



Red Hat OpenStack Platform 16.2

ハイパーコンバージドインフラストラクチャーガイド

Red Hat OpenStack Platform オーバークラウドにおけるハイパーコンバージドインフラストラクチャーの設定についての理解

Red Hat OpenStack Platform 16.2 ハイパーコンバージドインフラストラクチャーガイド

Red Hat OpenStack Platform オーバークラウドにおけるハイパーコンバージドインフラストラクチャーの設定についての理解

OpenStack Team
rhos-docs@redhat.com

法律上の通知

Copyright © 2023 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, the Red Hat 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 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 OpenStack Platform のハイパーコンバージェンスの実装について説明します。この実装では、Compute サービスと Ceph Storage サービスが同じホストに配置されます。

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 RED HAT OPENSTACK PLATFORM ハイパーコンバージドインフラストラクチャーの設定およびデプロイ	5
1.1. 前提条件	5
1.2. ハイパーコンバージドノード向けのオーバークラウドロールの準備	6
1.3. ハイパーコンバージドノード上におけるリソース分離の設定	10
1.4. 利用可能な RED HAT CEPH STORAGE パッケージの確認	13
1.5. HCI オーバークラウドのデプロイ	14
1.6. OPENSTACK WORKFLOW COMPUTE の CPU およびメモリーの計算	16
1.7. 関連情報	18
第2章 ハイパーコンバージドノードのスケーリング	19
2.1. HCI 環境におけるハイパーコンバージドノードのスケールアップ	19
2.2. HCI 環境におけるハイパーコンバージドノードのスケールダウン	19
付録A 追加情報	20
A.1. 設定ガイダンス	20

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。

ドキュメントへのダイレクトフィードバック (DDF) 機能の使用 (英語版のみ)

特定の文章、段落、またはコードブロックに対して直接コメントを送付するには、DDF の **Add Feedback** 機能を使用してください。なお、この機能は英語版のドキュメントでのみご利用いただけます。

1. **Multi-page HTML** 形式でドキュメントを表示します。
2. ドキュメントの右上隅に **Feedback** ボタンが表示されていることを確認してください。
3. コメントするテキスト部分をハイライト表示します。
4. **Add Feedback** をクリックします。
5. **Add Feedback** フィールドにコメントを入力します。
6. オプション: ドキュメントチームが問題の詳細を確認する際に使用できるメールアドレスを記入してください。
7. **Submit** をクリックします。

第1章 RED HAT OPENSTACK PLATFORM ハイパーコンバージドインフラストラクチャーの設定およびデプロイ

Red Hat OpenStack Platform (RHOSP) ハイパーコンバージドインフラストラクチャー (HCI) は、ハイパーコンバージドノードで設定されます。リソースの使用率を最適化するために、サービスがこのハイパーコンバージドノード上で共存します。RHOSP HCI では、Compute サービスとストレージサービスがハイパーコンバージドノード上で共存します。ハイパーコンバージドノードのみのオーバークラウド、またはハイパーコンバージドノードを通常のコンピュータードおよび Ceph Storage ノードと混在させたオーバークラウドをデプロイすることが可能です。



注記

Red Hat Ceph Storage をストレージプロバイダーとして使用する必要があります。

ヒント

- Ceph のメモリ設定を自動的に調整するには、ceph-ansible 3.2 以降を使用します。
- BlueStore のメモリ処理機能を利用するには、BlueStore を HCI デプロイメントのバックエンドとして使用します。

オーバークラウド上に HCI を作成およびデプロイし、オーバークラウドのネットワーク機能仮想化などのその他の機能と統合し、ハイパーコンバージドノード上の Compute サービスと Red Hat Ceph Storage サービス両方のパフォーマンスを最適な状態にするには、以下の手順を実施する必要があります。

1. ハイパーコンバージドノード向けの事前定義されたカスタムオーバークラウドロール **ComputeHCI** を準備する。
2. リソース分離を設定する。
3. 利用可能な Red Hat Ceph Storage パッケージを確認する。
4. HCI オーバークラウドをデプロイする。

HCI 設定のガイダンスについては、[設定ガイダンス](#) を参照してください。

1.1. 前提条件

- アンダークラウドをデプロイしている。アンダークラウドのデプロイ方法についての説明は、[Director Installation and Usage](#) を参照してください。
- お使いの環境で、RHOSP Compute および Red Hat Ceph Storage の要件を満たすノードをプロビジョニング可能である。詳しくは、[Basic Overcloud Deployment](#) を参照してください。
- 環境内の全ノードを登録している。詳しくは、[Registering Nodes](#) を参照してください。
- 環境内の全ノードがタグ付けされている。詳しくは、[Manually Tagging the Nodes](#) を参照してください。
- Compute サービスおよび Ceph OSD サービスに使用するノード上のディスクをクリーンアップしている。詳しくは、[Cleaning Ceph Storage Node Disks](#) を参照してください。

