Red Hat OpenStack Platform 16.2

アンダークラウドおよびコントロールプレーンノードのバックアップと復元

アンダークラウドおよびオーバークラウドコントロールプレーンノードのバックアップの作成と復元

Last Updated: 2024-04-04

Red Hat OpenStack Platform 16.2 アンダークラウドおよびコントロールプレーンノードのバックアップと復元

アンダークラウドおよびオーバークラウドコントロールプレーンノードのバックアップの作成と復元

法律上の通知

概要

このガイドでは、アンダークラウドおよびコントロールプレーンノードのバックアップを作成し、復元する方法を説明します。バックアップは、Red Hat OpenStack Platform をアップグレードまたは更新する際に必要です。また、任意で環境の定期的なバックアップを作成することで、問題発生時のダウンタイムを最小限に抑えることができます。

目次

多様性を受け入れるオープンソースの強化 **3**

RED HAT ドキュメントへのフィードバック (英語のみ) **4**

第1章 アンダークラウドノードのバックアップ **5**
1.1. サポート対象のバックアップ形式およびプロトコル 5
1.2. バックアップストレージの場所の設定 5
1.3. バックアップノードへの NFS サーバーのインストールと設定 6
1.4. アンダークラウドノードへの REAR のインストール 7
1.5. アンダークラウドノードのスタンドアロンデータベースバックアップの作成 7
1.6. バックアップ用の OPEN VSWITCH (OVS) インターフェイスの設定 8
1.7. アンダークラウドノードのバックアップの作成 8
1.8. CRON を使用したアンダークラウドノードバックアップのスケジューリング 9

第2章 コントロールプレーンノードのバックアップ **11**
2.1. サポート対象のバックアップ形式およびプロトコル 11
2.2. バックアップノードへの NFS サーバーのインストールと設定 11
2.3. コントロールプレーンノードへの REAR のインストール 12
2.4. バックアップ用の OPEN VSWITCH (OVS) インターフェイスの設定 13
2.5. コントロールプレーンノードのバックアップの作成 14
2.6. CRON を使用したコントロールプレーンノードバックアップのスケジューリング 15

第3章 アンダークラウドおよびコントロールプレーンノードの復元 **17**
3.1. 復元プロセス用に同じ場所に配置された CEPH モニターを備えたコントロールプレーンを準備する 17
3.2. アンダークラウドノードの復元 18
3.3. コントロールプレーンノードの復元 19
3.4. GALERA クラスターの手動による復元 21
3.5. アンダークラウドのノードデータベースを手動で復元する 24

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに対するご意見をお聞かせください。ドキュメントの改善点があればお知らせください。

Jira でドキュメントのフィードバックを提供する

ドキュメントに関するフィードバックを提供するには、Create Issue フォームを使用します。Red Hat OpenStack Platform Jira プロジェクトで Jira Issue が作成され、フィードバックの進行状況を追跡できます。

1. Jira にログインしていることを確認してください。Jira アカウントをお持ちでない場合は、アカウントを作成してフィードバックを送信してください。
2. Create Issue をクリックして、**Create Issue** ページを開きます。
3. **Summary** フィールドと **Description** フィールドに入力します。**Description** フィールドに、ドキュメントの URL、章またはセクション番号、および問題の詳しい説明を入力します。フォーム内の他のフィールドは変更しないでください。
4. **Create** をクリックします。

第1章 アンダークラウドノードのバックアップ

アンダークラウドノードをバックアップするには、バックアップノードを設定し、アンダークラウドノードに Relax-and-Recover ツールをインストールし、バックアップイメージを作成します。バックアップは、通常の環境メンテナンスの一環として作成できます。

さらに、更新またはアップグレードを実行する前にアンダークラウドノードをバックアップする必要があります。バックアップを使用して、更新またはアップグレード時にエラーが発生した場合に、アンダークラウドノードを以前の状態に復元することができます。

1.1. サポート対象のバックアップ形式およびプロトコル

アンダークラウドおよびバックアップ/復元プロセスは、オープンソースのツールである Relax-and-Recover (ReaR) を使用してブート可能なバックアップイメージを作成し、復元します。ReaR は Bash で記述されており、複数のイメージ形式と複数の転送プロトコルをサポートします。

次のリストは、ReaR を使用してアンダークラウドとコントロールプレーンをバックアップおよび復元するときに Red Hat OpenStack Platform がサポートするバックアップ形式とプロトコルを示しています。

ブート可能なメディア形式

- ISO

ファイルトランスポートプロトコル

- SFTP
- NFS

1.2. バックアップストレージの場所の設定

コントロールプレーンノードのバックアップを作成する前に、**bar-vars.yaml** 環境ファイルでバックアップの保存場所を設定します。このファイルには、バックアップの実行に渡す Key-Value パラメーターが格納されています。

手順

- **bar-vars.yaml** ファイルで、バックアップの保存場所を設定します。NFS サーバーまたは SFTP サーバーに適した手順を実行します。
  - NFS サーバーを使用する場合は、以下のパラメーターを **bar-vars.yaml** ファイルに追加します。

```
tripleo_backup_and_restore_server: <ip_address>
tripleo_backup_and_restore_shared_storage_folder: <backup_server_dir_path>
tripleo_backup_and_restore_output_url: "nfs://{{ tripleo_backup_and_restore_server }}{{
tripleo_backup_and_restore_shared_storage_folder }}"
tripleo_backup_and_restore_backup_url: "nfs://{{ tripleo_backup_and_restore_server }}{{
tripleo_backup_and_restore_shared_storage_folder }}"
```

<ip_address> と **<backup_server_dir_path>** を置き換えます。**tripleo_backup_and_restore_server** パラメーターのデフォルト値は **192.168.24.1** です。

- SFTP サーバーを使用する場合は、**tripleo_backup_and_restore_output_url** パラメーターを追加し、SFTP サーバーの URL と認証情報の値を設定します。

```
tripleo_backup_and_restore_output_url: sftp://<user>:<password>@<backup_node>/
tripleo_backup_and_restore_backup_url: iso:///backup/
```

<user>、**<password>**、および **<backup_node>** を、バックアップノードの URL および認証情報に置き換えます。

1.3. バックアップノードへの NFS サーバーのインストールと設定

バックアップファイルを保存するために、新しい NFS サーバーをインストールして設定できます。バックアップノードに NFS サーバーをインストールして設定するには、インベントリーファイルと SSH キーを順次作成し、NFS サーバーオプションを指定して **openstack undercloud backup** コマンドを実行します。

