



Red Hat OpenStack Platform 10

オーバークラウド向けの Red Hat Ceph Storage

オーバークラウドで Red Hat Ceph Storage を使用するための設定

OpenStack Team

Red Hat OpenStack Platform 10 オーバークラウド向けの Red Hat Ceph Storage

オーバークラウドで Red Hat Ceph Storage を使用するための設定

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、Red Hat Ceph Storage の環境の推奨事項や Ceph Storage ノードでオーバークラウドを実装する方法など、Red Hat OpenStack Platform director を使用して、Red Hat Ceph Storage を使用するオーバークラウドを作成する方法を説明します。

目次

第1章 はじめに	3
1.1. CEPH STORAGE の定義	3
1.2. RED HAT OPENSTACK PLATFORM での RED HAT CEPH STORAGE の使用	3
1.3. 設定要件	4
1.4. シナリオの定義	5
第2章 CEPH STORAGE ノードでのオーバークラウドの作成	6
2.1. STACK ユーザーの初期化	7
2.2. ノードの登録	7
2.3. ノードのハードウェアの検査	9
2.4. ノードの手動でのタグ付け	10
2.5. CEPH STORAGE ノードのルートディスクの定義	11
2.6. オーバークラウドでの CEPH STORAGE の有効化	13
2.7. CEPH STORAGE ノードディスクのレイアウトのマッピング	14
2.8. BACKUP サービスが CEPH にバックアップを保存するように設定する手順	15
2.9. CEPH STORAGE ノードディスクの GPT へのフォーマット	15
2.10. CEPH STORAGE CLUSTER のカスタマイズ	17
2.11. オーバークラウドの作成	18
2.12. オーバークラウドへのアクセス	19
2.13. CEPH STORAGE ノードの監視	19
2.14. 環境のリブート	20
2.15. CEPH STORAGE ノードの置き換え	20
第3章 既存の CEPH STORAGE CLUSTER のオーバークラウドとの統合	23
3.1. 既存の CEPH STORAGE CLUSTER の設定	23
3.2. STACK ユーザーの初期化	25
3.3. ノードの登録	25
3.4. ノードのハードウェアの検査	27
3.5. ノードの手動でのタグ付け	27
3.6. 既存の CEPH STORAGE CLUSTER との統合の有効化	28
3.7. 以前の RED HAT CEPH STORAGE バージョンとの後方互換性	29
3.8. オーバークラウドの作成	29
3.9. オーバークラウドへのアクセス	30
第4章 まとめ	31
付録A 環境ファイルのサンプル: CEPH クラスターの作成	32

第1章 はじめに

Red Hat OpenStack Platform director は、**オーバークラウド**と呼ばれるクラウド環境を作成します。director を使用して、オーバークラウドの追加機能を設定することができます。これらの追加機能の 1 つに、director で作成した Ceph Storage Cluster や、既存の Ceph Storage Cluster などの Red Hat Ceph Storage との統合が含まれます。本ガイドには、director を使用してオーバークラウドに Ceph Storage を統合する方法についての説明と設定例を記載します。

1.1. CEPH STORAGE の定義

Red Hat Ceph Storage は、優れたパフォーマンス、信頼性、スケーリングを提供するように設計された、分散型のデータオブジェクトストレージです。分散型のオブジェクトストアは、非構造化データに対応し、クライアントは新しいタイプのオブジェクトインターフェースと従来のインターフェースを同時に使用できるため、今後のストレージのあり方です。Ceph デプロイメントはすべて、2 種類のデーモンで構成される **Ceph Storage Cluster** を中心とします。

Ceph OSD (Object Storage Daemon)

Ceph OSD は、Ceph クライアントの代わりにデータを格納します。また、Ceph ノードの CPU とメモリーを使用して、データの複製、リバランス、復旧、監視、レポート作成を実行します。

Ceph モニター

Ceph モニターは、ストレージクラスターの現在のステータスを含む Ceph ストレージクラスターのマッピングのマスターコピーを管理します。

Red Hat Ceph Storage に関する詳しい情報は、『[Red Hat Ceph Storage Architecture Guide](#)』を参照してください。



重要

本ガイドでは、Ceph Block Storage のみを統合します。Ceph Object (RGW) または Ceph File (CephFS) ストレージの統合については記載していません。

1.2. RED HAT OPENSTACK PLATFORM での RED HAT CEPH STORAGE の使用

Red Hat OpenStack Platform director で Red Hat Ceph Storage をオーバークラウドに統合するには、主に 2 つの方法があります。

Ceph Storage Cluster でのオーバークラウドの作成

director には、オーバークラウドの作成中に Ceph Storage Cluster を作成する機能があります。director は、データの格納に Ceph OSD を使用する Ceph Storage ノードセットを作成します。さらに、director は、オーバークラウドのコントローラーノードに Ceph Monitor サービスをインストールします。このため、組織が高可用性のコントローラーノード 3 台で構成されるオーバークラウドを作成する場合には、Ceph Monitor も高可用性サービスになります。

既存の Ceph Storage のオーバークラウドへの統合

既存の Ceph Storage Cluster がある場合には、オーバークラウドのデプロイメント時に統合できます。これは、オーバークラウドの設定以外のクラスターの管理やスケーリングが可能であることを意味します。

1.3. 設定要件

本ガイドは、『[director のインストールと使用方法](#)』ガイドの補足情報としての役割を果たします。これは、「要件」で指定されているのと同じ要件が本ガイドにも適用されることを意味します。必要に応じて、この要件を実装してください。

Red Hat OpenStack Platform director を使用して Ceph Storage ノードを作成する場合は、それらのノードに対する以下の要件に注意してください。

プロセッサ

Intel 64 または AMD64 CPU 拡張機能のサポートがある 64 ビットの x86 プロセッサ

メモリー

メモリー要件はストレージ容量によって異なります。ハードディスク容量 1 TB あたり最小で 1 GB のメモリーを使用するのが理想的です。

ディスク領域

ストレージ要件はメモリーの容量によって異なります。ハードディスク容量 1 TB あたり最小で 1 GB のメモリーを使用するのが理想的です。

ディスクのレイアウト

推奨される Red Hat Ceph Storage ノードの設定には、以下のようなディスクレイアウトが必要です。

- ✳ **/dev/sda**: ルートディスク。director は、主なオーバークラウドイメージをディスクにコピーします。
- ✳ **/dev/sdb**: ジャーナルディスク。このディスクは、/dev/sdb1、/dev/sdb2、/dev/sdb3 などのように、Ceph OSD ジャーナル向けにパーティションを分割します。ジャーナルディスクは通常、システムパフォーマンス向上に役立つ Solid State Drive (SSD) です。
- ✳ **/dev/sdc** 以降: OSD ディスク。ストレージ要件で必要な数のディスクを使用します。

重要

オーバークラウドのデプロイ前に、ジャーナルおよび OSD の対象となるディスクで既存のパーティションをすべて消去します。さらに、Ceph Storage OSD およびジャーナルディスクは、デプロイメントの一部として設定可能な GPT ディスクラベルが必要です。詳しい情報は「[Ceph Storage ノードディスクの GPT へのフォーマット](#)」を参照してください。

ネットワークインターフェースカード

最小で 1 x 1 Gbps ネットワークインターフェースカード (実稼動環境では、最低でも NIC を 2 つ以上使用することを推奨します)。ボンディングインターフェース向けの場合や、タグ付けされた VLAN トラフィックを委譲する場合には、追加のネットワークインター

フェースを使用します。特に大量のトラフィックにサービスを提供する OpenStack Platform 環境を構築する場合には、ストレージノードには 10 Gbps インターフェースを使用することを推奨します。

Intelligent Platform Management Interface (IPMI)

各 Ceph ノードには、サーバーのマザーボード上に IPMI 機能が必要です。

本ガイドでは、以下の要件も満たす必要があります。

- ※ Red Hat OpenStack Platform director でインストールしたアンダークラウドホスト。[「アンダークラウドのインストール」](#)を参照してください。
- ※ Red Hat Ceph Storage のハードウェアの追加の推奨事項。これらの推奨事項については『[Red Hat Ceph Storage Hardware Guide](#)』を参照してください。



