Red Hat OpenStack Platform 17.1

16.2 から 17.1 へのアップグレードフレームワーク

Red Hat OpenStack Platform 16.2 から 17.1 へのインプレースアップグレード

Last Updated: 2024-06-12

Red Hat OpenStack Platform 17.1 16.2 から 17.1 へのアップグレードフレームワーク

Red Hat OpenStack Platform 16.2 から 17.1 へのインプレースアップグレード

OpenStack Team
rhos-docs@redhat.com

法律上の通知

概要

本ガイドでは、ロングライフバージョン間のインプレースアップグレードのフレームワークについて説明します。このフレームワークには、OpenStack Platform 環境をあるロングライフバージョンから次のロングライフバージョンにアップグレードするためのツールが含まれます。このドキュメントでは、OpenStack Platform 16.2 (Train) から 17.1 (Wallaby) へのアップグレードに重点を置いています。

目次

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、用語の置き換えは、今後の複数のリリースにわたって段階的に実施されます。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 RED HAT OPENSTACK PLATFORM のアップグレードフレームワークについて

アップグレード用の Red Hat OpenStack Platform (RHOSP) フレームワークは、RHOSP 環境を 1 つの長期バージョンから次の長期バージョンにアップグレードするためのワークフローです。このワークフローはインプレースのソリューションで、アップグレードは既存の環境内で実行されます。

1.1. RED HAT OPENSTACK PLATFORM 17.1 における変更点の概要

以下は、Red Hat OpenStack Platform (RHOSP) 17.1 へのアップグレード時に行われる変更の概要です。

- RHOSP のアップグレードとオペレーティングシステムのアップグレードは、2 つの異なるフェーズに分かれています。RHOSP を最初にアップグレードしてから、オペレーティングシステムをアップグレードします。
- コンピュートノードの一部を RHEL 9.2 にアップグレードし、残りのコンピュートノードは RHEL 8.4 のままにすることができます。これは Multi-RHEL 環境と呼ばれます。
- Red Hat Ceph Storage 5 へのアップグレードにより、**cephadm** が Red Hat Ceph Storage を管理するようになりました。Red Hat Ceph Storage の以前のバージョンは、**ceph-ansible** によって管理されていました。RHOSP 17.1 へのアップグレードの完了後、Red Hat Ceph Storage ノードをバージョン 5 からバージョン 6 にアップグレードできます。アップグレードしない場合、Red Hat Ceph Storage ノードは、Red Hat Ceph Storage 5 ライフサイクルが終了するまで RHOSP 17.1 でバージョン 5 のままにすることができます。Red Hat Ceph Storage バージョン 5 からバージョン 6 にアップグレードするには、ご使用の環境に応じて次のいずれかの手順を実行します。
  - director でデプロイされた Red Hat Ceph Storage 環境: **cephadm** クライアントの更新
  - 外部 Red Hat Ceph Storage クラスター環境: Red Hat Ceph Storage コンテナーイメージの更新
- デフォルトでは、RHOSP オーバークラウドは、バージョン 16.2 および 17.1 のデフォルトの ML2 メカニズムドライバーとして Open Virtual Network (OVN) を使用します。
  RHOSP 16.2 デプロイメントで OVS メカニズムドライバーを使用している場合は、OVS メカニズムドライバーを使用して 17.1 にアップグレードする必要があります。アップグレード中にメカニズムドライバーを変更しないでください。アップグレード後、OVS から OVN メカニズムドライバーに移行できます。OVN メカニズムドライバーへの移行 を参照してください。
- ML2/OVN デプロイメントでは、ハードウェアオフロードポートの egress 最小帯域幅および最大帯域幅のポリシーを有効にできます。
  詳細は、**Red Hat OpenStack Platform ネットワークの設定** の QoS ポリシーのネットワークサービスの設定 を参照してください。
- アンダークラウドおよびオーバークラウドは、共に Red Hat Enterprise Linux 9 上で動作します。

1.2. RED HAT ENTERPRISE LINUX 9 の変更点

Red Hat OpenStack Platform (RHOSP) 17.1 は、ベースオペレーティングシステムとして Red Hat Enterprise Linux (RHEL) 9.2 を使用します。アップグレードプロセスの一環として、ノードのベースオペレーティングシステムを RHEL 9.2 にアップグレードします。

アップグレードを開始する前に、以下の情報を確認して RHEL 9 についてよく理解してください。

- システムに RSA/SHA-1 署名付きのパッケージが含まれている場合は、RHOSP 17.1 にアップグレードする前に、それらを削除するか、ベンダーに連絡して RSA/SHA-256 署名付きのパッケージを入手する必要があります。詳細は、SHA-1 deprecation in Red Hat Enterprise Linux 9 を参照してください。
- RHEL 9 の最新の変更点に関する詳細は、 Red Hat Enterprise Linux 9.2 リリースノート を参照してください。
- Red Hat Enterprise Linux 8 と 9 の主な相違点は、RHEL 9 の採用における考慮事項 を参照してください。
- Red Hat Enterprise Linux 9 に関する一般的な情報は、Red Hat Enterprise Linux 9 の製品ドキュメント を参照してください。
- RHEL 8 から RHEL 9 へのアップグレードに関する詳細は、RHEL 8 から RHEL 9 へのアップグレード を参照してください。

1.3. ロングライフバージョンのアップグレードフレームワーク

Red Hat OpenStack Platform (RHOSP) の **アップグレードフレームワーク** を使用して、複数のオーバークラウドバージョンを経由するインプレースアップグレードパスを実施することができます。この機能は、**ロングライフバージョン** とされている特定の OpenStack のバージョンの使用を継続し、次のロングライフバージョンが提供された際にアップグレードする機会を提供することを目的としています。

Red Hat OpenStack Platform のアップグレードプロセスでは、ノード上の Red Hat Enterprise Linux (RHEL) のバージョンもアップグレードされます。

本ガイドは、以下のバージョン間のアップグレードフレームワークを提供します。

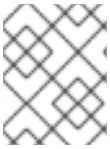
現在のバージョン	目的のバージョン
Red Hat OpenStack Platform 16.2.4 以降	Red Hat OpenStack Platform 17.1 (最新)

Red Hat OpenStack Platform のライフサイクルサポートに関する正確な日付けおよび詳細な情報は、Red Hat OpenStack Platform のライフサイクル を参照してください。

ロングライフリリースのアップグレードパス

アップグレードを開始する前に、可能な更新およびアップグレードのパスを把握してください。RHOSP 16.2.4 より前の環境を使用している場合は、メジャーバージョンからメジャーバージョンにアップグレードする前に、まずは既存の環境を最新のマイナーリリースに更新する必要があります。

たとえば、現在のデプロイメントが Red Hat Enterprise Linux (RHEL) 8.4 上の Red Hat OpenStack Platform (RHOSP) 16.2.1 である場合、RHOSP 17.1 にアップグレードする前に、最新の RHOSP 16.2 バージョンへのマイナー更新を実行する必要があります。



注記

RHOSP および RHEL の現行バージョンは、**/etc/rhosp-release** および **/etc/redhat-release** ファイルで確認できます。

表1.1 バージョンの更新パス

現行バージョン	更新後のバージョン
RHEL 8.4 / RHOSP 16.2.x	最新の RHEL 8.4 / 最新の RHOSP 16.2
RHEL 9.0 / RHOSP 17.0.x	最新の RHEL 9.0 / 最新の RHOSP 17.0
RHEL 9.0 / RHOSP 17.0.x	最新の RHEL 9.2 / 最新の RHOSP 17.1
RHEL 9.2 / RHOSP 17.1.x	最新の RHEL 9.2 / 最新の RHOSP 17.1

詳細は、Red Hat OpenStack Platform のマイナー更新の実行 を参照してください。

表1.2 バージョンのアップグレードパス

現行バージョン	更新後のバージョン
RHEL 8.4 上の RHOSP 16.2	最新の RHEL 9.2 / 最新の RHOSP 17.1

Red Hat では、お使いの環境を次のロングライフリリースにアップグレードするためのオプションを 2 つ提供しています。

インプレースアップグレード

既存の環境でサービスのアップグレードを実施します。本ガイドでは、主にこのオプションを中心に説明します。

並列移行

新しい RHOSP 17.1 環境を作成し、ワークロードを現在の環境から新しい環境に移行します。RHOSP の並行移行の詳細は、Red Hat Global Professional Services までお問い合わせください。

1.4. アップグレードの所要時間と影響

次の表に示された所要時間は、200 台のノードを備えたオーバークラウドと、それぞれ 17 個のオブジェクトストレージデーモン (OSD) を備えた 9 台の Ceph Storage ホストで構成されるテスト環境で記録されました。表に示された所要時間は、すべての実稼働環境に適用されるわけではありません。たとえば、ハードウェアのスペックが低い場合やブート時間が長い場合は、これらの時間に余裕を持たせてください。所要時間は、コンテナーおよびパッケージのコンテンツへのネットワーク I/O や、ホスト上のディスク I/O によっても異なります。

各タスクのアップグレードに要する時間を正確に推測するには、実稼働環境と類似したハードウェアを持つテスト環境でこれらの手順を実施してください。

表1.3 インプレースアップグレードの期間と影響

	所要時間	注記
アンダークラウドのアップグレード	• 30 分	• オーバークラウドは中断しません。

	所要時間	注記
オーバークラウドの導入と準備	• ベアメタルの導入に 10 分 • アップグレードの準備に 30 分	• 所要時間はオーバークラウドのサイズにより増減します。 • オーバークラウドは中断しません。
Red Hat Ceph Storage のアップグレード	• Ceph アップグレードの Ansible 実行: 合計 90 分、ノードごとに 10 分 • cephadm 導入のための Ceph ansible の実行: 合計 30 分、ノードごとに 3 分 • ceph のアップグレードと導入後に行うオーバークラウドのアップグレード準備: 20 分	• 所要時間は、ストレージホストと OSD の数により増減します。 • ストレージのパフォーマンスが低下します。
オーバークラウド OpenStack のアップグレード	• 120 分	• 所要時間はオーバークラウドのサイズにより増減します。 • このプロセス中にエージェントが再起動され、API トランザクションが失われる可能性があります。この段階では、OpenStack API へのクライアントアクセスを無効にしてください。
アンダークラウドシステムのアップグレード	• 40 分	• 複数回の再起動が含まれます。再起動時間はハードウェアにより異なります。 • SELinux の再ラベル付けが含まれます。多数のファイルがあるホストでは長時間かかります。 • オーバークラウドは中断しません。

	所要時間	注記
オーバークラウドの非 Compute ホストシステムのアップグレード	• アップグレードの準備に 30 分 • ホストシステムのアップグレードごとに 40 分	• 複数回の再起動が含まれます。再起動時間はハードウェアにより異なります。 • SELinux の再ラベル付けが含まれます。多数のファイルがあるホストでは長時間かかります。 • パフォーマンスが低下します。
オーバークラウド Compute ホストのアップグレード	• ホストのバッチごとに 40 分 • アップグレードの準備に 30 分	• Compute ノードの選択したバッチでアップグレードの準備を行います。所要時間は、各バッチ内の Compute ノードの数により異なります。停止時間はありません。 • 複数回の再起動が含まれます。再起動時間はハードウェアにより異なります。 • SELinux の再ラベル付けが含まれます。多数のファイルがあるホストでは長時間かかります。 • 再起動中にワークロードが停止することを防止するため、事前にワークロードを別のホストに移行できます。

1.5. インプレースアップグレードの計画および準備

OpenStack Platform 環境のインプレースアップグレードを実施する前に、アップグレードのプランを作成し、正常なアップグレードを妨げる可能性のある障害に対処してください。

1.5.1. Red Hat OpenStack Platform 17.1 の理解

アップグレードを実行する前に、Red Hat OpenStack Platform 17.1 をよく理解しておくことで、結果として生じる環境や、アップグレードに影響を与える可能性のあるバージョン間の変更点を理解することができます。Red Hat OpenStack Platform 17.1 の理解を深めるには、以下の推奨事項に従ってください。

- アップグレードパスにわたるすべてのバージョンのリリースノートを確認し、計画が必要になる可能性のある要素を識別します。
    - 新しい機能が含まれるコンポーネント

- 既知の問題

以下のリンクから、各バージョンのリリースノートを確認してください。

- Red Hat OpenStack Platform 16.2 (ソースバージョン)
- Red Hat OpenStack Platform 17.1 (ターゲットバージョン)

- バージョン 17.1 の director を使用した Red Hat OpenStack Platform のインストールと管理 ガイドを読み、このガイドに記載されている新しい要件とプロセスについて理解を深めてください。
- 概念実証用の Red Hat OpenStack Platform 17.1 アンダークラウドとオーバークラウドをインストールします。対象のバージョンの OpenStack Platform を実際に操作して経験を積み、対象のバージョンと現在のバージョンの違いを調査します。

1.5.2. マイナーバージョンの更新要件

Red Hat OpenStack Platform (RHOSP) 16.2 から 17.1 にアップグレードするには、使用している環境で RHOSP バージョン 16.2.4 以降が実行されている必要があります。16.2.4 より前の RHOSP バージョンを使用している場合は、環境を現行リリースの最新のマイナーバージョンに更新します。たとえば、Red Hat OpenStack Platform 17.1 にアップグレードする前に、Red Hat OpenStack Platform 16.2.3 環境を 16.2 の最新バージョンに更新します。

Red Hat OpenStack Platform 16.2 のマイナーバージョン更新を実行する手順は、Red Hat OpenStack Platform の最新状態の維持 を参照してください。

1.5.3. Red Hat OpenStack Platform での Leapp アップグレードの使用

ロングライフバージョンの Red Hat OpenStack Platform のアップグレードでは、ベースオペレーティングシステムを Red Hat Enterprise Linux (RHEL) 8.4 から RHEL 9.2 へアップグレードする必要があります。アップグレードプロセスでは、Leapp ユーティリティーを使用して RHEL 9.2 へのアップグレードを実行します。ただし、Leapp アップグレードの一部は、明確に RHEL 8.4 から RHEL 9.2 にアップグレードするようにカスタマイズされています。オペレーティングシステムを RHEL 9.2 にアップグレードするには、アンダークラウドシステムのアップグレードの実行 を参照してください。

制限事項

アップグレードに影響を及ぼす可能性のある制限の詳細は、**RHEL 8 から RHEL 9 へのアップグレード** の以下のセクションを参照してください。

- アップグレードの計画
- 既知の問題

既知の制限がお使いの環境に影響を及ぼす場合は、Red Hat Technical Support Team にアドバイスを求めてください。

トラブルシューティング

Leapp が原因と考えられる問題のトラブルシューティングに関する詳細は、**RHEL 8 から RHEL 9 へのアップグレード** の トラブルシューティング を参照してください。

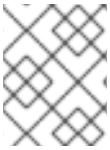
1.5.4. ストレージドライバーの認定

アップグレードする前に、デプロイされているストレージドライバーが Red Hat OpenStack Platform 17.1 で使用できると認定されていることを確認してください。

Red Hat OpenStack Platform 17.1 での使用が認定されたソフトウェアについては、Software certified for Red Hat OpenStack Platform 17 を参照してください。

1.5.5. サポート対象のアップグレードシナリオ

アップグレードを進める前に、ご自分のオーバークラウドがサポート対象であることを確認してください。



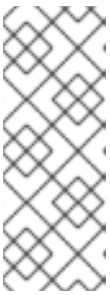
注記

以下のリストに記載されていない特定のシナリオがサポート対象かどうか不明な場合は、Red Hat Technical Support Team にアドバイスを求めてください。

サポート対象のシナリオ

以下のインプレースアップグレードシナリオは、テスト済みでサポートの対象です。

- デフォルトのロール種別を使用する標準環境: Controller、Compute、および Ceph Storage OSD
- 分割 Controller コンポーザブルロール
- Ceph Storage コンポーザブルロール
- ハイパーコンバージドインフラストラクチャー: 同一ノード上の Compute サービスおよび Ceph Storage OSD サービス
- ネットワーク機能仮想化 (NFV) 技術を使用する環境: Single-root Input/Output Virtualization (SR-IOV) および Data Plane Development Kit (DPDK)
- インスタンス HA が有効な環境



注記

アップグレードの際には、nova ライブマイグレーションがサポートされます。ただし、インスタンス HA が開始する避難はサポートされません。Compute ノードをアップグレードする場合、ノードはクリーンにシャットダウンされ、ノード上で実行されているワークロードはインスタンス HA によって自動的に退避されません。その代わり、手動でライブマイグレーションを行う必要があります。

テクノロジープレビューのシナリオ

アップグレードフレームワークを以下の機能と併用した場合は、テクノロジープレビューとみなされます。したがって、Red Hat では全面的にはサポートしていません。以下のシナリオは概念実証の環境でのみテストし、実稼働環境へのアップグレードは行わないでください。テクノロジープレビュー機能についての詳細は、対象範囲の詳細 を参照してください。

- エッジおよび分散コンピュートノード (DCN) のシナリオ

1.5.6. Red Hat Virtualization のアップグレードプロセス

コントロールプレーンを Red Hat Virtualization で実行している場合、Red Hat OpenStack Platform (RHOSP) のアップグレードプロセスには影響はありません。RHOSP のアップグレードは、環境が Red Hat Virtualization 上で実行しているかどうかに関係なく同じです。

1.5.7. アップグレードを妨げる可能性のある既知の問題

アップグレードの正常な完了に影響を及ぼす可能性のある、以下の既知の問題を確認してください。

BZ#2224085 - Leapp is stuck in Interim System when - -debug is specified

オペレーティングシステムを RHEL 7.x から RHEL 8.x に、または RHEL 8.x から RHEL 9.x にアップグレードする場合は、**--debug** オプションを指定して Leapp アップグレードを実行しないでください。システムは **early console in setup code** 状態のままとなり、自動的に再起動しません。この問題を回避するために、**UpgradeLeappDebug** パラメーターはデフォルトで **false** に設定されています。テンプレートではこの値を変更しないでください。

BZ#2203785 - Collectd sensubility stops working after overcloud node was rebooted.

オーバークラウドノードをリブートした後、パーミッションの問題により、collectd-sensubility が機能しなくなります。その結果、collectd-sensubility はコンテナーの健全性のレポートを停止します。Red Hat OpenStack Platform (RHOSP) 16.2 から RHOSP 17.1 へのアップグレード中に、Leapp アップグレードの一部としてオーバークラウドノードが再起動されます。collectd-sensubility が引き続き機能することを確認するには、以下のコマンドを実行します。

```
sudo podman exec -it collectd setfacl -R -m u:collectd:rwx /run/podman
```

BZ#2180542 - After upgrade, manila ceph-nfs fails to start with error: ceph-nfs start on leaf1-controller-0 returned 'error' because 'failed'

Pacemaker によって制御される **ceph-nfs** リソースには、一部のプロセスデータを保存するためのランタイムディレクトリーが必要です。このディレクトリーは、RHOSP をインストールまたはアップグレードするときに作成されます。現在、コントローラーノードを再起動するとディレクトリーが削除され、コントローラーノードを再起動しても **ceph-nfs** サービスは回復しません。すべてのコントローラーノードが再起動されると、**ceph-nfs** サービスは永続的に失敗します。

以下の回避策を適用できます。

1. コントローラーノードを再起動する場合は、コントローラーノードにログインし、**/var/run/ceph** ディレクトリーを作成します。
   $ mkdir -p /var/run/ceph

2. 再起動されたすべてのコントローラーノードでこの手順を繰り返します。**ceph-nfs-pacemaker** サービスが失敗としてマークされている場合は、ディレクトリーを作成した後、いずれかのコントローラーノードから以下のコマンドを実行します。
   $ pcs resource cleanup

BZ#2210873 - assimilate_{{ tripleo_cephadm_cluster }}.conf required if--crush-hierarchy is used

CephPools パラメーターが一連のプールオーバーライドを使用して定義されている場合は、RHOSP 16.2 から 17.1 へのアップグレード中にプールの作成で失敗が発生しないように、**rule_name:replicated_rule** を定義に追加する必要があります。

BZ#2245602 - Upgrade (OSP16.2 →OSP17.1) controller-0 does not perform leapp upgrade due to packages missing ovn2.15 openvswitch2.15

Red Hat OpenStack Platform (RHOSP) 13 から 16.1 または 16.2 にアップグレードする場合、または RHOSP 16.2 から 17.1 にアップグレードする場合は、**--answers-fileanswer-upgrade.yaml** ファイルに **system_upgrade.yaml** ファイルを含めないでください。そのファイルに **system_upgrade.yaml** ファイルが含まれていると、**environments/lifecycle/upgrade-prepare.yaml** ファイルによって

system_upgrade.yaml ファイル内のパラメーターが上書きされます。この問題を回避するには、**system_upgrade.yaml** ファイルを **openstack overcloud upgrade prepare** コマンドに追加します。以下に例を示します。

```
$ openstack overcloud upgrade prepare --answers-file answer-upgrade.yaml /
-r roles-data.yaml /
-n networking-data.yaml /
-e system_upgrade.yaml /
-e upgrade_environment.yaml /
```

この回避策を使用すると、**system_upgrade.yaml** ファイルで設定されているパラメーターによって 、**environments/lifecycle/upgrade-prepare.yaml** ファイルのデフォルトのパラメーターが上書きされます。

BZ#2246409 - (OSP16.2→17.1) Cinder volume NFS mounts on compute nodes are preventing leapp upgrade

Cinder ボリューム NFS マウントがコンピュートノード上に存在する場合、RHOSP 16.2 から 17.1 へのアップグレード中に、オペレーティングシステムの RHEL 8.4 から RHEL 9.2 へのアップグレードが失敗します。回避策については、Red Hat サポート担当者にお問い合わせください。

BZ#2266025 - FFU 16to17. System upgrade process is interrupted after undercloud reboot if OS was created from default RHEL 8.4 image

Red Hat Enterprise Linux (RHEL) 8.4 イメージの **/etc/default/grub** ファイルに **"GRUB_DEFAULT=saved"** 定義がないという問題があります。アンダークラウドをデプロイするために Red Hat カスタマーポータルから RHEL 8.4 KVM ゲストイメージをダウンロードし、Red Hat OpenStack Platform 16.2 から 17.1 にアップグレードすると、次の問題が発生します。

- システムのアップグレードで grub メニューが適切に更新されません。
- システムの再起動後、director はノード上で RHEL 9 ではなく RHEL 8 を起動します。

この問題の回避策は、Red Hat ナレッジベースのソリューション FFU 16to17 System upgrade process is interrupted after undercloud reboot を参照してください。

BZ#2277756 - rolling update fails unless mon_mds_skip_sanity=true is set

Red Hat Ceph Storage 4 から 5 へのアップグレード中に、既知の問題により Ceph Monitor ノードがアップグレードされません。最初の Ceph Monitor ノードがバージョン 5 にアップグレードされると、他の Ceph Monitor ノードは実行を停止し、次のメッセージを報告します。

```
"FAILED ceph_assert(fs->mds_map.compat.compare(compat) == 0)"
```

Red Hat Ceph Storage ノードをアップグレードする前に、Red Hat ナレッジベースソリューション RHCS during upgrade RHCS 4 → RHCS 5 ceph-mon is failing with "FAILED ceph_assert(fs→mds_map.compat.compare(compat) == 0)" の回避策を適用してください。アップグレードが完了すると、Red Hat Ceph Storage クラスターは **cephadm** によって採用されるため、この回避策は必要ありません。

BZ#2259891 - During upgrade 16.2-17.1 with not internet on UC overcloud_upgrade_prepare.sh fails pulling registry.access.redhat.com/ubi8/pause

アンダークラウドがインターネットに接続していない環境では、**infra_image** 値が定義されていないため、Red Hat OpenStack Platform 16.2 から 17.1 へのアップグレードは失敗しま