重要

- NFS サーバーまたは SFTP サーバーをインストールして設定している場合は、この手順を実行する必要はありません。バックアップするノードに ReaR を設定するときに、サーバー情報を入力します。
- デフォルトでは、NFS サーバーの Relax and Recover (ReaR) IP アドレスパラメーターは **192.168.24.1** です。使用している環境に一致する IP アドレス値を設定するには、**tripleo_backup_and_restore_server** パラメーターを追加する必要があります。

手順

1. アンダークラウドノードにおいて、source コマンドでアンダークラウドの認証情報を読み込みます。

```
[stack@undercloud ~]$ source stackrc
(undercloud) [stack@undercloud ~]$
```

2. アンダークラウドノードで、バックアップノードのインベントリーファイルを作成します。

```
(undercloud) [stack@undercloud ~]$ cat <<'EOF'> ~/nfs-inventory.yaml
[BackupNode]
<backup_node> ansible_host=<ip_address> ansible_user=<user>
EOF
```

<ip_address> および **<user>** を、実際の環境に該当する値に置き換えます。

3. 公開 SSH 鍵をアンダークラウドノードからバックアップノードにコピーします。

```
(undercloud) [stack@undercloud ~]$ ssh-copy-id -i ~/.ssh/id_rsa.pub <backup_node>
```

<backup_node> をバックアップノードのパスおよび名前に置き換えます。

4. バックアップノードに NFS サーバーを設定します。

```
(undercloud) [stack@undercloud ~]$ openstack undercloud backup --setup-nfs --extra-vars
/home/stack/bar-vars.yaml --inventory /home/stack/nfs-inventory.yaml
```

1.4. アンダークラウドノードへの REAR のインストール

アンダークラウドノードのバックアップを作成する前に、アンダークラウドに Relax and Recover (ReaR) をインストールして設定します。

前提条件

- バックアップノードに NFS または SFTP サーバーがインストールおよび設定されている。新しい NFS サーバーの作成方法は 「バックアップノードへの NFS サーバーのインストールと設定」 を参照してください。

手順

1. アンダークラウドノードにおいて、source コマンドでアンダークラウドの認証情報を読み込みます。

```
[stack@undercloud ~]$ source stackrc
```

カスタムのスタック名を使用する場合は、**--stack <stack_name>** オプションを **tripleo-ansible-inventory** コマンドに追加します。

2. 事前に実行していない場合は、インベントリーファイルを作成し、**tripleo-ansible-inventory** コマンドを使用して、すべてのオーバークラウドノードのホストおよび変数が含まれる静的なインベントリーファイルを生成します。

```
(undercloud) [stack@undercloud ~]$ tripleo-ansible-inventory \
--ansible_ssh_user heat-admin \
--static-yaml-inventory /home/stack/tripleo-inventory.yaml
```

3. アンダークラウドノードに ReaR をインストールします。

```
(undercloud) [stack@undercloud ~]$ openstack undercloud backup --setup-rear --extra-vars
/home/stack/bar-vars.yaml --inventory /home/stack/tripleo-inventory.yaml
```

4. システムで UEFI ブートローダーを使用している場合は、アンダークラウドノードで以下の手順を実施します。

   a. 以下のツールをインストールします。

   ```
   $ sudo dnf install dosfstools efibootmgr
   ```

   b. **USING_UEFI_BOOTLOADER** パラメーターの値 **0** を値 **1** に置き換えて、**/etc/rear/local.conf** にある ReaR 設定ファイルで UEFI バックアップを有効にします。

1.5. アンダークラウドノードのスタンドアロンデータベースバックアップの作成

Red Hat OpenStack Platform 環境を 13 から 16.2 にアップグレードする場合は、アンダークラウドのアップグレードを実行してから、アンダークラウドノードで Leapp によるアップグレードプロセスを実施する前に、スタンドアロンデータベースのバックアップを作成する必要があります。

オプションで、スタンドアロンのアンダークラウドデータベースバックアップを定期的なバックアップスケジュールに含めて、追加のデータセキュリティーを提供できます。アンダークラウドノードの完全バックアップには、アンダークラウドノードのデータベースバックアップが含まれます。ただし、完全なアンダークラウドの復元が失敗した場合、完全なアンダークラウドバックアップのデータベース部分にアクセスできなくなる可能性があります。この場合、スタンドアロンのアンダークラウドデータベースのバックアップからデータベースを復元できます。

手順

- アンダークラウドノードのデータベースのバックアップを作成します。

```
openstack undercloud backup --db-only
```

データベースのバックアップファイルは **/home/stack with the name openstack-backup-mysql-<timestamp>.sql** に保存されます。

関連情報

- Framework for Upgrades (13 to 16.2)
- 「アンダークラウドノードのバックアップの作成」
- 「アンダークラウドのノードデータベースを手動で復元する」

1.6. バックアップ用の OPEN VSWITCH (OVS) インターフェイスの設定

お使いの環境で Open vSwitch (OVS) ブリッジを使用する場合は、アンダークラウドまたはコントロールプレーンノードをバックアップする前に OVS インターフェイスを手動で設定する必要があります。復元プロセスでは、この情報を使用してネットワークインターフェイスを復元します。

手順

- **/etc/rear/local.conf** ファイルに、以下の形式で **NETWORKING_PREPARATION_COMMANDS** パラメーターを追加します。

```
NETWORKING_PREPARATION_COMMANDS=('<command_1>' '<command_2>' ...')
```

<command_1> および **<command_2>** を、ネットワークインターフェイス名または IP アドレスを設定するコマンドに置き換えます。たとえば、**ip link add br-ctlplane type bridge** コマンドを追加してコントロールプレーンのブリッジ名を設定するか、**ip link set eth0 up** コマンドを追加してインターフェイスの名前を設定できます。ネットワーク設定に基づいて、パラメーターにさらにコマンドを追加します。

1.7. アンダークラウドノードのバックアップの作成

アンダークラウドノードのバックアップを作成するには、**openstack undercloud backup** コマンドを使用します。その後、ノードが破損したりアクセスできなくなったりした場合に備えて、バックアップを使用して、アンダークラウドノードを以前の状態に復元できます。アンダークラウドノードのバックアップには、アンダークラウドノードで実行されるデータベースのバックアップが含まれます。