重要

Ceph Monitor サービスは、オーバークラウドのコントローラーノードにインストールされます。これは、パフォーマンスの問題を軽減するために、適切なリソースを提供する必要があるという意味です。お使いの環境のコントローラーノードでは、最低でもメモリー 16 GB を、Ceph Monitor データにはソリッドステートドライブ (SSD) を使用するようになっています。

1.4. シナリオの定義

本ガイドでは、2つのシナリオを使用します。

- ※ 最初のシナリオでは、Ceph Storage Cluster でオーバークラウドを作成します。これは、director が Ceph Storage Cluster をデプロイすることを意味します。
- ※ 2番目のシナリオでは、既存の Ceph Storage Cluster とオーバークラウドを統合します。これは、オーバークラウドの管理と Ceph Storage Cluster の管理を分けることを意味します。

第2章 CEPH STORAGE ノードでのオーバークラウドの作成

本章では、director を使用して、独自の Ceph Storage Cluster が含まれるオーバークラウドを作成する方法を説明します。オーバークラウドの作成方法や、既存の Ceph Storage Cluster との統合方法に関する説明は、「[3章 既存の Ceph Storage Cluster のオーバークラウドとの統合](#)」を参照してください。

本章のシナリオでは、オーバークラウドは 9 台のノードで構成されます。

- ✳ 高可用性のコントローラーノード 3 台。各ノードに Ceph Monitor サービスが含まれます。
- ✳ クラスター内に Red Hat Ceph Storage ノード 3 台。これらのノードには、Ceph OSD が含まれ、実際のストレージとしての役割を果たします。
- ✳ コンピュートノード 3 台

このシナリオのすべてのマシンは、電源管理に IPMI を使用したベアメタルマシンです。director より Red Hat Enterprise Linux 7 のイメージが各ノードにコピーされるため、これらのノードではオペレーティングシステムは必要ありません。

director は、イントロスペクションおよびプロビジョニングプロセス中に、プロビジョニングネットワークを使用して各ノードと通信します。すべてのノードは、ネイティブの VLAN 経由でネットワークに接続します。この例では、以下の IP アドレスの割り当てで、プロビジョニングサブネットとして 192.0.2.0/24 を使用します。

ノード名	IP アドレス	MAC アドレス	IPMI IP アドレス
director	192.0.2.1	aa:aa:aa:aa:aa:aa	
コントローラー 1	定義済みの DHCP	b1:b1:b1:b1:b1:b1	192.0.2.205
コントローラー 2	定義済みの DHCP	b2:b2:b2:b2:b2:b2	192.0.2.206
コントローラー 3	定義済みの DHCP	b3:b3:b3:b3:b3:b3	192.0.2.207
コンピュート 1	定義済みの DHCP	c1:c1:c1:c1:c1:c1	192.0.2.208
コンピュート 2	定義済みの DHCP	c2:c2:c2:c2:c2:c2	192.0.2.209
コンピュート 3	定義済みの DHCP	c3:c3:c3:c3:c3:c3	192.0.2.210
Ceph 1	定義済みの DHCP	d1:d1:d1:d1:d1:d1	192.0.2.211

ノード名	IP アドレス	MAC アドレス	IPMI IP アドレス
Ceph 2	定義済みの DHCP	d2:d2:d2:d2:d2:d2	192.0.2.212
Ceph 3	定義済みの DHCP	d3:d3:d3:d3:d3:d3	192.0.2.213

2.1. STACK ユーザーの初期化

stack ユーザーとして director ホストにログインし、以下のコマンドを実行して director の設定を初期化します。

```
$ source ~/stackrc
```

このコマンドでは、director の CLI ツールにアクセスする認証情報が含まれる環境変数を設定します。

2.2. ノードの登録

ノード定義のテンプレート (**instackenv.json**) は JSON ファイル形式で、ノード登録用のハードウェアおよび電源管理の情報が含まれています。以下に例を示します。

```
{
  "nodes": [
    {
      "mac": [
        "b1:b1:b1:b1:b1:b1"
      ],
      "cpu": "4",
      "memory": "6144",
      "disk": "40",
      "arch": "x86_64",
      "pm_type": "pxe_ipmitool",
      "pm_user": "admin",
      "pm_password": "p@55w0rd!",
      "pm_addr": "192.0.2.205"
    },
    {
      "mac": [
        "b2:b2:b2:b2:b2:b2"
      ],
      "cpu": "4",
      "memory": "6144",
      "disk": "40",
      "arch": "x86_64",
      "pm_type": "pxe_ipmitool",
      "pm_user": "admin",
      "pm_password": "p@55w0rd!",
      "pm_addr": "192.0.2.206"
    }
  ]
}
```

```

{
    "mac": [
        "b3:b3:b3:b3:b3:b3"
    ],
    "cpu": "4",
    "memory": "6144",
    "disk": "40",
    "arch": "x86_64",
    "pm_type": "pxe_ipmitool",
    "pm_user": "admin",
    "pm_password": "p@55w0rd!",
    "pm_addr": "192.0.2.207"
},
{
    "mac": [
        "c1:c1:c1:c1:c1:c1"
    ],
    "cpu": "4",
    "memory": "6144",
    "disk": "40",
    "arch": "x86_64",
    "pm_type": "pxe_ipmitool",
    "pm_user": "admin",
    "pm_password": "p@55w0rd!",
    "pm_addr": "192.0.2.208"
},
{
    "mac": [
        "c2:c2:c2:c2:c2:c2"
    ],
    "cpu": "4",
    "memory": "6144",
    "disk": "40",
    "arch": "x86_64",
    "pm_type": "pxe_ipmitool",
    "pm_user": "admin",
    "pm_password": "p@55w0rd!",
    "pm_addr": "192.0.2.209"
},
{
    "mac": [
        "c3:c3:c3:c3:c3:c3"
    ],
    "cpu": "4",
    "memory": "6144",
    "disk": "40",
    "arch": "x86_64",
    "pm_type": "pxe_ipmitool",
    "pm_user": "admin",
    "pm_password": "p@55w0rd!",
    "pm_addr": "192.0.2.210"
},
{
    "mac": [
        "d1:d1:d1:d1:d1:d1"
    ],

```

```

        "cpu": "4",
        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",
        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.211"
    },
    {
        "mac": [
            "d2:d2:d2:d2:d2:d2"
        ],
        "cpu": "4",
        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",
        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.212"
    },
    {
        "mac": [
            "d3:d3:d3:d3:d3:d3"
        ],
        "cpu": "4",
        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",
        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.213"
    }
]
}

```

テンプレートの作成後に、stack ユーザーのホームディレクトリー (`/home/stack/instackenv.json`) にファイルを保存してから、director にインポートします。これには、以下のコマンドを実行します。

```
$ openstack baremetal import --json ~/instackenv.json
```

このコマンドでテンプレートをインポートして、テンプレートから director に各ノードを登録します。

カーネルと ramdisk イメージを全ノードに割り当てます。

```
$ openstack baremetal configure boot
```

director でのノードの登録、設定が完了しました。

2.3. ノードのハードウェアの検査

ノードの登録後には、各ノードのハードウェア属性を検査します。各ノードのハードウェア属性を検査するには、以下のコマンドを実行します。

```
$ openstack baremetal introspection bulk start
```

重要

このプロセスが最後まで実行されて正常に終了したことを確認してください。ベアメタルの場合には、通常 15 分ほどかかります。

2.4. ノードの手動でのタグ付け

各ノードのハードウェアを登録、検査した後は、特定のプロファイルにノードをタグ付けします。これらのプロファイルタグによりノードとフレーバーが照合され、フレーバーがデプロイメントロールに割り当てられます。