す。**overcloud_upgrade_prepare.sh** スクリプトは **registry.access.redhat.com/ubi8/pause** をプルしようとしますが、エラーが発生します。

この問題を回避するには、Satellite Server に一時停止コンテナーを手動で追加します。

1. 一時停止コンテナーを Satellite Server にインポートします (例: **k8s.gcr.io/pause:3.5** または **registry.access.redhat.com/ubi8/pause**)。
2. **/usr/share/containers/containers.conf** ファイルで、ローカル Satellite URL の一時停止コンテナーを指定します。以下に例を示します。

```
infra_image="<LOCAL_SATELLITE_URL/pause:3.5>"
```

   - **<LOCAL_SATELLITE_URL/pause:3.5>** を、ローカルの Satellite URL とインポートした一時停止コンテナーに置き換えます。

3. Pod を起動できることを確認します。

```
$ podman pod create
```

BZ#2264543 - (FFU 16.2→17) leapp upgrade of ceph nodes is failing encrypted partition detected

Red Hat OpenStack Platform (RHOSP) 16.2 から RHOSP 17.1 にアップグレードすると、暗号化された **ceph-osd** が原因で、Red Hat Ceph Storage ノードの Leapp アップグレードが失敗します。Red Hat Ceph Storage ノードで Leapp アップグレードを実行する前に、Red Hat ナレッジベースソリューション (FFU 16.2→17) leapp upgrade of ceph nodes is failing encrypted partition detected を適用します。

BZ#2275097 - bridge_name が osp 17.1 では br-ex に変換されない (16.2 からの ffu)

bridge_name 変数は、RHOSP 17.1 の nic-config テンプレートでは有効ではなくなりました。RHOSP 16.2 から 17.1 にアップグレードした後、スタック更新を実行し、nic-config テンプレートに **bridge_name** 変数がまだ含まれていると、停止が発生します。RHOSP 17.1 にアップグレードする前に、**bridge_name** 変数の名前を変更する必要があります。

詳細は、Red Hat ナレッジベースソリューション **bridge_name** is still present in templates during and post FFU causing further updates failure を参照してください。

BZ#2269009 - cephadm の導入後、alertmanager がデプロイされていると haproxy が起動に失敗する

director によってデプロイされた Red Hat Ceph Storage 環境に Alertmanager をデプロイした場合、Red Hat Ceph Storage バージョン 4 からバージョン 5 へのアップグレードは失敗します。Red Hat Ceph Storage ノードで **cephadm** を設定するために次のコマンドを実行した後、HAProxy が再起動しないことが原因で失敗します。

```
$ openstack overcloud external-upgrade run \
--skip-tags ceph_ansible_remote_tmp \
--stack <stack> \
--tags cephadm_adopt  2>&1
```

コマンドを実行すると、Red Hat Ceph Storage クラスターのステータスは **HEALTH_WARN** になります。

この問題の回避策については、Red Hat ナレッジベースソリューション HAProxy does not restart during RHOSP upgrade when RHCS is director-deployed and Alertmanager is enabled を参照してください。

1.5.8. バックアップおよび復元

Red Hat OpenStack Platform (RHOSP) 16.2 環境をアップグレードする前に、次のオプションのいずれかを使用して、アンダークラウドおよびオーバークラウドのコントロールプレーンをバックアップします。

- アップグレードを実施する前にノードをバックアップします。アップグレード前のノードのバックアップに関する詳細は、**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** を参照してください。
- アンダークラウドのアップグレードの実行後、かつオーバークラウドのアップグレードの実行前に、アンダークラウドノードをバックアップします。アンダークラウドのバックアップに関する詳細は、**Red Hat OpenStack Platform 17.1 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** の 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。
- 使用中の環境に合ったサードパーティー製のバックアップおよび復元ツールを使用してください。認定済みのバックアップツールおよびリカバリーツールに関する詳細は、Red Hat Ecosystem Catalog を参照してください。

1.5.9. プロキシー設定

Red Hat OpenStack Platform 16.2 環境でプロキシーを使用する場合、**/etc/environment** ファイル内のプロキシー設定は、オペレーティングシステムのアップグレードおよび Red Hat OpenStack Platform 17.1 のアップグレード後も存続します。

- Red Hat OpenStack Platform 16.2 のプロキシー設定に関する詳細は、**director のインストールと使用方法** の プロキシーを使用してアンダークラウドを実行する際の考慮事項 を参照してください。
- Red Hat OpenStack Platform 17.1 のプロキシー設定に関する詳細は、**director を使用した Red Hat OpenStack Platform のインストールおよび管理** の プロキシーを使用してアンダークラウドを実行する際の考慮事項 を参照してください。

1.5.10. コンピュートノードのアップグレード計画

コンピュートノードを Red Hat OpenStack Platform (RHOSP) 16.2 から RHOSP 17.1 にアップグレードした後、以下のオプションのいずれかを選択し、コンピュートホストオペレーティングシステムをアップグレードできます。

- コンピュートノードの一部を Red Hat Enterprise Linux (RHEL) 8.4 上に残し、残りを RHEL 9.2 にアップグレードします。これは Multi-RHEL 環境と呼ばれます。
- すべてのコンピュートノードを RHEL 9.2 にアップグレードし、環境のアップグレードを完了します。
- すべてのコンピュートノードを RHEL 8.4 上に保持します。RHEL 8.4 のライフサイクルが適用されます。

Multi-RHEL 環境の利点

vTPM やセキュアブートなど、RHOSP 17.1 でのみサポートされているハードウェア関連機能を利用する

には、すべてのコンピュートノードを RHEL 9.2 にアップグレードする必要があります。ただし、コンピュートノードの一部またはすべてを RHEL 8.4 上に残すことが必要な場合があります。たとえば、アプリケーションを RHEL 8 用に認定した場合は、アップグレード全体をブロックすることなく、コンピュートノードを RHEL 8.4 上で実行し続けてアプリケーションをサポートできます。

コンピュートノードの一部を RHEL 9.2 にアップグレードするオプションを使用すると、アップグレードプロセスをより詳細に制御できるようになります。計画されたメンテナンス期間内で RHOSP 環境のアップグレードを優先し、オペレーティングシステムのアップグレードを別の時期に延期することができます。必要なダウンタイムが少なくなり、エンドユーザーへの影響が最小限に抑えられます。

注記

RHOSP 17.1 から RHOSP 18.0 にアップグレードする場合は、すべてのホストを RHEL 9.2 にアップグレードする必要があります。延長ライフサイクルサポートフェーズを過ぎてもホスト上で RHEL 8.4 を実行し続ける場合は、TUS サブスクリプションを取得する必要があります。

Multi-RHEL 環境の制限事項

Multi-RHEL 環境には以下の制限が適用されます。

- RHEL 8 を実行しているコンピュートノードは、NVMe-over-TCP Cinder ボリュームを消費できません。
- RHOSP 16.2 と 17.1 では、collectd モニタリング用のソケットファイルに異なるパスを使用できません。
- ML2/OVN メカニズムドライバーと ML2/OVS メカニズムドライバーを混在させることはできません。たとえば、RHOSP 16.2 デプロイメントに ML2/OVN が含まれている場合、Multi-RHEL 環境では ML2/OVN を使用する必要があります。
- FIPS は Multi-RHEL 環境ではサポートされません。FIPS のデプロイメントは Day 1 操作です。FIPS は RHOSP 16.2 ではサポートされていません。その結果、RHOSP 16.2 から 17.1 にアップグレードすると、17.1 環境には FIPS が含まれません。
- Edge トポロジーは現在サポートされていません。

重要

サポートされているハイパーコンバージドインフラストラクチャー環境内のすべての HCI ノードは、Red Hat OpenStack Platform コントローラーで使用されるバージョンと同じバージョンの Red Hat Enterprise Linux を使用する必要があります。同じハイパーコンバージドインフラストラクチャー環境内の HCI ノード上で複数の Red Hat Enterprise バージョンをハイブリッド状態で使用する場合は、サポート例外について Red Hat Customer Experience and Engagement チームにご相談ください。

Compute ノードのアップグレード

以下のいずれかのオプションを使用して、コンピュートノードをアップグレードします。

- コンピュートノードの Multi-RHEL アップグレードを実行するには、コンピュートノードの Multi-RHEL 環境へのアップグレード を参照してください。
- すべてのコンピュートノードを RHEL 9.2 にアップグレードするには、コンピュートノードの RHEL 9.2 へのアップグレード を参照してください。

- すべてのコンピュートノードを RHEL 8.4 上に維持する場合、追加の設定は必要ありません。

1.6. リポジトリー

本項では、アンダークラウドおよびオーバークラウドのリポジトリーについて説明します。特定の状況において、リポジトリーを有効にする必要がある場合は、本項を参照してください。

- Red Hat カスタマーポータルに登録する際にリポジトリーを有効にする。
- リポジトリーを有効にして Red Hat Satellite Server に同期させる。
- Red Hat Satellite Server に登録する際にリポジトリーを有効にする。

1.6.1. アンダークラウドのリポジトリー

Red Hat Enterprise Linux (RHEL) 9.2 で Red Hat OpenStack Platform (RHOSP) 17.1 を実行します。RHOSP 16.2 からアップグレードする場合、RHEL 8.4 コンピュートノードは Multi-RHEL 環境でもサポートされます。

注記

リポジトリーを Red Hat Satellite と同期する場合は、特定バージョンの Red Hat Enterprise Linux リポジトリーを有効にすることができます。ただし、選択したバージョンに関係なく、リポジトリーは同じままです。たとえば、BaseOS リポジトリーの 9.2 バージョンを有効にすることができますが、リポジトリー名は、選択した特定のバージョンに関係なく **rhel-9-for-x86_64-baseos-eus-rpms** のままになります。

警告

ここで指定されたもの以外のリポジトリーはサポートされていません。別途推奨されない限り、以下の表に記載されている以外の製品またはリポジトリーを有効にしないでください。有効にすると、パッケージの依存関係の問題が発生する可能性があります。Extra Packages for Enterprise Linux (EPEL) を有効にしないでください。

コアリポジトリー

以下の表には、アンダークラウドをインストールするためのコアリポジトリーをまとめています。

名前	リポジトリー	要件の説明
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs) Extended Update Support (EUS)	**rhel-9-for-x86_64-baseos-eus-rpms**	x86_64 システム用ベースオペレーティングシステムのリポジトリー
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)	**rhel-9-for-x86_64-appstream-eus-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。

名前	リポジトリー	要件の説明
Red Hat Enterprise Linux 9 for x86_64 - High Availability (RPMs) Extended Update Support (EUS)	**rhel-9-for-x86_64-highavailability-eus-rpms**	Red Hat Enterprise Linux の高可用性ツール。Controller ノードの高可用性に使用します。
Red Hat OpenStack Platform for RHEL 9 (RPMs)	**openstack-17.1-for-rhel-9-x86_64-rpms**	Red Hat OpenStack Platform のコアリポジトリー。Red Hat OpenStack Platform director のパッケージが含まれます。
Red Hat Fast Datapath for RHEL 9 (RPMS)	**fast-datapath-for-rhel-9-x86_64-rpms**	OpenStack Platform 用 Open vSwitch (OVS) パッケージを提供します。
Red Hat Enterprise Linux 8.4 for x86_64 - BaseOS (RPMs) Telecommunications Update Service (TUS)	**rhel-8-for-x86_64-baseos-tus-rpms**	x86_64 システム用ベースオペレーティングシステムのリポジトリー
Red Hat Enterprise Linux 8.4 for x86_64 - AppStream (RPMs)	**rhel-8-for-x86_64-appstream-tus-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat OpenStack Platform for RHEL 8 x86_64 (RPMs)	**openstack-17.1-for-rhel-8-x86_64-rpms**	Red Hat OpenStack Platform のコアリポジトリー

1.6.2. オーバークラウドのリポジトリー

Red Hat Enterprise Linux (RHEL) 9.2 で Red Hat OpenStack Platform (RHOSP) 17.1 を実行します。RHOSP 16.2 からアップグレードする場合、RHEL 8.4 コンピュートノードは Multi-RHEL 環境でもサポートされます。

注記

リポジトリーを Red Hat Satellite と同期する場合は、特定バージョンの Red Hat Enterprise Linux リポジトリーを有効にすることができます。ただし、選択したバージョンに関係なく、リポジトリーは同じままです。たとえば、BaseOS リポジトリーの 9.2 バージョンを有効にすることができますが、リポジトリー名は、選択した特定のバージョンに関係なく **rhel-9-for-x86_64-baseos-eus-rpms** のままになります。

警告

ここで指定されたもの以外のリポジトリーはサポートされていません。別途推奨されない限り、以下の表に記載されている以外の製品またはリポジトリーを有効にしないでください。有効にすると、パッケージの依存関係の問題が発生する可能性があります。Extra Packages for Enterprise Linux (EPEL) を有効にしないでください。

Controller ノード用リポジトリー

以下の表には、オーバークラウドの Controller ノード用コアリポジトリーをまとめています。

名前	リポジトリー	要件の説明
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs) Extended Update Support (EUS)	**rhel-9-for-x86_64-baseos-eus-rpms**	x86_64 システム用ベースオペレーティングシステムのリポジトリー
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)	**rhel-9-for-x86_64-appstream-eus-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat Enterprise Linux 9 for x86_64 - High Availability (RPMs) Extended Update Support (EUS)	**rhel-9-for-x86_64-highavailability-eus-rpms**	Red Hat Enterprise Linux の高可用性ツール。
Red Hat OpenStack Platform for RHEL 9 x86_64 (RPMs)	**openstack-17.1-for-rhel-9-x86_64-rpms**	Red Hat OpenStack Platform のコアリポジトリー
Red Hat Fast Datapath for RHEL 9 (RPMS)	**fast-datapath-for-rhel-9-x86_64-rpms**	OpenStack Platform 用 Open vSwitch (OVS) パッケージを提供します。
Red Hat Ceph Storage Tools 6 for RHEL 9 x86_64 (RPMs)	**rhceph-6-tools-for-rhel-9-x86_64-rpms**	Red Hat Ceph Storage 6 for Red Hat Enterprise Linux 9 のツール
Red Hat Enterprise Linux 8.4 for x86_64 - BaseOS (RPMs) Telecommunications Update Service (TUS)	**rhel-8-for-x86_64-baseos-tus-rpms**	x86_64 システム用ベースオペレーティングシステムのリポジトリー
Red Hat Enterprise Linux 8.4 for x86_64 - AppStream (RPMs)	**rhel-8-for-x86_64-appstream-tus-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat Enterprise Linux 8 for x86_64 - High Availability (RPMs) Telecommunications Update Service (TUS)	**rhel-8-for-x86_64-highavailability-tus-rpms**	Red Hat Enterprise Linux の高可用性ツール。

名前	リポジトリー	要件の説明
Red Hat OpenStack Platform for RHEL 8 x86_64 (RPMs)	**openstack-17.1-for-rhel-8-x86_64-rpms**	Red Hat OpenStack Platform のコアリポジトリー

Compute ノードおよび ComputeHCI ノードのリポジトリー

以下の表に、オーバークラウド内の Compute ノードおよび ComputeHCI ノードのコアリポジトリーを示します。

名前	リポジトリー	要件の説明
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs) Extended Update Support (EUS)	**rhel-9-for-x86_64-baseos-eus-rpms**	x86_64 システム用ベースオペレーティングシステムのリポジトリー
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)	**rhel-9-for-x86_64-appstream-eus-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat Enterprise Linux 9 for x86_64 - High Availability (RPMs) Extended Update Support (EUS)	**rhel-9-for-x86_64-highavailability-eus-rpms**	Red Hat Enterprise Linux の高可用性ツール。
Red Hat OpenStack Platform for RHEL 9 x86_64 (RPMs)	**openstack-17.1-for-rhel-9-x86_64-rpms**	Red Hat OpenStack Platform のコアリポジトリー
Red Hat Fast Datapath for RHEL 9 (RPMS)	**fast-datapath-for-rhel-9-x86_64-rpms**	OpenStack Platform 用 Open vSwitch (OVS) パッケージを提供します。
Red Hat Ceph Storage Tools 6 for RHEL 9 x86_64 (RPMs)	**rhceph-6-tools-for-rhel-9-x86_64-rpms**	Red Hat Ceph Storage 6 for Red Hat Enterprise Linux 9 のツール
Red Hat Enterprise Linux 8.4 for x86_64 - BaseOS (RPMs) Telecommunications Update Service (TUS)	**rhel-8-for-x86_64-baseos-tus-rpms**	x86_64 システム用ベースオペレーティングシステムのリポジトリー
Red Hat Enterprise Linux 8.4 for x86_64 - AppStream (RPMs)	**rhel-8-for-x86_64-appstream-tus-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat OpenStack Platform for RHEL 8 x86_64 (RPMs)	**openstack-17.1-for-rhel-8-x86_64-rpms**	Red Hat OpenStack Platform のコアリポジトリー

Ceph Storage ノード用リポジトリー

以下の表には、オーバークラウド用の Ceph Storage 関連のリポジトリーをまとめています。

名前	リポジトリー	要件の説明
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)	**rhel-9-for-x86_64-baseos-rpms**	x86_64 システム用ベースオペレーティングシステムのリポジトリー
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)	**rhel-9-for-x86_64-appstream-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat OpenStack Platform Deployment Tools for RHEL 9 x86_64 (RPMs)	**openstack-17.1-deployment-tools-for-rhel-9-x86_64-rpms**	director が Ceph Storage ノードを設定するのに役立つパッケージ。このリポジトリーは、スタンドアロンの Ceph Storage サブスクリプションに含まれています。OpenStack Platform と Ceph Storage サブスクリプションを組み合わせて使用する場合は、**openstack-17.1-for-rhel-9-x86_64-rpms** リポジトリーを使用します。
Red Hat OpenStack Platform for RHEL 9 x86_64 (RPMs)	**openstack-17.1-for-rhel-9-x86_64-rpms**	director が Ceph Storage ノードを設定するのに役立つパッケージ。このリポジトリーは、Red Hat OpenStack Platform と Red Hat Ceph Storage を組み合わせたサブスクリプションに含まれています。スタンドアロンの Red Hat Ceph Storage サブスクリプションを使用する場合は、**openstack-17.1-deployment-tools-for-rhel-9-x86_64-rpms** リポジトリーを使用します。
Red Hat Ceph Storage Tools 6 for RHEL 9 x86_64 (RPMs)	**rhceph-6-tools-for-rhel-9-x86_64-rpms**	Ceph Storage クラスターと通信するためのノード用のツールを提供します。
Red Hat Fast Datapath for RHEL 9 (RPMS)	**fast-datapath-for-rhel-9-x86_64-rpms**	OpenStack Platform 用 Open vSwitch (OVS) パッケージを提供します。Ceph Storage ノードで OVS を使用している場合は、このリポジトリーをネットワークインターフェイス設定 (NIC) テンプレートに追加します。

1.6.3. Red Hat Satellite Server 6 に関する考慮事項

Red Hat Satellite Server 6 を使用して Red Hat OpenStack Platform (RHOSP) 環境の RPM とコンテナーイメージをホストし、RHOSP 17.1 のアップグレード中に Satellite 6 を使用してコンテンツを配信する予定の場合は、以下の条件を満たす必要があります。

- Satellite Server は、RHOSP 16.2 RPM とコンテナーイメージをホストしている。
- RHOSP 16.2 環境内のすべてのノードを Satellite Server に登録している。
  たとえば、RHOSP 16.2 コンテンツビューにリンクされたアクティベーションキーを使用して、ノードを RHOSP 16.2 コンテンツに登録した場合など。

注記

アンダークラウドがインターネットにアクセスできない分離された環境を使用している場合は、既知の問題により、Red Hat OpenStack Platform 16.2 から 17.1 へのアップグレードが失敗します。回避策については、アップグレードを妨げる可能性がある既知の問題 の BZ2259891 の既知の問題を参照してください。

RHOSP アップグレードの推奨事項

- RHOSP 16.2 アンダークラウドとオーバークラウドの両方に必要な RPM リポジトリーを有効にして同期します。これには、必要な Red Hat Enterprise Linux (RHEL) 9.2 リポジトリーが含まれます。
- RHOSP 17.1 のコンテナーイメージをホストするために、Satellite Server 上にカスタム製品を作成します。
- RHOSP 17.1 アップグレード用のコンテンツビューを作成してプロモートし、コンテンツビューに次のコンテンツを含めます。
  - RHEL 8 のリポジトリー:
    - Red Hat Enterprise Linux 8 for x86_64 - AppStream (RPMs)

      ```
      rhel-8-for-x86_64-appstream-tus-rpms
      ```

    - Red Hat Enterprise Linux 8 for x86_64 - BaseOS (RPMs)

      ```
      rhel-8-for-x86_64-baseos-tus-rpms
      ```

    - Red Hat Enterprise Linux 8 for x86_64 - High Availability (RPMs)

      ```
      rhel-8-for-x86_64-highavailability-tus-rpms
      ```

    - Red Hat Fast Datapath for RHEL 8 (RPMs)

      ```
      fast-datapath-for-rhel-8-x86_64-rpms
      ```

  - RHEL 9 のリポジトリー:
    - Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)

      ```
      rhel-9-for-x86_64-appstream-eus-rpms
      ```

    - Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)

      ```
      rhel-9-for-x86_64-baseos-eus-rpms
      ```

- RHEL 9.2 リポジトリーを含む、すべてのアンタークラウドおよびオーバークラウド RPM リポジトリー。RHEL リポジトリーを有効にする際の問題を回避するには、RHEL リポジトリーの正しいバージョン (9.2) が含まれていることを確認してください。
- RHOSP 17.1 コンテナーイメージ。

- アクティベーションキーを、RHOSP 17.1 アップグレード用に作成した RHOSP 17.1 コンテンツビューに関連付けます。
- どのノードにも **katello-host-tools-fact-plugin** パッケージがインストールされていないことを確認します。Leapp のアップグレードでは、このパッケージはアップグレードされません。このパッケージを RHEL 9.2 システム上に残すと、**subscription-manager** がエラーを報告します。
- RHOSP 17.1 コンテナーイメージをホストするように Satellite Server を設定できます。RHOSP 16.2 から 17.1 にアップグレードするには、次のコンテナーイメージが必要です。
  - **rhosp-rhel8** namespace でホストされるコンテナーイメージ:
    - **rhosp-rhel8/openstack-collectd**
    - **rhosp-rhel8/openstack-nova-libvirt**
  - **rhosp-rhel9** namespace でホストされるコンテナーイメージ。**rhosp-rhel9** namespace のコンテナーイメージの設定については、**director を使用した Red Hat OpenStack Platform のインストールと管理** の コンテナーイメージ管理用 Satellite サーバーの準備 を参照してください。
- Red Hat Ceph Storage のサブスクリプションを使用し、Red Hat Ceph Storage ノード用に **overcloud-minimal** イメージを使用するように director を設定している場合、Satellite Server でコンテンツビューを作成し、以下の RHEL 9.2 リポジトリーを追加する必要があります。
  - Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)

    ```
    rhel-9-for-x86_64-appstream-eus-rpms
    ```

  - Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)

    ```
    rhel-9-for-x86_64-baseos-eus-rpms
    ```

    詳細は、**Red Hat Satellite コンテンツの管理** ガイドの コンテンツのインポート と コンテンツビューの管理 を参照してください。

第2章 アンダークラウドのアップグレード

アンダークラウドを Red Hat OpenStack Platform 17.1 にアップグレードします。アンダークラウドのアップグレードでは、実行中の Red Hat OpenStack Platform 16.2 アンダークラウドを使用します。アップグレードプロセスでは、ノード上の残りのサービスをアップグレードしながら、heat スタックをファイルにエクスポートし、heat を一時的な heat に変換します。