- オーバークラウドノードを Red Hat コンテンツ配信ネットワークまたは Red Hat Satellite サーバーに登録するための準備を行っている。詳しくは、[Ansible-based Overcloud Registration](#) を参照してください。

1.2. ハイパーコンバージドノード向けのオーバークラウドロールの準備

ノードをハイパーコンバージドとして指定するには、ハイパーコンバージドロールを定義する必要があります。Red Hat OpenStack Platform (RHOSP) は、ハイパーコンバージドノード向けの事前定義されたロール **ComputeHCI** を提供します。このロールにより、Compute サービスと Ceph オブジェクトストレージデーモン (OSD) サービスを共存させ、同じハイパーコンバージドノード上にまとめてデプロイすることができます。

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. **stackrc** ファイルを取得します。

```
[stack@director ~]$ source ~/stackrc
```

3. オーバークラウドに使用するその他のロールに加えて **ComputeHCI** ロールが含まれる新たなカスタムロールデータファイルを生成します。以下の例では、ロールデータファイル **roles_data_hci.yaml** を生成します。これには、**Controller**、**ComputeHCI**、**Compute**、および **CephStorage** ロールが含まれます。

```
(undercloud)$ openstack overcloud roles \
generate -o /home/stack/templates/roles_data_hci.yaml \
Controller ComputeHCI Compute CephStorage
```

注記

生成されたカスタムロールデータファイルの **ComputeHCI** ロール用に一覧表示されるネットワークには、Compute サービスと Storage サービスの両方に必要なネットワークが含まれます。以下に例を示します。

```
- name: ComputeHCI
  description: |
    Compute node role hosting Ceph OSD
  tags:
    - compute
  networks:
    InternalApi:
      subnet: internal_api_subnet
    Tenant:
      subnet: tenant_subnet
    Storage:
      subnet: storage_subnet
    StorageMgmt:
      subnet: storage_mgmt_subnet
```

4. **network_data.yaml** ファイルのローカルコピーを作成し、コンポーザブルネットワークをオーバークラウドに追加します。**network_data.yaml** ファイルは、デフォルトのネットワーク環境ファイル **/usr/share/openstack-tripleo-heat-templates/environments/*** と対話

し、**ComputeHCI** ロール用に定義したネットワークをハイパーコンバージドノードに関連付けます。詳しい情報は、**Advanced Overcloud Customization**の [Adding a composable network](#) を参照してください。

- Red Hat Ceph Storage のパフォーマンスを向上させるには、**network_data.yaml** のローカルコピーでジャンボフレーム用に **Storage** および **StorageMgmt** ネットワーク両方の MTU 設定を **9000** に更新します。詳細は、[Configuring MTU Settings in Director](#) および [Configuring jumbo frames](#) を参照してください。
- ハイパーコンバージドノード向けの **computeHCI** オーバークラウドフレーバーを作成します。

```
(undercloud)$ openstack flavor create --id auto \
--ram <ram_size_mb> --disk <disk_size_gb> \
--vcpus <no_vcpus> computeHCI
```

- **<ram_size_mb>** をベアメタルノードの RAM (MB 単位) に置き換えます。
- **<disk_size_gb>** をベアメタルノード上のディスク容量 (GB 単位) に置き換えます。
- **<no_vcpus>** をベアメタルノードの CPU 数に置き換えます。



注記

これらの属性は、インスタンスのスケジューリングには使用されません。ただし Compute スケジューラーは、ディスク容量を使用してルートパーティションのサイズを決定します。

- ノード一覧を取得して UUID を把握します。

```
(undercloud)$ openstack baremetal node list
```

- ハイパーコンバージドとして指定する各ベアメタルノードに、カスタムの HCI リソースクラスをタグ付けします。

```
(undercloud)$ openstack baremetal node set \
--resource-class baremetal.HCI <node>
```

<node> をベアメタルノードの ID に置き換えてください。

- computeHCI** フレーバーをカスタムの HCI リソースクラスに関連付けます。

```
(undercloud)$ openstack flavor set \
--property resources:CUSTOM_BAREMETAL_HCI=1 \
computeHCI
```

Bare Metal サービスノードのリソースクラスに対応するカスタムリソースクラスの名前を指定するには、リソースクラスを大文字に変換し、すべての句読点をアンダースコアに置き換え、**CUSTOM_** の接頭辞を追加します。



注記

フレーバーが要求できるのは、ベアメタルリソースクラスの1つのインスタンスだけです。

10. 以下のフレーバー属性を設定して、Compute スケジューラーがインスタンスのスケジューリングにベアメタルフレーバー属性を使用するのを防ぎます。

```
(undercloud)$ openstack flavor set \
--property resources:VCPU=0 \
--property resources:MEMORY_MB=0 \
--property resources:DISK_GB=0 computeHCI
```

