマネージャーに転送する API サービス。Trove API へのコマンドラインインターフェースは、**python-troveclient** パッケージの一部である **trove** クライアントによって提供されます。
- ※ ホスト上で実行され、ゲストインスタンスからのメッセージ (例: ステータスの更新など) をリッスンしてホストに渡すコンダクターサービス

以下のデータベースがサポートされています。

- ※ MySQL 5.5 および 5.6
- ※ MariaDB 5.5 および 10

## 第1章 TROVE の管理

### 1.1. TROVE コントローラーのインストールと設定

Trove コントローラーに必要なパッケージのインストールと環境の設定方法に関する詳しい説明は、『[インストールリファレンス](#)』を参照してください。

### 1.2. TROVE のセキュリティー保護

Red Hat OpenStack Platform の以前のバージョンでは、Trove ゲストエージェントは、RabbitMQ メッセージバスへの認証情報を使用して実行されていました。RabbitMQ メッセージバスは、管理コントローラー全体で、機密データを含む全トラフィックに使用されていたので、Trove のインスタンスは、メッセージバスで送信される全データを表示することが可能でした。このため、インスタンスがこのバスを共有する他のサービスを誤用したり、攻撃したりできてしまう可能性があります。Red Hat OpenStack Platform 8 では、Trove は、分離されたテナントに配置され、管理コントローラーの他の部分にはアクセスできなくなりましたが、DBaaS のセキュリティーをさらに強化するには、別のホストと別の RabbitMQ メッセージバスに Trove インスタンスを設定することが可能です。この設定を使用するには、以下の手順を実行してください。

#### 1. 管理コントローラー上での操作

1. RabbitMQ のユーザーとパスワードをカスタムの値に設定します。この値は、Trove テナントとは共有されません。たとえば、**guest** アカウントと / デフォルト vhost に変更したパスワードを使用することができます。

```
# rabbitmqctl change_password guest password
```

このパスワードは、サービスがメッセージバスに接続するように設定している全設定ファイルで変更されます。

2. Trove データベースエンドポイントを更新して、サービスカタログルックアップが別のホスト上で Trove を使用するようにします。まず現在のエンドポイント UUID を特定してから削除し、最後に別の Trove ホストの IP アドレスを使用して新規エンドポイントを作成します。

```
# keystone endpoint-list
# keystone endpoint-delete current_trove_endpoint_uuid
# keystone endpoint-create --service trove --publicurl
http://IP:8779/v1.0/\$(tenant_id)s --region RegionOne
```

#### 2. Trove コントローラー上での操作

1. リモート管理コントローラーホストの WSGI 設定で認証トークンフィルターをポイントします。**/etc/trove/api-paste.ini** ファイルで、以下のオプションと値を指定します。**IP** は管理コントローラーホストの IP アドレスに置き換えます。

```
[filter:authtoken]
paste.filter_factory =
keystonemiddleware.auth_token:filter_factory
service_protocol = http
service_host = IP
```



```

service_port = 5000
auth_host = IP
auth_port = 35357
auth_protocol = http
auth_uri = http://IP:35357/v2.0/
signing_dir = /tmp/keystone-signing-trove
admin_tenant_name = admin
admin_user = admin
admin_password = admin

```

2. 管理ユーザーを追加して、Trove ゲストエージェントには、管理コントローラーで  
使用しているのとは異なるシークレットが設定されるようにします。

```

# rabbitmqctl add_user isolated isolated
# rabbitmqctl set_permissions isolated ".*" ".*" ".*"

```

3. `/etc/trove/trove.conf` ファイルで以下オプションと値を指定して、Trove コ  
ントロールプレーンがローカルの RabbitMQ インスタンスに接続して認証を行う  
ように設定します。

```

# AMQP Connection info
rabbit_userid=isolated
rabbit_password=isolated
rabbit_host=TROVE_IP

# single instance tenant
nova_proxy_admin_user = user
nova_proxy_admin_pass = password
nova_proxy_admin_tenant_name = tenant
trove_auth_url = http://MGMT_IP:5000/v2.0
nova_compute_service_type = compute
cinder_service_type = volumev2
os_region_name = RegionOne
nova_compute_url = http://MGMT_IP:8774/v3

remote_nova_client =
trove_ext.cloudos.remote.nova_client_trove_admin
remote_cinder_client =
trove_ext.cloudos.remote.cinder_client_trove_admin
remote_neutron_client =
trove_ext.cloudos.remote.neutron_client_trove_admin

