Red Hat OpenStack Platform 10
カスタムの Block Storage バックエンドのデプロイメントガイド

Red Hat OpenStack Platform オーバークラウドでカスタムの Block Storage バックエンドをデプロイするためのガイド

OpenStack Team

Red Hat OpenStack Platform 10 カスタムの Block Storage バックエンドのデプロイメントガイド

Red Hat OpenStack Platform オーバークラウドでカスタムの Block Storage バックエンドをデプロイするためのガイド

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

本書では、Block Storage サービス向けに、統合されていないカスタムのバックエンドを Red Hat OpenStack Platform 10 のオーバークラウドにデプロイする方法について説明します。

目次

1. はじめに **2**
1.1. カスタムのバックエンド 2
1.2. 要件 3

2. プロセスの説明 **3**

3. 環境ファイルの作成 **3**

4. 設定したバックエンドのデプロイ **6**

5. 設定したバックエンドのテスト **6**

A. 付録 **8**
A.1. stack ユーザー 8
A.2. サンプルの環境ファイルに基づいた設定 8

1. はじめに

Red Hat OpenStack Platform director は、完全な OpenStack 環境をインストールおよび管理するためのツールセットで、主にアップストリームの TripleO (**OpenStack-on-OpenStack**) プロジェクトをベースとしています。director の主な目的は、手動での設定を最低限に抑えて、機能的なエンタープライズレベルの OpenStack デプロイメントを完全にオーケストレーションすることです。director は、個別の OpenStack コンポーネントの手動設定に内在する数多くの問題に対処するのに役立ちます。

director が作成して最終的に構築される OpenStack デプロイメントは、**オーバークラウド** と呼ばれます。オーバークラウドには、Block Storage など、エンドユーザーにサービスを提供するコンポーネントがすべて含まれます。本書は、オーバークラウドの Block Storage サービスにカスタムのバックエンドをデプロイする方法を説明します。

本書は、Block Storage の手動設定についての知識があることを前提とします。Packstack などを利用した OpenStack のテストデプロイメントでは、このサービスを設定するにはホストノードの **`/etc/cinder/cinder.conf`** を編集する必要があります。このファイルでの Block Storage の設定は、別のドキュメントで詳しく説明されていますが、本書では、**カスタムのバックエンド** をアタッチするために、これらの設定をオーバークラウドに適用する方法を記載しています。

> **警告**
>
> 以下の手順は、検証に成功していますが、使用したユースケースは限られています。予定のデプロイメントは、最初に実稼働環境以外の環境でテストするようにしてください。質問がある場合は、Red Hat のサポートにお問い合わせください。

1.1. カスタムのバックエンド

本書では、ストレージサーバー/アプライアンスまたは Red Hat OpenStack Platform director にまだ完全統合されていない設定として、カスタムのバックエンドを定義します。サポートされる Block Storage バックエンドの一部はすでに director に統合されているため、事前設定済みの director ファイルが追加の設定なしに提供されます。統合済みのバックエンドは、これらのファイルを使用して、オーバークラウドに設定、デプロイすることができます。たとえば、統合済みのバックエンドには、Red Hat Ceph や、Dell EqualLogic、Dell Storage Center、NetApp アプライアンスの単一バックエンド設定などが含まれます。

また、director にすでに組み込まれている一部のストレージアプライアンスは、単一インスタンスのバックエンドのみをサポートします。たとえば、Dell EqualLogic 用に事前設定済みの director ファイルは、単一バックエンドのデプロイメントにしか利用できません。このアプライアンスの複数のバックエンドインスタンスをデプロイするには、本書で実例をあげて説明しているように、カスタムの設定が必要です。

Block Storage サービスは、ノードの **`/etc/cinder/cinder.conf`** を直接編集して、手動で設定することが可能ですが、**openstack overcloud deploy** を実行すると director により設定が上書きされてしまいます (このオーバークラウドのデプロイメントのステップは、後ほど「設定したバックエンドのデプロイ」で行います)。このため、Block Storage バックエンドは director でデプロイして、オーバークラウドのデプロイメントやアップグレードを実行しても設定が変更されないようにすることを推奨します