11. 以下のパラメーターを **node-info.yaml** ファイルに追加して、ハイパーコンバージドノードおよびコントローラーノードの数およびハイパーコンバージドおよびコントローラーに指定するノード用に使用するフレーバーを指定します。

```
parameter_defaults:
  OvercloudComputeHCIFlavor: computeHCI
  ComputeHCICount: 3
  Controller: control
  ControllerCount: 3
```

関連情報

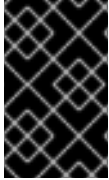
- [Composable Services and Custom Roles](#)
- [Examining the roles_data file](#)
- [Assigning Nodes and Flavors to Roles](#)

1.2.1. マルチディスククラスターのルートディスクの定義

ノードで複数のディスクが使用されている場合には、director はプロビジョニング時にルートディスクを特定する必要があります。たとえば、ほとんどの Ceph Storage ノードでは、複数のディスクが使用されます。デフォルトのプロビジョニングプロセスでは、director はルートディスクにオーバークラウドイメージを書き込みます。

以下の属性を定義すると、director がルートディスクを特定するのに役立ちます。

- **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
- **hctl** (文字列): SCSI のホスト、チャンネル、ターゲット、Lun
- **size** (整数): デバイスのサイズ (GB 単位)
- **wwn** (文字列): 一意のストレージ ID
- **wwn_with_extension** (文字列): ベンダー拡張子を追加した一意のストレージ ID
- **wwn_vendor_extension** (文字列): 一意のベンダーストレージ ID
- **rotational** (ブール値): 回転式デバイス (HDD) には true、そうでない場合 (SSD) には false
- **name** (文字列): デバイス名 (例: /dev/sdb1)



重要

name プロパティは、永続デバイス名が付いたデバイスにのみ使用します。他のデバイスのルートディスクを設定する際に、**name** を使用しないでください。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。

シリアル番号を使用してルートデバイスを指定することができます。

手順

1. 各ノードのハードウェアイントロスペクションからのディスク情報を確認します。以下のコマンドを実行して、ノードのディスク情報を表示します。

```
(undercloud)$ openstack baremetal introspection data save 1a4e30da-b6dc-499d-ba87-0bd8a3819bc0 | jq ".inventory.disks"
```

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。

```
[
  {
    "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"
  }
]
```

2. **openstack baremetal node set --property root_device=** を入力して、ノードのルートディスクを設定します。ルートディスクを定義するのに最も適切なハードウェア属性値を指定します。

```
(undercloud)$ openstack baremetal node set --property root_device='{ "serial": "  
<serial_number>" }' <node-uuid>
```

たとえば、ルートデバイスをシリアル番号が **61866da04f380d001ea4e13c12e36ad6** の disk 2 に設定するには、以下のコマンドを実行します。

```
(undercloud)$ openstack baremetal node set --property root_device='{ "serial": "  
"61866da04f380d001ea4e13c12e36ad6" }' 1a4e30da-b6dc-499d-ba87-0bd8a3819bc0
```



注記

各ノードの BIOS を設定して、選択したルートディスクからの起動を含めるようにします。最初にネットワークからのブートを試み、次にルートディスクからのブートを試みるように、ブート順序を設定します。

director は、ルートディスクとして使用する特定のディスクを把握します。**openstack overcloud deploy** コマンドを実行すると、director はオーバークラウドをプロビジョニングし、ルートディスクにオーバークラウドのイメージを書き込みます。

1.3. ハイパーコンバージドノード上におけるリソース分離の設定

ハイパーコンバージドノード上で Ceph OSD サービスと Compute サービスを共存させると、お互いが同じホスト上に存在することを認識しないため、Red Hat Ceph Storage サービスと Compute サービス間でリソースの競合が発生するリスクがあります。リソースの競合が発生すると、サービスのパフォーマンスが低下し、ハイパーコンバージェンスの利点が打ち消される可能性があります。

Ceph サービスと Compute サービスの両方にリソースの分離を設定して、競合が発生を防ぐ必要があります。

手順

1. (オプション) Compute 環境ファイルに以下のパラメーターを追加して、自動生成される Compute 設定を上書きします。

```
parameter_defaults:  
  ComputeHCIParameters:  
    NovaReservedHostMemory: <ram>  
    NovaCPUAllocationRatio: <ratio>
```

- **<ram>** を、ハイパーコンバージドノード上で Ceph OSD サービスおよびインスタンスのオーバーヘッド用に確保する RAM 容量 (MB 単位) に置き換えます。
- **<ratio>** を、インスタンスをデプロイするコンピュートノードを選択する際に Compute スケジューラーが使用すべき比率に置き換えます。
自動生成される Compute 設定についての詳細は、[Compute サービス用に確保する CPU およびメモリーリソースの自動生成プロセス](#) を参照してください。