ノード一覧を取得して UUID を識別します。

```
$ ironic node-list
```

特定のプロファイルにノードを手動でタグ付けする場合には、各ノードの **properties/capabilities** パラメーターに **profile** オプションを追加します。たとえば、2 台のノードをタグ付けしてコントローラープロファイルとコンピュータープロファイルをそれぞれ使用するには、以下のコマンドを実行します。

```
$ ironic node-update 1a4e30da-b6dc-499d-ba87-0bd8a3819bc0 add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 6faba1a9-e2d8-4b7c-95a2-c7fbd12129a add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 5e3b2f50-fcd9-4404-b0a2-59d79924b38e add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 484587b2-b3b3-40d5-925b-a26a2fa3036f add
properties/capabilities='profile:compute,boot_option:local'
$ ironic node-update d010460b-38f2-4800-9cc4-d69f0d067efe add
properties/capabilities='profile:compute,boot_option:local'
$ ironic node-update d930e613-3e14-44b9-8240-4f3559801ea6 add
properties/capabilities='profile:compute,boot_option:local'
$ ironic node-update da0cc61b-4882-45e0-9f43-fab65cf4e52b add
properties/capabilities='profile:ceph-storage,boot_option:local'
$ ironic node-update b9f70722-e124-4650-a9b1-aade8121b5ed add
properties/capabilities='profile:ceph-storage,boot_option:local'
$ ironic node-update 68bf8f29-7731-4148-ba16-efb31ab8d34f add
properties/capabilities='profile:ceph-storage,boot_option:local'
```

各ノードのハードウェアを登録、検査した後は、特定のプロファイルにノードをタグ付けします。これらのプロファイルタグによりノードとフレーバーが照合され、フレーバーがデプロイメントロールに割り当てられます。

ノード一覧を取得して UUID を識別します。

```
$ ironic node-list
```

特定のプロファイルにノードを手動でタグ付けする場合には、各ノードの **properties/capabilities** パラメーターに **profile** オプションを追加します。たとえば、2 台のノードをタグ付けしてコントローラープロファイルとコンピュートプロファイルをそれぞれ使用するには、以下のコマンドを実行します。

2.5. CEPH STORAGE ノードのルートディスクの定義

Ceph Storage ノードの大半では、複数のディスクを使用します。つまり、Ceph Storage ノードをプロビジョニングする際に、**director** はルートディスクに使用するディスクを特定する必要があるという意味です。

- ✧ **model** (文字列): デバイスの ID
- ✧ **vendor** (文字列): デバイスのベンダー
- ✧ **serial** (文字列): ディスクのシリアル番号
- ✧ **wwn** (文字列): 一意のストレージ ID
- ✧ **size** (整数): デバイスのサイズ (GB)

以下の例では、ルートデバイスを特定するディスクのシリアル番号を使用して、オーバークラウドイメージをデプロイするドライブを指定します。

最初に、**director** がイントロスペクションで取得した各ノードのハードウェア情報のコピーを収集します。この情報は、OpenStack Object Storage (swift) サーバーに保管されています。この情報を新規ディレクトリーにダウンロードします。

```
$ mkdir swift-data
$ cd swift-data
$ export SWIFT_PASSWORD=`sudo crudini --get /etc/ironic-inspector/inspector.conf swift password`
$ for node in $(ironic node-list | grep -v UUID | awk '{print $2}'); do
  swift -U service:ironic -K $SWIFT_PASSWORD download ironic-inspector
  inspector_data-$node; done
```



注記

この例では **crudini** パッケージで入手可能な **crudini** コマンドを使用します。

この操作により、イントロスペクションで取得した各 **inspector_data** オブジェクトからデータがダウンロードされます。全オブジェクトのオブジェクト名の一部には、ノードの UUID が使用されます。

```
$ ls -l
inspector_data-15fc0edc-eb8d-4c7f-8dc0-a2a25d5e09e3
inspector_data-46b90a4d-769b-4b26-bb93-50eaefcdb3f4
inspector_data-662376ed-faa8-409c-b8ef-212f9754c9c7
inspector_data-6fc70fe4-92ea-457b-9713-ee499eda206
inspector_data-9238a73a-ec8b-4976-9409-3fcff9a8dca3
inspector_data-9cbfe693-8d55-47c2-a9d5-10e059a14e07
inspector_data-ad31b32d-e607-4495-815c-2b55ee04cdb1
inspector_data-d376f613-bc3e-4c4b-ad21-847c4ec850f8
```

各ノードのディスク情報をチェックします。以下のコマンドを実行すると、各ノードの ID とディスク情報が表示されます。

```
$ for node in $(ironic node-list | grep -v UUID | awk '{print $2}'); do
echo "NODE: $node" ; cat inspector_data-$node | jq '.inventory.disks' ;
echo "-----" ; done
```

たとえば、1つのノードのデータで3つのディスクが表示される場合があります。

```
NODE: 15fc0edc-eb8d-4c7f-8dc0-a2a25d5e09e3
[
  {
    "size": 299439751168,
    "rotational": true,
    "vendor": "DELL",
    "name": "/dev/sda",
    "wwn_vendor_extension": "0x1ea4dcc412a9632b",
    "wwn_with_extension": "0x61866da04f3807001ea4dcc412a9632b",
    "model": "PERC H330 Mini",
    "wwn": "0x61866da04f380700",
    "serial": "61866da04f3807001ea4dcc412a9632b"
  }
  {
    "size": 299439751168,
    "rotational": true,
    "vendor": "DELL",
    "name": "/dev/sdb",
    "wwn_vendor_extension": "0x1ea4e13c12e36ad6",
    "wwn_with_extension": "0x61866da04f380d001ea4e13c12e36ad6",
    "model": "PERC H330 Mini",
    "wwn": "0x61866da04f380d00",
    "serial": "61866da04f380d001ea4e13c12e36ad6"
  }
  {
    "size": 299439751168,
    "rotational": true,
    "vendor": "DELL",
    "name": "/dev/sdc",
    "wwn_vendor_extension": "0x1ea4e31e121cfb45",
    "wwn_with_extension": "0x61866da04f37fc001ea4e31e121cfb45",
    "model": "PERC H330 Mini",
    "wwn": "0x61866da04f37fc00",
    "serial": "61866da04f37fc001ea4e31e121cfb45"
  }
]
-----
```

以下の例では、ルートデバイスを、シリアル番号 **61866da04f37fc001ea4e31e121cfb45** の disk 2 に設定します。そのためには、ノードの定義に **root_device** パラメーターを追加する必要があります。

```
$ ironic node-update 15fc0edc-eb8d-4c7f-8dc0-a2a25d5e09e3 add
properties/root_device='{ "serial": "61866da04f37fc001ea4e31e121cfb45" }'
```


これにより、director がルートディスクとして使用する特定のディスクを識別しやすくなります。オーバークラウドの作成の開始時には、director はこのノードをプロビジョニングして、オーバークラウドのイメージをこのディスクに書き込みます。その他のディスクは、Ceph Storage ノードのマッピングに使用されます。



重要

name でルートディスクを設定しないでください。この値は、ノードのブート時に変更される可能性があります。

2.6. オーバークラウドでの CEPH STORAGE の有効化

オーバークラウドのイメージにはすでに、Ceph サービスと必要な Puppet モジュールが含まれており、自動的に Ceph OSD ノードと Ceph Monitor をコントローラークラスター上に設定します。オーバークラウドの Heat テンプレートコレクションには、Ceph Storage 設定を有効化するのに必要な手順も含まれていますが、director では Ceph Storage を有効化して、対象の設定に指定するための情報が必要です。この情報を指定するには、**storage-environment.yaml** の環境ファイルを stack ユーザーの templates ディレクトリーにコピーします。

```
$ cp /usr/share/openstack-tripleo-heat-templates/environments/storage-environment.yaml ~/templates/.
```

storage-environment.yaml のコピーで以下のオプションを変更します。

CinderEnableIscsiBackend

iSCSI バックエンドを有効にするパラメーター。 **false** に設定します。

CinderEnableRbdBackend

Ceph Storage バックエンドを有効にするパラメーター。 **true** に設定します。

CinderEnableNfsBackend

NFS バックエンドを有効にするパラメーター。 **false** に設定します。

NovaEnableRbdBackend

Nova エフェメラルストレージ用に Ceph Storage を有効にするパラメーター。 **true** に設定します。

GlanceBackend

Glance で使用するバックエンドを定義します。イメージに Ceph Storage を使用するには、**rbd** に設定します。

次に **resource_registry** の各エントリーを各リソースの絶対パスを参照するように変更します。

```
resource_registry:
  OS::TripleO::Services::CephMon: /usr/share/openstack-tripleo-heat-templates/puppet/services/ceph-mon.yaml
  OS::TripleO::Services::CephOSD: /usr/share/openstack-tripleo-heat-
```

```
templates/puppet/services/ceph-osd.yaml
OS::TripleO::Services::CephClient: /usr/share/openstack-tripleo-heat-
templates/puppet/services/ceph-client.yaml
```

