



Red Hat OpenStack Platform 16.1

13 から 16.1 へのアップグレードフレームワーク

Red Hat OpenStack Platform 13 から 16.1 へのインプレースアップグレード

Red Hat OpenStack Platform 16.1 13 から 16.1 へのアップグレードフレームワーク

Red Hat OpenStack Platform 13 から 16.1 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

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

本ガイドでは、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。今回、本書では Red Hat OpenStack Platform 13 (Queens) から 16.1 (Train) 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	6
RED HAT ドキュメントへのフィードバック (英語のみ)	7
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	8
1.1. 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	8
1.2. ロングライフバージョンのライフサイクルサポート	8
1.3. ロングライフリリースのアップグレードパス	8
第2章 インプレースアップグレードの計画および準備	11
2.1. RED HAT OPENSTACK PLATFORM 16.1 の理解	11
2.2. RED HAT OPENSTACK PLATFORM 16.1 における変更点の概要	11
2.3. RED HAT ENTERPRISE LINUX 8 の変更点	12
2.4. RED HAT OPENSTACK PLATFORM での LEAPP アップグレードの使用	13
2.5. サポート対象のアップグレードシナリオ	14
2.6. 外部の CEPH デプロイメントと組み合わせたアップグレードに関する考慮事項	15
2.7. アップグレードを妨げる可能性のある既知の問題	16
2.8. バックアップと復元	19
2.9. マイナーバージョンの更新	19
2.10. プロキシ設定	19
2.11. アップグレード前の RED HAT OPENSTACK PLATFORM 13 の検証	19
第3章 リポジトリ	21
3.1. アンダークラウドのリポジトリ	21
3.2. オーバークラウドのリポジトリ	23
3.3. RED HAT SATELLITE SERVER 6 に関する考慮事項	28
第4章 アンダークラウドアップグレードの準備	31
4.1. 外部の CEPH と組み合わせたアップグレードの前提条件	31
4.2. 新たなメモリー要件	31
4.3. 予測可能なアンダークラウドノード NIC 名の使用	31
4.4. 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	35
4.5. 次世代電源管理ドライバーへの移行	35
第5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	37
5.1. RED HAT OPENSTACK PLATFORM DIRECTOR パッケージの削除	37
5.2. アンダークラウドでの LEAPP アップグレードの実施	37
第6章 DIRECTOR のアップグレード	41
6.1. 環境の RED HAT ENTERPRISE LINUX リリースへのロック	41
6.2. アンダークラウド用リポジトリの有効化	41
6.3. DIRECTOR パッケージのインストール	42
6.4. コンテナイメージの準備	42
6.5. コンテナイメージ準備のパラメーター	43
6.6. コンテナイメージタグ付けのガイドライン	46
6.7. プライベートレジストリーからのコンテナイメージの取得	47
6.8. アップグレード用の移行コンテナの取得	49
6.9. UNDERCLOUD.CONF ファイルの更新	51
6.10. DIRECTOR の設定パラメーター	51
6.11. DIRECTOR のアップグレードの実施	58
第7章 オーバークラウド準備の初期手順	60
7.1. オーバークラウドサービスのダウンタイムの準備	60

7.2. アップグレードテスト用のコンピュータノードの選択	60
7.3. オーバークラウドインベントリーファイルの作成	61
7.4. アップグレード前の要件の検証	61
7.5. オーバークラウドでのフェンシングの無効化	62
7.6. カスタムの PUPPET パラメーターの確認	62
7.7. アンダークラウドノードデータベースのバックアップ	63
第8章 LEAPP アップグレードのためのオーバークラウド設定	65
8.1. アップグレード環境ファイルの作成	65
8.2. アップグレードのパラメーター	66
8.3. オーバークラウドノードでの LEAPP データのコピー	67
8.4. オーバークラウドノードでの予測可能な NIC 名の使用	67
第9章 コンポーザブルサービスおよびパラメーターの更新	69
9.1. 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	69
9.2. カスタム環境ファイルのコンポーザブルサービスの更新	73
9.3. アンダークラウドレジストリーへのアクセスの設定	74
9.4. 非推奨化および廃止された NOVASCHEDULERDEFAULTFILTERS パラメーターのフィルター	75
9.5. COMPUTE 名の形式の設定	76
9.6. インスタンスのシリアル番号の設定	76
9.7. SSL/TLS 設定の更新	77
9.8. 設定後テンプレートの更新	78
9.9. 従来のテレメトリーサービスを維持する場合の考慮事項	78
第10章 RED HAT カスタマーポータルへのオーバークラウド登録の更新	79
10.1. RED HAT SUBSCRIPTION MANAGER (RHSM) コンポーザブルサービス	79
10.2. RHSMVARS サブパラメーター	79
10.3. RHSM コンポーザブルサービスへの切り替え	81
10.4. RHEL-REGISTRATION から RHSM へのマッピング	82
10.5. RHSM コンポーザブルサービスを使用したオーバークラウドの登録	82
10.6. 異なるロールに対する RHSM コンポーザブルサービスの適用	83
第11章 RED HAT SATELLITE SERVER へのオーバークラウド登録の更新	86
11.1. RED HAT SUBSCRIPTION MANAGER (RHSM) コンポーザブルサービス	86
11.2. RHSMVARS サブパラメーター	86
11.3. RHSM コンポーザブルサービスへの切り替え	88
11.4. RHEL-REGISTRATION から RHSM へのマッピング	89
11.5. RED HAT SATELLITE SERVER へのオーバークラウドの登録	89
11.6. SATELLITE SERVER を使用するための LEAPP の準備	90
第12章 DIRECTOR でデプロイされた CEPH STORAGE のアップグレードの準備	92
12.1. CEPH STORAGE ノードのアップグレードプロセスの概要	92
12.2. CEPH-ANSIBLE バージョンの確認	93
12.3. CEPH-ANSIBLE リポジトリーの設定	94
12.4. アップグレード前の CEPH クラスタステータスの確認	95
第13章 外部の CEPH デプロイメントと組み合わせたアップグレードの準備	96
13.1. CEPH-ANSIBLE のインストール	96
13.2. CEPH-ANSIBLE リポジトリーの設定	96
第14章 ネットワーク設定の更新	98
14.1. ネットワークインターフェイステンプレートの更新	98
14.2. アップグレード中の OPEN VSWITCH との互換性の維持	99
14.3. アップグレード中のコンポーザブルネットワーク互換性の維持	100