```

3. 管理コントローラーに戻り、`/etc/trove/trove-guestagent.conf` ファイルを以下の  
ように編集します。

```

# AMQP Connection info
rabbit_userid=isolated
rabbit_password=isolated
rabbit_host=TROVE_IP

# single tenant config
nova_proxy_admin_user = user

```

```
nova_proxy_admin_pass = password
nova_proxy_admin_tenant_name = tenant
trove_auth_url = http://MGMT_IP:5000/v2.0
swift_url = http://MGMT_IP:8080/v1/AUTH_
```

また、RabbitMQ vhost の提供する ACL ポリシーを活用してセキュリティを強化することができます。この場合には、vhost ごとに異なるパーミッションのある異なるユーザーを割り当てることができます。たとえば、**/isolated** という名前の vhost を定義して、**isolated** ユーザーに vhost の適切なパーミッションを割り当てます。このように設定するには、以下の手順を実行します。

1. デフォルトの / vhost 用の **guest** アカウントのパスワードを変更します。

```
# rabbitmqctl change_password guest password
```

2. 新規ユーザーと vhost を追加して、パーミッションを適切に設定します。

```
# rabbitmqctl add_user isolated isolated
# rabbitmqctl add_vhost /isolated
# rabbitmqctl -p /isolated set_permissions isolated ".*" ".*"
"."
```

3. Trove の設定を編集します。編集するファイルは 2 つあります。**/etc/trove/trove-guestmanager.conf** は、**puppet-trove** から **packstack** によって生成され、ゲストインスタンスに挿入されます。**/etc/trove/trove-conductor.conf** は、Trove インスタンスからの非同期ステータス更新のための Trove コントロールプレーンサービスを設定します。

```
[oslo_messaging_rabbit]
rabbit_host=IP
rabbit_virtual_host=/isolated
rabbit_userid=isolated
rabbit_password=isolated
rabbit_port=5672
rabbit_ha_queues=False
rabbit_hosts=IP:5672
rabbit_use_ssl=False
```

この方法の利点は、別のホストとメッセージキューを使用する場合のシナリオよりもシンプルであることです。これは、Red Hat OpenStack Platform director を使用するなどの方法で、プロビジョニングを自動的に実行する場合に役立てることができます。

## 第2章 TROVE の使用

### 2.1. TROVE ゲストデータベースイメージの作成および設定の自動化

**trove-image-create** ツールは、サポートされているデータストア向けの Trove 互換イメージの生成を自動化します。このツールの使用は、Red Hat が推奨しています。

**trove-image-create** ツールを取得して、**openstack-trove-images** パッケージをインストールします。

```
# yum install openstack-trove-images
```

以下の基本オプションを利用することができます。

オプション	説明、パラメーター
<b>-i, --image</b>	使用するベースイメージ。QEMU イメージ (qcow2) がサポートされています。イメージファイル名 (およびオプションでそのパス) をパラメーターとして指定します。
<b>-r, --release</b>	使用する OpenStack バージョン。 <b>kilo</b> または <b>liberty</b> のいずれかをパラメーターとして指定します。
<b>-s, --datastore</b>	デプロイするデータストア。サポート対象のデータストアは、 <a href="#">前書き</a> に一覧表示しています。以下のようなパラメーターを使用する可能性があります。 <ul style="list-style-type: none"> <li>✳ <b>mysql</b>: ディストリビューションで <b>mysql</b> を提供するパッケージ。RHEL 7 の場合には MariaDB 5.5 が使用されます。</li> <li>✳ <b>mysql155</b>: mysql.com から提供されている MySQL 5.5</li> <li>✳ <b>mysql156</b>: mysql.com から提供される MySQL 5.6</li> <li>✳ <b>mariadb10</b>: mariadb.org から提供されている MariaDB 10.0</li> </ul>

#### 例2.1 イメージのカスタマイズ

たとえば、以下のようにツールを使用することができます。

```
# trove-image-create -s mysql -r liberty -i myimage.qcow2
```

このコマンドにより、Red Hat OpenStack Platform 8 (Liberty) から MariaDB 5.5 および Trove が追加され、現在の作業ディレクトリー内の **myimage.qcow2** ファイルに保管されているイメージがカスタマイズされます