Red Hat OpenStack Platform 環境を 13 から 16.2 にアップグレードする場合は、アンダークラウドのアップグレードを実行してから、オーバークラウドノードで Leapp によるアップグレードプロセスを実施する前に、データベースのバックアップを別に作成する必要があります。詳細は、「アンダークラウドノードのスタンドアロンデータベースバックアップの作成」 を参照してください。

前提条件

- バックアップノードに NFS または SFTP サーバーがインストールおよび設定されている。新しい NFS サーバーの作成方法は 「バックアップノードへの NFS サーバーのインストールと設定」 を参照してください。
- アンダークラウドノードに ReaR がインストールされている。詳細は、「アンダークラウドノードへの ReaR のインストール」 を参照してください。
- ネットワークインターフェイスに OVS ブリッジを使用する場合は、OVS インターフェイスを設定している。詳細は、「バックアップ用の Open vSwitch (OVS) インターフェイスの設定」を参照してください。

手順

1. アンダークラウドに **stack** ユーザーとしてログインします。
2. MySQL の root パスワードを取得します。

```
[stack@undercloud ~]$ PASSWORD=$(sudo /bin/hiera -c /etc/puppet/hiera.yaml
mysql::server::root_password)
```

3. アンダークラウドノードのデータベースのバックアップを作成します。

```
[stack@undercloud ~]$ sudo podman exec mysql bash -c "mysqldump -uroot -
p$PASSWORD --opt --all-databases" | sudo tee /root/undercloud-all-databases.sql
```

4. source コマンドでアンダークラウドの認証情報を読み込みます。

```
[stack@undercloud ~]$ source stackrc
```

5. 事前に実行していない場合は、インベントリーファイルを作成し、**tripleo-ansible-inventory** コマンドを使用して、すべてのオーバークラウドノードのホストおよび変数が含まれる静的なインベントリーファイルを生成します。

```
(undercloud) [stack@undercloud ~]$ tripleo-ansible-inventory \
--ansible_ssh_user heat-admin \
--static-yaml-inventory /home/stack/tripleo-inventory.yaml
```

6. アンダークラウドノードのバックアップを作成します。

```
(undercloud) [stack@undercloud ~]$ openstack undercloud backup --inventory
/home/stack/tripleo-inventory.yaml
```

1.8. CRON を使用したアンダークラウドノードバックアップのスケジューリング

Ansible **backup-and-restore** ロールを使用して、ReaR でアンダークラウドノードのバックアップをスケジュールできます。**/var/log/rear-cron** ディレクトリーでログを確認できます。

前提条件

- バックアップノードに NFS または SFTP サーバーがインストールおよび設定されている。新しい NFS サーバーの作成方法は 「バックアップノードへの NFS サーバーのインストールと設定」 を参照してください。
- アンダークラウドおよびコントロールプレーンノードに ReaR がインストールされている。詳細は、「コントロールプレーンノードへの ReaR のインストール」 を参照してください。
- バックアップの保存用に、バックアップの場所に十分なディスク領域がある。

手順

1. コントロールプレーンノードのバックアップをスケジュールするには、以下のコマンドを実行します。デフォルトのスケジュールは日曜日の午前 0 時です。

```
openstack undercloud backup --cron
```

2. オプション: デプロイメントに応じてスケジュールが設定されたバックアップをカスタマイズします。
   - デフォルトのバックアップスケジュールを変更するには、**tripleo_backup_and_restore_cron** パラメーターに別の cron スケジュールを渡します。

```
openstack undercloud backup --cron --extra-vars
'{"tripleo_backup_and_restore_cron": "0 0 * * 0"}'
```

   - cron がスケジュールされたバックアップを実行する際にバックアップコマンドに追加されるパラメーターを定義するには、以下の例のように **tripleo_backup_and_restore_cron_extra** パラメーターをバックアップコマンドに渡します。

```
openstack undercloud backup --cron --extra-vars
'{"tripleo_backup_and_restore_cron_extra":"--extra-vars bar-vars.yaml --inventory
/home/stack/tripleo-inventory.yaml"}'
```

   - バックアップを実行するデフォルトユーザーを変更するには、以下の例のように **tripleo_backup_and_restore_cron_user** パラメーターを backup コマンドに渡します。

```
openstack undercloud backup --cron --extra-vars
'{"tripleo_backup_and_restore_cron_user": "root"}
```

第2章 コントロールプレーンノードのバックアップ

コントロールプレーンノードをバックアップするには、バックアップノードを設定し、コントロールプレーンノードに Relax-and-Recover ツールをインストールし、バックアップイメージを作成します。バックアップは、通常の環境メンテナンスの一環として作成できます。

さらに、更新またはアップグレードを実行する前にコントロールプレーンノードをバックアップする必要があります。更新またはアップグレード時にエラーが発生した場合は、バックアップを使用して、コントロールプレーンノードを以前の状態に復元できます。

2.1. サポート対象のバックアップ形式およびプロトコル

アンダークラウドおよびバックアップ/復元プロセスは、オープンソースのツールである Relax-and-Recover (ReaR) を使用してブート可能なバックアップイメージを作成し、復元します。ReaR は Bash で記述されており、複数のイメージ形式と複数の転送プロトコルをサポートします。

次のリストは、ReaR を使用してアンダークラウドとコントロールプレーンをバックアップおよび復元するときに Red Hat OpenStack Platform がサポートするバックアップ形式とプロトコルを示しています。

ブート可能なメディア形式

- ISO

ファイルトランスポートプロトコル

- SFTP
- NFS

2.2. バックアップノードへの NFS サーバーのインストールと設定

バックアップファイルを保存するために、新しい NFS サーバーをインストールして設定できます。バックアップノードに NFS サーバーをインストールして設定するには、インベントリーファイルと SSH キーを順次作成し、NFS サーバーオプションを指定して **openstack undercloud backup** コマンドを実行します。