注記

storage-environment.yaml には、Heat を直接使用して Ceph Storage を設定するためのオプションも含まれています。ただし、director はこれらのノードを作成して、このシナリオでは、自動的に設定値を定義するので、これらのオプションは必要ありません。

2.7. CEPH STORAGE ノードディスクのレイアウトのマッピング

デフォルトマッピングは、Ceph Storage にルートディスクを使用しますが、実稼動環境の多くは、ストレージやジャーナリング用のパーティションに別個のディスクを複数使用します。このような状況では、以前にコピーした **storage-environment.yaml** ファイルの一部として、ストレージマップを定義します。

storage-environment.yaml ファイルと以下のスニペットを **parameter_defaults** に編集します。

```
ExtraConfig:
  ceph::profile::params::osds:
```

このセクションにより、オーバークラウドに Hiera データが追加されます。Puppet は、設定時にこのデータをカスタムのパラメーターとして使用します。**ceph::profile::params::osds** パラメーターを使用して、適切なディスクとジャーナルパーティションをマッピングします。たとえば、4 つのディスクがある Ceph ノードでは以下のように割り当てられます。

- ※ **/dev/sda**: オーバークラウドのイメージを含むルートディスク
- ※ **/dev/sdb**: ジャーナルのパーティションが含まれるディスク。これは通常、システムパフォーマンスをソリッドステートドライブ (SSD) です。
- ※ **/dev/sdc** および **/dev/sdd**: OSD のディスク

この例では、以下のような内容が含まれるマッピングとなります。

```
ceph::profile::params::osds:
  '/dev/sdc':
    journal: '/dev/sdb'
  '/dev/sdd':
    journal: '/dev/sdb'
```

ジャーナル用に別のディスクを使用しない場合には、OSD ディスクに併置されているジャーナルを使用します。journal パラメーターには空の値を渡します。

```
ceph::profile::params::osds:
  '/dev/sdb': {}
  '/dev/sdc': {}
  '/dev/sdd': {}
```

注記

director は、異なるタイプのディスクを混在させた Ceph ノードのデプロイメントもサポートするようになりました (例: 同じ物理ホスト上に SSD と SATA のディスクを混在させるなど)。通常の Ceph デプロイメントでは、これは「[Placing Different Pools on Different OSDs](#)」に記載のように、CRUSH マップで設定されます。このようなデプロイメントをマッピングする場合には、**storage-environment.yaml** の **ExtraConfig** セクションに以下の行を追加します。

```
ceph::osd_crush_update_on_start: false
```

次に、オーバークラウドのデプロイ時に Ceph Storage ノードにディスクマッピングが使用されるように **~/templates/storage-environment.yaml** ファイルを保存します。ストレージで必要な設定が開始されるように、デプロイメント内にこのファイルを追加します。

2.8. BACKUP サービスが CEPH にバックアップを保存するように設定する手順

Block Storage Backup サービス (**cinder-backup**) はデフォルトで無効になっています。環境ファイル (**~/templates/storage-environment.yaml**) の **resource_registry** に以下の行を追加して、このサービスを有効化することができます。

```
OS::TripleO::Services::CinderBackup: /usr/share/openstack-tripleo-heat-templates/puppet/services/pacemaker/cinder-backup.yaml
```

注記

このリソースは **/usr/share/openstack-tripleo-heat-templates/environments/cinder-backup.yaml** でも定義されていますが、直接呼び出すこともできます。本ガイドでは、**/home/stack/templates/storage-environment.yaml** で直接このリソースを定義して、リソースとパラメーターすべてが 1 つの環境ファイルに集約されるようにします。

次に **cinder-backup** サービスが Ceph にバックアップを保存するように設定します。これには、サービスで Ceph RBD ドライバーを使用するように設定する必要があります。そのためには、環境ファイルの **parameter_defaults** に以下の行を追加します。

```
CinderBackupBackend: ceph
```

これで、デフォルトのバックアップ先である Object Storage サービス (**swift**) よりも Ceph が優先されるようになります。

2.9. CEPH STORAGE ノードディスクの GPT へのフォーマット

Ceph Storage OSD およびジャーナルのパーティションには、GPT ディスクラベルが必要です。そのため、Ceph Storage にディスクを追加すると、Ceph OSD をインストールする前に GPT ラベルへ変換する必要があります。これには、ノードでスクリプトを実行して、初回起動時にこの操作が実行されるようにする必要があります。このスクリプトは、オーバークラウドの作成時に Heat

テンプレートの一部として追加します。たとえば、以下の Heat テンプレート (**wipe-disks.yaml**) は、Ceph Storage ノードの全ディスクをチェックして、全ノード (root ファイルシステムを含むディスク以外) を GPT に変換します。

```
heat_template_version: 2014-10-16

description: >
  Wipe and convert all disks to GPT (except the disk containing the
  root file system)

resources:
  userdata:
    type: OS::Heat::MultipartMime
    properties:
      parts:
        - config: {get_resource: wipe_disk}

  wipe_disk:
    type: OS::Heat::SoftwareConfig
    properties:
      config: {get_file: wipe-disk.sh}

outputs:
  OS::stack_id:
    value: {get_resource: userdata}
```

この Heat テンプレートは、**wipe-disk.sh** と呼ばれる Bash スクリプトを参照します。このスクリプトには、ルートディスク以外のディスクを消去する手順が含まれます。以下のスクリプトは、ルートディスク以外のディスクをすべて消去する **wipe-disk.sh** の例です。

```
#!/bin/bash
if [[ `hostname` = *"ceph"* ]]
then
  echo "Number of disks detected: $(lsblk -no NAME,TYPE,MOUNTPOINT |
grep "disk" | awk '{print $1}' | wc -l)"
  for DEVICE in `lsblk -no NAME,TYPE,MOUNTPOINT | grep "disk" | awk
'{print $1}'`
  do
    ROOTFOUND=0
    echo "Checking /dev/$DEVICE..."
    echo "Number of partitions on /dev/$DEVICE: $(expr $(lsblk -n
/dev/$DEVICE | awk '{print $7}' | wc -l) - 1)"
    for MOUNTS in `lsblk -n /dev/$DEVICE | awk '{print $7}'`
    do
      if [ "$MOUNTS" = "/" ]
      then
        ROOTFOUND=1
      fi
    done
    if [ $ROOTFOUND = 0 ]
    then
      echo "Root not found in /dev/${DEVICE}"
      echo "Wiping disk /dev/${DEVICE}"
      sgdisk -Z /dev/${DEVICE}
      sgdisk -g /dev/${DEVICE}
    else
```

```

        echo "Root found in /dev/${DEVICE}"
    fi
done
fi

```

環境内に Heat テンプレートを追加するには、**storage-environment.yaml** ファイルにテンプレートを **NodeUserData** リソースとして登録します。

```

resource_registry:
    OS::TripleO::NodeUserData: /home/stack/templates/firstboot/wipe-
    disks.yaml

```

2.10. CEPH STORAGE CLUSTER のカスタマイズ

ExtraConfig フックを使用して Ceph Storage ノードのデフォルト設定パラメーターを上書きして、Puppet 設定を渡すようにデータを定義できます。このデータを渡す方法は 2 種類存在します。