2. Red Hat Ceph Storage 用にメモリーリソースを確保するには、**/home/stack/templates/storage-container-config.yaml** で **is_hci** を **true** に設定します。

```
parameter_defaults:
  CephAnsibleExtraConfig:
    is_hci: true
```

この設定により、**ceph-ansible** は Red Hat Ceph Storage 用のメモリーリソースを確保することができ、HCI デプロイメントの **osd_memory_target** パラメーター設定を自動的に調整することで、Ceph OSD によるメモリー増大を低減することができます。



警告

Red Hat では、**ceph_osd_docker_memory_limit** パラメーターを直接上書きすることは推奨しません。



注記

FileStore または BlueStore どちらのバックエンドが使用されていても、ceph-ansible 3.2 では、**ceph_osd_docker_memory_limit** は Ansible の検出したホストの最大メモリーに自動的に設定されます。

3. (オプション) デフォルトでは、**ceph-ansible** は Ceph OSD ごとに1つの仮想 CPU を確保します。Ceph OSD ごとに複数の CPU が必要な場合は、以下の設定を **/home/stack/templates/storage-container-config.yaml** に追加します。

```
parameter_defaults:
  CephAnsibleExtraConfig:
    ceph_osd_docker_cpu_limit: <cpu_limit>
```

<cpu_limit> を各 Ceph OSD 用に確保する CPU 数に置き換えます。

ハードウェアおよびワークロードに基づいて CPU リソースを調整する方法の詳細は、[Red Hat Ceph Storage Hardware Selection Guide](#) を参照してください。

4. (オプション) 以下のパラメーターを Ceph 環境ファイルに追加することで、Ceph OSD の削除時に Red Hat Ceph Storage のバックフィルとリカバリーの操作の優先度を低くします。

```
parameter_defaults:
  CephConfigOverrides:
    osd_recovery_op_priority: <priority_value>
    osd_recovery_max_active: <no_active_recovery_requests>
    osd_max_backfills: <max_no_backfills>
```

- **<priority_value>** を、OSD クライアント OP の優先度と相対的に、リカバリーの操作の優先度に置き換えます。
- **<no_active_recovery_requests>** を、1 OSD あたりの1回にアクティブなリカバリー要求の件数に置き換えます。
- **<max_no_backfills>** を、単一の OSD との間で許容されるバックフィルの最大数に置き換えます。

デフォルトの Red Hat Ceph Storage のバックフィルおよびリカバリーオプションに関する詳細は、[Red Hat Ceph Storage のバックフィルとリカバリーの操作](#) を参照してください。

1.3.1. Compute サービス用に確保する CPU およびメモリーリソースの自動生成プロセス

director の提供するデフォルトのプラン環境ファイルにより、デプロイメント時のハイパーコンバージドノードのリソース制約が設定されます。このプラン環境ファイルは、OpenStack Workflow に以下のプロセスを実施するように指示します。

1. ハードウェアノードの検査時に収集したハードウェアイントロスペクションデータを取得する。
2. そのデータに基づき、ハイパーコンバージドノード上の Compute の最適な CPU およびメモリー割り当て負荷を算出する。
3. これらの制約を設定し Compute に CPU/メモリーリソースを確保するのに必要なパラメーターを自動生成する。これらのパラメーターは、**plan-environment-derived-params.yaml** ファイルの **hci_profile_config** セクションで定義されます。



注記

Compute の **reserved_host_memory** および **cpu_allocation_ratio** の設定値を算出するのに、各ワークロードプロファイルの **average_guest_memory_size_in_mb** および **average_guest_cpu_utilization_percentage** パラメーターが使用されます。

Compute 環境ファイルに以下のパラメーターを追加して、自動生成される Compute 設定を上書きすることができます。

自動生成される nova.conf パラメーター	Compute 環境ファイルのオーバーライド	説明
reserved_host_memory	<pre>parameter_defaults: ComputeHCIParameters: NovaReservedHostMemory: 181000</pre>	ハイパーコンバージドノード上で、Ceph OSD サービスおよびゲストインスタンスごとのオーバーヘッドに確保する RAM 容量を設定します。
cpu_allocation_ratio	<pre>parameter_defaults: ComputeHCIParameters: NovaCPUAllocationRatio: 8.2</pre>	インスタンスをデプロイするコンピュータノードを選択する際に Compute スケジューラーが使用すべき比率を設定します。

これらのオーバーライドは、ComputeHCI ロールを使用するすべてのノード (つまり、すべてのハイパーコンバージドノード) に適用されます。**NovaReservedHostMemory** および **NovaCPUAllocationRatio** の最適値を手動で決定する方法についての詳細は、[OpenStack Workflow Compute の CPU およびメモリーの計算](#) を参照してください。

ヒント

以下のスクリプトを使用して、ハイパーコンバージドノードの **NovaReservedHostMemory** および **NovaCPUAllocationRatio** の適切な基準値を算出することができます。