重要

- NFS サーバーまたは SFTP サーバーをインストールして設定している場合は、この手順を実行する必要はありません。バックアップするノードに ReaR を設定するときに、サーバー情報を入力します。
- デフォルトでは、NFS サーバーの Relax and Recover (ReaR) IP アドレスパラメーターは **192.168.24.1** です。使用している環境に一致する IP アドレス値を設定するには、**tripleo_backup_and_restore_server** パラメーターを追加する必要があります。

手順

1. アンダークラウドノードにおいて、source コマンドでアンダークラウドの認証情報を読み込みます。

```
[stack@undercloud ~]$ source stackrc
(undercloud) [stack@undercloud ~]$
```

2. アンダークラウドノードで、バックアップノードのインベントリーファイルを作成します。

```
(undercloud) [stack@undercloud ~]$ cat <<'EOF'> ~/nfs-inventory.yaml
[BackupNode]
<backup_node> ansible_host=<ip_address> ansible_user=<user>
EOF
```

<ip_address> および **<user>** を、実際の環境に該当する値に置き換えます。

3. 公開 SSH 鍵をアンダークラウドノードからバックアップノードにコピーします。

```
(undercloud) [stack@undercloud ~]$ ssh-copy-id -i ~/.ssh/id_rsa.pub <backup_node>
```

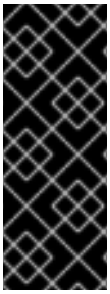
<backup_node> をバックアップノードのパスおよび名前に置き換えます。

4. バックアップノードに NFS サーバーを設定します。

```
(undercloud) [stack@undercloud ~]$ openstack undercloud backup --setup-nfs --extra-vars
/home/stack/bar-vars.yaml --inventory /home/stack/nfs-inventory.yaml
```

2.3. コントロールプレーンノードへの REAR のインストール

オーバークラウドコントロールプレーンのバックアップを作成する前に、各コントロールプレーンノードに Relax and Recover (ReaR) をインストールして設定します。



重要

既知の問題が原因でコントローラーノードがダウンしても、オーバークラウドノードの ReaR バックアップは継続されます。ReR バックアップを実行する前に、すべてのコントローラーノードが実行されていることを確認してください。今後の Red Hat OpenStack Platform (RHOSP) リリースで修正される予定です。詳細は BZ#2077335 - Back up of the overcloud ctlplane keeps going even if one controller is unreachable を参照してください。

前提条件

- バックアップノードに NFS または SFTP サーバーがインストールおよび設定されている。新しい NFS サーバーの作成方法は 「バックアップノードへの NFS サーバーのインストールと設定」 を参照してください。

手順

1. アンダークラウドノードにおいて、source コマンドでアンダークラウドの認証情報を読み込みます。

```
[stack@undercloud ~]$ source stackrc
```

2. 事前に実行していない場合は、インベントリーファイルを作成し、**tripleo-ansible-inventory** コマンドを使用して、すべてのオーバークラウドノードのホストおよび変数が含まれる静的なインベントリーファイルを生成します。

```
(undercloud) [stack@undercloud ~]$ tripleo-ansible-inventory \
--ansible_ssh_user heat-admin \
--static-yaml-inventory /home/stack/tripleo-inventory.yaml
```

3. **bar-vars.yaml** ファイルで、バックアップの保存場所を設定します。NFS サーバーまたは SFTP サーバーに適した手順を実行します。

   a. NFS サーバーを使用する場合は、以下のパラメーターを **bar-vars.yaml** ファイルに追加します。

   ```
   tripleo_backup_and_restore_server: <ip_address>
   tripleo_backup_and_restore_shared_storage_folder: <backup_server_dir_path>
   tripleo_backup_and_restore_output_url: "nfs://{{ tripleo_backup_and_restore_server }}{{
   tripleo_backup_and_restore_shared_storage_folder }}"
   tripleo_backup_and_restore_backup_url: "nfs://{{ tripleo_backup_and_restore_server }}{{
   tripleo_backup_and_restore_shared_storage_folder }}"
   ```

   <ip_address>`and `<backup_server_dir_path> を置き換えます。**tripleo_backup_and_restore_server** パラメーターのデフォルト値は **192.168.24.1** です。

   b. SFTP サーバーを使用する場合は、**tripleo_backup_and_restore_output_url** パラメーターを追加し、SFTP サーバーの URL と認証情報の値を設定します。

   ```
   tripleo_backup_and_restore_output_url: sftp://<user>:<password>@<backup_node>/
   tripleo_backup_and_restore_backup_url: iso:///backup/
   ```

   <user>、**<password>**、および **<backup_node>** を、バックアップノードの URL および認証情報に置き換えます。

4. コントロールプレーンノードに ReaR をインストールします。

```
(undercloud) [stack@undercloud ~]$ openstack overcloud backup --setup-rear --extra-vars
/home/stack/bar-vars.yaml --inventory /home/stack/tripleo-inventory.yaml
```

5. システムで UEFI ブートローダーを使用している場合は、コントロールプレーンノードで以下の手順を実行します。

   a. 以下のツールをインストールします。

   ```
   $ sudo dnf install dosfstools efibootmgr
   ```

   b. **USING_UEFI_BOOTLOADER** パラメーターの値 **0** を値 **1** に置き換えて、**/etc/rear/local.conf** にある ReaR 設定ファイルで UEFI バックアップを有効にします。

2.4. バックアップ用の OPEN VSWITCH (OVS) インターフェイスの設定

お使いの環境で Open vSwitch (OVS) ブリッジを使用する場合は、アンダークラウドまたはコントロールプレーンノードをバックアップする前に OVS インターフェイスを手動で設定する必要があります。復元プロセスでは、この情報を使用してネットワークインターフェイスを復元します。

手順

- **/etc/rear/local.conf** ファイルに、以下の形式で **NETWORKING_PREPARATION_COMMANDS** パラメーターを追加します。

```
NETWORKING_PREPARATION_COMMANDS=('<command_1>' '<command_2>' ...')
```

<command_1> および **<command_2>** を、ネットワークインターフェイス名または IP アドレスを設定するコマンドに置き換えます。たとえば、**ip link add br-ctlplane type bridge** コマンドを追加してコントロールプレーンのブリッジ名を設定するか、**ip link set eth0 up** コマンドを追加してインターフェイスの名前を設定できます。ネットワーク設定に基づいて、パラメーターにさらにコマンドを追加します。