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。

2.1. アンダークラウド用リポジトリーの有効化

アンダークラウドに必要なリポジトリーを有効にし、システムパッケージを最新バージョンに更新します。

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. デフォルトのリポジトリーをすべて無効にしてから、必要な Red Hat Enterprise Linux (RHEL) リポジトリーを有効にします。

```
[stack@director ~]$ sudo subscription-manager repos --disable=*
[stack@director ~]$ sudo subscription-manager repos \
--enable=rhel-8-for-x86_64-baseos-tus-rpms \
--enable=rhel-8-for-x86_64-appstream-tus-rpms \
--enable=rhel-8-for-x86_64-highavailability-tus-rpms \
--enable=openstack-17.1-for-rhel-8-x86_64-rpms \
--enable=fast-datapath-for-rhel-8-x86_64-rpms
```

3. すべてのノードで **container-tools** モジュールのバージョンを RHEL 8 に切り替えます。

```
[stack@director ~]$ sudo dnf -y module switch-to container-tools:rhel8
```

4. director のインストールと設定を行うためのコマンドラインツールをインストールします。

```
[stack@director ~]$ sudo dnf install -y python3-tripleoclient
```

2.2. アップグレード前の RHOSP の検証

Red Hat OpenStack Platform (RHOSP) 17.1 にアップグレードする前に、**tripleo-validations** Playbook を使用してアンダークラウドとオーバークラウドを検証してください。RHOSP 16.2 では、OpenStack Workflow Service (mistral) を通じてこれらの Playbook を実行します。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. **/var/lib/mistral/.ssh** ディレクトリーの権限を調整します。

```
$ sudo chmod +x /var/lib/mistral/.ssh/
```

4. 検証用にパッケージをインストールします。

```
$ sudo dnf -y update openstack-tripleo-validations python3-validations-libs validations-common
```

5. mistral からインベントリーをコピーします。

```
$ sudo chown stack:stack /var/lib/mistral/.ssh/tripleo-admin-rsa
$ sudo cat /var/lib/mistral/<stack>/tripleo-ansible-inventory.yaml > inventory.yaml
```

- <stack> をスタックの名前に置き換えます。

6. 検証を実行します。

```
$ validation run -i inventory.yaml --group pre-upgrade
```

7. スクリプトの出力を確認し、成功した検証と失敗した検証を確認します。

```
=== Running validation: "check-ftype" ===

Success! The validation passed for all hosts:
* undercloud
```

2.3. コンテナーイメージの準備

アンダークラウドのインストールには、コンテナーイメージの取得先およびその保存方法を定義するための環境ファイルが必要です。コンテナーイメージの準備に使用できる環境ファイルを生成およびカスタマイズします。

注記

アンダークラウド用に特定のコンテナーイメージバージョンを設定する必要がある場合は、イメージを特定のバージョンに固定する必要があります。詳細は、アンダークラウド用コンテナーイメージのピニング を参照してください。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. オプション: 16.2 **containers-prepare-parameter.yaml** ファイルをバックアップします。

```
$ cp containers-prepare-parameter.yaml \
  containers-prepare-parameter.yaml.orig
```

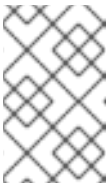
3. デフォルトのコンテナーイメージ準備ファイルを生成します。

```
$ openstack tripleo container image prepare default \
  --local-push-destination \
  --output-env-file containers-prepare-parameter.yaml
```

上記のコマンドでは、以下の追加オプションを使用しています。

- **--local-push-destination**: コンテナーイメージの保管場所として、アンダークラウド上のレジストリーを設定します。つまり、director は必要なイメージを Red Hat Container Catalog からプルし、それをアンダークラウド上のレジストリーにプッシュします。director はこのレジストリーをコンテナーイメージのソースとして使用します。Red Hat Container Catalog から直接プルする場合には、このオプションを省略します。

- **--output-env-file**: 環境ファイルの名前です。このファイルには、コンテナーイメージを準備するためのパラメーターが含まれます。ここでは、ファイル名は **containers-prepare-parameter.yaml** です。



注記

アンダークラウドとオーバークラウド両方のコンテナーイメージのソースを定義するのに、同じ **containers-prepare-parameter.yaml** ファイルを使用することができます。

4. 要件に合わせて **containers-prepare-parameter.yaml** を変更します。コンテナーイメージのパラメーターの詳細は、コンテナーイメージ準備のパラメーター を参照してください。

5. デプロイメントに Red Hat Ceph Storage が含まれている場合は、デプロイメントで使用する Red Hat Ceph Storage のバージョンに応じて **containers-prepare-parameter.yaml** ファイル内の Red Hat Ceph Storage コンテナーイメージパラメーターを更新します。

```
ceph_namespace: registry.redhat.io/rhceph
ceph_image: <ceph_image_file>
ceph_tag: latest
ceph_grafana_image: <grafana_image_file>
ceph_grafana_namespace: registry.redhat.io/rhceph
ceph_grafana_tag: latest
```

- **<ceph_image_file>** は、デプロイメントで使用する Red Hat Ceph Storage のバージョンのイメージファイルの名前に置き換えます。
  - Red Hat Ceph Storage 5: **rhceph-5-rhel8**
  - Red Hat Ceph Storage 6: **rhceph-6-rhel9**

- **<grafana_image_file>** をデプロイメントで使用する Red Hat Ceph Storage のバージョンのイメージファイルの名前に置き換えます。
  - Red Hat Ceph Storage 5: **rhceph-5-dashboard-rhel8**
  - Red Hat Ceph Storage 6: **rhceph-6-dashboard-rhel9**

2.4. コンテナーイメージタグ付けのガイドライン

Red Hat コンテナーレジストリーでは、すべての Red Hat OpenStack Platform コンテナーイメージをタグ付けするのに、特定のバージョン形式が使用されます。この形式は、**version-release** のように各コンテナーのラベルのメタデータに従います。

version

Red Hat OpenStack Platform のメジャーおよびマイナーバージョンに対応します。これらのバージョンは、1 つまたは複数のリリースが含まれるストリームとして機能します。

release

バージョンストリーム内の、特定コンテナーイメージバージョンのリリースに対応します。

たとえば、Red Hat OpenStack Platform の最新バージョンが 17.1.0 で、コンテナーイメージのリリースが **5.161** の場合、コンテナーイメージのタグは 17.1.0-5.161 となります。

Red Hat コンテナーレジストリーでは、コンテナーイメージバージョンの最新リリースとリンクするメジャーおよびマイナー **version** タグのセットも使用されます。たとえば、17.1 と 17.1.0 の両方が、17.1.0 コンテナーストリームの最新 **release** とリンクします。17.1 の新規マイナーリリースが公開されると、17.1 タグは新規マイナーリリースストリームの最新 **release** とリンクします。一方、17.1.0 タグは、引き続き 17.1.0 ストリーム内の最新の **release** とリンクします。

ContainerImagePrepare パラメーターには 2 つのサブパラメーターが含まれ、これを使用してダウンロードするコンテナーイメージを定義することができます。これらのサブパラメーターは、**set** ディクショナリー内の **tag** パラメーターおよび **tag_from_label** パラメーターです。以下のガイドラインを使用して、**tag** または **tag_from_label** のどちらを使用するかを判断してください。

- **tag** のデフォルト値は、お使いの OpenStack Platform のメジャーバージョンです。本バージョンでは、17.1 です。これは常に最新のマイナーバージョンおよびリリースに対応します。

```
parameter_defaults:
  ContainerImagePrepare:
  - set:
      ...
      tag: 17.1
      ...
```

- OpenStack Platform コンテナーイメージの特定マイナーバージョンに変更するには、タグをマイナーバージョンに設定します。たとえば、17.1.2 に変更するには、**tag** を 17.1.2 に設定します。

```
parameter_defaults:
  ContainerImagePrepare:
  - set:
      ...
      tag: 17.1.2
      ...
```

- **tag** を設定すると、インストールおよび更新時に、director は必ず **tag** で設定したバージョンの最新のコンテナーイメージ **release** をダウンロードします。

- **tag** を設定しないと、director は最新のメジャーバージョンと共に **tag_from_label** の値を使用します。

```
parameter_defaults:
  ContainerImagePrepare:
  - set:
      ...
      # tag: 17.1
      ...
    tag_from_label: '{version}-{release}'
```

- **tag_from_label** パラメーターは、Red Hat コンテナーレジストリーから検査する最新コンテナーイメージリリースのラベルメタデータからタグを生成します。たとえば、特定のコンテナーのラベルは、以下の **version** および **release** メタデータを使用します。

```
"Labels": {
  "release": "5.161",
  "version": "17.1.0",
  ...
}
```

- **tag_from_label** のデフォルト値は **{version}-{release}** です。これは、各コンテナーイメージのバージョンおよびリリースのメタデータラベルに対応します。たとえば、コンテナーイメージの **version** に 17.1.0 が、**release** に 5.161 が、それぞれ設定されている場合、コンテナーイメージのタグは 17.1.0-5.161 となります。

- **tag** パラメーターは、常に **tag_from_label** パラメーターよりも優先されます。**tag_from_label** を使用するには、コンテナー準備の設定で **tag** パラメーターを省略します。

- **tag** および **tag_from_label** の主な相違点は、次のとおりです。director が **tag** を使用してイメージをプルする場合は、Red Hat コンテナーレジストリーがバージョンストリーム内の最新イメージリリースとリンクするメジャーまたはマイナーバージョンのタグだけに基づきます。一方、**tag_from_label** を使用する場合は、director がタグを生成して対応するイメージをプルできるように、各コンテナーイメージのメタデータの検査を行います。

2.5. プライベートレジストリーからのコンテナーイメージの取得

registry.redhat.io レジストリーにアクセスしてイメージをプルするには、認証が必要です。**registry.redhat.io** およびその他のプライベートレジストリーで認証するには、**containers-prepare-parameter.yaml** ファイルに **ContainerImageRegistryCredentials** および **ContainerImageRegistryLogin** パラメーターを含めます。

ContainerImageRegistryCredentials

一部のコンテナーイメージレジストリーでは、イメージにアクセスするのに認証が必要です。そのような場合には、**containers-prepare-parameter.yaml** 環境ファイルの **ContainerImageRegistryCredentials** パラメーターを使用します。**ContainerImageRegistryCredentials** パラメーターは、プライベートレジストリーの URL に基づくキーのセットを使用します。それぞれのプライベートレジストリーの URL は、独自のキーと値のペアを使用して、ユーザー名 (キー) およびパスワード (値) を定義します。これにより、複数のプライベートレジストリーに対して認証情報を指定することができます。

```
parameter_defaults:
  ContainerImagePrepare:
  - push_destination: true
    set:
      namespace: registry.redhat.io/...
      ...
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      my_username: my_password
```

上記の例の **my_username** および **my_password** を、実際の認証情報に置き換えてください。Red Hat では、個人のユーザー認証情報を使用する代わりに、レジストリーサービスアカウントを作成し、それらの認証情報を使用して **registry.redhat.io** コンテンツにアクセスすることを推奨します。

複数のレジストリーの認証情報を指定するには、**ContainerImageRegistryCredentials** でレジストリーごとに複数のキー/ペアの値を設定します。

```
parameter_defaults:
```

```
  ContainerImagePrepare:
  - push_destination: true
    set:
      namespace: registry.redhat.io/...
      ...
  - push_destination: true
    set:
      namespace: registry.internalsite.com/...
      ...
  ...
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      myuser: 'p@55w0rd!'
    registry.internalsite.com:
      myuser2: '0th3rp@55w0rd!'
    '192.0.2.1:8787':
      myuser3: '@n0th3rp@55w0rd!'
```

重要

デフォルトの **ContainerImagePrepare** パラメーターは、認証が必要な **registry.redhat.io** からコンテナーイメージをプルします。

詳細は、Red Hat コンテナーレジストリーの認証 を参照してください。

ContainerImageRegistryLogin

ContainerImageRegistryLogin パラメーターを使用して、コンテナーイメージを取得するために、オーバークラウドノードシステムがリモートレジストリーにログインする必要があるかどうかを制御します。このような状況は、アンダークラウドを使用してイメージをホストする代わりに、オーバークラウドノードがイメージを直接プルする場合に発生します。

特定の設定について、**push_destination** が false に設定されている、または使用されていない場合は、**ContainerImageRegistryLogin** を **true** に設定する必要があります。

```
parameter_defaults:
  ContainerImagePrepare:
  - push_destination: false
    set:
      namespace: registry.redhat.io/...
      ...
  ...
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      myuser: 'p@55w0rd!'
  ContainerImageRegistryLogin: true
```

ただし、オーバークラウドノードに **ContainerImageRegistryCredentials** で定義されたレジストリーホストへのネットワーク接続がなく、**ContainerImageRegistryLogin** を **true** に設定すると、ログインを試みる際にデプロイメントが失敗する可能性があります。オーバークラウドノードに **ContainerImageRegistryCredentials** で定義されたレジストリーホストへのネットワーク接続がない場合、**push_destination** を **true** に、**ContainerImageRegistryLogin** を **false** に設定して、オーバークラウドノードがアンダークラウドからイメージをプルできるようにします。

```
parameter_defaults:
```

```
ContainerImagePrepare:
- push_destination: true
  set:
    namespace: registry.redhat.io/...
    ...
...
ContainerImageRegistryCredentials:
  registry.redhat.io:
    myuser: 'p@55w0rd!'
ContainerImageRegistryLogin: false
```

2.6. UNDERCLOUD.CONF ファイルの更新

Red Hat OpenStack Platform 16.2 環境の元の **undercloud.conf** ファイルを引き続き使用できますが、Red Hat OpenStack Platform 17.1 との互換性を維持するにはファイルを変更する必要があります。**undercloud.conf** ファイルを設定するためのパラメーターの詳細は、**director を使用した Red Hat OpenStack Platform のインストールと管理** の アンダークラウド設定パラメーター を参照してください。

手順

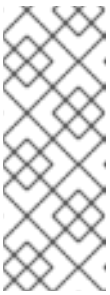
1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **skip_rhel_release.yaml** というファイルを作成し、**SkipRhelEnforcement** パラメーターを **true** に設定します。

```
parameter_defaults:
  SkipRhelEnforcement: true
```

3. **undercloud.conf** ファイルを開き、ファイルの **DEFAULT** セクションに、以下のパラメーターを追加します。

```
container_images_file = /home/stack/containers-prepare-parameter.yaml
custom_env_files = /home/stack/skip_rhel_release.yaml
```

- 追加のカスタム環境ファイルを **custom_env_files** パラメーターに追加します。**custom_env_files** パラメーターは、アップグレードに必要な **skip_rhel_release.yaml** ファイルの場所を定義します。
- **container_images_file** パラメーターは、director が正しい場所からアンダークラウドのコンテナーイメージをプルできるように、**containers-prepare-parameter.yaml** 環境ファイルの場所を定義します。



注記

元の **undercloud.conf** ファイルの **/home/stack/custom-kerberos-params.yaml** ファイルに **CertmongerKerberosRealm** パラメーターが含まれている場合は、**CertmongerKerberosRealm** パラメーターを **HAProxyCertificatePrincipal** パラメーターに置き換える必要があります。**CertmongerKerberosRealm** パラメーターが原因で、アンダークラウドのアップグレードに失敗します。

4. ファイル内の他のすべてのパラメーターが変更されていないか確認します。

5. ファイルを保存します。

2.7. ネットワーク設定ファイルの変換

ネットワーク設定テンプレートに以下の機能が含まれている場合は、アンダークラウドをアップグレードする前に、NIC テンプレートを Jinja2 Ansible 形式に手動で変換する必要があります。次の関数は自動変換ではサポートされていません。

- **'get_file'**
- **'get_resource'**
- **'digest'**
- **'repeat'**
- **'resource_facade'**
- **'str_replace'**
- **'str_replace_strict'**
- **'str_split'**
- **'map_merge'**
- **'map_replace'**
- **'yaql'**
- **'equals'**
- **'if'**
- **'not'**
- **'and'**
- **'or'**
- **'filter'**
- **'make_url'**
- **'contains'**

NIC テンプレートの手動変換の詳細は、**Red Hat OpenStack Platform デプロイメントのカスタマイズ** の NIC テンプレートの Jinja2 Ansible 形式への手動変換 を参照してください。

2.8. DIRECTOR のアップグレードの実施

アンダークラウド上の director をアップグレードします。

前提条件

- **tripleo_mysql.service** サービスが実行していることを確認します。

```
$ systemctl status tripleo_mysql
```

サービスが実行されていない場合は、サービスを開始します。

```
$ sudo systemctl start tripleo_mysql
```

- ネットワーク設定テンプレートに特定の機能が含まれている場合は、NIC テンプレートを必ず手動で Jinja2 Ansible 形式に変換してください。該当する機能のリストと手動手順へのリンクについては、ネットワーク設定ファイルの変換 を参照してください。

手順

- director 設定スクリプトを起動して director をアップグレードします。

```
$ openstack undercloud upgrade
```

director 設定スクリプトは、director パッケージをアップグレードし、**undercloud.conf** ファイルの設定と一致するように director サービスを設定します。このスクリプトは、完了までに数分かかります。

注記

director の設定スクリプトでは、次の設定に進む前に確認が求められます。この確認を省略するには、**-y** オプションを使用します。

```
$ openstack undercloud upgrade -y
```

第3章 外部 CEPH デプロイメントと組み合わせたアップグレード

Red Hat OpenStack Platform (RHOSP) デプロイメントが外部にデプロイされた Red Hat Ceph Storage クラスターを使用している場合は、RHOSP のアップグレードを続行する前に Red Hat Ceph Storage クラスターをアップグレードする必要がある場合があります。

Red Hat Ceph Storage クラスターが現在、リリース 4 である場合は、次のタスクを実行します。

1. Red Hat Ceph Storage クラスターをリリース 4 からリリース 5 にアップグレードします。
2. RHOSP デプロイメントを Release 16.2 から Release 17.1 にアップグレードします。
3. Red Hat Ceph Storage クラスターをリリース 5 からリリース 6 にアップグレードします。

Red Hat Ceph Storage クラスターが現在、リリース 5 である場合は、次のタスクを実行します。

1. RHOSP デプロイメントを Release 16.2 から Release 17.1 にアップグレードします。
2. Red Hat Ceph Storage クラスターをリリース 5 からリリース 6 にアップグレードします。

Red Hat Ceph Storage クラスターのアップグレードの詳細は、次のガイドを参照してください。

- Red Hat Ceph Storage 5 アップグレードガイド
- Red Hat Ceph Storage 6 アップグレードガイド

Red Hat Ceph Storage クラスターをアップグレードした後、ceph-ansible **ceph-client** ロールから tripleo-ansible **tripleo_ceph_client** ロールに移行する必要があります。

3.1. RHOSP 17.1 の CEPH クライアント設定の更新

Red Hat OpenStack Platform (RHOSP) 17.1 より前は、外部 Red Hat Ceph Storage 環境の場合、OpenStack Ceph クライアントは ceph-ansible **ceph-client** ロールによって設定されていました。RHOSP 17.1 では、OpenStack Ceph クライアントは tripleo-ansible **Tripleo_ceph_client** ロールによって設定されます。オーバークラウドの導入と準備の実行 でオーバークラウドのアップグレードを実行する前に、OpenStack サービスの設定に使用される tripleo-heat-templates 環境ファイルを外部 Ceph クラスターに置き換える必要があります。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. 次のコマンドに **environment/ceph-ansible/ceph-ansible-external.yaml** ファイルを含めた場合は、そのファイルを **environment/external-ceph.yaml** ファイルに置き換える必要があります。
   - **openstack overcloud upgrade prepare**
   - **openstack overcloud deploy**
     たとえば、以下のように指定したとします。

```
$ openstack overcloud deploy
...
-e environments/ceph-ansible/ceph-ansible-external.yaml
...
```

以下に置き換えます。

```
$ openstack overcloud deploy
...
-e environments/external-ceph.yaml
...
```

4. **ceph_params.yaml** というファイルを作成し、次の内容を含めます。

```
parameter_defaults:
  CephClusterFSID: <fsid>
  CephClientKey: <key>
  CephExternalMonHost: <mon ip addresses>
  CephSpecFqdn: <true/false>
  CephConfigPath: "/etc/ceph"
  DeployedCeph: false
```

- **<fsid>** は、Red Hat Ceph Storage クラスターの UUID に置き換えます。
- **<key>** は、Ceph クライアントキーに置き換えます。
- **<mon ip address>** は、Ceph Mon Host IP のリストに置き換えます。
- **<true/false>** は、環境に応じて適切な値に置き換えます。

注記

Red Hat Ceph Storage デプロイメントに短縮名が含まれている場合は、**CephSpecFqdn** パラメーターを false に設定する必要があります。true に設定すると、短縮名とドメイン名の両方を使用してインベントリーが生成されるため、Red Hat Ceph Storage のアップグレードが失敗します。

5. オーバークラウドのデプロイコマンドに **ceph_params.yaml** ファイルを含めます。

```
$ openstack overcloud deploy \
...
-e ~/environments/ceph_params.yaml \
```

重要

RHOSP のアップグレード完了後に **ceph_params.yaml** ファイルを削除しないでください。このファイルは外部 Red Hat Ceph Storage 環境に存在する必要があります。さらに、**openstack overcloud deploy** を実行するときは、**-e ceph_params.yaml** を指定するなどして、常に **ceph_params.yaml** ファイルを含める必要があります。

次のステップ

オーバークラウドの導入と準備の手順を実行するときに作成するオーバークラウドのアップグレード準備スクリプトに **ceph_params.yaml** ファイルを含めます。詳細は、オーバークラウドの導入と準備の実行 を参照してください。

第4章 オーバークラウドのアップグレードの準備

オーバークラウドのアップグレードに向けた準備を行うには、いくつかの初期手順を完了する必要があります。

4.1. オーバークラウドサービスのダウンタイムの準備

オーバークラウドのアップグレードプロセスでは、主要なポイントでメインのコントロールプレーンサービスを無効します。これらの主要なポイントに達すると、オーバークラウドサービスを使用して新規リソースを作成することはできません。オーバークラウドで実行されているワークロードは、アップグレードプロセス中もアクティブなままであるため、インスタンスはコントロールプレーンのアップグレード中も引き続き実行されます。Compute ノードのアップグレード中に、これらのワークロードは、すでにアップグレードされている Compute ノードにライブマイグレーションできます。

アップグレード中にはユーザーがオーバークラウドのサービスにアクセスできないように、メンテナンスの時間帯を計画することが重要となります。

オーバークラウドのアップグレードによる影響を受ける項目

- OpenStack Platform サービス

オーバークラウドのアップグレードによる影響を受けない項目

- アップグレード中に実行するインスタンス
- Ceph Storage OSD (インスタンス向けのバックエンドストレージ)
- Linux ネットワーク
- Open vSwitch ネットワーク
- アンダークラウド

4.2. オーバークラウドでのフェンシングの無効化

オーバークラウドをアップグレードする前に、フェンシングが無効になっていることを確認します。

オーバークラウドをアップグレードする場合、高可用性機能を維持するために、各コントローラーノードを個別にアップグレードします。フェンシングが環境にデプロイされると、オーバークラウドは特定ノードが無効であることを検出し、フェンシング操作を試みる場合があります。これにより、意図しない結果が生じる可能性があります。

オーバークラウドでフェンシングを有効にしている場合には、意図しない結果を防ぐために、アップグレード期間中フェンシングを一時的に無効にする必要があります。

注記

Red Hat OpenStack Platform 環境のアップグレードが完了したら、オーバークラウドでフェンシングを再度有効にする必要があります。フェンシングの再有効化に関する詳細は、オーバークラウドでのフェンシングの再有効化 を参照してください。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。

2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. 各コントローラーノードにログインし、Pacemaker コマンドを実行してフェンシングを無効にします。

```
$ ssh tripleo-admin@<controller_ip> "sudo pcs property set stonith-enabled=false"
```