[nova_mem_cpu_calc.py](#)

関連情報

- [ベアメタルノードハードウェアのインベントリーの作成](#)

1.3.2. Red Hat Ceph Storage のバックフィルとリカバリーの操作

Ceph OSD が削除されると、Red Hat Ceph Storage はバックフィルおよびリカバリー操作を使用してクラスターをリバランスします。Red Hat Ceph Storage は配置グループポリシーに従って、データのコピーを複数保管するためにこの操作を実行します。これらの操作は、システムリソースを使用します。Red Hat Ceph Storage クラスターに負荷がかかっている場合、リソースがバックフィルおよびリカバリーに回されるので、パフォーマンスが低下します。

OSD 削除時のこのパフォーマンスへの影響を軽減するには、バックフィルおよびリカバリー操作の優先度を低くすることができます。この手法のマイナス面は、データのレプリカがより少ない状態が長くなるので、データがリスクにさらされる可能性が若干高くなることです。

以下の表で説明するパラメーターが、バックフィルおよびリカバリー操作の優先度を設定するのに使用されます。

パラメーター	説明	デフォルト値
osd_recovery_op_priority	OSD クライアントの操作に関して、リカバリー操作の優先度を設定します。	3
osd_recovery_max_active	同時に要求できる1 OSD あたりのアクティブなリカバリー要求の数を設定します。要求が増えるにつれてリカバリーは加速されますが、それらの要求によりクラスターにかかる負荷が増大します。レイテンシーを低くするには、このパラメーターを1に設定します。	3
osd_max_backfills	単一の OSD との間で許容されるバックフィルの最大数を設定します。	1

1.4. 利用可能な RED HAT CEPH STORAGE パッケージの確認

オーバークラウドのデプロイメントが失敗しないようにするには、必要なパッケージがサーバーに存在することを確認します。

1.4.1. ceph-ansible パッケージバージョンの確認

アンダークラウドには Ansible ベースの検証が含まれ、これを実行してオーバークラウドをデプロイする前に潜在的な問題を特定することができます。これらの検証は、典型的な問題が発生する前にそれらを特定し、オーバークラウドのデプロイメントの失敗を回避するのに役立ちます。

手順

- 必要な **ceph-ansible** パッケージバージョンがインストールされていることを確認します。

```
$ ansible-playbook -i /usr/bin/tripleo-ansible-inventory /usr/share/ansible/validation-
playbooks/ceph-ansible-installed.yaml
```

1.4.2. 事前にプロビジョニングされたノード用のパッケージの確認

Red Hat Ceph Storage (RHCS) は、特定のパッケージセットを持つオーバークラウドノードにのみサービスを提供することができます。事前にプロビジョニングされたノードを使用する場合には、これらのパッケージが存在することを確認することができます。

事前にプロビジョニングされたノードの詳細は、[Configuring a basic overcloud with pre-provisioned nodes](#) を参照してください。

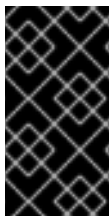
手順

- 事前にプロビジョニングされたノードに必要なパッケージが含まれていることを確認します。

```
ansible-playbook -i /usr/bin/tripleo-ansible-inventory /usr/share/ansible/validation-
playbooks/ceph-dependencies-installed.yaml
```

1.5. HCI オーバークラウドのデプロイ

HCI 設定が完了した後に、オーバークラウドをデプロイする必要があります。



重要

Red Hat OpenStack Platform (RHOSP) HCI 環境をデプロイする際には、インスタンス HA を有効にしないでください。Red Hat Ceph Storage を使用したハイパーコンバージド RHOSP デプロイメントでインスタンス HA を使用する場合は、Red Hat の担当者にお問い合わせください。

前提条件

- その他すべての Red Hat Ceph Storage の設定に、別のベース環境ファイルを 1 つ (または複数) 使用している (例: **/home/stack/templates/storage-config.yaml**)。詳細は、[Customizing the Storage service](#) および [Appendix A. Sample environment file: creating a Ceph Storage cluster](#) を参照してください。
- ベース環境ファイルで、各ロールに割り当てるノード数を定義している。詳細は、[Assigning nodes and flavors to roles](#) を参照してください。
- アンダークラウドのインストール時に、**undercloud.conf** ファイルで **generate_service_certificate=false** と設定している。設定していない場合は、[Enabling SSL/TLS on Overcloud Public Endpoints](#) で説明するように、オーバークラウドのデプロイ時にトラストアンカーを挿入する必要があります。

手順

- その他の環境ファイルと共に新しいロールファイルおよび環境ファイルをスタックに追加して、HCI オーバークラウドをデプロイします。

■

```
(undercloud)$ openstack overcloud deploy --templates \
-e [your environment files] \
-r /home/stack/templates/roles_data_hci.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-
ansible.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-environment.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/net-single-nic-with-vlans.yaml \
-e /home/stack/templates/storage-config.yaml \
-e /home/stack/templates/storage-container-config.yaml \
-n /home/stack/templates/network_data.yaml \
[-e /home/stack/templates/ceph-backfill-recovery.yaml \]
--ntp-server pool.ntp.org
```