2.5. コントロールプレーンノードのバックアップの作成

コントロールプレーンノードのバックアップを作成するには、**openstack overcloud backup** コマンドを使用します。その後、ノードが破損したりアクセスできなくなったりした場合に備えて、バックアップを使用して、コントロールプレーンノードを以前の状態に復元できます。コントロールプレーンのバックアップには、コントロールプレーンノードで実行されるデータベースのバックアップが含まれます。

前提条件

- バックアップノードに NFS または SFTP サーバーがインストールおよび設定されている。新しい NFS サーバーの作成方法は 「バックアップノードへの NFS サーバーのインストールと設定」 を参照してください。
- コントロールプレーンノードに ReaR がインストールされている。詳細は、「コントロールプレーンノードへの ReaR のインストール」 を参照してください。
- ネットワークインターフェイスに OVS ブリッジを使用する場合は、OVS インターフェイスを設定している。詳細は、「バックアップ用の Open vSwitch (OVS) インターフェイスの設定」を参照してください。

手順

1. 各コントロールプレーンノードで **config-drive** パーティションを見つけます。

```
[stack@undercloud ~]$ lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda    253:0    0   55G  0 disk
├─vda1 253:1    0    1M  0 part          (1)
├─vda2 253:2    0  100M  0 part /boot/efi
└─vda3 253:3    0 54.9G  0 part /
```

(1) **config-drive** パーティションは、マウントされていない 1M パーティションです。

2. 各コントロールプレーンノードで、各ノードの **config-drive** パーティションを **root** ユーザーとしてバックアップします。

```
[root@controller-x ~]# dd if=<config_drive_partition> of=/mnt/config-drive
```

<config_drive_partition> を、手順 1 で見つけた **config-drive** パーティションの名前に置き換えます。

3. アンダークラウドノードにおいて、source コマンドでアンダークラウドの認証情報を読み込みます。

```
[stack@undercloud ~]$ source stackrc
```

4. 事前に実行していない場合は、**tripleo-ansible-inventory** コマンドを使用して、すべてのオーバークラウドノードのホストおよび変数が含まれる静的なインベントリーファイルを生成します。

```
(undercloud) [stack@undercloud ~]$ tripleo-ansible-inventory \
--ansible_ssh_user heat-admin \
--static-yaml-inventory /home/stack/tripleo-inventory.yaml
```

5. コントロールプレーンノードのバックアップを作成します。

```
(undercloud) [stack@undercloud ~]$ openstack overcloud backup --inventory
/home/stack/tripleo-inventory.yaml
```

バックアッププロセスは、環境へのサービスを中断することなく、各コントロールプレーンノードで順番に実行されます。

2.6. CRON を使用したコントロールプレーンノードバックアップのスケジューリング

Ansible **backup-and-restore** ロールを使用して、ReaR でコントロールプレーンノードのバックアップをスケジュールできます。**/var/log/rear-cron** ディレクトリーでログを確認できます。

前提条件

- バックアップノードに NFS または SFTP サーバーがインストールおよび設定されている。新しい NFS サーバーの作成方法は 「バックアップノードへの NFS サーバーのインストールと設定」 を参照してください。
- アンダークラウドおよびコントロールプレーンノードに ReaR がインストールされている。詳細は、「コントロールプレーンノードへの ReaR のインストール」 を参照してください。
- バックアップの保存用に、バックアップの場所に十分なディスク領域がある。

手順

1. コントロールプレーンノードのバックアップをスケジュールするには、以下のコマンドを実行します。デフォルトのスケジュールは日曜日の午前 0 時です。

```
openstack overcloud backup --cron
```

2. オプション: デプロイメントに応じてスケジュールが設定されたバックアップをカスタマイズします。
   - デフォルトのバックアップスケジュールを変更するには、**tripleo_backup_and_restore_cron** パラメーターに別の cron スケジュールを渡します。

```
openstack overcloud backup --cron --extra-vars
'{"tripleo_backup_and_restore_cron": "0 0 * * 0"}'
```

- cron がスケジュールされたバックアップを実行する際にバックアップコマンドに追加されるパラメーターを定義するには、以下の例のように **tripleo_backup_and_restore_cron_extra** パラメーターをバックアップコマンドに渡します。

```
openstack overcloud backup --cron --extra-vars
'{"tripleo_backup_and_restore_cron_extra":"--extra-vars bar-vars.yaml --inventory
/home/stack/tripleo-inventory.yaml"}'
```

- バックアップを実行するデフォルトユーザーを変更するには、以下の例のように **tripleo_backup_and_restore_cron_user** パラメーターを backup コマンドに渡します。

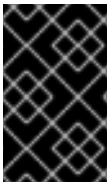
```
openstack overcloud backup --cron --extra-vars
'{"tripleo_backup_and_restore_cron_user": "root"}
```

第3章 アンダークラウドおよびコントロールプレーンノードの復元

アンダークラウドまたはコントロールプレーンノードが破損している場合、もしくは更新またはアップグレード中にエラーが発生した場合は、コントロールプレーンノードをバックアップから以前の状態に復元できます。復元プロセスが Galera クラスターまたは共存する Ceph モニターを持つノードを自動的に復元できない場合は、これらのコンポーネントを手動で復元できます。

3.1. 復元プロセス用に同じ場所に配置された CEPH モニターを備えたコントロールプレーンを準備する

同じ場所に配置された Ceph モニターを使用してコントロールプレーンノードを復元する前に、Ceph モニターのバックアップファイルをノードファイルシステムにマウントするスクリプトと、ReaR がバックアップファイルを見つけるために使用する別のスクリプトを作成して、環境を準備します。



重要

/var/lib/ceph ディレクトリーのバックアップが作成できない場合は、Red Hat テクニカルサポートチームに連絡して **ceph-mon** インデックスを再ビルドする必要があります。詳細は、Red Hat Technical Support Team を参照してください。

前提条件

- アンダークラウドノードのバックアップを作成している。詳細は、「アンダークラウドノードのバックアップの作成」 を参照してください。
- コントロールプレーンノードのバックアップを作成している。詳細は、「コントロールプレーンノードのバックアップの作成」 を参照してください。
- バックアップノードにアクセスできる。
- **NETWORKING_PREPARATION_COMMANDS** パラメーターで設定したネットワーク設定情報にアクセスできる (ネットワークインターフェイスの OVS ブリッジを使用する場合)。詳細は、「バックアップ用の Open vSwitch (OVS) インターフェイスの設定」 を参照してください。