   - **<controller_ip>** は、コントローラーノードの IP アドレスに置き換えます。コントローラーノードの IP アドレスは、**/etc/hosts** または **/var/lib/mistral** で確認できます。

4. **fencing.yaml**環境ファイルで、**EnableFencing**パラメーターを**false**に設定し、アップグレードプロセス中にフェンシングが無効のままとなるようにします。

関連情報

- STONITH を使用したコントローラーノードのフェンシング

4.3. アンダークラウドノードデータベースのバックアップ

openstack undercloud backup --db-only コマンドを使用して、アンダークラウドノード上で実行されるスタンドアロンデータベースバックアップを作成できます。データベースが破損した場合には、そのバックアップを使用してデータベースの状態を回復することもできます。アンダークラウドデータベースのバックアップに関する詳細は、**Red Hat OpenStack Platform 17.1 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** の 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。

4.4. カスタム ROLES_DATA ファイルのコンポーザブルサービスの更新

本項では、新しいコンポーザブルサービスおよび非推奨のコンポーザブルサービスについて説明します。

全ノード

すべてのノードで、以下のサービスが非推奨になっています。すべてのロールからこれらのサービスを削除します。

サービス	理由
OS::TripleO::Services::CinderBackendDellEMCXTREMIOISCSI **OS::TripleO::Services::CinderBackendDellPs** **OS::TripleO::Services::CinderBackendVRTSHyperScale**	このサービスは非推奨となりました。
OS::TripleO::Services::Ec2Api	このサービスは非推奨となりました。
OS::TripleO::Services::Fluentd	**OS::TripleO::Services::Rsyslog** を優先し、非推奨になったため。

サービス	理由
OS::TripleO::Services::FluentdAlt	このサービスは非推奨となりました。
OS::TripleO::Services::Keepalived	このサービスは非推奨となりました。
OS::TripleO::Services::MistralApi **OS::TripleO::Services::MistralEngine** **OS::TripleO::Services::MistralEventEngine** **OS::TripleO::Services::MistralExecutor**	このサービスは非推奨となりました。
OS::TripleO::Services::NeutronLbaasv2Agent **OS::TripleO::Services::NeutronLbaasv2Api**	Octavia を優先し、OpenStack Networking (neutron) Load Balancing as a Service は非推奨になったため。
OS::TripleO::Services::NeutronML2FujitsuCfab **OS::TripleO::Services::NeutronML2FujitsuFossw**	このサービスは非推奨となりました。
OS::TripleO::Services::NeutronSriovHostConfig	このサービスは非推奨となりました。
OS::TripleO::Services::NovaConsoleauth	このサービスは削除されています。
OS::TripleO::Services::Ntp	**OS::TripleO::Services::Timesync** を優先し、非推奨となりました。
OS::TripleO::Services::OpenDaylightApi **OS::TripleO::Services::OpenDaylightOvs**	OpenDaylight はサポートされなくなったため。
OS::TripleO::Services::OpenShift::GlusterFS **OS::TripleO::Services::OpenShift::Infra** **OS::TripleO::Services::OpenShift::Master** **OS::TripleO::Services::OpenShift::Worker**	このサービスは非推奨となりました。
OS::TripleO::Services::PankoApi	メトリクスおよびモニタリングには Service Telemetry Framework (STF) が優先されるため、OpenStack Telemetry サービスは非推奨になりました。従来のテレメトリーサービスは、STF への移行を促進するために RHOSP 17.1 でのみ利用可能で、RHOSP の今後のバージョンでは削除される予定です。

サービス	理由
OS::TripleO::Services::Rear	このサービスは非推奨となりました。
OS::TripleO::Services::SaharaApi **OS::TripleO::Services::SaharaEngine**	このサービスは非推奨となりました。
OS::TripleO::Services::SensuClient **OS::TripleO::Services::SensuClientAlt**	このサービスは非推奨となりました。
OS::TripleO::Services::SkydiveAgent **OS::TripleO::Services::SkydiveAnalyzer**	Skydive はサポートされなくなったため。
OS::TripleO::Services::Tacker	Tacker はサポートされなくなったため。
OS::TripleO::Services::TripleoUI	このサービスは非推奨となりました。
OS::TripleO::Services::UndercloudMinionMessaging **OS::TripleO::Services::UndercloudUpgradeEphemeralHeat**	このサービスは非推奨となりました。
OS::TripleO::Services::Zaqar	このサービスは非推奨となりました。

コントローラーノード

コントローラーノードの新規サービスは以下のとおりです。Controller ロールにこれらのサービスを追加します。

サービス	理由
OS::TripleO::Services::GlanceApiInternal	イメージサービス (glance) API の内部インスタンスのサービス。位置データを管理者とそれを必要とするサービス (Block Storage サービス (cinder) やコンピュートサービス (nova) など) に提供します。

Compute nodes

デフォルトでは、17.1 のコンピュートノードは **OS::TripleO::Services::NovaLibvirt** サービスを実行します。ただし、**OS::TripleO::Services::NovaLibvirt** サービスを実行しているコンピュートノードで RHOSP アップグレードを実行すると、仮想マシンインスタンスはシャットオフされた状態として表示されます。この問題を回避するには、RHEL 8.4 上のすべてのコンピュートノードで **OS::TripleO::Services::NovaLibvirtLegacy** サービスを実行し、コンテナーイメージを UBI-8 に基づいて作成する必要があります。

RHOSP のアップグレード後、コンピュートノードを RHEL 9.2 にアップグレードする場合、コンピュートノードは **OS::TripleO::Services::NovaLibvirt** サービスを実行し、コンテナーイメージは UBI-9 に基づいている必要があります。そうしないと、仮想マシンインスタンスがシャットオフされて

いるように表示されます。

コンピュートノード上のオペレーティングシステムのアップグレードに関する詳細は RHEL 9.2 への全コンピュートノードのアップグレード および コンピュートノードの Multi-RHEL 環境へのアップグレード を参照してください。

4.5. カスタムの PUPPET パラメーターの確認

Puppet パラメーターのカスタマイズに **ExtraConfig** インターフェイスを使用する場合には、アップグレード中に、Puppet が重複した宣言のエラーを報告する可能性があります。これは、Puppet モジュール自体によって提供されるインターフェイスの変更が原因です。

この手順では、環境ファイル内のカスタムの **ExtraConfig** hieradata パラメーターを確認する方法を説明します。

手順

1. 環境ファイルを選択して、**ExtraConfig** パラメーターが設定されているかどうかを確認します。

```
$ grep ExtraConfig ~/templates/custom-config.yaml
```

2. このコマンドの結果に、選択したファイル内のいずれかのロールの **ExtraConfig** パラメーター (例: **ControllerExtraConfig**) が表示される場合には、そのファイルの完全なパラメーター構造を確認してください。

3. **SECTION/parameter** 構文で **value** が続くいずれかの Puppet hieradata がパラメーターに含まれている場合には、実際の Puppet クラスのパラメーターに置き換えられている可能性があります。以下に例を示します。

```
parameter_defaults:
  ExtraConfig:
    neutron::config::dhcp_agent_config:
      'DEFAULT/dnsmasq_local_resolv':
        value: 'true'
```

4. director の Puppet モジュールを確認して、パラメーターが Puppet クラス内に存在しているかどうかを確認します。以下に例を示します。

```
$ grep dnsmasq_local_resolv
```

その場合には、新規インターフェイスに変更します。

5. 構文の変更の実例を以下に示します。

- 例 1:

```
parameter_defaults:
  ExtraConfig:
    neutron::config::dhcp_agent_config:
      'DEFAULT/dnsmasq_local_resolv':
        value: 'true'
```

変更後

```
parameter_defaults:
  ExtraConfig:
    neutron::agents::dhcp::dnsmasq_local_resolv: true
```

- 例 2:

```
parameter_defaults:
  ExtraConfig:
    ceilometer::config::ceilometer_config:
      'oslo_messaging_rabbit/rabbit_qos_prefetch_count':
        value: '32'
```

変更後

```
parameter_defaults:
  ExtraConfig:
    oslo::messaging::rabbit::rabbit_qos_prefetch_count: '32'
```

4.6. アップグレード前の最終確認

アップグレードを開始する前に、すべての準備手順の最終確認を行います。

4.6.1. アップグレードコマンドの概要

アップグレードプロセスには、プロセスの特定の段階で実行するさまざまなコマンドが含まれます。

重要

本セクションでは、それぞれのコマンドについてのみ説明します。これらのコマンドは特定の順序で実行し、オーバークラウドに固有のオプションを指定する必要があります。適切なステップでこれらのコマンドを実行するよう指示されるまで待ちます。

4.6.1.1. openstack overcloud upgrade prepare

このコマンドにより、オーバークラウドのアップグレードの初期準備の手順が実行されます。これには、アンダークラウド上の現在のオーバークラウドプランを新しい OpenStack Platform 17.1 オーバークラウドプランおよび更新された環境ファイルに置き換えることが含まれます。このコマンドは、**openstack overcloud deploy** コマンドと同じように機能し、多くの同一オプションが使用されます。

openstack overcloud upgrade prepare コマンドを実行する前に、オーバークラウドの導入を実行する必要があります。オーバークラウドの導入について、詳細は オーバークラウドの導入と準備の実行 を参照してください。

4.6.1.2. openstack overcloud upgrade run

このコマンドにより、アップグレードプロセスが実施されます。director は、新しい OpenStack Platform 17.1 オーバークラウドプランに基づいて Ansible Playbook のセットを作成し、オーバークラウド全体で Fast Forward タスクを実行します。これには、16.2 から 17.1 までの各 OpenStack Platform バージョンを通じてアップグレードプロセスを実行することが含まれます。

標準のアップグレードプロセスに加えて、このコマンドによりオーバークラウドノード上のオペレーティングシステムの Leapp アップグレードを実施することができます。**--tags** オプションを使用して、これらのタスクを実行します。

Leapp のアップグレードタスクタグ

system_upgrade

system_upgrade_prepare、**system_upgrade_run**、および **system_upgrade_reboot** のタスクを組み合わせるタスク

system_upgrade_prepare

Leapp を使用したオペレーティングシステムのアップグレードに向けた準備を行うタスク

system_upgrade_run

Leapp を実行し、オペレーティングシステムをアップグレードするタスク

system_upgrade_reboot

システムをリブートし、オペレーティングシステムのアップグレードを完了するタスク

4.6.1.3. openstack overcloud external-upgrade run

このコマンドにより、標準のアップグレードプロセス以外のアップグレードタスクが実行されます。director は新しい OpenStack Platform 17.1 オーバークラウドプランに基づいて Ansible Playbook のセットを作成するので、**--tags** オプションを使用して特定のタスクを実行します。

コンテナー管理の外部タスクタグ

container_image_prepare

アンダークラウドレジストリーにコンテナーイメージをプルし、オーバークラウドが使用するようにイメージを準備するタスク

4.6.2. アップグレードのパラメーター

アップグレードパラメーターを使用してアップグレードプロセスの動作を変更できます。

パラメーター	説明
UpgradeInitCommand	アップグレードプロセスを初期化するためにすべてのオーバークラウドノード上で実行するコマンドまたはスクリプトのスニペット。たとえば、リポジトリーの切り替えなど。
UpgradeInitCommonCommand	アップグレードプロセスに必要な共通のコマンド。操作者は通常このパラメーターを変更する必要はなく、major-upgrade-composable-steps.yaml および major-upgrade-converge.yaml 環境ファイルで設定および設定解除されます。
UpgradeLeappCommandOptions	Leapp コマンドに追加するその他のコマンドラインオプション
UpgradeLeappDebug	Leapp の実行中にデバッグのアウトプットを出力します。デフォルト値は **false** です。

パラメーター	説明
UpgradeLeappDevelSkip	開発/テスト環境で Leapp を実行する場合は、環境変数を設定して Leapp の確認を省略します。たとえば、LEAPP_DEVEL_SKIP_RHSM=1 と設定します。
UpgradeLeappEnabled	オペレーティングシステムのアップグレードに Leapp を使用します。デフォルト値は **false** です。
UpgradeLeappPostRebootDelay	マシンがリブートしてテストコマンドに応答するのを待つ最大の時間 (秒)。デフォルト値は **120** です。
UpgradeLeappRebootTimeout	Leapp による OS アップグレードフェーズのタイムアウト時間 (秒単位)。デフォルト値は **3600** です。
UpgradeLeappToInstall	Leapp によるアップグレード後にインストールするパッケージのリスト
UpgradeLeappToRemove	Leapp によるアップグレード時に削除するパッケージのリスト

4.6.3. デプロイメントに含めるカスタムファイル

デプロイメント内のオーバークラウドノードが Object Storage (swift) 専用ノードの場合は、デフォルトの **roles_data.yaml** ファイルをコピーし、**ObjectStorage** を編集して deprecated_server_resource_name: **'SwiftStorage'の行を削除する必要があります**。次に、**--roles-file** オプションを使用して、そのファイルを **openstack overcloud upgrade prepare** コマンドに渡します。

4.6.4. デプロイメントに追加する新たな環境ファイル

Red Hat OpenStack Platform (RHOSP) 17.1 へのアップグレードを円滑に行うために、通常のオーバークラウドの環境ファイルに加えて、新しい環境ファイルを追加する必要があります。

ファイル	注記
/home/stack/templates/upgrades-environment.yaml	このファイルには、アップグレードに固有のパラメーターが含まれます。このファイルは、アップグレード期間中にのみ必要です。
/home/stack/containers-prepare-parameter.yaml	ソースおよび準備の手順が含まれるファイルです。これは、アンダークラウドのアップグレードに使用するファイルと同じです。
/home/stack/templates/ceph.yaml	このファイルには、Ceph Storage がオーバーライドする必要があるパラメーターが含まれています。

以下のコマンドを実行する際に、環境ファイルリストの最後にこれらのファイルを追加します。

- **openstack overcloud upgrade prepare**
- **openstack overcloud deploy**

4.6.5. デプロイメントから削除する環境ファイル

Red Hat OpenStack Platform 16.2 に固有の環境ファイルをすべて削除します。

- Red Hat OpenStack Platform 16.2 コンテナーイメージのリスト
- Red Hat OpenStack Platform 16.2 カスタマーポータルまたは Satellite **rhel-registration** スクリプト

以下のコマンドを実行する際に指定する環境ファイルのリストから、これらのファイルを削除します。

- **openstack overcloud upgrade prepare**
- **openstack overcloud deploy**

4.6.6. IPA サービスのアップグレード

環境内で TLS everywhere が有効になっている場合は、Nova Host Manager ロールにパーミッションを追加して、DNS ゾーンエントリーの作成を許可します。

前提条件

Nova Host Management パーミッションが使用中の環境に含まれているかどうかを確認する。

```
$ ipa privilege-show "Nova Host Management"
```

すでにこのパーミッションを持っている場合は、以下の手順はスキップしてください。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. **Nova Host Management** パーミッションを追加します。

```
$ kinit admin
$ ipa privilege-add-permission 'Nova Host Management' --permission 'System: Modify Realm Domains'
```

4. **ipa_environment.yaml** という環境ファイルを作成し、以下の設定を含めます。

```
resource_registry:
  OS::TripleO::Services::IpaClient: /usr/share/openstack-tripleo-heat-templates/deployment/ipa/ipaservices-baremetal-ansible.yaml

parameter_defaults:
```

```
IdMServer: $IPA_FQDN
IdMDomain: $IPA_DOMAIN
IdMInstallClientPackages: False
```

5. 環境ファイルを保存します。

4.6.7. アップグレードのチェックリスト

以下のチェックリストを使用して、オーバークラウドをアップグレードする準備ができているかどうかを判断します。

確認項目	実施状況
動作中のオーバークラウドを検証済みである。	はい / いいえ
オーバークラウドコントロールプレーンの Relax-and-Recover (ReaR) バックアップを実施済みである。詳細は、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を 参照してください。	はい / いいえ
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を作成済みである。詳細は、**Red Hat OpenStack Platform 17.1 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** の 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。	はい / いいえ
登録情報を Red Hat OpenStack Platform 17.1 リポジトリーに更新し、Ansible ベースの手法を使用するように環境ファイルを変更済みである。	はい / いいえ
ネットワーク設定テンプレートを更新した。	はい / いいえ
環境ファイルの一覧を Red Hat OpenStack Platform 17.1 用の新しい環境ファイルで更新済みである。	はい / いいえ
(オプション) デプロイメントに専用の Object Storage (swift) ノードが含まれている場合は、 **role_data.yaml** ファイルをコピーして **deprecated_server_resource_name: 'SwiftStorage'** を削除し、そのファイルを **openstack overcloud upgrade prepare** コマンドに渡している。	はい / いいえ
古い Red Hat 登録やコンテナーイメージの場所に関するファイルなど、Red Hat OpenStack Platform 16.2 にしか該当しない古い環境ファイルを削除済みである。	はい / いいえ

第5章 オーバークラウドの導入と準備

環境内の各スタックでオーバークラウドの導入とアップグレードの準備を実行します。DCN 環境でオーバークラウドの導入とアップグレードの準備を実行するには、DCN 環境でのオーバークラウドの導入と準備 を参照してください。

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。

5.1. オーバークラウドの導入と準備の実行

オーバークラウドを導入するには、次のタスクを実行する必要があります。

- 各スタックで、ネットワークとホストのプロビジョニング設定エクスポートをオーバークラウドに導入します。
- 新しいコンテナーと追加の互換性設定を定義します。

導入後、次のタスクを実行するアップグレード準備スクリプトを実行する必要があります。

- オーバークラウドのプランを OpenStack Platform 17.1 に更新する。
- アップグレードに向けてノードを準備する。

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。

前提条件

1. すべてのノードが **ACTIVE** 状態にあることを確認します。

```
$ openstack baremetal node list
```

2. いずれかのノードが **MAINTENANCE** 状態にある場合は、次のコマンドを実行して **last_error** フィールドを確認し、**MAINTENANCE** 状態にあるノードの根本原因を特定してトラブルシューティングします。

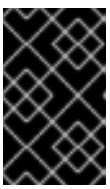
```
$ openstack baremetal node show <node_uuid>
```

   - **<node_uuid>** をノードの UUID に置き換えます。

3. **MAINTENANCE** の状態の設定を解除します。

```
$ openstack baremetal node maintenance unset <node_uuid>
```

   ノードが **MAINTENANCE** 状態に戻るかどうかを確認するために 3 - 5 分待ちます。



重要

いずれかのノードが **MAINTENANCE** 状態のままの場合、アップグレードを続行できません。ノードを **MAINTENANCE** から削除できない場合は、Red Hat サポートにお問い合わせください。

手順

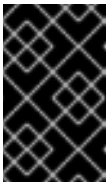
1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. アンダークラウドのアップグレード中にエクスポートされた以下のファイルに、オーバークラウドのアップグレードで想定される設定が含まれていることを確認します。**~/overcloud-deploy** ディレクトリーには以下のファイルがあります。
   - **tripleo-<stack>-passwords.yaml**
   - **tripleo-<stack>-network-data.yaml**
   - **tripleo-<stack>-virtual-ips.yaml**
   - **tripleo-<stack>-baremetal-deployment.yaml**

注記

ファイルがアンダークラウドのアップグレード後に生成されなかった場合は、Red Hat サポートにお問い合わせください。



重要

マルチセル環境をお使いの場合は、マルチセル環境でのオーバークラウドの導入 を参照し、ファイルを各セルスタックにコピーする例を確認してください。

4. メインスタックで、**passwords.yaml** ファイルを **~/overcloud-deploy/$(<stack>)** ディレクトリーにコピーします。環境内の各スタックでこの手順を繰り返します。

```
$ cp ~/overcloud-deploy/<stack>/tripleo-<stack>-passwords.yaml ~/overcloud-
deploy/<stack>/<stack>-passwords.yaml
```

   - **<stack>** は、スタックの名前に置き換えます。

5. メインスタックで、**network-data.yaml** ファイルをスタックユーザーのホームディレクトリーにコピーし、ネットワークをデプロイします。環境内の各スタックでこの手順を繰り返します。

```
$ cp ~/overcloud-deploy/<stack>/tripleo-<stack>-network-data.yaml ~/
$ mkdir ~/overcloud_adopt
$ openstack overcloud network provision --debug \
--output /home/stack/overcloud_adopt/generated-networks-deployed.yaml tripleo-<stack>-
network-data.yaml
```

   詳細は、**director を使用した Red Hat OpenStack Platform のインストールと管理**の オーバークラウドのプロビジョニングとデプロイ を参照してください。

6. メインスタックで、**virtual-ips.yaml** ファイルをスタックユーザーのホームディレクトリーにコピーし、ネットワーク VIP をプロビジョニングします。環境内の各スタックでこの手順を繰り返します。

```
$ cp ~/overcloud-deploy/<stack>/tripleo-<stack>-virtual-ips.yaml ~/
$ openstack overcloud network vip provision --debug \
--stack <stack> --output \
/home/stack/overcloud_adopt/generated-vip-deployed.yaml tripleo-<stack>-virtual-ips.yaml
```

7. メインスタックで、**baremetal-deployment.yaml** ファイルをスタックユーザーのホームディレクトリーにコピーし、オーバークラウドノードをプロビジョニングします。環境内の各スタックでこの手順を繰り返します。

```
$ cp ~/overcloud-deploy/<stack>/tripleo-<stack>-baremetal-deployment.yaml ~/
$ openstack overcloud node provision --debug --stack <stack> \
--output /home/stack/overcloud_adopt/baremetal-deployment.yaml \
tripleo-<stack>-baremetal-deployment.yaml
```

注記

これはオーバークラウド導入の最終ステップです。オーバークラウドの導入が完了するまでに 10 分以上かかる場合は、Red Hat サポートにお問い合わせください。

8. コンテナーを準備するには、以下の手順を実行します。

   a. アンダークラウドのアップグレードに使用した **containers-prepare-parameter.yaml** ファイルをバックアップします。

```
$ cp containers-prepare-parameter.yaml \
containers-prepare-parameter.yaml.orig
```

   b. スクリプトを実行して **containers-prepare-parameter.yaml** ファイルを更新する前に、以下の環境変数を定義します。

   - **NAMESPACE**: UBI9 イメージの名前空間。たとえば、**NAMESPACE='"namespace":"example.redhat.com:5002",'** など。
   - **EL8_NAMESPACE**: UBI8 イメージの名前空間。
   - **NEUTRON_DRIVER**: 使用する OpenStack Networking (neutron) コンテナーを定義するために使用するドライバー。元のスタックのデプロイに使用したコンテナーのタイプに設定します。たとえば、OVN ベースのコンテナーを使用するには、**NEUTRON_DRIVER='"neutron_driver":"ovn",'** に設定します。
   - **EL8_TAGS**: UBI8 イメージのタグ (例: **EL8_TAGS='"tag":"17.1",'**)。
     - **17.1** は、コンテンツビューで使用するタグに置き換えます。
   - **EL9_TAGS**: UBI9 イメージのタグ (例: **EL9_TAGS='"tag":"17.1",'**)。
     - **17.1** は、コンテンツビューで使用するタグに置き換えます。
       tag パラメーターの詳細は、**Red Hat OpenStack Platform デプロイメントのカスタマイズ** の コンテナーイメージ準備のパラメーター を参照してください。
   - **CONTROL_PLANE_ROLES**: **--role** オプションを使用したコントロールプレーンロールのリスト (例: **--role ControllerOpenstack, --role Database, --role Messaging, --role Networker, --role CephStorage**)。環境内のコントロールプレーンのロールのリストを表示するには、以下のコマンドを実行します。

```
$ export STACK=<stack> \
$ sudo awk '/tripleo_role_name/ {print "--role " $2}' \
/var/lib/mistral/${STACK}/tripleo-ansible-inventory.yaml \
| grep -vi compute
```