デプロイメントコマンドに **/usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml** を含めることで、すべてのデフォルト設定でコンテナ化された Red Hat Ceph クラスタをデプロイするベース環境ファイルが追加されます。詳しくは、[Deploying an Overcloud with Containerized Red Hat Ceph](#) を参照してください。

注記

デプロイメントで Single Root Input/Output Virtualization (SR-IOV) を使用する場合は、デプロイメントコマンドに以下のオプションを追加します。

デプロイで ML2/OVS メカニズムドライバーを使用する場合は、次のオプションを指定します。

```
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-
sriov.yaml
-e /home/stack/templates/network-environment.yaml
```

デプロイで ML2/OVN メカニズムドライバーを使用する場合は、次のオプションを指定します。

```
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovn-
sriov.yaml
-e /home/stack/templates/network-environment.yaml
```

ヒント

アンサー ファイルを使用して、デプロイメントに追加する環境ファイルを指定することも可能です。詳しくは、[director のインストールと使用方法のオーバークラウドデプロイメントへの環境ファイルの追加](#)を参照してください。

1.5.1. ceph-ansible を実行するノードの限定

ceph-ansible を実行するノードを限定することで、デプロイメントの更新時間を短縮することができます。Red Hat OpenStack Platform (RHOSP) で Ceph の設定に **config-download** が使用されている場合、デプロイメント全体に対して **config-download** および **ceph-ansible** を実行する代わりに、**--limit** オプションを使用してノードの一覧を指定することができます。この機能は、たとえばオーバークラウドをスケールアップする場合や障害の発生したディスクを置き換える場合に役立ちます。このようなシナリオでは、環境に追加する新規ノードでのみデプロイメントを実行することができます。

障害の発生したディスクの置き換え時に --limit を使用するシナリオの例

以下の手順例では、Ceph ストレージノード **oc0-cephstorage-0** でディスク障害が発生したため、ベンダーでフォーマット済みの新規ディスクを受け入れます。新しいディスクを OSD として使用できるように、Ansible を **oc0-cephstorage-0** ノード上で実行する必要があります。しかし、その他のすべての Ceph Storage ノードで実行する必要はありません。例として記述した環境ファイルおよびノードの名前を、実際の環境に適した名前に置き換えてください。

手順

1. アンダークラウドノードに **stack** ユーザーとしてログインし、source コマンドで **stackrc** 認証情報ファイルを読み込みます。

```
# source stackrc
```

2. 新規ディスクが不足している OSD を起動するのに使用されるように、以下の手順の1つを実施します。

- **--limit** オプションを使用して **ceph-ansible** を実行するノードを指定し、スタックの更新を実行する。

```
$ openstack overcloud deploy --templates \
-r /home/stack/roles_data.yaml \
-n /usr/share/openstack-tripleo-heat-templates/network_data_dashboard.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml \
-e ~/my-ceph-settings.yaml \
-e <other-environment_files> \
--limit oc0-controller-0:oc0-controller-2:oc0-controller-1:oc0-cephstorage-0:undercloud
```

この例では、Ceph mon の OSD 定義を変更するのに Ansible が必要なため、コントローラーが含まれています。

- **config-download** が **ansible-playbook-command.sh** スクリプトを生成した場合は、**--limit** オプションを指定してスクリプトを実行し、指定されたノードを **ceph-ansible** に渡すこともできます。

```
./ansible-playbook-command.sh --limit oc0-controller-0:oc0-controller-2:oc0-controller-1:oc0-cephstorage-0:undercloud
```

Warning

必ずアンダークラウドを制限一覧に含める必要があります。含めないと、**--limit** を使用する際に **ceph-ansible** を実行することができません。この注意が必要なのは、アンダークラウドでのみ実行される **external_deploy_steps_tasks** Playbook により **ceph-ansible** が実行されるためです。

1.6. OPENSTACK WORKFLOW COMPUTE の CPU およびメモリーの計算

OpenStack Workflow は CPU とメモリーに最適な設定を計算し、その結果を使用してパラメーター **NovaReservedHostMemory** および **NovaCPUAllocationRatio** に反映させます。

NovaReservedHostMemory

NovaReservedHostMemory パラメーターは、ホストノードに確保するメモリー容量 (MB 単位) を設定します。ハイパーコンバージドノードの適切な値を決定するには、各 OSD が 3 GB のメモリーを消

費すると仮定します。メモリーが 256 GB で OSD が 10 のノードの場合には、Ceph に 30 GB のメモリーを割り当てて、226 GB を Compute に残します。このメモリー容量のノードでは、たとえば、それぞれ 2 GB のメモリーを使用するインスタンスを 113 台ホストすることができます。