手順

1. 復元する各ノードで、スクリプト **/usr/share/rear/setup/default/011_backup_ceph.sh** を作成し、次のコンテンツを追加します。

```
mount -t <file_type> <device_disk> /mnt/local
cd /mnt/local
[ -d "var/lib/ceph" ] && tar cvfz /tmp/ceph.tar.gz var/lib/ceph --xattrs --xattrs-include='.' --acls
cd /
umount <device_disk>
```

   <file_type> および **<device_disk>** は、バックアップファイルの種類と場所に置き換えてください。通常、ファイルタイプは **xfs** で、場所は **/dev/vda2** です。

2. 同じノードで、スクリプト **/usr/share/rear/wrapup/default/501_restore_ceph.sh** を作成し、以下の出力を追加します。

```
if [ -f "/tmp/ceph.tar.gz" ]; then
  rm -rf /mnt/local/var/lib/ceph/*
```

```
  tar xvC /mnt/local -f /tmp/ceph.tar.gz var/lib/ceph --xattrs --xattrs-include='.'
fi
```

関連情報

- 「アンダークラウドノードの復元」
- 「コントロールプレーンノードの復元」

3.2. アンダークラウドノードの復元

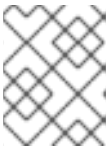
ReaR を使用して作成したバックアップの ISO イメージを使用し、アンダークラウドノードを以前の状態に復元できます。バックアップの ISO イメージは、バックアップノードにあります。ブート可能な ISO イメージを DVD に書き込むか、Integrated Lights-Out (iLO) リモートアクセスを通じてアンダークラウドノードにダウンロードします。

前提条件

- アンダークラウドノードのバックアップを作成している。詳細は、「コントロールプレーンノードのバックアップの作成」を参照してください。
- バックアップノードにアクセスできる。
- **NETWORKING_PREPARATION_COMMANDS** パラメーターで設定したネットワーク設定情報にアクセスできる (ネットワークインターフェイスの OVS ブリッジを使用する場合)。詳細は、「バックアップ用の Open vSwitch (OVS) インターフェイスの設定」 を参照してください。

手順

1. アンダークラウドノードの電源をオフにします。次のステップに進む前に、アンダークラウドノードの電源が完全にオフになっていることを確認します。
2. バックアップの ISO イメージでアンダークラウドノードをブートします。
3. **Relax-and-Recover** ブートメニューが表示されたら、**Recover <undercloud_node>** を選択します。**<undercloud_node>** をアンダークラウドノードの名前に置き換えます。



注記

システムで UEFI を使用している場合は、**Relax-and-Recover (no Secure Boot)** オプションを選択します。

4. **root** ユーザーとしてログインし、ノードを復元します。
以下のメッセージが表示されます。

```
Welcome to Relax-and-Recover. Run "rear recover" to restore your system!
RESCUE <undercloud_node>:~ # rear recover
```

アンダークラウドノードの復元プロセスが完了すると、コンソールに以下のメッセージが表示されます。

```
Finished recovering your system
Exiting rear recover
Running exit tasks
```

5. ノードの電源を切ります。

```
RESCUE <undercloud_node>:~ #  poweroff
```

ノードをブートすると、以前の状態で再開されます。

3.3. コントロールプレーンノードの復元

更新またはアップグレード中にエラーが発生した場合は、ReaR を使用して作成したバックアップの ISO イメージを使用して、コントロールプレーンノードを以前の状態に復元できます。

コントロールプレーンを復元する場合は、状態の整合性を確保するために、すべてのコントロールプレーンノードを復元する必要があります。

バックアップの ISO イメージは、バックアップノードにあります。ブート可能な ISO イメージを DVD に書き込むか、Integrated Lights-Out (iLO) リモートアクセスを通じてアンダークラウドノードにダウンロードします。

注記

Red Hat は、Open vSwitch (OVS) およびデフォルトの Open Virtual Network (OVN) などのネイティブ SDN を使用する Red Hat OpenStack Platform のバックアップをサポートします。サードパーティーの SDN の詳細は、サードパーティーの SDN ドキュメントを参照してください。

前提条件

- コントロールプレーンノードのバックアップを作成している。詳細は、「コントロールプレーンノードのバックアップの作成」 を参照してください。
- バックアップノードにアクセスできる。
- **NETWORKING_PREPARATION_COMMANDS** パラメーターで設定したネットワーク設定情報にアクセスできる (ネットワークインターフェイスの OVS ブリッジを使用する場合)。詳細は、「バックアップ用の Open vSwitch (OVS) インターフェイスの設定」 を参照してください。

手順

1. 各コントロールプレーンノードの電源をオフにします。次のステップに進む前に、コントロールプレーンノードの電源が完全にオフになっていることを確認します。
2. 対応するバックアップの ISO イメージで各コントロールプレーンノードをブートします。
3. **Relax-and-Recover** ブートメニューが表示されたら、各コントロールプレーンノードで **Recover <control_plane_node>** を選択します。**<control_plane_node>** を対応するコントロールプレーンノードの名前に置き換えます。

注記

システムで UEFI を使用している場合は、**Relax-and-Recover (no Secure Boot)** オプションを選択します。

4. それぞれのコントロールプレーンノードで **root** ユーザーとしてログインし、ノードを復元します。
以下のメッセージが表示されます。

```
Welcome to Relax-and-Recover. Run "rear recover" to restore your system!
RESCUE <control_plane_node>:~ # rear recover
```

コントロールプレーンノードの復元プロセスが完了すると、コンソールに以下のメッセージが表示されます。

```
Finished recovering your system
Exiting rear recover
Running exit tasks
```

5. コマンドラインコンソールが利用可能な場合は、各コントロールプレーンノードの **config-drive** パーティションを復元します。

```
# once completed, restore the config-drive partition (which is ISO9660)
RESCUE <control_plane_node>:~ $ dd if=/mnt/local/mnt/config-drive of=
<config_drive_partition>
```

6. ノードの電源を切ります。

```
RESCUE <control_plane_node>:~ #  poweroff
```

7. ブートシーケンスを通常のブートデバイスに設定します。ノードをブートすると、以前の状態で再開されます。

8. サービスが正常に実行されていることを確認するには、pacemaker のステータスを確認します。**root** ユーザーとしてコントローラーノードにログインし、以下のコマンドを入力します。

```
# pcs status
```