  - **<stack>** は、スタックの名前に置き換えます。

- **COMPUTE_ROLES**: **--role** オプションを使用したコンピュートロールのリスト (**--Compute-1** など)。環境内のコンピュートロールのリストを表示するには、以下のコマンドを実行します。

```
$ sudo awk '/tripleo_role_name/ {print "--role " $2}' \
/var/lib/mistral/${STACK}/tripleo-ansible-inventory.yaml \
| grep -i compute
```

- **CEPH_OVERRIDE**: Red Hat Ceph Storage をデプロイした場合は、Red Hat Ceph Storage 5 コンテナーイメージを指定します。以下に例を示します。
CEPH_OVERRIDE='"ceph_image":"rhceph-5-rhel8","ceph_tag":"<latest>",'

  - **<latest>** は、最新の **ceph_tag** バージョン (**5-499** など) に置き換えます。
  以下は、**containers-prepare-parameter.yaml** ファイル設定の例です。

```
NAMESPACE='"namespace":"registry.redhat.io/rhosp-rhel9",'
EL8_NAMESPACE='"namespace":"registry.redhat.io/rhosp-rhel8",'
NEUTRON_DRIVER='"neutron_driver":"ovn",'
EL8_TAGS='"tag":"17.1",'
EL9_TAGS='"tag":"17.1",'
CONTROL_PLANE_ROLES="--role Controller"
COMPUTE_ROLES="--role Compute"
CEPH_TAGS='"ceph_tag":"5",'
```

c. 以下のスクリプトを実行して、**containers-prepare-parameter.yaml** ファイルを更新します。

警告

Red Hat Ceph Storage をデプロイした場合は、次のコマンドを実行する前に、**CEPH_OVERRIDE** 環境変数が正しい値に設定されていることを確認してください。これを行わないと、Red Hat Ceph Storage のアップグレード時に問題が発生します。

```
$ python3 /usr/share/openstack-tripleo-heat-templates/tools/multi-rhel-container-image-prepare.py \
    ${COMPUTE_ROLES} \
    ${CONTROL_PLANE_ROLES} \
    --enable-multi-rhel \
    --excludes collectd \
    --excludes nova-libvirt \
    --minor-override "
```

```
{${EL8_TAGS}${EL8_NAMESPACE}${CEPH_OVERRIDE}${NEUTRON_DRIVER}\"no_tag\":\"not_used\"}" \
    --major-override "
{${EL9_TAGS}${NAMESPACE}${CEPH_OVERRIDE}${NEUTRON_DRIVER}\"no_tag\":\"not_used\"}" \
    --output-env-file \
   /home/stack/containers-prepare-parameter.yaml
```

multi-rhel-container-image-prepare.py スクリプトは、次のパラメーターをサポートしています。

--output-env-file

デフォルトの **ContainerImagePrepare** 値を含む環境ファイルを書き込みます。

--local-push-destination

ローカルレジストリーへのアップロードをトリガーします。

--enable-registry-login

コンテナーをプルする前に、システムがリモートレジストリーへのログインを試行できるようにするフラグを有効にします。このフラグは、**--local-push-destination** が使用されておらず、ターゲットシステムにリモートレジストリーへのネットワーク接続がある場合に使用します。このフラグは、リモートレジストリーへのネットワーク接続がない可能性があるオーバークラウドには使用しないでください。

--enable-multi-rhel

multi-rhel を有効にします。

--excludes

除外するサービスをリストします。

--major-override

メジャーリリースのオーバーライドパラメーターをリストします。

--minor-override

マイナーリリースのオーバーライドパラメーターをリストします。

--role

ロールのリスト。

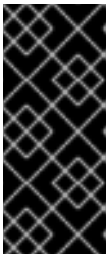
--role-file

role_data.yaml ファイル。

d. Red Hat Ceph Storage をデプロイした場合は、**containers-prepare-parameter.yaml** ファイルを開いて、Red Hat Ceph Storage 5 コンテナーイメージが指定されていること、および Red Hat Ceph Storage 6 コンテナーイメージへの参照がないことを確認します。

9. director でデプロイされた Red Hat Ceph Storage デプロイメントがある場合は、**ceph_params.yaml** というファイルを作成し、次の内容を含めます。

```
parameter_defaults:
  CephSpecFqdn: true
  CephConfigPath: "/etc/ceph"
  CephAnsibleRepo: "rhceph-5-tools-for-rhel-8-x86_64-rpms"
  DeployedCeph: true
```



重要

RHOSP のアップグレード完了後に **ceph_params.yaml** ファイルを削除しないでください。このファイルは、director でデプロイされた Red Hat Ceph Storage 環境に存在する必要があります。さらに、**openstack overcloud deploy** を実行するときは、**-e ceph_params.yaml** を指定するなどして、常に **ceph_params.yaml** ファイルを含める必要があります。

注記

Red Hat Ceph Storage デプロイメントに短縮名が含まれている場合は、**CephSpecFqdn** パラメーターを **false** に設定する必要があります。**true** に設定すると、短縮名とドメイン名の両方を使用してインベントリーが生成され、Red Hat Ceph Storage のアップグレードが失敗します。

10. テンプレートディレクトリーに **upgrades-environment.yaml** という環境ファイルを作成し、以下の内容を含めます。

```
parameter_defaults:
  ExtraConfig:
    nova::workarounds::disable_compute_service_check_for_ffu: true
  DnsServers: ["<dns_servers>"]
  DockerInsecureRegistryAddress: <undercloud_FQDN>
  UpgradeInitCommand: |
    sudo subscription-manager repos --disable=*
      if $( grep -q  9.2  /etc/os-release )
      then
        sudo subscription-manager repos --enable=rhel-9-for-x86_64-baseos-eus-rpms --enable=rhel-9-for-x86_64-appstream-eus-rpms --enable=rhel-9-for-x86_64-highavailability-eus-rpms --enable=openstack-17.1-for-rhel-9-x86_64-rpms --enable=fast-datapath-for-rhel-9-x86_64-rpms
        sudo podman ps | grep -q ceph && subscription-manager repos --enable=rhceph-5-tools-for-rhel-9-x86_64-rpms
        sudo subscription-manager release --set=9.2
      else
        sudo subscription-manager repos --enable=rhel-8-for-x86_64-baseos-tus-rpms --enable=rhel-8-for-x86_64-appstream-tus-rpms --enable=rhel-8-for-x86_64-highavailability-tus-rpms --enable=openstack-17.1-for-rhel-8-x86_64-rpms --enable=fast-datapath-for-rhel-8-x86_64-rpms
        sudo podman ps | grep -q ceph && subscription-manager repos --enable=rhceph-5-tools-for-rhel-8-x86_64-rpms
        sudo subscription-manager release --set=8.4
      fi

      if $(sudo podman ps | grep -q ceph )
      then
        sudo dnf -y install cephadm
      fi
```

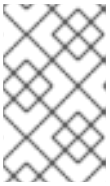
- **<dns_servers>** を、DNS サーバーの IP アドレスのコンマ区切りのリスト (**["10.0.0.36", "10.0.0.37"]** など) に置き換えます。
- **<undercloud_FQDN>** をアンダークラウドホストの完全修飾ドメイン名 (FQDN) に置き換えます (例: **"undercloud-0.ctlplane.redhat.local:8787"**)。

環境ファイルで設定できるアップグレードパラメーターに関する詳細は、アップグレード

環境ファイルで設定できるアップグレードパラメーターに関する詳細は、アップグレードパラメーター を参照してください。

11. アンダークラウドで、テンプレートディレクトリーに **overcloud_upgrade_prepare.sh** というファイルを作成します。このファイルは、環境内のスタックごとに作成する必要があります。このファイルには、オーバークラウドのデプロイファイルの元の内容と、使用中の環境に関連する環境ファイルが含まれています。以下に例を示します。

```
#!/bin/bash
openstack overcloud upgrade prepare --yes \
  --timeout 460 \
  --templates /usr/share/openstack-tripleo-heat-templates \
  --ntp-server 192.168.24.1 \
  --stack <stack> \
  -r /home/stack/roles_data.yaml \
  -e /home/stack/templates/internal.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  -e /home/stack/templates/network/network-environment.yaml \
  -e /home/stack/templates/inject-trust-anchor.yaml \
  -e /home/stack/templates/hostnames.yml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-
ansible.yaml \
  -e /home/stack/templates/nodes_data.yaml \
  -e /home/stack/templates/debug.yaml \
  -e /home/stack/templates/firstboot.yaml \
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/overcloud-params.yaml \
  -e /home/stack/overcloud-deploy/overcloud/overcloud-network-environment.yaml \
  -e /home/stack/overcloud_adopt/baremetal-deployment.yaml \
  -e /home/stack/overcloud_adopt/generated-networks-deployed.yaml \
  -e /home/stack/overcloud_adopt/generated-vip-deployed.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/nova-hw-machine-type-
upgrade.yaml \
  -e /home/stack/skip_rhel_release.yaml \
  -e ~/containers-prepare-parameter.yaml
```



注記

マルチセル環境をお使いの場合は、マルチセル環境でのオーバークラウドの導入 を参照し、セルスタックごとに **overcloud_upgrade_prepare.sh** ファイルを作成する例を確認してください。

a. 元の **network-environment.yaml** ファイル (**/home/stack/templates/network/network-environment.yaml**) で、**OS::TripleO::*::Net::SoftwareConfig** を指す resource_registry リソースをすべて削除します。

b. **overcloud_upgrade_prepare.sh** ファイルに、環境に関連する以下のオプションを含めます。

- アップグレード固有のパラメーターを持つ環境ファイル (**upgrades-environment.yaml**) (**-e**)
- 新しいコンテナーイメージの場所を定義した環境ファイル (**containers-prepare-parameter.yaml**) (**-e**)。多くの場合、アンダークラウドが使用する環境ファイルと同じファイルです。

- リリースパラメーターを持つ環境ファイル (**skip_rhel_release.yaml**) (-e)。
- デプロイメントに関連するカスタム設定環境ファイル (**-e**)
- 該当する場合は、**--roles-file** を使用してカスタムロール (**roles_data**) ファイルを指定します。
- Ceph デプロイメントの場合、Ceph パラメーターを持つ環境ファイル (**ceph_params.yaml**) (-e)。
- オーバークラウドの導入中に生成されたファイル (**network-deployed.yaml**、**vip-deployed.yaml**、**baremetal-deployment.yaml**) (-e)。
- 該当する場合、環境ファイル (**ipa-environment.yaml**) と IPA サービス (-e)。
- コンポーザブルネットワークを使用している場合は、**--network-file** を使用して (**network_data**) ファイルを指定します。

注記

オーバークラウドのデプロイファイルや **overcloud_upgrade_prepare.sh** ファイルに **network-isolation.yaml** ファイルを含めないでください。ネットワークの分離は **network_data.yaml** ファイルで定義します。

- カスタムのスタック名を使用する場合は、**--stack** オプションでその名前を渡します。

注記

環境内のすべての RHEL 8 コンピュートノードが RHEL 9 にアップグレードされるまで、テンプレートに **nova-hw-machine-type-upgrade.yaml** ファイルを含める必要があります。このファイルを除外すると、**/var/log/containers/nova** ディレクトリーの **nova_compute.log** にエラーが表示されます。すべての RHEL 8 コンピュートノードを RHEL 9 にアップグレードした後、このファイルを設定から削除してスタックを更新できます。

c. director でデプロイされた Red Hat Ceph Storage のユースケースでは、アップグレードするデプロイメントで、CephFS NFS を使用する Shared File Systems サービス (manila) を有効にしていた場合、**overcloud_upgrade_prepare.sh** スクリプトファイルの最後に追加の環境ファイルを指定する必要があります。環境ファイルは、スクリプトの前の方で指定されている別の環境ファイルをオーバーライドするため、スクリプトの最後に追加する必要があります。

```
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/manila-cephfsganesha-config.yaml
```

d. 外部 Red Hat Ceph Storage のユースケースでは、アップグレードするデプロイメントで、CephFS NFS を使用する Shared File Systems サービス (manila) を有効にしていた場合、**overcloud_upgrade_prepare.sh** スクリプト内の関連する環境ファイルが tripleo ベースの **ceph-nfs** ロールを参照することを確認する必要があります。次の環境ファイルが存在する場合は、削除します。

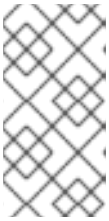
```
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/manila-cephfsganesha-config.yaml
```

次の環境ファイルを追加します。

```
-e /usr/share/openstack-tripleo-heat-templates/environments/manila-cephfsganesha-config.yaml
```

12. 環境内のスタックごとにアップグレード準備スクリプトを実行します。

```
$ source stackrc
$ chmod 755 /home/stack/overcloud_upgrade_prepare.sh
$ sh /home/stack/overcloud_upgrade_prepare.sh
```



注記

マルチセル環境をお使いの場合は、各セルスタック用に作成した **overcloud_upgrade_prepare.sh** ファイルごとにスクリプトを実行する必要があります。例については、マルチセル環境でのオーバークラウドの導入 を参照してください。

13. アップグレードの準備が完了するまで待ちます。

14. コンテナーイメージをダウンロードします。

```
$ openstack overcloud external-upgrade run --stack <stack> --tags container_image_prepare
```

5.2. マルチセル環境でのオーバークラウドの導入

オーバークラウドを導入するには、アンダークラウドのアップグレード時にエクスポートした次のファイルを、スタックユーザーのホームディレクトリーにコピーする必要があります。

- **network-data.yaml**
- **virtual-ips.yaml**
- **baremetal-deployment.yaml**

まずファイルをオーバークラウドスタックにコピーしてから、各セルスタックにコピーする必要があります。

注記

network-data.yaml ファイルは、オーバークラウドスタックでのみ使用できます。このファイルをオーバークラウドスタックから他のすべてのセルスタックにコピーする必要があります。

次の例では、**virtual-ips.yaml** ファイルをコピーします。

- オーバークラウドスタック:

```
$ cp ~/overcloud-deploy/<overcloud>/tripleo-<overcloud>-virtual-ips.yaml ~/ \
$ cd ~/ \
$ openstack overcloud network vip provision \
--debug --stack <overcloud> \
--output /home/stack/overcloud_adopt/generated-vip-deployed.yaml \ tripleo-<overcloud>-virtual-ips.yaml
```

- セルスタック 1:

```
$ cp ~/overcloud-deploy/<stack1>/tripleo-<stack1>-virtual-ips.yaml ~/ \
$ cd ~/ \
$ openstack overcloud network vip provision \
--debug --stack <stack1> \
--output /home/stack/overcloud_adopt/generated-<stack1>-vip-deployed.yaml \ tripleo-<stack1>-virtual-ips.yaml
```

- セルスタック 2:

```
$ cp ~/overcloud-deploy/<stack2>/tripleo-<stack2>-virtual-ips.yaml ~/ \
$ cd ~/ \
$ openstack overcloud network vip provision \
--debug --stack <stack2> \
--output /home/stack/overcloud_adopt/generated-<stack2>-vip-deployed.yaml \ tripleo-<stack2>-virtual-ips.yaml
```

アップグレードの準備

マルチセル環境のアップグレード準備手順を実行するには、次のステップが必要です。

1. 各セルスタック用の **overcloud_upgrade_prepare.sh** ファイルを作成します。オーバークラウドスタックから始めます。
2. セルスタック用に作成して生成した出力ファイルを **overcloud_upgrade_prepare.sh** ファイルに含めます。各セルスタック固有の環境ファイルを **overcloud_upgrade_prepare.sh** ファイルに必ず含めてください。
3. 各セルスタック用の **overcloud_upgrade_prepare.sh** スクリプトを実行します。

以下に、オーバークラウドの導入時に各セルスタック用に生成した **generated-vip-deployed.yaml** ファイルを追加する例を示します。

- オーバークラウドスタック:

```
#!/bin/bash
openstack overcloud upgrade prepare --yes \
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/overcloud-params.yaml \
-e/home/stack/overcloud_adopt/generated-vip-deployed.yaml \
...
```

  オーバークラウドスタック用の **overcloud_upgrade_prepare.sh** スクリプトを実行します。

- セルスタック 1:

```
#!/bin/bash
openstack overcloud upgrade prepare --yes \
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/stack1-params.yaml \
-e/home/stack/overcloud_adopt/generated-<stack1>-vip-deployed.yaml \
...
```

セルスタック 1 用の **overcloud_upgrade_prepare.sh** スクリプトを実行します。

- セルスタック 2:

```
#!/bin/bash
openstack overcloud upgrade prepare --yes \
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/stack2-params.yaml \
-e/home/stack/overcloud_adopt/generated-<stack2>-vip-deployed.yaml \
...
```

セルスタック 2 用の **overcloud_upgrade_prepare.sh** スクリプトを実行します。

オーバークラウドの導入と準備のプロセスの詳細は、オーバークラウドアップグレード準備の実行 を参照してください。

第6章 DIRECTOR でデプロイされた CEPH デプロイメントを使用したオーバークラウドのアップグレード

ハイパーコンバージドインフラストラクチャー (HCI) ノードの有無にかかわらず、ご使用の環境に director でデプロイされた Red Hat Ceph Storage デプロイメントが含まれている場合は、デプロイメントを Red Hat Ceph Storage 5 にアップグレードする必要があります。バージョン 5 へのアップグレードにより、**cephadm** は、**ceph-ansible** の代わりに Red Hat Ceph Storage を管理するようになりました。

6.1. CEPH-ANSIBLE のインストール

director を使用して Red Hat Ceph Storage をデプロイした場合は、この手順を完了する必要があります。Red Hat Ceph Storage を Red Hat OpenStack Platform でアップグレードするには、**ceph-ansible** パッケージが必要です。

手順

1. Ceph 5 Tools リポジトリーを有効化します。

```
[stack@director ~]$ sudo subscription-manager repos --enable=rhceph-5-tools-for-rhel-8-x86_64-rpms
```

2. **ceph-ansible** がインストールされているかどうかを確認します。

```
[stack@director ~]$ sudo rpm -q ceph-ansible
```

3. **ceph-ansible** パッケージをインストールするか、更新します。
   a. **ceph-ansible** がインストールされていない場合は、**ceph-ansible** パッケージをインストールします。

```
[stack@director ~]$ sudo dnf install -y ceph-ansible
```

   b. **ceph-ansible** がインストールされている場合には、**ceph-ansible** パッケージを最新版に更新します。

```
[stack@director ~]$ sudo dnf update -y ceph-ansible
```

6.2. RED HAT CEPH STORAGE 5 へのアップグレード

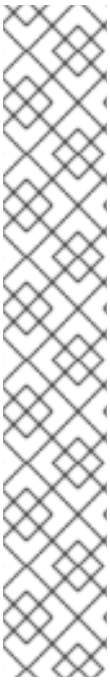
次のノードを、Red Hat Ceph Storage バージョン 4 からバージョン 5 にアップグレードします。

- Red Hat Ceph Storage ノード
- ハイパーコンバージドインフラストラクチャー (HCI) ノード。Compute サービスと Ceph OSD サービスの組み合わせが含まれています。

重要

overcloud-minimal イメージの使用による Red Hat サブスクリプションエンタイトルメントの使用回避 の説明に従って、Red Hat Ceph Storage ノードを **overcloud-minimal** イメージを使用してデプロイした場合は、この手順を実行しないでください。BZ#2257738 が完了次第、この手順を **overcloud-minimal** イメージのデプロイに対応させる予定です。

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。



注記

Red Hat Ceph Storage 5 は Prometheus v4.10 を使用します。Prometheus v4.10 には、Red Hat Ceph Storage ダッシュボードを有効にすると、ダッシュボードに 2 つのデータソースが設定されるという既知の問題があります。この既知の問題の詳細は、BZ#2054852 を参照してください。

Red Hat Ceph Storage 6 は Prometheus v4.12 を使用します。Prometheus v4.12 には、この既知の問題はありません。Red Hat は、Red Hat OpenStack Platform (RHOSP) 16.2 から 17.1 へのアップグレードが完了した後、Red Hat Ceph Storage 5 から Red Hat Ceph Storage 6 にアップグレードすることを推奨します。Red Hat Ceph Storage バージョン 5 からバージョン 6 にアップグレードするには、ご使用の環境に応じて次のいずれかの手順を実行します。

- director でデプロイされた Red Hat Ceph Storage 環境: **cephadm** クライアントの更新
- 外部 Red Hat Ceph Storage クラスター環境: Red Hat Ceph Storage コンテナーイメージの更新

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

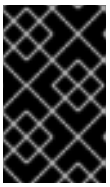
```
$ source ~/stackrc
```

3. **ceph** タグを使用して、Red Hat Ceph Storage の外部アップグレードプロセスを実行します。

```
$ openstack overcloud external-upgrade run \
  --skip-tags "ceph_ansible_remote_tmp" \
  --stack <stack> \
  --tags ceph,facts 2>&1
```

   - **<stack>** は、スタックの名前に置き換えます。
   - DCN がデプロイされたサイトでこのコマンドを実行する場合は、**--skip-tags** パラメーターに指定されたコンマ区切りの値リストに、skip-tag **cleanup_cephansible** の値を追加します。

4. **ceph versions** コマンドを実行して、すべての Red Hat Ceph Storage デーモンがバージョン 5 にアップグレードされたことを確認します。このコマンドは、コントローラーノードでデフォルトでホストされる **ceph monitor** コンテナーで使用できます。



重要

前の手順のコマンドは、**ceph-ansible rolling_update.yaml** Playbook を実行して、クラスターをバージョン 4 から 5 に更新します。この手順を続行する前に、すべてのデーモンが更新されていることを確認することが重要です。

次の例は、このコマンドの使用方法と出力を示しています。例に示されているように、デプロイメント内のすべてのデーモンには、パッケージバージョン **16.2.*** とキーワード **pacific** が表示されます。

```
$ sudo podman exec ceph-mon-$(hostname -f) ceph versions
{
    "mon": {
        "ceph version 16.2.10-248.el8cp (0edb63afd9bd3edb333364f2e0031b77e62f4896)
pacific (stable)": 3
    },
    "mgr": {
        "ceph version 16.2.10-248.el8cp (0edb63afd9bd3edb333364f2e0031b77e62f4896)
pacific (stable)": 3
    },
    "osd": {
        "ceph version 16.2.10-248.el8cp (0edb63afd9bd3edb333364f2e0031b77e62f4896)
pacific (stable)": 180
    },
    "mds": {},
    "rgw": {
        "ceph version 16.2.10-248.el8cp (0edb63afd9bd3edb333364f2e0031b77e62f4896)
pacific (stable)": 3
    },
    "overall": {
        "ceph version 16.2.10-248.el8cp (0edb63afd9bd3edb333364f2e0031b77e62f4896)
pacific (stable)": 189
    }
}
```

注記

Red Hat Ceph Storage をホストしている任意のサーバー上で **sudo podman ps | grep ceph** コマンドを実行すると、バージョン 5 のコンテナーが返されるはずです。

5. **ceph-admin** ユーザーを作成し、適切なキーリングを配布します。

```
ANSIBLE_LOG_PATH=/home/stack/cephadm_enable_user_key.log \
ANSIBLE_HOST_KEY_CHECKING=false \
ansible-playbook -i /home/stack/overcloud-deploy/<stack>/config-download/<stack>/tripleo-
ansible-inventory.yaml \
  -b -e ansible_python_interpreter=/usr/libexec/platform-python /usr/share/ansible/tripleo-
playbooks/ceph-admin-user-playbook.yml \
 -e tripleo_admin_user=ceph-admin \
 -e distribute_private_key=true \
  --limit
Undercloud,ceph_mon,ceph_mgr,ceph_rgw,ceph_mds,ceph_nfs,ceph_grafana,ceph_osd
```