方法 1: Puppet のデフォルト設定の変更

オーバークラウドの設定時に **ceph** Puppet モジュールに渡すパラメーターをカスタマイズします。これらのパラメーターは、**/etc/puppet/modules/ceph/manifests/profile/params.conf** で定義される **ceph::profile::params** Puppet クラスに含まれます。たとえば、以下の環境ファイルのスニペットは、**ceph::profile::params** クラスからのデフォルトの **osd_journal_size** パラメーターをカスタマイズし、デフォルト設定を上書きします。

```

parameter_defaults:
    ExtraConfig:
        ceph::profile::params::osd_journal_size: 2048

```

この内容を環境ファイル (例: **ceph-settings.yaml**) に追加して、「[オーバークラウドの作成](#)」で **openstack overcloud deploy** コマンドを実行する際に追加します。以下に例を示します。

```

$ openstack overcloud deploy --templates --ceph-storage-scale <number
of nodes> -e /home/stack/templates/storage-environment.yaml -e
/home/stack/templates/ceph-settings.yaml

```

方法 2: 任意の設定デフォルト

方法 1 で、設定の必要な固有のパラメーターが追加されない場合には、**ceph::conf::args** Puppet クラスを使用して、任意の Ceph Storage パラメーターを渡すことができます。このクラスでは、**stanza/key** 形式とパラメーターの値を定義する **value** を使用してパラメーターの名前を受け入れます。これらの設定値で、各ノード上の **ceph.conf** ファイルが設定されます。たとえば、**ceph.conf** の **global** セクションにある **max_open_files** パラメーターを変更するには、環境ファイルで以下の構造を使用します。

```

parameter_defaults:
    ExtraConfig:
        ceph::conf::args:
            global/max_open_files:

```

```
value: 131072
```

この内容を環境ファイル (例: **ceph-settings.yaml**) に追加して、「[オーバークラウドの作成](#)」で **openstack overcloud deploy** コマンドを実行する際に追加します。以下に例を示します。

```
$ openstack overcloud deploy --templates --ceph-storage-scale <number of nodes> -e /home/stack/templates/storage-environment.yaml -e /home/stack/templates/ceph-settings.yaml
```

この **ceph.conf** ファイルには最終的に、以下のパラメーターが自動的に入力されるはずです。

```
[global]
max_open_files = 131072
```

2.11. オーバークラウドの作成

オーバークラウドの作成には、**openstack overcloud deploy** コマンドに追加の引数を指定する必要があります。以下に例を示します。

```
$ openstack overcloud deploy --templates -e /home/stack/templates/storage-environment.yaml --control-scale 3 --compute-scale 3 --ceph-storage-scale 3 --control-flavor control --compute-flavor compute --ceph-storage-flavor ceph-storage --neutron-network-type vxlan --ntp-server pool.ntp.org
```

上記のコマンドは、以下のオプションを使用します。

- ※ **--templates**: デフォルトの Heat テンプレートコレクションからオーバークラウドを作成します。
- ※ **-e /home/stack/templates/storage-environment.yaml**: 別の環境ファイルをオーバークラウドデプロイメントに追加します。この場合は、Ceph Storage の設定を含むストレージ環境ファイルです。
- ※ **--control-scale 3**: コントローラーノードを 3 台にスケーリングします。
- ※ **--compute-scale 3**: コンピュートノードを 3 台にスケーリングします。
- ※ **--ceph-storage-scale 3**: Ceph Storage ノードを 3 台にスケーリングします。
- ※ **--control-flavor control**: 対象のコントローラーノードに特定のフレーバーを使用します。
- ※ **--compute-flavor compute**: コンピュートノードに特定のフレーバーを使用します。
- ※ **--ceph-storage-flavor ceph-storage**: コンピュートノードに特定のフレーバーを使用します。
- ※ **--neutron-network-type vxlan**: ネットワーク種別を **neutron** に設定します。
- ※ **--ntp-server pool.ntp.org**: NTP サーバーを設定します。

/home/stack/templates/storage-environment.yaml で使用する全設定の概要については、「[付録A 環境ファイルのサンプル: Ceph クラスターの作成](#)」を参照してください。

注記

オプションの完全な一覧を表示するには、以下を実行します。

```
$ openstack help overcloud deploy
```

詳しい情報は、『[director のインストールと使用方法](#)』ガイドの「[オーバークラウドのパラメーター設定](#)」を参照してください。

オーバークラウドの作成プロセスが開始され、director によりノードがプロビジョニングされます。このプロセスは完了するまで多少時間がかかります。オーバークラウドの作成のステータスを確認するには、**stack** ユーザーとして別のターミナルを開き、以下を実行します。

```
$ source ~/stackrc
$ heat stack-list --show-nested
```

2.12. オーバークラウドへのアクセス

director は、director ホストがオーバークラウドと対話するための設定と認証を行うスクリプトを生成します。このファイル (**overcloudrc**) は、**stack** ユーザーのホームディレクトリーに保存されます。このファイルを使用するには、以下のコマンドを実行します。

```
$ source ~/overcloudrc
```

これにより、director ホストの CLI からオーバークラウドと対話するために必要な環境変数が読み込まれます。director ホストとの対話に戻るには、以下のコマンドを実行します。

```
$ source ~/stackrc
```

2.13. CEPH STORAGE ノードの監視

オーバークラウドの作成が完了したら、正しく機能していることを確認するため、Ceph Storage Cluster のステータスをチェックすることを推奨します。これには、director から **heat-admin** ユーザーとしてコントローラーノードにログインします。

```
$ nova list
$ ssh heat-admin@192.168.0.25
```

クラスターの正常性を確認します。

```
$ sudo ceph health
```

クラスターに問題がない場合は、上記のコマンドにより、**HEALTH_OK** というレポートが返されます。これは、クラスターを安全に使用できることを意味します。

Ceph モニタークォーラムのステータスを確認します。

```
$ sudo ceph quorum_status
```

これにより、クォーラムに参加するモニターとどれがリーダーであるかが表示されます。

全 Ceph OSD が実行中であるかどうかを確認します。

```
$ ceph osd stat
```

Ceph Storage Cluster の監視に関する詳しい情報は、『**Red Hat Ceph Storage Administration Guide**』の「[Monitoring](#)」を参照してください。

2.14. 環境のリブート

環境を再起動する必要がある状況が発生する場合があります。たとえば、物理サーバーを変更する必要がある場合や、停電から復旧する必要がある場合などです。このような状況では、Ceph Storage ノードが正しく起動されることが重要です。

以下の順序でノードを起動してください。

- ※ **Boot all Ceph Monitor nodes first** これにより、Ceph Monitor サービスが高可用性クラスター内でアクティブとなります。以前のリリースでは、Ceph Monitor サービスは常にコントローラーノード上にインストールされていました。
- ※ **Ceph Storage ノードすべてを起動します。**これにより、Ceph OSD クラスターはコントローラーノード上のアクティブな Ceph Monitor クラスターに接続できるようになります。

オーバークラウドすべてが同時に起動する状況が発生した場合には、Ceph OSD サービスが Ceph Storage ノード上で正しく起動されない場合があります。そのような場合には、Ceph Storage OSD を再起動して、Ceph Monitor サービスに接続できるようにします。各 Ceph Storage ノードで以下のコマンドを実行します。

```
$ sudo systemctl restart 'ceph*'
```

以下のコマンドを使用して、Ceph Storage ノードクラスターの **HEALTH_OK** のステータスを検証します。

```
$ sudo ceph status
```

2.15. CEPH STORAGE ノードの置き換え

Ceph Storage ノードに障害が発生する可能性があります。このような状況では、データが失われないように、問題のあるノードを無効化してリバランスしてから、オーバークラウドから削除するようにしてください。以下の手順では、Ceph Storage ノードを置き換えるプロセスについて説明します。