バックエンドの設定がすでに完全に統合されている場合には、単に編集して、パッケージされている環境ファイルを呼び出すことができます。ただし、カスタムのバックエンドでは、独自の「環境ファイル」を記述する必要があります。本ガイドには、/home/stack/templates/custom-env.yaml という注釈付きのサンプルが含まれており、独自のデプロイメント用に編集することができます。このサンプルファイルは、Block Storage サービスが NetApp バックエンドを 2 つ使用するように設定するのに適しています。

1.2. 要件

本書では、Block Storage とデプロイするバックエンドを手動で設定する予備知識に加えて、以下の条件を満たしていることを前提とします。

- サードパーティー製のバックエンドのアプライアンスを使用する場合は、そのアプライアンスがストレージリポジトリーとして正しく設定済みであること。
- オーバークラウドは、『director のインストールと使用方法』の手順に従って、director を使用してデプロイ済みであること。
- 管理者権限のあるアカウントのユーザー名とパスワードが用意されていること。「director のインストールユーザーの作成」でオーバークラウドのデプロイ用に作成したのと同じアカウントを使用することができます。stack ユーザーはこの目的のために作成、使用します。
- **`/etc/cinder/cinder.conf`** の Block Storage バックエンドで使用する設定をすでに策定済みであること。これで後は、予定されている設定を director でオーケストレーションするだけとなります。

2. プロセスの説明

Block Storage サービスの設定は **`/etc/cinder/cinder.conf`** に保存されます。これらの設定には、**バックエンドの定義** が含まれます。Block Storage サービスで利用可能な (または、サポートされている) サードパーティー製のバックエンドの多くで提供されている設定手順では、**`/etc/cinder/cinder.conf`** の設定を編集する必要があります。「はじめに」の項に記載したように、この設定を行うことで、Block Storage サービスが設定されますが、これらの設定は、それ以降のオーバークラウドの更新で上書きされてしまいます。

いずれにしても、**`/etc/cinder/cinder.conf`** での手動設定に関する説明は、オーバークラウドのデプロイメントに役立ちます。director は結局のところ、**heat** を使用して、**`/etc/cinder/cinder.conf`** に同じ設定を適用します。そのため、バックエンドの設定をプランニングする際には、以下の作業を行う必要があります。

- Block Storage バックエンドの必要な構成の綿密な計画を立てます。
- この構成に基づいて、**`/etc/cinder/cinder.conf`** ファイルの内容を策定します。

`/etc/cinder/cinder.conf` ファイルの内容の策定が完了したら、バックエンドの設定をオーケストレーションするための環境ファイルを作成します。「環境ファイル」では、サンプルファイル **`/home/stack/templates/custom-env.yaml`** を使用して、このステップを詳しく説明しています。環境ファイルを用意しておくと、今後オーバークラウドを更新の際に設定を永続させるのに役立ちます。

3. 環境ファイルの作成

環境ファイルには、定義する各バックエンドの設定が記載されます。また、カスタムのバックエンドのデプロイメントに関連したその他の設定も含まれます。環境ファイルに関する詳しい情報は、『Advanced Overcloud Customization』の「Environment Files」のセクションを参照してください。