6. Red Hat Ceph Storage ノード上のパッケージを更新します。

```
$ openstack overcloud upgrade run \
    --stack <stack> \
    --skip-tags ceph_ansible_remote_tmp \
    --tags setup_packages --limit
Undercloud,ceph_mon,ceph_mgr,ceph_rgw,ceph_mds,ceph_nfs,ceph_grafana \
    --playbook /home/stack/overcloud-deploy/<stack>/config-
download/<stack>/upgrade_steps_playbook.yaml 2>&1
```

- DCN がデプロイされたサイトでこのコマンドを実行する場合は、**--skip-tags** パラメーターに指定されたコンマ区切りの値リストに、skip-tag **cleanup_cephansible** の値を追加します。

注記

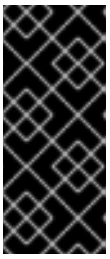
デフォルトでは、コンポーザブルロール機能を使用して他の場所でホストしていない限り、Ceph Monitor サービス (CephMon) はコントローラーノード上で実行し　ます。このコマンドには **ceph_mon** タグが含まれており、Ceph Monitor サービスをホストしているノード (デフォルトではコントローラーノード) 上のパッケージも更新されます。

7. **cephadm** を使用するように Red Hat Ceph Storage ノードを設定します。

```
$ openstack overcloud external-upgrade run \
    --skip-tags ceph_ansible_remote_tmp \
    --stack <stack> \
    --tags cephadm_adopt  2>&1
```

- DCN がデプロイされたサイトでこのコマンドを実行する場合は、**--skip-tags** パラメーターに指定されたコンマ区切りの値リストに、skip-tag **cleanup_cephansible** の値を追加します。

8. **ceph -s** コマンドを実行して、すべてのプロセスが Red Hat Ceph Storage オーケストレーターによって管理されていることを確認します。このコマンドは、コントローラーノードでデフォルトでホストされる **ceph monitor** コンテナーで使用できます。



重要

前の手順のコマンドは、**ceph-ansible cephadm-adopt.yaml** Playbook を実行して、クラスターの今後の管理を **ceph-ansible** から **cephadm** および Red Hat Ceph Storage オーケストレーターに移行します。この手順を続行する前に、すべてのプロセスがオーケストレータによって管理されていることを確認することが重要です。

次の例は、このコマンドの使用方法と出力を示しています。この例で示されているように、**cephadm** によって管理されていないデーモンは 63 個あります。これは、**ceph-ansible cephadm-adopt.yml** Playbook の実行に問題があったことを示しています。アップグレードを続行する前に、Red Hat Ceph Storage サポートに連絡してこれらのエラーのトラブルシューティングを行ってください。導入プロセスが正常に完了すると、**cephadm** によって管理されていない迷子のデーモンに関する警告は表示されなくなります。

```
$ sudo cephadm shell -- ceph -s
  cluster:
```

```
    id:     f5a40da5-6d88-4315-9bb3-6b16df51d765
    health: HEALTH_WARN
            63 stray daemon(s) not managed by cephadm
```

9. **overcloud_upgrade_prepare.sh** ファイルを変更して、**ceph-ansible** ファイルを **cephadm** heat 環境ファイルに置き換えます。

```
#!/bin/bash
openstack overcloud upgrade prepare  --yes \
  --timeout 460 \
 --templates /usr/share/openstack-tripleo-heat-templates \
  --ntp-server 192.168.24.1 \
  --stack <stack> \
  -r /home/stack/roles_data.yaml \
  -e /home/stack/templates/internal.yaml \
  …
  -e /usr/share/openstack-tripleo-heat-templates/environments/cephadm/cephadm-rbd-
only.yaml \
  -e ~/containers-prepare-parameter.yaml
```

注記

この例では、RGW がデプロイされていないため、**environments/cephadm/cephadm-rbd-only.yaml** ファイルを使用します。RGW のデプロイを計画している場合は、RHOSP 環境のアップグレードが完了した後に **environments/cephadm/cephadm.yaml** を使用し、スタックの更新を実行します。

10. オーバークラウドのアップグレード準備用のコマンドを実行した際に以下の環境ファイルを追加した場合は、**overcloud_upgrade_prepare.sh** ファイルを変更してこの環境ファイルを削除します。

```
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/manila-
cephfsganesha-config.yaml
```

11. ファイルを保存します。

12. upgrade prepare コマンドを実行します。

```
$ source stackrc
$ chmod 755 /home/stack/overcloud_upgrade_prepare.sh
sh /home/stack/overcloud_upgrade_prepare.sh
```

13. デプロイメントに HCI ノードが含まれている場合は、コントローラーノードの **cephadm** コンテナーに一時的な **hci.conf** ファイルを作成します。

    a. コントローラーノードにログインします。

    ```
    $ ssh cloud-admin@<controller_ip>
    ```

    - **<controller_ip>** をコントローラーノードの IP アドレスに置き換えます。

    b. コントローラーノードから **cephadm** シェルを取得します。

例

```
[cloud-admin@controller-0 ~]$ sudo cephadm shell
```

c. **cephadm** シェルで、一時的な **hci.conf** ファイルを作成します。

例

```
[ceph: root@edpm-controller-0 /]# cat <<EOF > hci.conf
[osd]
osd_memory_target_autotune = true
osd_numa_auto_affinity = true
[mgr]
mgr/cephadm/autotune_memory_target_ratio = 0.2
EOF
…
```

d. 設定を適用します。

例

```
[ceph: root@edpm-controller-0 /]# ceph config assimilate-conf -i hci.conf
```

HCI デプロイメントの設定の調整について、詳しくは**ハイパーコンバージドインフラストラクチャーをデプロイする** の HCI の Ceph 設定オーバーライド を参照してください。

重要

すべての HCI ノードのオペレーティングシステムを、RHEL 9 にアップグレードする必要があります。コンピュートノードと HCI ノードのアップグレードの詳細は、コンピュートノードを RHEL 9.2 にアップグレードする を参照してください。

Red Hat Ceph Storage クラスターはバージョン 5 にアップグレードされました。これには次の意味があります。

- 今後は、Red Hat Ceph Storage の管理に **ceph-ansible** を使用しません。代わりに、Ceph Orchestrator が Red Hat Ceph Storage クラスターを管理します。Ceph Orchestrator の詳細は、Red Hat Ceph Storage オペレーションガイド を参照してください。

- 今後は、ほとんどの場合において、Red Hat Ceph Storage クラスターに変更を加えるためにスタック更新を実行する必要がなくなります。代わりに、 Red Hat Ceph Storage オペレーションガイド で説明されているように、Day Two の Red Hat Ceph Storage 操作をクラスター上で直接実行できます。また、**director を使用した Red Hat Ceph Storage および Red Hat OpenStack Platform のデプロイ** の Ceph Storage クラスターのスケールアップ で説明されているように、Red Hat Ceph Storage クラスターノードをスケールアップまたはスケールダウンすることもできます。

- Red Hat Ceph Storage クラスターの健全性を検査します。クラスターの健全性のモニタリングに関する詳細は、**director を使用した Red Hat Ceph Storage および Red Hat OpenStack Platform のデプロイ** の Red Hat Ceph Storage ノードのモニタリング を参照してください。

- **openstack overcloud deploy** などの openstack デプロイメントコマンドに、**environments/ceph-ansible/ceph-ansible.yaml** などの環境ファイルを含めないでください。デプロイメントに **ceph-ansible** 環境ファイルが含まれている場合は、それらを以下のオプ

ションのいずれかに置き換えます。

Red Hat Ceph Storage のデプロイメント	元の ceph-ansible ファイル	Cephadm ファイルの置換
Ceph RADOS ブロックデバイス (RBD) のみ	任意の **ceph-ansible** 環境ファイル	**environments/cephadm/cephadm-rbd-only.yaml**
RBD と Ceph Object Gateway (RGW)	任意の **ceph-ansible** 環境ファイル	**environments/cephadm/cephadm.yaml**
Ceph Dashboard	**environments/ceph-ansible/ceph-dashboard.yaml**	**environments/cephadm/** 内のそれぞれのファイル
Ceph MDS	**environments/ceph-ansible/ceph-mds.yaml**	**environments/cephadm/** 内のそれぞれのファイル

第7章 ネットワーク機能仮想化 (NFV) の準備

ネットワーク機能仮想化 (NFV) を使用する場合は、オーバークラウドのアップグレードに向けて準備タスクを完了する必要があります。

7.1. ネットワーク機能仮想化 (NFV) 用環境ファイル

典型的な NFV ベースの環境では、以下のようなサービスを有効にすることができます。

- Single-root input/output virtualization (SR-IOV)
- Data Plane Development Kit (DPDK)

Red Hat OpenStack Platform 17.1 へのアップグレードに対応するために、これらのサービスに対して特定の再設定を行う必要はありません。ただし、NFV 機能を有効にする環境ファイルは、以下の要件を満たすようにしてください。

- NFV 機能を有効にするデフォルトの環境ファイルは、Red Hat OpenStack Platform 17.1 **openstack-tripleo-heat-templates** コレクションの **environments/services** ディレクトリーにあります。Red Hat OpenStack Platform 16.2 デプロイメントに **openstack-tripleo-heat-templates** からのデフォルト NFV 環境ファイルを追加している場合は、Red Hat OpenStack Platform 17.1 での該当機能の正しい環境ファイルの場所を確認してください。
  - Open vSwitch (OVS) ネットワークおよび SR-IOV: **/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-sriov.yaml**
  - Open vSwitch (OVS) ネットワークおよび DPDK: **/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs-dpdk.yaml**
- Red Hat OpenStack Platform 16.2 から Red Hat OpenStack Platform 17.1 へのアップグレード中に OVS の互換性を維持するには、**/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml** 環境ファイルを含める必要があります。環境ファイルを指定してデプロイメントおよびアップグレードのコマンドを実行する際には、**neutron-ovs.yaml** ファイルの後に NFV 関連の環境ファイルをすべて追加する必要があります。たとえば、OVS および NFV 環境ファイルを指定して **openstack overcloud upgrade prepare** を実行する場合は、以下の順序でファイルを追加します。
- OVS 用環境ファイル
- SR-IOV 用環境ファイル
- DPDK 用環境ファイル

```
$ openstack overcloud upgrade prepare \
  ...
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-sriov.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs-dpdk.yaml \
  ...
```

注記

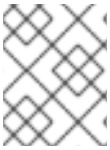
NFV ワークロードには、移行に関する制約があります。アップグレード中に、OVS-DPDK コンピュートノードからのインスタンスのライブマイグレーションを行うことはできません。代替の手段として、アップグレード中に、OVS-DPDK Compute ノードからのインスタンスのコールドマイグレーションを行うことができます。

第8章 オーバークラウドのアップグレード

環境内の各スタック上のオーバークラウド全体で Red Hat OpenStack Platform コンテンツをアップグレードします。

8.1. 各スタック内のすべてのノードでの RHOSP のアップグレード

メインスタックから開始して、スタックごとにすべてのオーバークラウドノードを Red Hat OpenStack Platform (RHOSP) 17.1 にアップグレードします。



注記

オーバークラウドノードをアップグレードする前に、Pacemaker がすべてのコントローラーで実行されていることを確認する必要があります。

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。

手順

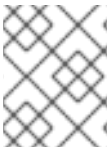
1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. メインスタック内のすべてのノードで RHOSP をアップグレードします。

```
$ openstack overcloud upgrade run --yes --stack <stack> --debug --limit
allovercloud,undercloud --playbook all
```

- <stack> をノードをアップグレードするオーバークラウドスタックの名前に置き換えます。RHOSP デプロイメント内のスタックごとにこの手順を繰り返します。



注記

マルチセル環境をお使いの場合は、オーバークラウドスタック上の RHOSP をアップグレードする前に、セルスタック上の RHOSP をアップグレードする必要があります。

第9章 アンダークラウドのオペレーティングシステムのアップグレード

アンダークラウドオペレーティングシステムを Red Hat Enterprise Linux 8.4 から Red Hat Enterprise Linux 9.2 にアップグレードする必要があります。システムのアップグレードでは次のタスクが実行されます。

- システムのアップグレード後も、ネットワークインターフェイスの命名が一貫していることを確認します。
- Leapp を使用して RHEL をインプレースアップグレードします。
- アンダークラウドを再起動します。

9.1. アンダークラウドでの SSH ROOT パーミッションパラメーターの設定

Leapp によるアップグレードでは、**PermitRootLogin** パラメーターが **/etc/ssh/sshd_config** ファイルに存在するかどうかを確認します。このパラメーターを、明示的に **yes** または **no** のいずれかに設定する必要があります。

セキュリティー上の理由から、アンダークラウドで root ユーザーへの SSH アクセスを無効にするには、このパラメーターを **no** に設定します。

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. **/etc/ssh/sshd_config** ファイルに **PermitRootLogin** パラメーターがあるかどうかを確認します。

```
$ sudo grep PermitRootLogin /etc/ssh/sshd_config
```

3. **/etc/ssh/sshd_config** ファイルにパラメーターがない場合は、ファイルを編集して **PermitRootLogin** パラメーターを設定します。

```
PermitRootLogin no
```

4. ファイルを保存します。

9.2. SSH キーのサイズの検証

Red Hat Enterprise Linux (RHEL) 9.1 以降では、最小 2048 ビットの SSH キーサイズが必要です。Red Hat OpenStack Platform (RHOSP) director 上の現在の SSH キーが 2048 ビット未満の場合、オーバークラウドにアクセスできなくなる可能性があります。SSH キーが必要なビットサイズを満たしていることを確認する必要があります。

手順

1. SSH キーのサイズを検証します。

```
ssh-keygen -l -f /home/stack/overcloud-deploy/overcloud/ssh_private_key
```

出力例:

```
1024 SHA256:Xqz0Xz0/aJua6B3qRD7VsLr6n/V3zhmnGSkcFR6FlJw
stack@director.example.local (RSA)
```

2. SSH キーが 2048 ビット未満の場合は、次に進む前に、SSH キーをローテーションする必要があります。詳細は、**Red Hat OpenStack Platform の強化** の OpenStack 環境での SSH キーの更新 を参照してください。

9.3. アンダークラウドシステムのアップグレードの実行

アンダークラウドオペレーティングシステムを Red Hat Enterprise Linux (RHEL) 9.2 にアップグレードします。このアップグレードの一環として、**system_upgrade.yaml** という名前のファイルを作成します。このファイルを使用して、Leapp をインストールするための適切なリポジトリー、必要な Red Hat OpenStack Platform オプションおよびコンテンツを有効化します。このファイルを使用して、コントロールプレーンノードとコンピュートノードもアップグレードします。

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。

手順

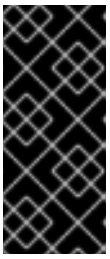
1. アンダークラウドに **stack** ユーザーとしてログインします。
2. テンプレートディレクトリーに **system_upgrade.yaml** という名前のファイルを作成し、以下の内容を含めます。

```
parameter_defaults:
  UpgradeLeappDevelSkip: "LEAPP_UNSUPPORTED=1
LEAPP_DEVEL_SKIP_CHECK_OS_RELEASE=1 LEAPP_NO_NETWORK_RENAMING=1
LEAPP_DEVEL_TARGET_RELEASE=9.2"
  UpgradeLeappDebug: false
  UpgradeLeappEnabled: true
  LeappActorsToRemove:
['checkifcfg','persistentnetnamesdisable','checkinstalledkernels','biosdevname']
  LeappRepoInitCommand: |
     subscription-manager repos --disable=*
     subscription-manager repos --enable rhel-8-for-x86_64-baseos-tus-rpms --enable rhel-8-
for-x86_64-appstream-tus-rpms --enable openstack-17.1-for-rhel-8-x86_64-rpms
     subscription-manager release --set=8.4
  UpgradeLeappCommandOptions: "--enablerepo=rhel-9-for-x86_64-baseos-eus-rpms --
enablerepo=rhel-9-for-x86_64-appstream-eus-rpms --enablerepo=rhel-9-for-x86_64-
highavailability-eus-rpms --enablerepo=openstack-17.1-for-rhel-9-x86_64-rpms --
enablerepo=fast-datapath-for-rhel-9-x86_64-rpms"
  UpgradeLeappToRemove: ['irb']
```

注記

デプロイメントに Red Hat Ceph Storage ノードが含まれている場合は、**CephLeappRepoInitCommand** パラメーターを追加し、Red Hat Ceph Storage ノードのソース OS バージョンを指定する必要があります。以下に例を示します。

```
CephLeappRepoInitCommand:
...
  subscription-manager release --set=8.6
```



重要

RHEL 8 の ruby-irb ディレクトリーと RHEL 9 のシンボリックリンクの間の競合を避けるために、**ruby-irb** パッケージを削除する必要があります。詳細は、Red Hat ナレッジベースのソリューション leapp upgrade RHEL8 to RHEL9 fails with error "rubygem-irb-1.3.5-160.el9_0.noarch conflicts with file from package ruby-irb-2.5.9-110.module+el8.6.0+15956+aa803fc1.noarch" を参照してください。

3. **LeappInitCommand** パラメーターを **system_upgrade.yaml** ファイルに追加して、環境に適用される追加の要件を指定します。たとえば、ロールベースのオーバーライドを定義する必要がある場合は、以下のようになります。

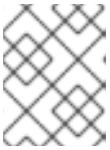
```
  LeappInitCommand: |
    subscription-manager repos --disable=*
    subscription-manager release --unset
    subscription-manager repos --enable=rhel-9-for-x86_64-baseos-eus-rpms --enable=rhel-9-for-x86_64-appstream-eus-rpms --enable=rhel-9-for-x86_64-highavailability-eus-rpms --enable=openstack-17.1-for-rhel-9-x86_64-rpms --enable=fast-datapath-for-rhel-9-x86_64-rpms
    leapp answer --add --section check_vdo.confirm=True
```

4. カーネルベースの NIC 名を使用する場合は、以下のパラメーターを **system_upgrade.yaml** ファイルに追加して、アップグレードプロセス全体にわたって NIC 名が保持されるようにします。

```
parameter_defaults:
  NICsPrefixesToUdev: ['en']
...
```

5. Leapp アップグレードを実行します。

```
$ openstack undercloud upgrade --yes --system-upgrade \
/home/stack/system_upgrade.yaml
```



注記

Leapp アップグレードを再度実行する必要がある場合は、まずリポジトリーを RHEL 8 にリセットする必要があります。

6. アンダークラウドをリブートします。

```
$ sudo reboot
```

第10章 コントロールプレーンオペレーティングシステムのアップグレード

コントロールプレーンノード上のオペレーティングシステムをアップグレードします。アップグレードには以下のタスクが含まれます。

- システムアップグレードパラメーターを指定した overcloud upgrade prepare コマンドの実行
- オーバークラウドシステムのアップグレードを実行します。これは、Leapp を使用して RHEL をインプレースでアップグレードします。
- ノードの再起動

10.1. コントロールプレーンノードのアップグレード

環境内のコントロールプレーンノードを Red Hat Enterprise Linux 9.2 にアップグレードするには、ブートストラップノードから開始して、コントロールプレーンノードの 3 分の 1 を一度にアップグレードする必要があります。

コントロールプレーンノードをアップグレードするには、**openstack overcloud upgrade run** コマンドを使用します。このコマンドにより、以下のアクションが行われます。

- Leapp によるオペレーティングシステムのアップグレードを実施する。
- Leapp によるアップグレードの一部としてリブートを実施する。

システムのアップグレード中に各ノードが再起動されます。このダウンタイム中に Pacemaker クラスターと Red Hat Ceph Storage クラスターのパフォーマンスは低下しますが、停止することはありません。

この例には、コンポーザブルロールを持つ以下のノードが含まれています。

- **controller-0**
- **controller-1**
- **controller-2**
- **database-0**
- **database-1**
- **database-2**
- **networker-0**
- **networker-1**
- **networker-2**
- **ceph-0**
- **ceph-1**
- **ceph-2**

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。

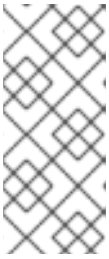
手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. **CONTROL_PLANE_ROLES** パラメーターを指定せずに以下のスクリプトを実行します。オーバークラウドアップグレード準備の実行 でコンテナーを準備するために使用した変数を必ず含めてください。

```
python3 \
/usr/share/openstack-tripleo-heat-templates/tools/multi-rhel-container-image-prepare.py \
    ${COMPUTE_ROLES} \
    --enable-multi-rhel \
    --excludes collectd \
    --excludes nova-libvirt \
    --minor-override \
"
{${EL8_TAGS}${EL8_NAMESPACE}${CEPH_OVERRIDE}${NEUTRON_DRIVER}\"no_tag\":
\"not_used\"}" \
    --major-override \
    "
{${EL9_TAGS}${NAMESPACE}${CEPH_OVERRIDE}${NEUTRON_DRIVER}\"no_tag\":\"not
_used\"}" \
    --output-env-file \
/home/stack/containers-prepare-parameter.yaml
```



注記

CONTROL_PLANE_ROLES パラメーターは、コントロールプレーンロールのリストを定義します。このパラメーターをスクリプトから削除すると、RHEL 9.2 へのアップグレード用のコントロールプレーンロールが準備されます。**CONTROL_PLANE_ROLES** パラメーターがスクリプトに含まれている場合は、コントロールプレーンのロールは RHEL 8.4 に残ります。

4. **skip_rhel_release.yaml** ファイルで、**SkipRhelEnforcement** パラメーターを **false** に設定します。

```
parameter_defaults:
  SkipRhelEnforcement: false
```

5. **overcloud_upgrade_prepare.sh** ファイルを更新します。

```
$ openstack overcloud upgrade prepare --yes \
    ...
    -e /home/stack/system_upgrade.yaml \
    -e /home/stack/containers-prepare-parameter.yaml \
    -e /home/stack/skip_rhel_release.yaml \
    ...
```

–

- アップグレード固有のパラメーターを持つ **system_upgrade.yaml** ファイルを含めます (-e)。
- コントロールプレーンロールを削除した **containers-prepare-parameter.yaml** ファイルを含めます (-e)。
- リリースパラメーターを持つ **skip_rhel_release.yaml** ファイルを含めます (-e)。

6. **overcloud_upgrade_prepare.sh** スクリプトを実行します。

```
$ sh /home/stack/overcloud_upgrade_prepare.sh
```

7. システムのアップグレードに必要な新しいコンテナーまたは変更されたコンテナーを取得します。

```
$ openstack overcloud external-upgrade run  \
    --stack <stack> \
    --tags container_image_prepare 2>&1
```

8. コントロールプレーンノードの最初の 3 分の 1 をアップグレードします。

```
$ openstack overcloud upgrade run --yes \
    --stack <stack> \
    --tags system_upgrade \
    --limit <controller-0>,<database-0>,<messaging-0>,<networker-0>,<ceph-0>
```

- **<stack>** は、スタックの名前に置き換えます。
- **<controller-0>**、**<database-0>**、**<messaging-0>**、**<networker-0>**、**<ceph-0>** を独自のノード名に置き換えます。

9. アップグレードされた各ノードにログインし、各ノードのクラスターが実行されていることを確認します。

```
$ sudo pcs status
```

コントロールプレーンノードの次の 3 分の 1 をアップグレードした後、そしてコントロールプレーンノードの最後の 3 分の 1 をアップグレードした後に、この検証手順を繰り返します。

10. コントロールプレーンノードの次の 3 分の 1 をアップグレードします。

```
$ openstack overcloud upgrade run --yes \
    --stack <stack> \
    --tags system_upgrade \
    --limit <controller-1>,<database-1>,<messaging-1>,<networker-1>,<ceph-1>
```

- **<controller-1>**、**<database-1>**、**<messaging-1>**、**<networker-1>**、**<ceph-1>** を独自のノード名に置き換えます。