注記

以下の手順では、『**Red Hat Ceph Storage Administration Guide**』からの手順を使用して、手動で Ceph Storage ノードを削除します。Ceph Storage ノードの手動での削除に関する詳しい情報は、『**Red Hat Ceph Storage Administration Guide**』の「[Removing OSDs \(Manual\)](#)」を参照してください。

heat-admin ユーザーとして、コントローラーノードまたは Ceph Storage ノードにログインします。director の **stack** ユーザーには、**heat-admin** ユーザーにアクセスするための SSH キーがあります。

OSD ツリーを一覧表示して、ノードの OSD を検索します。たとえば、削除するノードには、以下の OSD が含まれる場合があります。

```
-2 0.09998 host overcloud-cephstorage-0 0 0.04999 osd.0 up 1.00000 1.00000 1 0.04999 osd.1 up
1.00000 1.00000
```

Ceph Storage ノードの OSD を無効化します。今回は、OSD ID は 0 と 1 です。

```
[heat-admin@overcloud-controller-0 ~]$ sudo ceph osd out 0
[heat-admin@overcloud-controller-0 ~]$ sudo ceph osd out 1
```

Ceph Storage Cluster がリバランスを開始します。このプロセスが完了するまで待機してください。以下のコマンドを使用して、ステータスを確認できます。

```
[heat-admin@overcloud-controller-0 ~]$ sudo ceph -w
```

Ceph クラスターのリバランスが完了したら、**heat-admin** ユーザーとして、問題のある Ceph Storage ノードにログインして、このノードを停止します。

```
[heat-admin@overcloud-cephstorage-0 ~]$ sudo /etc/init.d/ceph stop
osd.0
[heat-admin@overcloud-cephstorage-0 ~]$ sudo /etc/init.d/ceph stop
osd.1
```

これ以上データを受信しないように、CRUSH マップからこの Ceph Storage ノードを削除します。

```
[heat-admin@overcloud-cephstorage-0 ~]$ sudo ceph osd crush remove
osd.0
[heat-admin@overcloud-cephstorage-0 ~]$ sudo ceph osd crush remove
osd.1
```

OSD 認証キーを削除します。

```
[heat-admin@overcloud-cephstorage-0 ~]$ sudo ceph auth del osd.0
[heat-admin@overcloud-cephstorage-0 ~]$ sudo ceph auth del osd.1
```

クラスターから OSD を削除します。

```
[heat-admin@overcloud-cephstorage-0 ~]$ sudo ceph osd rm 0
[heat-admin@overcloud-cephstorage-0 ~]$ sudo ceph osd rm 1
```

ノードからログアウトして、**stack** ユーザーとして director ホストに戻ります。

```
[heat-admin@overcloud-cephstorage-0 ~]$ exit
[stack@director ~]$
```

director が再度プロビジョニングしないように、Ceph Storage ノードを無効にします。

```
[stack@director ~]$ ironic node-list
[stack@director ~]$ ironic node-set-maintenance [UUID] true
```

Ceph Storage ノードを削除するには、ローカルのテンプレートファイルを使用して **overcloud** スタックへの更新が必要です。最初に、オーバークラウドスタックの UUID を特定します。

```
$ heat stack-list
```

削除する Ceph Storage ノードの UUID を特定します。

```
$ nova list
```

以下のコマンドを実行してスタックからノードを削除し、それに応じてプランを更新します。

```
$ openstack overcloud node delete --stack [STACK_UUID] --templates -e  
[ENVIRONMENT_FILE] [NODE_UUID]
```

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** または **--environment-file** オプションを使用して環境ファイルを再度渡します。

stack が更新を完了するまで待機します。**heat stack-list --show-nested** コマンドを使用して、stack の更新を監視します。

director のノードプールに新しいノードを追加して、Ceph Storage ノードとしてデプロイします。**--ceph-storage-scale** オプションを使用して、オーバークラウド内の合計 Ceph Storage ノード数を定義します。たとえば、3 つのノードから構成されるクラスターから問題のあるノードを削除して置き換える場合は、**--ceph-storage-scale 3** を使用して、Ceph Storage ノードの数を元の値に戻します。

```
$ openstack overcloud deploy --templates --ceph-storage-scale 3 -e  
[ENVIRONMENT_FILES]
```

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** または **--environment-file** オプションを使用して環境ファイルを再度渡します。

director は、新しいノードをプロビジョニングして、新しいノードの詳細を用いて stack 全体を更新します。

heat-admin ユーザーとしてコントローラーノードにログインして、Ceph Storage ノードのステータスを確認します。以下に例を示します。

```
[heat-admin@overcloud-controller-0 ~]$ sudo ceph status
```

osdmap セクションの値が、クラスターに設定するノード数と一致していることを確認します。

エラーの発生した Ceph Storage ノードが新規ノードに置き換えられました。

第3章 既存の CEPH STORAGE CLUSTER のオーバークラウドとの統合

本章では、オーバークラウドを作成して、既存の Ceph Storage Cluster と統合する方法について説明します。オーバークラウドの作成方法と既存の Ceph Storage Cluster との統合方法に関する説明はいずれも、「[2章 Ceph Storage ノードでのオーバークラウドの作成](#)」を参照してください。

本章のシナリオでは、オーバークラウドは 6 台のノードで構成されます。

※ 高可用性のコントローラーノード 3 台

※ コンピュートノード 3 台

director は、別個の Ceph Storage Cluster と独自のノードをオーバークラウドに統合します。このクラスターは、オーバークラウドから独立して、管理されます。たとえば、Ceph Storage Cluster は、OpenStack Platform director ではなく Ceph 管理ツールを使用してスケーリングします。

すべての OpenStack マシンは、電源管理に IPMI を使用したベアメタルマシンです。director により Red Hat Enterprise Linux 7 のイメージが各ノードにコピーされるため、これらのノードではオペレーティングシステムは必要ありません。

director は、イントロスペクションおよびプロビジョニングプロセス中に、プロビジョニングネットワークを使用してコントローラーノードとコンピュートノードと通信します。すべてのノードは、ネイティブの VLAN 経由でネットワークに接続します。この例では、以下の IP アドレスの割り当てで、プロビジョニングサブネットとして 192.0.2.0/24 を使用します。

ノード名	IP アドレス	MAC アドレス	IPMI IP アドレス
director	192.0.2.1	aa:aa:aa:aa:aa:aa	
コントローラー 1	定義済みの DHCP	b1:b1:b1:b1:b1:b1	192.0.2.205
コントローラー 2	定義済みの DHCP	b2:b2:b2:b2:b2:b2	192.0.2.206
コントローラー 3	定義済みの DHCP	b3:b3:b3:b3:b3:b3	192.0.2.207
コンピュート 1	定義済みの DHCP	c1:c1:c1:c1:c1:c1	192.0.2.208
コンピュート 2	定義済みの DHCP	c2:c2:c2:c2:c2:c2	192.0.2.209
コンピュート 3	定義済みの DHCP	c3:c3:c3:c3:c3:c3	192.0.2.210

3.1. 既存の CEPH STORAGE CLUSTER の設定

1. お使いの環境に適した Ceph クラスターに以下のプールを作成します。

- ✧ **volumes**: OpenStack Block Storage (cinder) のストレージ
- ✧ **images**: OpenStack Image Storage (glance) のストレージ
- ✧ **vms**: インスタンスのストレージ
- ✧ **backups**: OpenStack Block Storage Backup (cinder-backup) のストレージ
- ✧ **metrics**: OpenStack Telemetry Metrics (gnocchi) のストレージ

以下のコマンドは指標として使用してください。

```
[root@ceph ~]# ceph osd pool create volumes PGNUM
[root@ceph ~]# ceph osd pool create images PGNUM
[root@ceph ~]# ceph osd pool create vms PGNUM
[root@ceph ~]# ceph osd pool create backups PGNUM
[root@ceph ~]# ceph osd pool create metrics PGNUM
```