ただし、**ハイパーバイザー** 用に、インスタンス 1 台あたりの追加のオーバーヘッドを考慮する必要があります。このオーバーヘッドが 0.5 GB と仮定すると、同じノードでは、90 インスタンスしかホストできません。これは、226 GB を 2.5 GB で割ったものです。ホストノードに確保するメモリー容量 (Compute サービスが使用してはならないメモリー) は以下のように算出します。

$$(In * Ov) + (Os * RA)$$

ここで、

- **In**: インスタンス数
- **Ov**: インスタンス 1 台あたりに必要なオーバーヘッド用のメモリー容量
- **Os**: ノード上の OSD 数
- **RA**: 各 OSD に割り当てる必要のある RAM 容量

90 台のインスタンスの場合には、 $(90 * 0.5) + (10 * 3) = 75$ GB という計算になります。Compute サービスには、この値を MB 単位で指定します (75000)。

以下の Python コードにより、この計算を行うことができます。

```
left_over_mem = mem - (GB_per_OSD * osds)
number_of_guests = int(left_over_mem /
    (average_guest_size + GB_overhead_per_guest))
nova_reserved_mem_MB = MB_per_GB * (
    (GB_per_OSD * osds) +
    (number_of_guests * GB_overhead_per_guest))
```

NovaCPUAllocationRatio

Compute スケジューラーは、インスタンスをデプロイするコンピュートノードを選択する際に **NovaCPUAllocationRatio** を使用します。デフォルトでは、この値は **16.0** (16:1) です。ノードに 56 のコアがある場合には、Compute スケジューラーは 896 の仮想 CPU を使用するのに十分なインスタンスをノードにスケジューリングすることになります。この値を超えると、そのノードはそれ以上インスタンスをホストできないと見なされます。

ハイパーコンバージドノードの適切な **NovaCPUAllocationRatio** を決定するには、各 Ceph OSD が少なくとも 1 コアを使用すると仮定します (ワークロードが I/O 集中型で、SSD を使用しないノード上にある場合を除く)。56 コア、10 OSD のノードでは、この計算で 46 コアが Compute に確保されます。各インスタンスが割り当てられた CPU を 100 パーセント使用すると仮定すると、この比率は単にインスタンスの仮想 CPU 数をコア数で除算した値となります ($46 / 56 = 0.8$)。ただし、通常インスタンスは割り当てられた CPU を 100 パーセント使用することはないため、必要なゲスト仮想 CPU の数を決定する際には、予想される使用率を考慮に入れて **NovaCPUAllocationRatio** を高くすることができます。

したがって、インスタンスが仮想 CPU の 10 パーセント (または 0.1) のみを使用すると予想できる場合には、インスタンス用の仮想 CPU 数は $46 / 0.1 = 460$ の式で表すことができます。この値をコア数 (56) で除算すると、比率は約 8 に増えます。

以下の Python コードにより、この計算を行うことができます。

```
cores_per_OSD = 1.0
average_guest_util = 0.1 # 10%
nonceph_cores = cores - (cores_per_OSD * osds)
guest_vCPUs = nonceph_cores / average_guest_util
cpu_allocation_ratio = guest_vCPUs / cores
```

1.7. 関連情報

Red Hat OpenStack Platform (RHOSP) に関する詳しい情報は、以下のガイドを参照してください。

- [Director Installation and Usage](#): このガイドは、RHOSP 環境 (アンダークラウドおよびオーバークラウドの両方) のエンドツーエンドのデプロイメントに関するガイダンスを提供します。
- [Advanced Overcloud Customization](#): このガイドでは、director を使用して RHOSP の高度な機能を設定する方法について記載しています (例: カスタムロールの使用方法)。
- [Deploying an Overcloud with Containerized Red Hat Ceph](#) : このガイドでは、Red Hat Ceph Storage をストレージプロバイダーとして使用するオーバークラウドのデプロイ方法について記載しています。
- [Networking Guide](#): このガイドでは、RHOSP のネットワーク設定タスクについて詳細に説明しています。

第2章 ハイパーコンバージドノードのスケーリング

HCI ノードをスケールアップまたはスケールダウンする場合、コンピュータードまたは Red Hat Ceph Storage ノードのスケーリングと同じ原則および手法が適用されます。

2.1. HCI 環境におけるハイパーコンバージドノードのスケールアップ

HCI 環境のハイパーコンバージドノードをスケールアップするには、ハイパーコンバージドノードではないノードをスケールアップするのと同じ手順に従います。詳しくは、[Adding nodes to the overcloud](#) を参照してください。



注記

新規ノードをタグ付けする場合には、必ず適切なフレーバーを使用するようにしてください。

OSD を Red Hat Ceph Storage クラスターに追加して HCI ノードをスケールアップする方法は、**Deploying an Overcloud with Containerized Red Hat Ceph** の [Deploying an Overcloud with Containerized Red Hat Ceph](#) を参照してください。