以下のサンプルの環境ファイルは、2 つの NetApp バックエンド (**netapp1** および **netapp2**) を定義します。

/home/stack/templates/custom-env.yaml

```
parameters: # (1)
  CinderEnableIscsiBackend: false
  CinderEnableRbdBackend: false
```

```
  CinderEnableNfsBackend: false
  NovaEnableRbdBackend: false
  GlanceBackend: file # 2

parameter_defaults:
  ControllerExtraConfig: # 3
    cinder::config::cinder_config:
        netapp1/volume_driver: # 4
            value: cinder.volume.drivers.netapp.common.NetAppDriver
        netapp1/netapp_storage_family:
            value: ontap_7mode
        netapp1/netapp_storage_protocol:
            value: iscsi
        netapp1/netapp_server_hostname:
            value: 10.35.64.11
        netapp1/netapp_server_port:
            value: 80
        netapp1/netapp_login:
            value: root
        netapp1/netapp_password:
            value: p@$$w0rd
        netapp1/volume_backend_name:
            value: netapp1
        netapp2/volume_driver: # 5
            value: cinder.volume.drivers.netapp.common.NetAppDriver # 6
        netapp2/netapp_storage_family:
            value: ontap_7mode
        netapp2/netapp_storage_protocol:
            value: iscsi
        netapp2/netapp_server_hostname:
            value: 10.35.64.11
        netapp2/netapp_server_port:
            value: 80
        netapp2/netapp_login:
            value: root
        netapp2/netapp_password:
            value: p@$$w0rd
        netapp2/volume_backend_name:
            value: netapp2
    cinder_user_enabled_backends: ['netapp1','netapp2'] # 7
```

1

以下のパラメーターは、**false** に設定されているので、必要のない他のバックエンド種別は無効になります。

- **CinderEnableIscsiBackend**: その他の iSCSI バックエンド
- **CinderEnableRbdBackend**: Red Hat Ceph
- **CinderEnableNfsBackend**: NFS
- **NovaEnableRbdBackend**: Red Hat Ceph の一時ストレージ

2

GlanceBackend パラメーターは、Image サービスがイメージの保管に使用すべきバックエンドを設定します。以下の値がサポートされています。

- **`file`**: 各コントローラーノード上の **`/var/lib/glance/images`** にイメージを保管します。
- **`swift`**: イメージの保管に Object Storage サービスを使用します。
- **`cinder`**: イメージの保存に Block Storage サービスを使用します。

3

`ControllerExtraConfig` は、全コントローラーノードに適用されるカスタムの設定を定義します。**`cinder::config::cinder_config`** クラスは、この設定が Block Storage (**`cinder`**) サービスに適用される必要のあることを意味します。このため、バックエンドの設定は最終的に、各コントローラーノードの **`/etc/cinder/cinder.conf`** ファイルに指定されることになります。

4

`netapp1/volume_driver` および **`netapp2/volume_driver`** の設定は、**section/setting** の構文に従います。Block Storage サービスでは、各バックエンドは、**`/etc/cinder/cinder.conf`** 内の独自のセクションで定義されます。**`netapp1`** で始まる行の各設定は、新しい **`[netapp1]`** バックエンドのセクションで定義されます。

5

同様に、**`netapp2`** の設定は **`[netapp2]`** という別のセクションで定義されます。

6

`value` で始まる行は、直上の行の設定値を設定します。

7

`cinder_user_enabled_backends` クラスは、カスタムのバックエンドを設定して有効にします。名前が示すように、このクラスは、ユーザーが有効化したバックエンドにのみ使用する必要があります。対象となるバックエンドは、具体的には、**`cinder::config::cinder_config`** クラスで定義されています。

director を使用してネイティブで有効化できるバックエンドをリストする際に、**`cinder_user_enabled_backends`** は使用しないでください。これには、Red Hat Ceph、NFS、およびサポート対象の NetApp または Dell アプライアンスの単一バックエンドが含まれます。たとえば、Red Hat Ceph バックエンドを有効にする場合には、**`cinder_user_enabled_backends`** にはリストせずに、**`CinderEnableRbdBackend: true`** を使用して有効化します。

注記

Red Hat Ceph back end for OpenStack Block Storage の定義に関する詳しい情報は、『オーバークラウド向けの Red Hat Ceph Storage』を参照してください。

「設定したバックエンドのデプロイ」は、環境ファイル /home/stack/templates/custom-env.yaml を使用して、カスタムのバックエンドのデプロイメントをオーケストレーションする方法を説明します。**/home/stack/templates/custom-env.yaml** から最終的に作成される **/etc/cinder/cinder.conf** の設定を確認するには、「サンプルの環境ファイルに基づいた設定」を参照してください。