RHEL 7 イメージを使用して作業を行う場合には、使用する必要のある追加のオプションがあります。

オプション	認識されるパラメーターおよび構文	説明
<b>--sm-register</b>	<b>USER:password:PASSWORD</b> <b>USER:file:FILE_CONTAINING_PASSWORD</b>	Red Hat の認証情報を使用して、サブスクリプションマネージャーに登録します。
<b>--sm-pool</b>	<b>pool:POOL_ID</b> <b>file:FILE_CONTAINING_POOL_ID</b> <b>auto</b>	指定または自動的に決定されたサブスクリプションプールをシステムにアタッチします。

## 例2.2 RHEL 7 イメージのカスタマイズ

### 例

```
# trove-image-create -s mysql -r liberty -i ../../images/rhel-
mariadb55.qcow2 --sm-register admin@example.com:password:123456 --
sm-pool auto
```

このコマンドは、上記の例と同様の方法でイメージをカスタマイズするとともに、Red Hat ログイン名 **admin@example.com**、パスワード **123456**、および最も適したサブスクリプションを使用してシステムを登録します。

### 2.1.1. Trove 管理へのイメージの読み込み

イメージのカスタマイズが完了したら、以下のステップを実行してください。

1. イメージを Glance にアップロードします。そのためには、以下と同様のコマンドを実行します。

```
# openstack image create rhel7-mariadb55 --disk-format qcow2 --
container-format bare --public < myimage.qcow2
```

2. 上記のコマンドの出力から、アップロードしたイメージの ID を取得します。以下のような出力が表示されるはずです。

```
+-----+-----+
+-----+
| Field          | Value |
+-----+-----+
+-----+
| checksum       | dec3f16054739459d03984b7a552cd9c |
| container_format | bare |
| created_at     | 2016-01-27T20:10:36Z |
+-----+-----+
```

```

| disk_format      | qcow2
|
| file             | /v2/images/c637391b-e00f-47fb-adb5-
e8dfc4e224d4/file |
| id              | c637391b-e00f-47fb-adb5-e8dfc4e224d4
|
| min_disk        | 0
|
| min_ram         | 0
|
| name            | rhel7-mariadb55
|
| owner           | 483cae7de00c4f029e19eef5983c67a9
|
| protected       | False
|
| schema          | /v2/schemas/image
|
| size            | 1910767616
|
| status          | active
|
| updated_at      | 2016-01-27T20:10:46Z
|
| virtual_size    | None
|
| visibility      | public
|
+-----+-----+
-----+

```

この場合、ID は **c637391b-e00f-47fb-adb5-e8dfc4e224d4** です。

3. Trove 管理データストアを更新して、必要なデータストアとバージョンのインスタンスを起動するのに使用される新規イメージのレコードを追加します。

```

# export DATASTORE=mariadb
# export DATASTORE_VERSION=5.5
# export IMAGE_ID=c637391b-e00f-47fb-adb5-e8dfc4e224d4
# export PACKAGES=mariadb-server
#
# trove-manage datastore_update ${DATASTORE} ""
# trove-manage datastore_version_update ${DATASTORE}
${DATASTORE_VERSION} ${DATASTORE} ${IMAGE_ID} ${PACKAGES} 1
# trove-manage datastore_update ${DATASTORE} ${DATASTORE_VERSION}

```

### 重要

**PACKAGES** 変数は、使用するデータストアによって異なります。MySQL (任意のバージョン) の場合には、**mysql-community-server** を使用します。MariaDB 10.0 の場合には、**MariaDB-server** を使用します。

## 2.1.2. トラブルシューティング

インスタンスの起動時にエラーが発生した場合には、SSH キーを使用してイメージを作成し、トラブルシューティングに使用することができます。**trove-image-create** ツールには、この目的のために **--root-ssh-key** オプションがあります。このオプションは、公開鍵へのパスをパラメーターとして取り、イメージに鍵を挿入します。以下に例を示します。

```
# trove-image-create -i myimage.qcow2 -r liberty -s mysql --root-ssh-key ~/.ssh/id_rsa.pub
```

このイメージをベースとしたインスタンスにアクセスするには、インスタンスに関連付けられたセキュリティグループを編集して、SSH ポートを開放します。ICMP を許可することも検討してください。また、インスタンスがプライベートネットワークにある場合には、Floating IP をインスタンスに追加する必要があります。これらのステップが完了した後は、以下のコマンドを実行するとインスタンスにログインできるはずです。

```
# ssh root@INSTANCE_IP
```