11. コントロールプレーンノードの最後の 3 分の 1 をアップグレードします。

```
$ openstack overcloud upgrade run --yes \
    --stack <stack> \
    --tags system_upgrade \
```

```
    --limit <controller-2>,<database-2>,<messaging-2>,<networker-2>,<ceph-2>
```

- **<controller-2>**、**<database-2>**、**<messaging-2>**、**<networker-2>**、**<ceph-2>** を独自のノード名に置き換えます。

12. STF を有効にした場合は、タグを付けずにアップグレードコマンドを実行します。オペレーティングシステムのアップグレード後にこのコマンドを実行して、すべてのノードの **collectd** コンテナーを更新します。

```
$ openstack overcloud upgrade run --yes \
    --stack <stack> \
    --limit <controller-0>,<controller-1>,<controller-2>,<database-0>,<database-1>,
<database-2>,<networker-0>,<networker-1>,<networker-2>,<ceph-0>,<ceph-1>,<ceph-2>
```

- **<controller-0>**、**<controller-1>**、**<controller-2>**、**<database-0>**、**<database-1>**、**<database-2>**、**<networker-0>**、**<networker-1>**、**<networker-2>**、**<ceph-0>**、**<ceph-1>**、**<ceph-2>** を、独自のノード名に置き換えます。

第11章 コンピュートノードのオペレーティングシステムのアップグレード

すべてのコンピュートノードのオペレーティングシステムを RHEL 9.2 にアップグレードすることも、一部のコンピュートノードをアップグレードし、残りのコンピュートノードは RHEL 8.4 のままにすることもできます。

重要

デプロイメントにハイパーコンバージドインフラストラクチャー (HCI) ノードが含まれている場合は、すべての HCI ノードを RHEL 9 にアップグレードする必要があります。RHEL 9 へのアップグレードの詳細は、コンピュートノードを RHEL 9.2 にアップグレードする を参照してください。

このアップグレード手順の所要時間と影響については、アップグレードの所要時間と影響 を参照してください。

前提条件

- コンピュートノードのアップグレード計画 を確認している。

11.1. アップグレードテスト用の COMPUTE ノードの選択

オーバークラウドのアップグレードプロセスでは、次のいずれかを行うことができます。

- 1 つのロールのノードをすべてアップグレードする。
- 個々のノードを個別にアップグレードする。

オーバークラウドのアップグレードプロセスを円滑にするには、全 Compute ノードをアップグレードする前に、環境内にある個々のCompute ノードのいくつかでアップグレードをテストすると役立ちます。これにより、アップグレード中に大きな問題が発生しなくなり、ワークロードのダウンタイムを最小限に抑えることができます。

アップグレードをテストするノードを選択するにあたっては、以下の推奨事項を参考にしてください。

- アップグレードのテストには、2 台または 3 台のコンピュートノードを選択します。
- クリティカルなインスタンスが実行されていないノードを選択します。
- 必要に応じて、選択したテストコンピュートノードからクリティカルインスタンスを他のコンピュートノードに移行します。どの移行シナリオがサポートされているかを確認してください。

ソースコンピュートノードの RHEL バージョン	宛先コンピュートノードの RHEL バージョン	サポート対象/サポート対象外
RHEL 8	RHEL 8	サポート対象
RHEL 8	RHEL 9	サポート対象
RHEL 9	RHEL 9	サポート対象

ソースコンピュートノードの RHEL バージョン	宛先コンピュートノードの RHEL バージョン	サポート対象/サポート対象外
RHEL 9	RHEL 8	サポート対象外

11.2. すべてのコンピュートノードを RHEL 9.2 にアップグレードする

すべてのコンピュートノードを RHEL 9.2 にアップグレードして、最新の機能を利用し、ダウンタイムを削減します。

前提条件

- デプロイメントにハイパーコンバージドインフラストラクチャー (HCI) ノードが含まれている場合は、ホストをメンテナンスモードにして、各 HCI ノード上で Red Hat Ceph Storage クラスターを再起動できるように準備します。詳細は、**Ceph オペレーションガイド** の Ceph Orchestrator を使用してホストをメンテナンスモードにする を参照してください。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. **container-image-prepare.yaml** ファイルには、**ContainerImagePrepare** パラメーターで指定されたタグのみが含まれ、**MultiRhelRoleContainerImagePrepare** パラメーターが削除されていることを確認します。以下に例を示します。

```
parameter_defaults:
  ContainerImagePrepare:
  - tag_from_label: "{version}-{release}"
    set:
      namespace:
      name_prefix:
      name_suffix:
      tag:
      rhel_containers: false
      neutron_driver: ovn
      ceph_namespace:
      ceph_image:
      ceph_tag:
```

4. **role_data.yaml** ファイルで、**OS::TripleO::Services::NovaLibvirtLegacy** サービスを、RHEL 9.2 に必要な **OS::TripleO::Services::NovaLibvirt** サービスに置き換えます。
5. 次の例に示すように、**-e system_upgrade.yaml** 引数と必要なその他の **-e** 環境ファイル引数を **overcloud_upgrade_prepare.sh** スクリプトに含めます。

```
$ openstack overcloud upgrade prepare --yes
…
-e /home/stack/system_upgrade.yaml
```

```
...
```

6. **overcloud_upgrade_prepare.sh** スクリプトを実行します。

7. コンピュートノード上のオペレーティングシステムを RHEL 9.2 にアップグレードします。アップグレードするノードのコンマ区切りリストを指定して、**--limit** オプションを使用します。以下の例では、**compute-0**、**compute-1**、および **compute-2** ノードをアップグレードします。

```
$ openstack overcloud upgrade run --yes --tags system_upgrade --stack <stack> --limit
compute-0,compute-1,compute-2
```

   - **<stack>** は、スタックの名前に置き換えます。

8. コンピュートノード上のコンテナーを RHEL 9.2 にアップグレードします。アップグレードするノードのコンマ区切りリストを指定して、**--limit** オプションを使用します。以下の例では、**compute-0**、**compute-1**、および **compute-2** ノードをアップグレードします。

```
$ openstack overcloud upgrade run --yes --stack <stack>  --limit compute-0,compute-
1,compute-2
```

11.3. コンピュートノードの MULTI-RHEL 環境へのアップグレード

コンピュートノードの一部を RHEL 9.2 にアップグレードし、残りのコンピュートノードは RHEL 8.4 のままにすることができます。このアップグレードプロセスには、以下の基本的な手順が含まれます。

1. どのノードを RHEL 9.2 にアップグレードするか、そしてどのノードを RHEL 8.4 に残しておきたいかを計画します。ノードのバッチごとに作成する各ロールのロール名を選択します (例: **ComputeRHEL-9.2** および **ComputeRHEL-8.4**)。

2. RHEL 9.2 にアップグレードするノード、または RHEL 8.4 に残しておきたいノードを保存するロールを作成します。これらのロールは、コンピュートノードを新しいロールに移動する準備ができるまで空のままにすることができます。必要な数のロールを作成し、任意の方法でロール間でノードを分割できます。以下に例を示します。

   - 環境で **ComputeSRIOV** というロールが使用されており、カナリアテストを実行して RHEL 9.2 にアップグレードする必要がある場合は、新しい **ComputeSRIOVRHEL9** ロールを作成し、カナリアノードを新しいロールに移動できます。

   - 環境で **ComputeOffload** というロールを使用しており、そのロール内のほとんどのノードを RHEL 9.2 にアップグレードするが、いくつかのノードを RHEL 8.4 に残しておきたい場合は、新しい **ComputeOffloadRHEL8** ロールを作成して RHEL 8.4 ノードを保存できます。その後、元の **ComputeOffload** ロール内のノードを選択して、RHEL 9.2 にアップグレードできます。

3. ノードを各コンピュートロールから新しいロールに移動します。

4. 特定のコンピュートノード上のオペレーティングシステムを RHEL 9.2 にアップグレードします。同じロールまたは複数のロールから、ノードをバッチでアップグレードできます。

注記

Multi-RHEL 環境では、デプロイメントでは引き続き pc-i440fx マシンタイプを使用する必要があります。デフォルトを Q35 に更新しないでください。Q35 マシンタイプへの移行は、すべてのコンピュートノードを RHEL 9.2 にアップグレードした後に実行する、別のアップグレード後の手順です。Q35 マシンタイプの移行に関する詳細は、RHOSP 17 へのアップグレード後のホストのデフォルトマシンタイプの更新 を参照してください。

次の手順を使用して、コンピュートノードを Multi-RHEL 環境にアップグレードします。

- Multi-RHEL コンピュートノードのロールの作成
- コンピュートノードのオペレーティングシステムのアップグレード

11.3.1. Multi-RHEL コンピュートノードのロールの作成

RHEL 9.2 にアップグレードするノード、または RHEL 8.4 に留まるノードを保存する新しいロールを作成し、ノードを新しいロールに移動します。

手順

1. 環境に関連するロールを作成します。**role_data.yaml** ファイルで、新しいロールに使用するソースのコンピュートロールをコピーします。
   必要な追加のロールごとにこの手順を繰り返します。コンピュートノードを新しいロールに移動する準備ができるまで、ロールは空のままにすることができます。

   - RHEL 8 ロールを作成している場合は、以下のようになります。

```
name: <ComputeRHEL8>
 description: |
  Basic Compute Node role
 CountDefault: 1
 rhsm_enforce_multios: 8.4
...
ServicesDefault:
...
- OS::TripleO::Services::NovaLibvirtLegacy
```

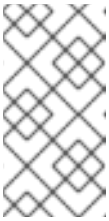
注記

RHEL 8.4 に残っているノードを含むロールには、**NovaLibvirtLegacy** サービスが含まれている必要があります。

   - **<ComputeRHEL8>** を RHEL 8.4 ロールの名前に置き換えます。
   - RHEL 9 ロールを作成している場合は、以下のようになります。

```
name: <ComputeRHEL9>
 description: |
Basic Compute Node role
 CountDefault: 1
...
```

```
ServicesDefault:
...
- OS::TripleO::Services::NovaLibvirt
```



注記

RHEL 9.2 にアップグレードされるノードを含むロールには、**NovaLibvirt** サービスが含まれている必要があります。**OS::TripleO::Services::NovaLibvirtLegacy** を **OS::TripleO::Services::NovaLibvirt** に置き換えます。

- <ComputeRHEL9> を RHEL 9.2 ロールの名前に置き換えます。

2. **overcloud_upgrade_prepare.sh** ファイルを **copy_role_Compute_param.sh** ファイルにコピーします。

```
$ cp overcloud_upgrade_prepare.sh copy_role_Compute_param.sh
```

3. **copy_role_Compute_param.sh** ファイルを編集して、**copy_role_params.py** スクリプトを含めます。このスクリプトは、新しいロールの追加パラメーターとリソースを含む環境ファイルを生成します。以下に例を示します。

```
/usr/share/openstack-tripleo-heat-templates/tools/copy_role_params.py --rolename-src
<Compute_source_role> --rolename-dst <Compute_destination_role> \
 -o <Compute_new_role_params.yaml> \

-e /home/stack/templates/internal.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
-e /home/stack/templates/network/network-environment.yaml \
-e /home/stack/templates/inject-trust-anchor.yaml \
-e /home/stack/templates/hostnames.yml \
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml
\
-e /home/stack/templates/nodes_data.yaml \
-e /home/stack/templates/debug.yaml \
-e /home/stack/templates/firstboot.yaml \
   -e /home/stack/overcloud-params.yaml \
 -e /home/stack/overcloud-deploy/overcloud/overcloud-network-environment.yaml \
-e /home/stack/overcloud_adopt/baremetal-deployment.yaml \
-e /home/stack/overcloud_adopt/generated-networks-deployed.yaml \
-e /home/stack/overcloud_adopt/generated-vip-deployed.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/nova-hw-machine-type-
upgrade.yaml \
-e ~/containers-prepare-parameter.yaml
```

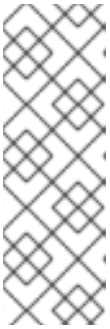
- **<Compute_source_role>** をコピーするソースのコンピュートロールの名前に置き換えます。
- **<Compute_destination_role>** を新しいロールの名前に置き換えます。
- -o オプションを使用して、新しいロールのソースコンピュートロールのデフォルト以外の値をすべて含む出力ファイルの名前を定義します。**<Compute_new_role_params.yaml>** を出力ファイルの名前に置き換えます。

4. **copy_role_Compute_param.sh** スクリプトを実行します。

```
$ sh /home/stack/copy_role_Compute_param.sh
```

5. コンピュートノードをソースロールから新しいロールに移動します。

```
python3
/usr/share/openstack-tripleo-heat-templates/tools/baremetal_transition.py  --baremetal-
deployment /home/stack/tripleo-<stack>-baremetal-deployment.yaml  --src-role
<Compute_source_role>  --dst-role <Compute_destination_role> <Compute-0> <Compute-
1> <Compute-2>
```



注記

このツールには、アンダークラウドのアップグレード中にエクスポートした元の **/home/stack/tripleo-<stack>-baremetal-deployment.yaml** ファイルが含まれています。このツールは、**/home/stack/tripleo-<stack>-baremetal-deployment.yaml** ファイル内のソースロール定義をコピーし、名前を変更します。次に、新しく作成された宛先ロールとの競合を防ぐため に、**hostname_format** を変更します。続いて、ツールはノードをソースロールから宛先ロールに移動し、**count** 値を変更します。

- **<stack>** は、スタックの名前に置き換えます。
- **<Compute_source_role>** を新しいロールに移動するノードを含むソースコンピュートロールの名前に置き換えます。
- **<Compute_destination_role>** を新しいロールの名前に置き換えます。
- **<Compute-0> <Compute-1> <Compute-2>** を、新しいロールに移動するノードの名前に置き換えます。

6. ノードを再プロビジョニングして、スタック内の環境ファイルを新しいロールの場所で更新します。

```
$ openstack overcloud node provision --stack <stack> --output
/home/stack/overcloud_adopt/baremetal-deployment.yaml /home/stack/tripleo-<stack>-
baremetal-deployment.yaml
```

注記

出力された **baremetal-deployment.yaml** ファイルは、オーバークラウドの導入中に **overcloud_upgrade_prepare.sh** ファイルで使用されたファイルと同じものです。

7. RHEL 8.4 に残っているコンピュートロールを **COMPUTE_ROLES** パラメーターに含めて、次のスクリプトを実行します。たとえば、RHEL 8.4 に残っているノードを含む **ComputeRHEL8** というロールがある場合は、**COMPUTE_ROLES = --role ComputeRHEL8** となります。

```
python3
/usr/share/openstack-tripleo-heat-templates/tools/multi-rhel-container-image-prepare.py \
    ${COMPUTE_ROLES} \
    --enable-multi-rhel \
    --excludes collectd \
    --excludes nova-libvirt \
```

```
    --minor-override "
{${EL8_TAGS}${EL8_NAMESPACE}${CEPH_OVERRIDE}${NEUTRON_DRIVER}\"no_tag\":
\"not_used\"}" \
    --major-override "
{${EL9_TAGS}${NAMESPACE}${CEPH_OVERRIDE}${NEUTRON_DRIVER}\"no_tag\":\"not
_used\"}" \
    --output-env-file \
    /home/stack/containers-prepare-parameter.yaml
```

8. この手順を繰り返して追加のロールを作成し、追加のコンピュートノードをそれらの新しいロールに移動します。

11.3.2. コンピュートノードのオペレーティングシステムのアップグレード

選択したコンピュートノードのオペレーティングシステムを RHEL 9.2 にアップグレードします。異なるロールから複数のノードを同時にアップグレードできます。

前提条件

環境に必要なロールが作成されていることを確認してください。Multi-RHEL 環境のロールの作成に関する詳細は、Multi-RHEL コンピュートノードのロールの作成 を参照してください。

手順

1. **skip_rhel_release.yaml** ファイルで、**SkipRhelEnforcement** パラメーターを **false** に設定します。

```
parameter_defaults:
  SkipRhelEnforcement: false
```

2. 次の例に示すように、**-e system_upgrade.yaml** 引数と必要なその他の **-e** 環境ファイル引数を **overcloud_upgrade_prepare.sh** スクリプトに含めます。

```
$ openstack overcloud upgrade prepare --yes \
...
-e /home/stack/system_upgrade.yaml  \
-e /home/stack/<Compute_new_role_params.yaml> \
...
```

   - アップグレード固有のパラメーターを持つ **system_upgrade.yaml** ファイルを含めます (-e)。
   - 新しいロールに必要なパラメーターを含む環境ファイルを含めます (-e)。**<Compute_new_role_params.yaml>** を新しいロール用に作成した環境ファイルの名前に置き換えます。
   - 複数のロールからノードを同時にアップグレードする場合は、作成した新しいロールごとに環境ファイルを含めます。

3. オプション: インスタンスを移行します。移行計画の詳細は、コンピュートノード間の仮想マシンの移行 および 移行の準備 を参照してください。
4. **overcloud_upgrade_prepare.sh** スクリプトを実行します。
5. 特定のコンピュートノード上のオペレーティングシステムをアップグレードします。アップグ

レードするノードのコンマ区切りリストを指定して、**--limit** オプションを使用します。以下の例では、**computerhel9-0**、**computerhel9-1**、**computerhel9-2**、および **computesriov-42** ノードを **ComputeRHEL9** および **ComputeSRIOV** ロールからアップグレードします。

```
$ openstack overcloud upgrade run --yes --tags system_upgrade --stack <stack> --limit
computerhel9-0,computerhel9-1,computerhel9-2,computesriov-42
```

- <stack> は、スタックの名前に置き換えます。

6. コンピュートノード上のコンテナーを RHEL 9.2 にアップグレードします。アップグレードするノードのコンマ区切りリストを指定して、**--limit** オプションを使用します。以下の例では、**computerhel9-0**、**computerhel9-1**、**computerhel9-2**、および **computesriov-42** ノードを **ComputeRHEL9** および **ComputeSRIOV** ロールからアップグレードします。

```
$ openstack overcloud upgrade run --yes --stack <stack>  --limit computerhel9-
0,computerhel9-1,computerhel9-2,computesriov-42
```

第12章 アップグレード後操作の実施

オーバークラウドのアップグレードが完了したら、アップグレード後の設定を実施して、環境が完全にサポートされ、これ以降の操作を行う準備が整っている状態にする必要があります。

重要

Red Hat OpenStack Platform 16.2 から 17.1 にアップグレードした後に追加のオーバークラウドコマンドを実行する場合は、以下の点を考慮する必要があります。

- アップグレード後に実行するオーバークラウドコマンドには、アップグレードプロセス中に作成または更新した **YAML** ファイルを含める必要があります。たとえば、スケールアップ操作中にオーバークラウドノードをプロビジョニングするには、**/home/stack/templates/overcloud-baremetal-deployed.yaml** ファイルではなく、**/home/stack/tripleo-[stack]-baremetal-deploy.yaml** ファイルを使用します。
- **system_upgrade.yaml** ファイルと **upgrades-environment.yaml** ファイルを除き、**openstack overcloud upgrade prepare** コマンドの最後の実行に渡したすべてのオプションを含めます。

12.1. オーバークラウドイメージのアップグレード

現在のオーバークラウドイメージを新しいバージョンに置き換える必要があります。新しいイメージにより、director は最新バージョンの Red Hat OpenStack Platform ソフトウェアを使用して、ノードのイントロスペクションとプロビジョニングを行うことができるようになります。

注記

オーバークラウドを再デプロイする場合は、新しいバージョンのオーバークラウドイメージを使用する必要があります。オーバークラウドイメージのインストールに関する詳細は、**director を使用した Red Hat OpenStack Platform のインストールと管理**の オーバークラウドイメージのインストール を参照してください。

前提条件

- アンダークラウドが最新バージョンにアップグレードされていること

手順

1. **stack** ユーザーのホーム下の **images** ディレクトリー (**/home/stack/images**) から既存のイメージを削除します。

```
$ rm -rf ~/images/*
```

2. アーカイブをデプロイメントします。

```
$ cd ~/images
$ for i in /usr/share/rhosp-director-images/overcloud-full-latest-17.1.tar /usr/share/rhosp-director-images/ironic-python-agent-latest-17.1.tar; do tar -xvf $i; done
$ cd ~
```

3. イメージを director にインポートします。

```
(undercloud) [stack@director images]$ openstack overcloud image upload --image-path
/home/stack/images/ --update-existing
```

このコマンドは次のタスクを完了します。

- イメージフォーマットを QCOW から RAW に変換します。
- イメージのアップロードに関するステータスの最新情報を提供します。

12.2. CPU ピニングパラメーターの更新

Red Hat OpenStack Platform 17.1 へのアップグレードが完了した後、CPU ピニング設定を **NovaVcpuPinSet** パラメーターから以下のパラメーターに移行する必要があります。

NovaComputeCpuDedicatedSet

専用の (ピニングされた) CPU を設定します。

NovaComputeCpuSharedSet

共有の (ピニングされていない) CPU を設定します。

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. Compute ノードが同時マルチスレッド (SMT) をサポートするが **hw:cpu_thread_policy=isolated** ポリシーでインスタンスを作成している場合は、以下のオプションのいずれかを実施する必要があります。
   - **hw:cpu_thread_policy** スレッドポリシーを設定しない新しいフレーバーを作成し、そのフレーバーでインスタンスのサイズを変更します。
     i. source コマンドでオーバークラウドの認証ファイルを読み込みます。

     ```
     $ source ~/overcloudrc
     ```

     ii. デフォルトのスレッドポリシー **prefer** を使用してフレーバーを作成します。

     ```
     (overcloud) $ openstack flavor create <flavor>
     ```

     注記

     インスタンスのサイズを変更する場合は、新しいフレーバーを使用する必要があります。現在のフレーバーを再利用することはできません。詳細は、**インスタンスの作成と管理** ガイドの インスタンスのサイズ変更 を参照してください。

     iii. 新しいフレーバーを使用するようにインスタンスを変換します。

     ```
     (overcloud) $ openstack server resize --flavor <flavor> <server>
     (overcloud) $ openstack server resize confirm <server>
     ```

     iv. **hw:cpu_thread_policy=isolated** ポリシーを使用するすべての固定インスタンスに対して、このステップを繰り返します。

- Compute ノードからインスタンスを移行して、Compute ノードの SMT を無効にする。

  i. source コマンドでオーバークラウドの認証ファイルを読み込みます。

  ```
  $ source ~/overcloudrc
  ```

  ii. Compute ノードが新しい仮想マシンを受け入れるのを無効にします。

  ```
  (overcloud) $ openstack compute service list
  (overcloud) $ openstack compute service set <hostname> nova-compute --disable
  ```

  iii. Compute ノードからすべてのインスタンスを移行します。インスタンスの移行の詳細は、コンピュートノ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。

  iv. Compute ノードをリブートし、Compute ノードの BIOS で SMT を無効にします。

  v. Compute ノードをブートします。

  vi. Compute ノードを再度有効にします。

  ```
  (overcloud) $ openstack compute service set <hostname> nova-compute --enable
  ```

3. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

4. **NovaVcpuPinSet** パラメーターが含まれる環境ファイルを編集します。

5. CPU ピニングの設定を **NovaVcpuPinSet** パラメーターから **NovaComputeCpuDedicatedSet** と **NovaComputeCpuSharedSet** に移行します。

   - これまでピニングされたインスタンス用に使用されていたホストの場合には、**NovaVcpuPinSet** の値を **NovaComputeCpuDedicatedSet** に変更します。
   - これまでピニングされていないインスタンス用に使用されていたホストの場合には、**NovaVcpuPinSet** の値を **NovaComputeCpuSharedSet** に変更します。
   - NovaVcpuPinSet の値が設定されていない場合には、ノード上でホストするインスタンスの種別に応じて、すべての Compute ノードのコアを **NovaComputeCpuDedicatedSet** または **NovaComputeCpuSharedSet** のどちらかに割り当てる必要があります。