4. 設定したバックエンドのデプロイ

/home/stack/templates/ に custom-env.yaml ファイルを作成したら、**stack** ユーザーとしてログインし、以下のコマンドを実行してカスタムのバックエンド設定をデプロイします。

```
$ openstack overcloud deploy --templates -e /home/stack/templates/custom-
env.yaml
```

重要

オーバークラウドの作成時に追加の環境ファイルを渡した場合には、予定外の変更がオーバークラウドに加えられないように、ここで **-e** オプションを使用して環境ファイルを再度渡します。詳しい情報は、『director のインストールと使用方法』の「オーバークラウド環境の変更」を参照してください。

director がオーケストレーションを完了したら、バックエンドをテストします。手順は、「設定したバックエンドのテスト」を参照してください。

5. 設定したバックエンドのテスト

バックエンドをオーバークラウドにデプロイした後には、そのバックエンド上でボリュームを正常に作成できるかどうかを検証します。このテストには、必要な環境変数が最初に読み込まれている必要があります。これらの変数は、デフォルトでは **/home/stack/overcloudrc** で定義されています。

これらの変数を読み込むには、**stack** ユーザーとして以下のコマンドを実行します。

```
$ source /home/stack/overcloudrc
```

注記

詳しい情報は、「オーバークラウドへのアクセス」を参照してください。

次に、各バックエンド用の **ボリューム種別** を作成します。オーバークラウドのコントローラーノードに **stack** ユーザーとしてログインし、以下のコマンドを実行します。

```
$ cinder type-create backend1
$ cinder type-create backend2
```

上記のコマンドにより、xref:envfile の **cinder::config::cinder_config** クラスで定義されている各バックエンドに対して 1 つずつ、**backend1** および **backend2** というボリューム種別が作成されます。

最後に、xref:envfile の **cinder_user_enabled_backends** クラスで有効化されているバックエンドの **volume_backend_name** に、各ボリューム種別をマッピングします。以下のコマンドを実行すると、ボリューム種別 **backend1** が **netapp1** に、**backend2** が **netapp2** にマッピングされます。

```
$ cinder type-key backend1 set volume_backend_name=netapp1
$ cinder type-key backend2 set volume_backend_name=netapp2
```

この時点で、各バックエンドをテストできるはずです。テストを開始するには、**netapp1** バックエンドに、ボリューム種別 **backend1** を指定して **netapp_volume_1** という名前の 1 GB のボリュームを作成します。

```
$ cinder create --volume-type backend1 --display_name netappvolume_1 1
```

同じように、**netapp2** バックエンドに、ボリューム種別 **backend2** を指定して、同様のボリュームを作成します。

```
$ cinder create --volume-type backend2 --display_name netappvolume_2 1
```

A. 付録

A.1. stack ユーザー

「director のインストールユーザーの作成」の項は、オーバークラウドのデプロイに使用する **stack** という名前のユーザーの作成方法を記載しています。**stack** アカウントは、バックエンドのデプロイ (「設定したバックエンドのデプロイ」) や、オーバークラウドにアクセスするために必要な環境変数の読み込み (「設定したバックエンドのテスト」) など、管理者特権が必要なコマンドを実行するのに使用することができます。

『director のインストールと使用方法』の手順に従って操作を行った場合には、**stack** ユーザーはすでに存在しているはずなので、必要に応じて使用すると便利です。

A.2. サンプルの環境ファイルに基づいた設定

xref:envfile 内の環境は、Block Storage サービスが 2 つの NetApp バックエンドを使用するように設定します。以下のスニペットには、必要な設定が示されています。

```
enabled_backends = netapp1,netapp2

[netapp1]
volume_backend_name=netapp_1
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_login=root
netapp_storage_protocol=iscsi
netapp_password=p@$$w0rd
netapp_storage_family=ontap_7mode
netapp_server_port=80
netapp_server_hostname=10.35.64.11

[netapp2]
volume_backend_name=netapp_2
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_login=root
netapp_storage_protocol=iscsi
netapp_password=p@$$w0rd
netapp_storage_family=ontap_7mode
netapp_server_port=80
netapp_server_hostname=10.35.64.11
```