PGNUM は **配置グループ** の数に置き換えます。1 OSD につき 100 程度を推奨します。たとえば、OSD の合計数を 100 で乗算して、レプリカ数で除算します (**osd pool default size**)。適切な値を判断するには [Ceph Placement Groups \(PGs\) per Pool Calculator](#) を使用することを推奨します。

2. Ceph クラスターに、以下のケーパビリティを指定して **client.openstack** ユーザーを作成します。

- ✧ **cap_mon: allow r**
- ✧ **cap_osd: allow class-read object_prefix rbd_children, allow rwx pool=volumes, allow rwx pool=vms, allow rwx pool=images, allow rwx pool=backups, allow rwx pool=metrics**

以下のコマンドは指標として使用してください。

```
[root@ceph ~]# ceph auth add client.openstack mon 'allow r' osd
'allow class-read object_prefix rbd_children, allow rwx
pool=volumes, allow rwx pool=vms, allow rwx pool=images, allow
rwx pool=backups, allow rwx pool=metrics'
```

3. 次に、**client.openstack** ユーザー向けに作成された **Ceph client key** をメモします。

```
[root@ceph ~]# ceph auth list
...
client.openstack
  key: AQDLOh1VgEp6FRAAFzT7Zw+Y9V6JJExQAsRnRQ==
  caps: [mon] allow r
  caps: [osd] allow class-read object_prefix rbd_children, allow
rwx pool=volumes, allow rwx pool=vms, allow rwx pool=images,
allow rwx pool=backups, allow rwx pool=metrics
...
```

ここでの **key** の値は (AQDLOh1VgEp6FRAAFzT7Zw+Y9V6JJExQAsRnRQ==)、Ceph のクライアントキーです。

- 最後に、Ceph Storage Cluster の **file system ID** をメモします。この値は、クラスターの設定ファイルにある **fsid** の設定で指定されています (**[global]** セクションの配下)。

```
[global]
fsid = 4b5c8c0a-ff60-454b-a1b4-9747aa737d19
...
```

注記

Ceph Storage Cluster の設定ファイルに関する詳しい情報は、『[Red Hat Ceph Storage Configuration Guide](#)』の「[Configuration Reference](#)」を参照してください。

Ceph クライアントキーおよびファイルシステム ID はいずれも、「[既存の Ceph Storage Cluster との統合の有効化](#)」で後ほど使用します。

3.2. STACK ユーザーの初期化

stack ユーザーとして director ホストにログインし、以下のコマンドを実行して director の設定を初期化します。

```
$ source ~/stackrc
```

このコマンドでは、director の CLI ツールにアクセスする認証情報が含まれる環境変数を設定します。

3.3. ノードの登録

ノード定義のテンプレート (**instackenv.json**) は JSON ファイル形式で、ノード登録用のハードウェアおよび電源管理の情報が含まれています。以下に例を示します。

```
{
  "nodes": [
    {
      "mac": [
        "bb:bb:bb:bb:bb:bb"
      ],
      "cpu": "4",
      "memory": "6144",
      "disk": "40",
      "arch": "x86_64",
      "pm_type": "pxe_ipmitool",
      "pm_user": "admin",
      "pm_password": "p@55w0rd!",
      "pm_addr": "192.0.2.205"
    },
    {
      "mac": [
        "cc:cc:cc:cc:cc:cc"
      ],
      "cpu": "4",
```

```

        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",
        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.206"
    },
    {
        "mac": [
            "dd:dd:dd:dd:dd:dd"
        ],
        "cpu": "4",
        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",
        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.207"
    },
    {
        "mac": [
            "ee:ee:ee:ee:ee:ee"
        ],
        "cpu": "4",
        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",
        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.208"
    }
    {
        "mac": [
            "ff:ff:ff:ff:ff:ff"
        ],
        "cpu": "4",
        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",
        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.209"
    }
    {
        "mac": [
            "gg:gg:gg:gg:gg:gg"
        ],
        "cpu": "4",
        "memory": "6144",
        "disk": "40",
        "arch": "x86_64",
        "pm_type": "pxe_ipmitool",

```

```

        "pm_user": "admin",
        "pm_password": "p@55w0rd!",
        "pm_addr": "192.0.2.210"
    }
}

```

テンプレートの作成後に、stack ユーザーのホームディレクトリー (`/home/stack/instackenv.json`) にファイルを保存してから、director にインポートします。これには、以下のコマンドを実行します。

```
$ openstack baremetal import --json ~/instackenv.json
```

このコマンドでテンプレートをインポートして、テンプレートから director に各ノードを登録します。

カーネルと ramdisk イメージを全ノードに割り当てます。

```
$ openstack baremetal configure boot
```

director でのノードの登録、設定が完了しました。

3.4. ノードのハードウェアの検査

ノードの登録後には、各ノードのハードウェア属性を検査します。各ノードのハードウェア属性を検査するには、以下のコマンドを実行します。

```
$ openstack baremetal introspection bulk start
```

重要

このプロセスが最後まで実行されて正常に終了したことを確認してください。ベアメタルの場合には、通常 15 分ほどかかります。

3.5. ノードの手動でのタグ付け

各ノードのハードウェアを登録、検査した後は、特定のプロファイルにノードをタグ付けします。これらのプロファイルタグによりノードとフレーバーが照合され、フレーバーがデプロイメントロールに割り当てられます。

ノード一覧を取得して UUID を識別します。

```
$ ironic node-list
```

特定のプロファイルにノードを手動でタグ付けする場合には、各ノードの **properties/capabilities** パラメーターに `profile` オプションを追加します。たとえば、2 台のノードをタグ付けしてコントローラープロファイルとコンピュートプロファイルをそれぞれ使用するには、以下のコマンドを実行します。

```

$ ironic node-update 1a4e30da-b6dc-499d-ba87-0bd8a3819bc0 add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 6faba1a9-e2d8-4b7c-95a2-c7fbdc12129a add

```

```
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 5e3b2f50-fcd9-4404-b0a2-59d79924b38e add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 484587b2-b3b3-40d5-925b-a26a2fa3036f add
properties/capabilities='profile:compute,boot_option:local'
$ ironic node-update d010460b-38f2-4800-9cc4-d69f0d067efe add
properties/capabilities='profile:compute,boot_option:local'
$ ironic node-update d930e613-3e14-44b9-8240-4f3559801ea6 add
properties/capabilities='profile:compute,boot_option:local'
```

profile オプションを追加すると、適切なプロファイルにノードをタグ付けします。



注記

手動でのタグ付けの代わりに、Automated Health Check (AHC) ツールを使用し、ベンチマークデータに基づいて、多数のノードに自動でタグ付けします。

3.6. 既存の CEPH STORAGE CLUSTER との統合の有効化

stack ユーザーのホームディレクトリー内のディレクトリーに `/usr/share/openstack-tripleo-heat-templates/environments/puppet-ceph-external.yaml` のコピーを作成します。

```
[stack@director ~]# mkdir templates
[stack@director ~]# cp /usr/share/openstack-tripleo-heat-
templates/environments/puppet-ceph-external.yaml ~/templates/.
```

このファイルを編集して、下記のパラメーターを設定します。

- ※ 絶対パスに **CephExternal** のリソース定義を設定します。

```
OS::TripleO::Services::CephExternal: /usr/share/openstack-tripleo-heat-
templates/puppet/services/ceph-external.yaml
```

- ※ Ceph Storage 環境の情報を使用して、以下の 3 つのパラメーターを設定します。

- **CephClientKey**: Ceph Storage Cluster の Ceph クライアントキー。これは、「[既存の Ceph Storage Cluster の設定](#)」で先ほど取得した **key** の値です (例: **AQDL0h1VgEp6FRAAFzT7Zw+Y9V6JJExQAsRnRQ==**)。
- **CephExternalMonHost**: Ceph Storage Cluster の全 MON ホストの IP をコンマ区切りにしたリスト
- **CephClusterFSID**: Ceph Storage Cluster のファイルシステム ID。これは、「[既存の Ceph Storage Cluster の設定](#)」で先ほど取得した Ceph Storage Cluster の設定ファイルにある **fsid** の値です (例: **4b5c8c0a-ff60-454b-a1b4-9747aa737d19**)。