   たとえば、以前の環境ファイルに以下のピニング設定が定義されていたとします。

```
parameter_defaults:
  ...
  NovaVcpuPinSet: 1,2,3,5,6,7
  ...
```

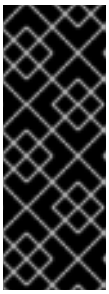
   設定をピニング設定に移行するには、NovaCompute **CpuDedicatedSet** パラメーターを設定し、**NovaVcpuPinSet** パラメーターの設定を解除します。

```
parameter_defaults:
  ...
  NovaComputeCpuDedicatedSet: 1,2,3,5,6,7
```

```
    NovaVcpuPinSet: ""
    ...
```

設定をピニングしない設定に移行するには、**NovaComputeCpuSharedSet** パラメーターを設定し、**NovaVcpuPinSet** パラメーターの設定を解除します。

```
parameter_defaults:
  ...
  NovaComputeCpuSharedSet: 1,2,3,5,6,7
  NovaVcpuPinSet: ""
  ...
```



重要

NovaComputeCpuDedicatedSet または **NovaComputeCpuSharedSet** のいずれかが、**NovaVcpuPinSet** で定義した設定と一致するようにします。**NovaComputeCpuDedicatedSet** または **NovaComputeCpuSharedSet** のいずれかの設定を変更する、またはその両方を設定するには、設定を更新する前にピニング設定の Compute ノードが 1 つのインスタンスも実行していないようにします。

6. ファイルを保存します。
7. デプロイメントコマンドを実行して、新しい CPU ピニングパラメーターでオーバークラウドを更新します。

```
(undercloud) $ openstack overcloud deploy \
  --stack _STACK NAME_ \
  --templates \
  ...
  -e /home/stack/templates/<compute_environment_file>.yaml
  ...
```

関連情報

- Compute ノードでの CPU ピニングの設定

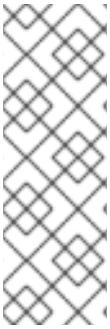
12.3. RHOSP 17 へのアップグレード後のホストのデフォルトマシンタイプの更新

インスタンスのマシンタイプは、PCIe グラフィックスカードやイーサネットコントローラーなどの特定のデフォルトデバイスを提供する仮想チップセットです。クラウドユーザーは、必要な **hw_machine_type** メタデータプロパティーを持つイメージを使用して、インスタンスのマシンタイプを指定できます。

クラウド管理者は、コンピュートパラメーター **NovaHWMachineType** を使用して、各コンピュートノードアーキテクチャーをデフォルトのマシンタイプで設定し、そのアーキテクチャーでホストされているインスタンスに適用できます。インスタンスの起動時に **hw_machine_type** イメージプロパティーが指定されていない場合は、ホストアーキテクチャーのデフォルトのマシンタイプがインスタンスに適用されます。Red Hat OpenStack Platform (RHOSP) 17 は RHEL 9 をベースにしています。**pc-i440fx** QEMU マシンタイプは RHEL 9 で非推奨となったため、RHEL 9 で実行される **x86_64** インスタンスのデフォルトのマシンタイプは **pc** から **q35** に変更されました。RHEL 9 でのこの変更に基づいて、マシンタイプ **x86_64** のデフォルト値も、RHOSP 16 の **pc** から RHOSP 17 の **q35** に変更されました。

RHOSP 16.2 以降では、Compute サービスは、インスタンスの起動時にインスタンスのシステムメタデータ内にインスタンスのマシンタイプを記録します。これは、既存のインスタンスのマシンタイプに影響を及ぼすことなく、RHOSP デプロイメントの有効期間中に **NovaHWMachineType** を変更できるようになったことを意味します。

Compute サービスは、**SHELVED_OFFLOADED** 状態にないインスタンスのマシンタイプを記録します。したがって、RHOSP 17 にアップグレードした後は、**SHELVED_OFFLOADED** 状態にあるインスタンスのマシンタイプを手動で記録し、環境または特定のセル内におけるすべてのインスタンスのマシンタイプが記録されていることを確認する必要があります。マシンタイプを使用して各インスタンスのシステムメタデータを更新した後、既存のインスタンスのマシンタイプに影響を及ぼすことなく、**NovaHWMachineType** パラメーターを RHOSP 17 のデフォルト (**q35**) に更新できます。



注記

RHOSP OSP17.0 以降では、Q35 がデフォルトのマシンタイプです。Q35 マシンタイプは PCIe ポートを使用します。heat パラメーター **NovaLibvirtNumPciePorts** を設定すると、PCIe ポートデバイスの数を管理できます。PCIe ポートに接続できるデバイスの数は、以前のバージョンで実行いているインスタンスよりも少なくなります。より多くのデバイスを使用する場合は、イメージ属性 **hw_disk_bus=scsi** または **hw_scsi_model=virtio-scsi** を使用する必要があります。詳細は、仮想ハードウェアのメタデータプロパティー を参照してください。

前提条件

- すべてのコンピュートノードを RHEL 9.2 にアップグレードします。コンピュートノードのアップグレードに関する詳細は、すべてのコンピュートノードを RHEL 9.2 にアップグレードする 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. source コマンドで **stackrc** ファイルを読み込みます。

```
$ source ~/stackrc
```

3. コントローラーノードに **heat-admin** ユーザーとしてログインします。

```
(undercloud)$ metalsmith list
$ ssh heat-admin@<controller_ip>
```

<controller_ip> をコントローラーノードの IP アドレスに置き換えます。

4. マシンタイプが設定されていないインスタンスのリストを取得します。

```
[heat-admin@<controller_ip> ~]$ sudo podman exec -i -u root nova_api \
  nova-manage libvirt list_unset_machine_type
```

5. インスタンスホストのデフォルトのマシンタイプについては、**nova-hw-machine-type-upgrade.yaml** ファイルの **NovaHWMachineType** パラメーターを確認します。RHOSP 16.2 の **NovaHWMachineType** パラメーターのデフォルト値は次のとおりです。
x86_64=pc-i440fx-rhel7.6.0,aarch64=virt-rhel7.6.0,ppc64=pseries-rhel7.6.0,ppc64le=pseries-rhel7.6.0

6. 各インスタンスのシステムメタデータをデフォルトのインスタンスマシンタイプで更新します。

```
[heat-admin@<controller_ip> ~]$ sudo podman exec -i -u root nova_api \
  nova-manage libvirt update_machine_type <instance_uuid> <machine_type>
```

- **<instance_uuid>** をインスタンスの UUID に置き換えます。
- **<machine_type>** をインスタンスを記録するマシンタイプに置き換えます。

7. すべてのインスタンスのマシンタイプが記録されていることを確認します。

```
[heat-admin@<controller_ip> ~]$ sudo podman exec -i -u root nova_api \
  nova-status upgrade check
```

このコマンドは、マシンタイプのないインスタンスが見つかった場合に警告を返します。この警告が表示された場合は、手順 4 からこの手順を繰り返します。

8. コンピュート環境ファイルの **NovaHWMachineType** のデフォルト値を **x86_64=q35** に変更し、オーバークラウドをデプロイします。

検証

1. デフォルトのマシンタイプを持つインスタンスを作成します。

```
(overcloud)$ openstack server create --flavor <flavor> \
  --image <image> --network <network> \
  --wait defaultMachineTypeInstance
```

- **<flavor>** をインスタンスのフレーバーの名前または ID に置き換えます。
- **<image>** を **hw_machine_type** を設定しないイメージの名前または ID に置き換えます。
- **<network>** をインスタンスの接続先となるネットワークの名前または ID に置き換えます。

2. インスタンスのマシンタイプがデフォルト値に設定されていることを確認します。

```
[heat-admin@<controller_ip> ~]$ sudo podman exec -i -u root nova_api \
  nova-manage libvirt get_machine_type <instance_uuid>
```

<instance_uuid> をインスタンスの UUID に置き換えます。

3. マシンタイプ **x86_64=pc-i440fx** のインスタンスをハードリブートします。

```
(overcloud)$ openstack server reboot --hard <instance_uuid>
```

<instance_uuid> をインスタンスの UUID に置き換えます。

4. インスタンスのマシンタイプが変更されていないことを確認します。

```
[heat-admin@<controller_ip> ~]$ sudo podman exec -i -u root nova_api \
  nova-manage libvirt get_machine_type <instance_uuid>
```

<instance_uuid> をインスタンスの UUID に置き換えます。

12.4. オーバークラウドでのフェンシングの再有効化

オーバークラウドをアップグレードする前に、オーバークラウドでのフェンシングの無効化 で、フェンシングを無効化しました。環境をアップグレードした後、ノードに障害が発生した場合にデータを保護するためにフェンシングを再度有効にします。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. コントローラーノードにログインし、Pacemaker コマンドを実行してフェンシングを再度有効にします。

```
$ ssh tripleo-admin@<controller_ip> "sudo pcs property set stonith-enabled=true"
```

   - **<controller_ip>** は、コントローラーノードの IP アドレスに置き換えます。コントローラーノードの IP アドレスは、**openstack server list** コマンドで確認できます。

4. **fencing.yaml** 環境ファイルで、**EnableFencing** パラメーターを **true** に設定します。

関連情報

- STONITH を使用したコントローラーノードのフェンシング

第13章 RED HAT CEPH STORAGE 5 から 6 へのアップグレード

他のすべてのアップグレードタスクが完了したら、Red Hat Ceph Storage クラスターをリリース 5 から 6 にアップグレードできます。

前提条件

- Red Hat OpenStack Platform 16.2 から 17.1 へのアップグレードが完了している。
- すべてのコントローラーノードが Red Hat Enterprise Linux 9 にアップグレードされている。HCI 環境では、すべてのコンピュートノードも RHEL 9 にアップグレードする必要があります。
- 現在の Red Hat Ceph Storage 5 クラスターが健全な状態である。

13.1. DIRECTOR でデプロイされた RED HAT CEPH STORAGE 環境

Red Hat Ceph Storage が director によって環境にデプロイされた場合は、次のタスクを実行します。

13.1.1. cephadm クライアントの更新

Red Hat Ceph Storage クラスターをアップグレードする前に、オーバークラウドノードの **cephadm** パッケージをリリース 6 に更新する必要があります。

前提条件

Red Hat Ceph Storage クラスターの健全性ステータスが **HEALTH_OK** であることを確認します。コントローラーノードにログインし、コマンド **sudo cephadm shell — ceph -s** を使用してクラスターの健全性を確認します。ステータスが **HEALTH_OK** でない場合は、この手順を続行する前に問題を修正してください。

手順

1. コントローラーノードで Red Hat Ceph Storage (ツールのみ) リポジトリーを有効にするための Playbook を作成します。次の情報を含める必要があります。

```
- hosts: all
  gather_facts: false
  tasks:
    - name: Enable RHCS 6 tools repo
      ansible.builtin.command: |
          subscription-manager repos --disable=rhceph-5-tools-for-rhel-9-x86_64-rpms
          subscription-manager repos --enable=rhceph-6-tools-for-rhel-9-x86_64-rpms
      become: true
  - name: Update cephadm
    ansible.builtin.package:
      name: cephadm
      state: latest
    become: true
```

2. Playbook を実行します。
ansible-playbook -i ~/overcloud-deploy/<stack>/tripleo-ansible-inventory.yaml <playbook_file_name> --limit <controller_role>

- **<stack>** は、スタックの名前に置き換えます。
- **<playbook_file_name>** は、前の手順で作成した Playbook の名前に置き換えます。
- **<controller_role>** は、コントローラーノードに適用するロールに置き換えます。
- **--limit** オプションを使用して、コンテンツをコントローラーノードにのみ適用します。

3. コントローラーノードにログインします。
4. **cephadm** パッケージがリリース 6 に更新されていることを確認します。
   $ sudo dnf info cephadm | grep -i version

13.1.2. Red Hat Ceph Storage コンテナーイメージの更新

container-image-prepare.yaml ファイルは、**ContainerImagePrepare** パラメーターを含むファイルであり、Red Hat Ceph Storage コンテナーを定義します。このファイルは、アンダークラウドとオーバークラウドのコンテナーイメージを取得するルールを定義するために、**tripleo-container-image prepare** コマンドで使用します。環境を更新する前に、このファイルを正しいイメージバージョンで更新してください。

手順

1. コンテナー準備ファイルを見つけます。このファイルのデフォルト名は、**containers-prepare-parameter.yaml** です。
2. コンテナー準備ファイルを編集します。
3. **ceph_tag** パラメーターを見つけます。現在のエントリーは次の例のようになっているはずです。

```
ceph_namespace: registry.redhat.io
ceph_image: rhceph
ceph_tag: '5'
```

4. Red Hat Ceph Storage 6 の **ceph_tag** パラメーターを更新します。

```
ceph_namespace: registry.redhat.io
ceph_image: rhceph-6-rhel9
ceph_tag: '6'
```

5. ファイルを保存します。

13.1.3. container image prepare の実行

director コンテナー準備コマンドを実行して、コンテナーイメージの準備プロセスを完了します。これにより、オーバークラウド用のすべてのコンテナーイメージ設定が準備され、最新の Red Hat Ceph Storage 6 コンテナーイメージが取得されます。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. コンテナー準備コマンドを実行します。
$ openstack tripleo container image prepare -e <container_preparation_file>

   - **<container_preparation_file>** は、ファイルの名前に置き換えます。デフォルトのファイルは、**containers-prepare-parameter.yaml** です。

4. 新しい Red Hat Ceph Storage イメージがアンダークラウドレジストリーに存在することを確認します。
$ openstack tripleo container image list -f value | awk -F '//' '/ceph/ {print $2}'

13.1.4. Orchestrator を使用した Red Hat Ceph Storage 6 へのアップグレード

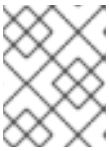
cephadm コマンドの Orchestrator 機能を使用して、Red Hat Ceph Storage 6 にアップグレードします。

前提条件

- **ceph-mon** サービスを実行している Monitor またはコントローラーノードで、**sudo cephadm --shell ceph status** コマンドを使用して、Red Hat Ceph Storage クラスターのステータスを確認する。このコマンドは、次の 3 つの応答のいずれかを返します。
  - **HEALTH_OK** - クラスターは健全な状態です。クラスターのアップグレードを続行してください。
  - **HEALTH_WARN** - クラスターは異常です。障害となっている問題が解決されるまでは、クラスターのアップグレードを続行しないでください。トラブルシューティングのガイダンスについては、Red Hat Ceph Storage 5 トラブルシューティングガイド を参照してください。
  - **HEALTH_ERR** - クラスターは異常です。障害となっている問題が解決されるまでは、クラスターのアップグレードを続行しないでください。トラブルシューティングのガイダンスについては、Red Hat Ceph Storage 5 トラブルシューティングガイド を参照してください。

手順

1. コントローラーノードにログインします。
2. **Red Hat Ceph Storage 6 アップグレードガイド**の cephadm を使用した Red Hat Ceph Storage クラスターのアップグレード を使用して、クラスターを最新の Red Hat Ceph Storage バージョンにアップグレードします。
3. Red Hat Ceph Storage コンテナーのアップグレードが完了するまで待ちます。



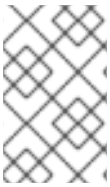
注記

コマンド **sudo cephadm shell — ceph orch upgrade status** を使用して、ステータスのアップグレードを監視します。

13.1.5. Red Hat Ceph Storage 5 から 6 に移行する場合の NFS Ganesha のアップグレード

Red Hat Ceph Storage をリリース 4 から 5 にアップグレードする場合、NFS Ganesha は Orchestrator によって導入されません。そのため、NFS Ganesha は director の管理下に残り、手動でリリース 6 に移動する必要があります。

Red Hat Ceph Storage 5 ベースの NFS Ganesha と Red Hat Ceph Storage 6 クラスターの併用は、アップグレード期間中にのみサポートされます。Red Hat Ceph Storage クラスターを 6 にアップグレードしたら、リリース 6 ベースのコンテナーイメージを使用するように NFS Ganesha をアップグレードする必要があります。



注記

この手順は、CephFS NFS を使用する Shared File Systems サービス (manila) を使用している環境にのみ適用されます。この環境では、NFS Ganesha に合わせた Red Hat Ceph Storage コンテナーのアップグレードが必須です。

手順

1. コントローラーノードにログインします。
2. **ceph-nfs** サービスを調べます。
   $ sudo pcs status | grep ceph-nfs
3. **ceph-nfs systemd** ユニットを調べて、Red Hat Ceph Storage 5 のコンテナーイメージとタグが含まれていることを確認します。

```
$ cat /etc/systemd/system/ceph-nfs@.service | grep -i container_image
```

4. 次の内容を含む **/home/stack/ganesha_update_extravars.yaml** というファイルを作成します。

```
tripleo_cephadm_container_image: <ceph_image_name>
tripleo_cephadm_container_ns: <ceph_image_namespace>
tripleo_cephadm_container_tag: <ceph_image_tag>
```

   - **<ceph_image_name>** は、Red Hat Ceph Storage コンテナーイメージの名前に置き換えます。
   - **<ceph_image_namespace>** は、Red Hat Ceph Storage コンテナーの名前空間の名前に置き換えます。
   - **<ceph_image_tag>** は、Red Hat Ceph Storage コンテナータグの名前に置き換えます。たとえば、一般的な環境では、このファイルの内容として次の値が含まれます。

```
tripleo_cephadm_container_image: rhceph
tripleo_cephadm_container_ns: undercloud-0.ctlplane.redhat.local:8787
tripleo_cephadm_container_tag: '6'
```

5. ファイルを保存します。
6. **ceph-update-genesha.yml** Playbook を実行します。追加のコマンドパラメーターとして **ganesha_update_extravars.yaml** Playbook を指定します。

```
ansible-playbook -i /usr/share/ansible/tripleo-playbooks/ceph-update-ganesha.yml \
 -e @$HOME/overcloud-deploy/<stack>/config-download/<stack>/global_vars.yaml \
 -e @$HOME/overcloud-deploy/<stack>/config-download/<stack>/cephadm/cephadm-extra-
```

```
vars-heat.yml \
  -e @$HOME/ganesha_update_extravars.yaml
```

- **<stack>** は、オーバークラウドスタックの名前に置き換えます。

7. **ceph-nfs** サービスが実行されていることを確認します。
   $ sudo pcs status | grep ceph-nfs

8. **ceph-nfs systemd** ユニットに Red Hat Ceph Storage 6 のコンテナーイメージとタグが含まれていることを確認します。
   $ cat /etc/systemd/system/ceph-nfs@.service | grep rhceph

13.2. 外部 RED HAT CEPH STORAGE クラスター環境

Red Hat Ceph Storage クラスターが環境内の Red Hat OpenStack Platform デプロイメントの外部にある場合は、次のタスクを実行します。

13.2.1. Red Hat Ceph Storage コンテナーイメージの更新

container-image-prepare.yaml ファイルは、**ContainerImagePrepare** パラメーターを含むファイルであり、Red Hat Ceph Storage コンテナーを定義します。このファイルは、アンダークラウドとオーバークラウドのコンテナーイメージを取得するルールを定義するために、**tripleo-container-image prepare** コマンドで使用します。環境を更新する前に、このファイルを正しいイメージバージョンで更新してください。

手順

1. コンテナー準備ファイルを見つけます。このファイルのデフォルト名は、**containers-prepare-parameter.yaml** です。

2. コンテナー準備ファイルを編集します。

3. **ceph_tag** パラメーターを見つけます。現在のエントリーは次の例のようになっているはずです。

```
ceph_namespace: registry.redhat.io
ceph_image: rhceph
ceph_tag: '5'
```

4. Red Hat Ceph Storage 6 の **ceph_tag** パラメーターを更新します。

```
ceph_namespace: registry.redhat.io
ceph_image: rhceph-6-rhel9
ceph_tag: '6'
```

5. ファイルを保存します。

13.2.2. container image prepare の実行

director コンテナー準備コマンドを実行して、コンテナーイメージの準備プロセスを完了します。これにより、オーバークラウド用のすべてのコンテナーイメージ設定が準備され、最新の Red Hat Ceph Storage 6 コンテナーイメージが取得されます。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. コンテナー準備コマンドを実行します。
   $ openstack tripleo container image prepare -e <container_preparation_file>
   - **<container_preparation_file>** は、ファイルの名前に置き換えます。デフォルトのファイルは、**containers-prepare-parameter.yaml** です。
4. 新しい Red Hat Ceph Storage イメージがアンダークラウドレジストリーに存在することを確認します。
   $ openstack tripleo container image list -f value | awk -F '//' '/ceph/ {print $2}'

13.2.3. Red Hat Ceph Storage 5 から 6 に移行する場合の NFS Ganesha のアップグレード

Red Hat Ceph Storage をリリース 4 から 5 にアップグレードする場合、NFS Ganesha は Orchestrator によって導入されません。そのため、NFS Ganesha は director の管理下に残り、手動でリリース 6 に移動する必要があります。

Red Hat Ceph Storage 5 ベースの NFS Ganesha と Red Hat Ceph Storage 6 クラスターの併用は、アップグレード期間中にのみサポートされます。Red Hat Ceph Storage クラスターを 6 にアップグレードしたら、リリース 6 ベースのコンテナーイメージを使用するように NFS Ganesha をアップグレードする必要があります。

注記

この手順は、CephFS NFS を使用する Shared File Systems サービス (manila) を使用している環境にのみ適用されます。この環境では、NFS Ganesha に合わせた Red Hat Ceph Storage コンテナーのアップグレードが必須です。

手順

1. コントローラーノードにログインします。
2. **ceph-nfs** サービスを調べます。
   $ sudo pcs status | grep ceph-nfs
3. **ceph-nfs systemd** ユニットを調べて、Red Hat Ceph Storage 5 のコンテナーイメージとタグが含まれていることを確認します。

```
$ cat /etc/systemd/system/ceph-nfs@.service | grep -i container_image
```

4. 次の内容を含む **/home/stack/ganesha_update_extravars.yaml** というファイルを作成します。

```
tripleo_cephadm_container_image: <ceph_image_name>
tripleo_cephadm_container_ns: <ceph_image_namespace>
tripleo_cephadm_container_tag: <ceph_image_tag>
```

- **<ceph_image_name>** は、Red Hat Ceph Storage コンテナーイメージの名前に置き換えます。
- **<ceph_image_namespace>** は、Red Hat Ceph Storage コンテナーの名前空間の名前に置き換えます。
- **<ceph_image_tag>** は、Red Hat Ceph Storage コンテナータグの名前に置き換えます。たとえば、一般的な環境では、このファイルの内容として次の値が含まれます。

```
tripleo_cephadm_container_image: rhceph
tripleo_cephadm_container_ns: undercloud-0.ctlplane.redhat.local:8787
tripleo_cephadm_container_tag: '6'
```

5. ファイルを保存します。
6. **ceph-update-genesha.yml** Playbook を実行します。追加のコマンドパラメーターとして **ganesha_update_extravars.yaml** Playbook を指定します。

```
ansible-playbook -i /usr/share/ansible/tripleo-playbooks/ceph-update-ganesha.yml \
 -e @$HOME/overcloud-deploy/<stack>/config-download/<stack>/global_vars.yaml \
 -e @$HOME/overcloud-deploy/<stack>/config-download/<stack>/cephadm/cephadm-extra-vars-heat.yml \
  -e @$HOME/ganesha_update_extravars.yaml
```

- **<stack>** は、オーバークラウドスタックの名前に置き換えます。

7. **ceph-nfs** サービスが実行されていることを確認します。
   $ sudo pcs status | grep ceph-nfs
8. **ceph-nfs systemd** ユニットに Red Hat Ceph Storage 6 のコンテナーイメージとタグが含まれていることを確認します。
   $ cat /etc/systemd/system/ceph-nfs@.service | grep rhceph