9. オーバークラウドのステータスを確認するには、OpenStack Integration Test Suite (tempest) を使用します。詳細は、Validating your OpenStack cloud with the Integration Test Suite (tempest) を参照してください。

トラブルシューティング

- **pcs status** で表示されるリソースアラームを以下のコマンドで解除します。

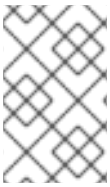
```
# pcs resource clean
```

- **pcs status** で表示される STONITH フェンシングの動作エラーを以下のコマンドで解除します。

```
# pcs resource clean
# pcs stonith history cleanup
```

3.4. GALERA クラスターの手動による復元

復元手順の一環として Galera クラスターが復元されない場合は、Galera を手動で復元する必要があります。



注記

以下の手順では、1 つのコントローラーノードでいくつかのステップを実施する必要があります。手順の実施と同じコントローラーノードで、これらのステップを実施してください。

手順

1. **Controller-0** で、Galera クラスターの仮想 IP を取得します。

```
$ sudo hiera -c /etc/puppet/hiera.yaml mysql_vip
```

2. すべてのコントローラーノードで、仮想 IP を通じたデータベース接続を無効にします。

```
$ sudo iptables -I INPUT  -p tcp --destination-port 3306 -d $MYSQL_VIP  -j DROP
```

3. **Controller-0** で MySQL の root パスワードを取得します。

```
$ sudo hiera -c /etc/puppet/hiera.yaml mysql::server::root_password
```

4. **Controller-0** で、Galera リソースを **unmanaged** モードに設定します。

```
$ sudo pcs resource unmanage galera-bundle
```

5. すべてのコントローラーノードで、MySQL コンテナーを停止します。

```
$ sudo podman container stop $(sudo podman container ls --all --format "{{.Names}}" --
filter=name=galera-bundle)
```

6. すべてのコントローラーノードで、現在のディレクトリーを移動します。

```
$ sudo mv /var/lib/mysql /var/lib/mysql-save
```

7. すべてのコントローラーノードで、新規ディレクトリー **/var/lib/mysq** を作成します。

```
$ sudo mkdir /var/lib/mysql
$ sudo chown 42434:42434 /var/lib/mysql
$ sudo chcon -t container_file_t /var/lib/mysql
$ sudo chmod 0755 /var/lib/mysql
$ sudo chcon -r object_r /var/lib/mysql
$ sudo chcon -u system_u /var/lib/mysql
```

8. すべてのコントローラーノードで、MySQL コンテナーを起動します。

```
$ sudo podman container start $(sudo podman container ls --all --format "{{ .Names }}" --
filter=name=galera-bundle)
```

9. すべてのコントローラーノードで、MySQL データベースを作成します。

```
$ sudo podman exec -i $(sudo podman container ls --all --format "{{ .Names }}" \
    --filter=name=galera-bundle) bash -c "mysql_install_db --datadir=/var/lib/mysql --
user=mysql --log_error=/var/log/mysql/mysql_init.log"
```

10. すべてのコントローラーノードで、データベースを起動します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}" \
    --filter=name=galera-bundle) bash -c "mysqld_safe --skip-networking --wsrep-on=OFF --
log-error=/var/log/mysql/mysql_safe.log" &
```

11. すべてのコントローラーノードで、**.my.cnf** Galera 設定ファイルを移動します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}" \
    --filter=name=galera-bundle) bash -c "mv /root/.my.cnf /root/.my.cnf.bck"
```

12. すべてのコントローラーノードで、Galera root パスワードをリセットします。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}"  \
    --filter=name=galera-bundle) bash -c "mysql -uroot -e'use mysql;update user set
password=PASSWORD(\"$ROOTPASSWORD\")where User=\"root\";flush privileges;'"
```

13. すべてのコントローラーノード上の Galera コンテナー内で、**.my.cnf** Galera 設定ファイルを復元します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}"   \
    --filter=name=galera-bundle) bash -c "mv /root/.my.cnf.bck /root/.my.cnf"
```

14. **Controller-0** で、バックアップデータベースファイルを **/var/lib/MySQL** にコピーします。

```
$ sudo cp $BACKUP_FILE /var/lib/mysql
$ sudo cp $BACKUP_GRANT_FILE /var/lib/mysql
```

注記

これらのファイルへのパスは /home/heat-admin/ です。

15. **Controller-0** で、MySQL データベースを復元します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}"    \
--filter=name=galera-bundle) bash -c "mysql -u root -p$ROOT_PASSWORD <
\"/var/lib/mysql/$BACKUP_FILE\"  "

$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}"    \
--filter=name=galera-bundle) bash -c "mysql -u root -p$ROOT_PASSWORD <
\"/var/lib/mysql/$BACKUP_GRANT_FILE\"  "
```

16. すべてのコントローラーノードで、データベースをシャットダウンします。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}"    \
    --filter=name=galera-bundle) bash -c "mysqladmin shutdown"
```

17. **Controller-0** で、ブートストラップノードを起動します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}"  --
filter=name=galera-bundle) \
      /usr/bin/mysqld_safe --pid-file=/var/run/mysql/mysqld.pid --
socket=/var/lib/mysql/mysql.sock --datadir=/var/lib/mysql \
      --log-error=/var/log/mysql/mysql_cluster.log  --user=mysql --open-files-limit=16384 \
      --wsrep-cluster-address=gcomm:// &
```

18. 検証: Controller-0 で、クラスターのステータスを確認します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}" \
       --filter=name=galera-bundle) bash -c "clustercheck"
```

Galera cluster node is synced のメッセージが表示されるのを確認してください。表示されない場合は、ノードを再作成する必要があります。

19. **Controller-0** で、設定からクラスターアドレスを取得します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}" \
--filter=name=galera-bundle) bash -c "grep wsrep_cluster_address /etc/my.cnf.d/galera.cnf" |
awk '{print $3}'
```

20. 残りの各コントローラーノードでデータベースを起動し、クラスターを検証します。

    a. データベースを起動します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}" \
    --filter=name=galera-bundle) /usr/bin/mysqld_safe --pid-
file=/var/run/mysql/mysqld.pid --socket=/var/lib/mysql/mysql.sock \
    --datadir=/var/lib/mysql --log-error=/var/log/mysql/mysql_cluster.log  --user=mysql --
open-files-limit=16384 \
    --wsrep-cluster-address=$CLUSTER_ADDRESS &
```

    b. MYSQL クラスターのステータスを確認します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}" \
       --filter=name=galera-bundle) bash -c "clustercheck"
```

Galera cluster node is synced のメッセージが表示されるのを確認してください。表示されない場合は、ノードを再作成する必要があります。

21. すべてのコントローラーノードで MySQL コンテナーを停止します。