- ※ 必要の場合は、以下のパラメーターと値を使用して、OpenStack プールとクライアントユーザーの名前を設定します。

- **CephClientUserName**: **openstack**
- **NovaRbdPoolName**: **vms**

- **CinderRbdPoolName:** volumes
- **GlanceRbdPoolName:** images
- **CinderBackupRbdPoolName:** backups
- **GnocchiRbdPoolName:** metrics

3.7. 以前の RED HAT CEPH STORAGE バージョンとの後方互換性

Red Hat OpenStack Platform が、以前のバージョン(Red Hat Ceph Storage 1.3) の外部の Ceph Storage Cluster と統合されている場合には、後方互換性を有効にする必要があります。そのためには、最初に環境ファイル (例: `/home/stack/templates/ceph-backwards-compatibility.yaml`) を作成して、以下の内容を記載します。

```
parameter_defaults:
  ExtraConfig:
    ceph::conf::args:
      client/rbd_default_features:
        value: "1"
```

「[オーバークラウドの作成](#)」に記載のとおり、オーバークラウドのデプロイメントにこのファイルを追加します。

3.8. オーバークラウドの作成

オーバークラウドの作成には、**openstack overcloud deploy** コマンドに追加の引数を指定する必要があります。以下に例を示します。

```
$ openstack overcloud deploy --templates -e
/home/stack/templates/puppet-ceph-external.yaml --control-scale 3 --
compute-scale 3 --ceph-storage-scale 0 --control-flavor control --
compute-flavor compute --neutron-network-type vxlan --ntp-server
pool.ntp.org
```

上記のコマンドは、以下のオプションを使用します。

- ※ **--templates:** デフォルトの Heat テンプレートコレクションからオーバークラウドを作成します。
- ※ **-e /home/stack/templates/puppet-ceph-external.yaml:** 別の環境ファイルをオーバークラウドデプロイメントに追加します。この場合は、既存の Ceph Storage Cluster の設定を含むストレージ環境ファイルです。
- ※ **--control-scale 3:** コントローラーノードを 3 台にスケーリングします。
- ※ **--compute-scale 3:** コンピュートノードを 3 台にスケーリングします。
- ※ **--ceph-storage-scale 0:** Ceph Storage ノードを 0 にスケーリングします。このように指定すると、director により Ceph Storage ノードが作成されないようになります。
- ※ **--control-flavor control:** 対象のコントローラーノードに特定のフレーバーを使用します。
- ※ **--compute-flavor compute:** コンピュートノードに特定のフレーバーを使用します。

※ **--neutron-network-type vxlan**: ネットワーク種別を **neutron** に設定します。

※ **--ntp-server pool.ntp.org**: NTP サーバーを設定します。

注記

オプションの完全な一覧を表示するには、以下を実行します。

```
$ openstack help overcloud deploy
```

詳しい情報は、『[director のインストールと使用方法](#)』ガイドの「[オーバークラウドのパラメーター設定](#)」を参照してください。

オーバークラウドの作成プロセスが開始され、director によりノードがプロビジョニングされます。このプロセスは完了するまで多少時間がかかります。オーバークラウドの作成のステータスを確認するには、**stack** ユーザーとして別のターミナルを開き、以下を実行します。

```
$ source ~/stackrc
$ heat stack-list --show-nested
```

この設定では、オーバークラウドが外部の Ceph Storage Cluster を使用するように設定します。このクラスターは、オーバークラウドから独立して、管理される点に注意してください。たとえば、Ceph Storage Cluster は、OpenStack Platform director ではなく Ceph 管理ツールを使用してスケールリングします。

3.9. オーバークラウドへのアクセス

director は、director ホストがオーバークラウドと対話するための設定と認証を行うスクリプトを生成します。このファイル (**overcloudrc**) は、**stack** ユーザーのホームディレクトリーに保存されます。このファイルを使用するには、以下のコマンドを実行します。

```
$ source ~/overcloudrc
```

これにより、director ホストの CLI からオーバークラウドと対話するために必要な環境変数が読み込まれます。director ホストとの対話に戻るには、以下のコマンドを実行します。

```
$ source ~/stackrc
```

第4章 まとめ

これで、Red Hat Ceph Storage を使用したオーバークラウドの作成および設定が完了しました。オーバークラウド作成後の一般的な機能については、[director インストールと使用方法](#)の「[オーバークラウドの作成後のタスク実行](#)」を参照してください。

付録A 環境ファイルのサンプル: CEPH クラスターの作成

以下のカスタム環境ファイルのサンプルは、「[2章 Ceph Storage ノードでのオーバークラウドの作成](#)」で説明したオプションの多くを使用しています。このサンプルには、コメントアウトされているオプションは含まれません。環境ファイルの概要については、『[Advanced Overcloud Customization](#)』ガイドの「[Environment Files](#)」を参照してください。

/home/stack/templates/storage-environment.yaml

```
resource_registry: // 1
  OS::TripleO::Services::CephMon: /usr/share/openstack-tripleo-heat-
  templates/puppet/services/ceph-mon.yaml
  OS::TripleO::Services::CephOSD: /usr/share/openstack-tripleo-heat-
  templates/puppet/services/ceph-osd.yaml
  OS::TripleO::Services::CephClient: /usr/share/openstack-tripleo-heat-
  templates/puppet/services/ceph-client.yaml
  OS::TripleO::Services::CinderBackup: /usr/share/openstack-tripleo-
  heat-templates/puppet/services/pacemaker/cinder-backup.yaml // 2
  OS::TripleO::NodeUserData: /home/stack/templates/firstboot/wipe-
  disks.yaml // 3

parameter_defaults: // 4
  CinderEnableIscsiBackend: false
  CinderEnableRbdBackend: true
  CinderEnableNfsBackend: false
  NovaEnableRbdBackend: true
  GlanceBackend: rbd
  CinderBackupBackend: ceph // 5
  ExtraConfig:
    ceph::profile::params::osds: // 6
      /dev/sdc:
        journal: /dev/sdb
      /dev/sdd:
        journal: /dev/sdb
```

1

resource_registry セクションは、Heat テンプレートにリンクするリソースを定義します。最初の3つのエントリー (**CephMon**、**CephOSD**、**CephClient**) は、Ceph クラスター (MON、OSD、クライアント) の異なるコンポーネントを定義するのに使用する Heat テンプレートにリンクします。

2

OS::TripleO::Services::CinderBackup のエントリーは、Block Storage Backup サービスをデプロイするのに必要な Heat テンプレート呼び出します。デフォルトでは、**ceph-client** Heat テンプレートは Object Storage サービス (**swift**) をバックアップ先として設定します。これは、**parameter_defaults** セクションで後ほど上書きすることができます。

3

OS::Triple0::NodeUserData: エントリーは、カスタムのスクリプトを使用し、(root ファイルシステムを含むディスク以外の) Ceph Storage ノードにある全ディスクをチェックしてすべてを GPT に変換するテンプレート (/home/stack/templates/firstboot/wipe-disks.yaml) を使用します。詳細は「[Ceph Storage ノードディスクの GPT へのフォーマット](#)」を参照してください。

4

parameter_defaults セクションは、全テンプレート内のパラメーターのデフォルト値を変更します。ここに記載のエントリーはすべて「[オーバークラウドでの Ceph Storage の有効化](#)」に記載しています。

5

デフォルトでは、**OS::Triple0::Services::CinderBackup:** で使用する Heat テンプレートは Object Storage サービスをバックアップ先として設定します。**CinderBackupBackend: ceph** エントリーは、**cinder-backup** がバックアップを Ceph に保存するように設定します (「[Backup サービスが Ceph にバックアップを保存するように設定する手順](#)」に記載)。

6

「[Ceph Storage ノードディスクのレイアウトのマッピング](#)」の説明にあるように、**ceph::profile::params::osds::** セクションは、カスタムのディスクレイアウトを定義します。