2.2. HCI 環境におけるハイパーコンバージドノードのスケールダウン

HCI 環境のハイパーコンバージドノードをスケールダウンするには、HCI ノード上の Ceph OSD サービスをリバランスし、HCI ノードからインスタンスを移行し、オーバークラウドからコンピュータードを削除する必要があります。

手順

1. HCI ノード上の Ceph OSD サービスを無効にして、リバランスします。HCI ノードまたは Red Hat Ceph Storage ノードを削除する際に、director は Red Hat Ceph Storage クラスターを自動的にリバランスしないので、このステップが必要となります。
2. HCI ノードからインスタンスを移行します。詳細については、**インスタンス作成のための Compute サービスの設定**の [コンピュータード間の仮想マシンインスタンスの移行](#) を参照してください。
3. オーバークラウドからコンピュータードを削除します。詳しくは、[Removing Compute nodes](#) を参照してください。

付録A 追加情報

A.1. 設定ガイドンス

次の設定ガイドンスは、ハイパーコンバージドインフラストラクチャー環境を作成するためのフレームワークを提供することを目的としています。このガイドンスは、すべての Red Hat OpenStack Platform インストールに最終的な設定パラメーターを提供することを目的としたものではありません。特定の環境に適した具体的なガイドンスや提案は、[Red Hat カスタマーエクスペリエンスおよびエンゲージメントチーム](#) にお問い合わせください。

- [クラスタのサイジングとスケールアウト](#)
- [容量のプランニングとサイジング](#)

A.1.1. クラスタのサイジングとスケールアウト

[Red Hat Ceph Storage ハードウェアガイド](#) では、IOPS が最適化され、スループットが最適化され、コストと容量が最適化された Ceph 導入シナリオに関する推奨事項が提供されます。デプロイメントシナリオを最もよく表す推奨事項に従い、コンピュータワークロードをサポートするために必要な NIC、CPU、RAM を追加します。

最適で設置面積が小さい設定は、7つのノードで設定されます。環境内で IOPS が最適化されたパフォーマンスの要件があり、オールフラッシュストレージを使用している場合を除き、スループット最適化導入シナリオを使用する必要があります。

3 ノード Ceph Storage クラスタ設定が可能です。この設定では、次のことを行う必要があります。

- オールフラッシュストレージを使用します。
- **ceph.conf** ファイルで、**replica_count** パラメーターを 3 に設定します。
- **ceph.conf** ファイルで **min_size** パラメーターを 2 に設定します。

この設定でノードがサービスを終了しても、IOPS は継続します。データの 3 つのコピーを保持するために、3 番目のノードへのレプリケーションは、サービスに戻るまでキューに入れられます。その後、データは 3 番目のノードにバックフィルされます。



注記

最大 64 ノードの HCI 設定がテストされています。一部の HCI 環境の例は、最大 128 ノードまで文書化されています。このような大規模なクラスタについては、サポート例外およびコンサルティングサービスの契約を検討できます。ガイドンスについては、[Red Hat Customer Experience and Engagement チーム](#) にお問い合わせください。

2 つの NUMA ノードを含むデプロイメントでは、1 つの NUMA ノードでレイテンシーの影響を受けやすいコンピューティングワークロードをホストし、もう 1 つの NUMA ノードで Ceph OSD サービスをホストできます。両方のノードにネットワークインターフェイスがあり、ディスクコントローラーがノード 0 にある場合は、ストレージネットワークにノード 0 のネットワークインターフェイスを使用し、ノード 0 で Ceph OSD ワークロードをホストします。ノード 1 でコンピューティングワークロードをホストし、ノード 1 のネットワークインターフェイスを使用するように設定します。導入用のハードウェアを購入するときは、どの NIC がどのノードを使用するかに注意し、ストレージとワークロードに分割するようにしてください。

A.1.2. 容量のプランニングとサイジング

[Red Hat Ceph Storage ハードウェアガイド](#) で定義されているスループット最適化 Ceph ソリューションは、IOPS の最適化を必要としないほとんどのデプロイメントにバランスの取れたソリューションを提供します。環境を作成するときは、ソリューションで提供される設定ガイドラインに加えて、次の点に注意してください。

- OSD ごとに 5 GB の RAM が割り当てられるため、OSD には十分な動作メモリーが確保されます。ハードウェアがこの要件をサポートできることを確認してください。
- CPU の速度は、使用している記憶媒体と一致する必要があります。SSD などの高速ストレージメディアの利点は、CPU が遅すぎてサポートできない場合に無効になる可能性があります。同様に、高速な CPU は、より高速な記憶媒体によってより効率的に使用できます。CPU とストレージの媒体速度のバランスをとり、どちらか一方が他方のボトルネックにならないようにします。