```
$ sudo podman exec $(sudo podman container ls --all --format "{{ .Names }}" --
filter=name=galera-bundle) \
      /usr/bin/mysqladmin -u root shutdown
```

22. すべてのコントローラーノードで以下のファイアウォールルールを削除して、仮想 IP アドレス経由のデータベース接続を許可します。

```
$ sudo iptables -D  INPUT  -p tcp --destination-port 3306 -d $MYSQL_VIP  -j DROP
```

23. すべてのコントローラーノードで MySQL コンテナーを再起動します。

```
$ sudo podman container restart $(sudo podman container ls --all --format  "{{ .Names }}" --filter=name=galera-bundle)
```

24. すべてのコントローラーノードで **clustercheck** コンテナーを再起動します。

```
$ sudo podman container restart $(sudo podman container ls --all --format  "{{ .Names }}" --filter=name=clustercheck)
```

25. **Controller-0** で、Galera リソースを **managed** モードに設定します。

```
$ sudo pcs resource manage galera-bundle
```

検証

1. サービスが正常に実行されていることを確認するには、pacemaker のステータスを確認します。

```
$ sudo pcs status
```

2. オーバークラウドのステータスを確認するには、OpenStack Integration Test Suite (tempest) を使用します。詳細は、Validating your OpenStack cloud with the Integration Test Suite (tempest) を参照してください。

3. 特定のノードで問題が疑われる場合は、**clustercheck** でクラスターの状態を確認します。

```
$ sudo podman exec clustercheck /usr/bin/clustercheck
```

3.5. アンダークラウドのノードデータベースを手動で復元する

アンダークラウドのデータベースがアンダークラウドの復元プロセスの一部として復元されない場合は、データベースを手動で復元できます。データベースを復元できるのは、以前にスタンドアロンのデータベースバックアップを作成した場合のみです。

前提条件

- アンダークラウドデータベースのスタンドアロンバックアップを作成している。詳細は、「アンダークラウドノードのスタンドアロンデータベースバックアップの作成」 を参照してください。

手順

1. director アンダークラウドノードに **root** ユーザーとしてログインします。

2. すべての tripleo サービスを停止します。

```
[root@director ~]# systemctl  stop  tripleo_*
```

3. 次のコマンドを入力して、サーバーでコンテナーが実行していないことを確認します。

```
[root@director ~]# podman ps
```

実行中のコンテナーがある場合は、次のコマンドを入力してコンテナーを停止します。

```
[root@director ~]# podman stop <container_name>
```

4. 現在の **/var/lib/mysql** ディレクトリーのバックアップを作成してから、そのディレクトリーを削除します。

```
[root@director ~]# cp -a /var/lib/mysql /var/lib/mysql_bck
[root@director ~]# rm -rf /var/lib/mysql
```

5. データベースディレクトリーを再作成し、新しいディレクトリーに SELinux の属性を設定します。

```
[root@director ~]# mkdir /var/lib/mysql
[root@director ~]# chown 42434:42434 /var/lib/mysql
[root@director ~]# chmod 0755 /var/lib/mysql
[root@director ~]# chcon -t container_file_t /var/lib/mysql
[root@director ~]# chcon -r object_r /var/lib/mysql
[root@director ~]# chcon -u system_u /var/lib/mysql
```

6. **mariadb** イメージのローカルタグを作成します。**<image_id>** および **<undercloud.ctlplane.example.com>** を、使用している環境の値に置き換えます。

```
[root@director ~]# podman images | grep mariadb
<undercloud.ctlplane.example.com>:8787/rh-osbs/rhosp16-openstack-mariadb
16.2_20210322.1   <image_id>   3 weeks ago   718 MB
```

```
[root@director ~]# podman tag <image_id> mariadb
```

```
[root@director ~]# podman images | grep maria
localhost/mariadb                                                    latest        <image_id>   3
weeks ago   718 MB
<undercloud.ctlplane.example.com>:8787/rh-osbs/rhosp16-openstack-mariadb
16.2_20210322.1   <image_id>   3 weeks ago   718 MB
```

7. **/var/lib/mysql** ディレクトリーをコンテナーで初期化します。

```
[root@director ~]# podman run --net=host -v /var/lib/mysql:/var/lib/mysql localhost/mariadb
mysql_install_db --datadir=/var/lib/mysql --user=mysql
```

8. データベースにインポートするデータベースのバックアップファイルをコピーします。

```
[root@director ~]# cp /root/undercloud-all-databases.sql /var/lib/mysql
```

9. データベースサービスを開始して、データをインポートします。

```
[root@director ~]# podman run --net=host -dt -v /var/lib/mysql:/var/lib/mysql
localhost/mariadb  /usr/libexec/mysqld
```

10. データを読み込み、**max_allowed_packet** パラメーターを設定します。

a. コンテナーにログインし、設定を行います。

```
[root@director ~]# podman exec -it <container_id> /bin/bash
    ()[mysql@5a4e429c6f40 /]$ mysql -u root -e "set global max_allowed_packet =
1073741824;"
    ()[mysql@5a4e429c6f40 /]$ mysql -u root < /var/lib/mysql/undercloud-all-
databases.sql
    ()[mysql@5a4e429c6f40 /]$ mysql -u root -e 'flush privileges'
    ()[mysql@5a4e429c6f40 /]$ exit
    exit
```

b. コンテナーを停止します。

```
[root@director ~]# podman stop <container_id>
```

c. コンテナーが動いていないことを確認します。

```
[root@director ~]# podman ps
CONTAINER ID  IMAGE  COMMAND  CREATED  STATUS  PORTS  NAMES
[root@director ~]#
```

11. すべての tripleo サービスを再起動します。

```
[root@director ~]# systemctl start multi-user.target
```