第15章 ネットワーク機能仮想化 (NFV) の準備	101
15.1. ネットワーク機能仮想化 (NFV) 用環境ファイル	101
第16章 アップグレード前の最終確認	103
16.1. デプロイメントに含めるカスタムファイル	103
16.2. デプロイメントに追加する新たな環境ファイル	103
16.3. デプロイメントから削除する環境ファイル	104
16.4. アップグレードのチェックリスト	104
第17章 アップグレードコマンドの概要	106
17.1. OPENSTACK OVERCLOUD UPGRADE PREPARE	106
17.2. OPENSTACK OVERCLOUD UPGRADE RUN	106
17.3. OPENSTACK OVERCLOUD EXTERNAL-UPGRADE RUN	106
17.4. OPENSTACK OVERCLOUD UPGRADE CONVERGE	107
17.5. オーバークラウドノードのアップグレードワークフロー	107
第18章 標準的なオーバークラウドのアップグレード	110
18.1. オーバークラウドアップグレード準備タスクの実行	110
18.2. DIRECTOR でデプロイされた CEPH STORAGE と組み合わせたコントローラーノードのアップグレード	111
18.3. CEPH STORAGE ノードのオペレーティングシステムのアップグレード	115
18.4. コンピュートノードのアップグレード	117
18.5. オーバークラウドスタックの同期	118
第19章 外部の CEPH デプロイメントと組み合わせたオーバークラウドのアップグレード	121
19.1. オーバークラウドアップグレード準備タスクの実行	121
19.2. 外部の CEPH デプロイメントと組み合わせたコントローラーノードのアップグレード	122
19.3. コンピュートノードのアップグレード	125
19.4. オーバークラウドスタックの同期	126
第20章 オーバークラウドアップグレードのスピードアップ	128
20.1. オーバークラウドアップグレード準備タスクの実行	128
20.2. コントロールプレーンノードのアップグレード	129
20.3. コンピュートノードの並列アップグレード	134
20.4. オーバークラウドスタックの同期	135
第21章 コントローラーが分割されたオーバークラウドのアップグレード	138
21.1. オーバークラウドアップグレード準備タスクの実行	138
21.2. PACEMAKER ベースのノードのアップグレード	139
21.3. PACEMAKER コントローラーノード以外のアップグレード	141
21.4. CEPH MON ノードのオペレーティングシステムのアップグレード	142
21.5. CEPH STORAGE ノードのオペレーティングシステムのアップグレード	145
21.6. コンピュートノードのアップグレード	147
21.7. オーバークラウドスタックの同期	148
第22章 ハイパーコンバージドインフラストラクチャーを持つオーバークラウドのアップグレード	151
22.1. オーバークラウドアップグレード準備タスクの実行	151
22.2. DIRECTOR でデプロイされた CEPH STORAGE と組み合わせたコントローラーノードのアップグレード	152
22.3. ハイパーコンバージドインフラストラクチャー (HCI) を持つコンピュートノードのアップグレード	156
22.4. オーバークラウドスタックの同期	157
第23章 DIRECTOR でデプロイされた CEPH STORAGE クラスターの RED HAT CEPH STORAGE 4 へのアップグレード	160
23.1. CEPH-ANSIBLE のインストール	160
23.2. CEPH STORAGE 4 へのアップグレード	161

第24章 FILESTORE から BLUESTORE への OSD の移行	162
24.1. クラスターが FILESTORE を実行している (したがって移行が必要である) ことの確認	162
24.2. FILESTORE から BLUESTORE への OSD の移行	162
24.3. FILESTORE から BLUESTORE への移行の確認	164
第25章 アップグレード後操作の実施	166
25.1. アンダークラウドからの不要なパッケージおよびディレクトリーの削除	166
25.2. アップグレード後の機能検証	166
25.3. オーバークラウドイメージのアップグレード	166
25.4. CPU ピニングパラメーターの更新	168
25.5. メンバーロールへのユーザーの移行	170
第26章 アップグレードに関する問題のトラブルシューティング	172
26.1. 環境ファイルの修正	172

多様性を受け入れるオープンソースの強化

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

RED HAT ドキュメントへのフィードバック (英語のみ)

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

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

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

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

第1章 RED HAT OPENSTACK PLATFORM のアップグレードフレームワークについて

Red Hat OpenStack Platform のアップグレードフレームワークは、Red Hat 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. 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 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。



重要

Red Hat OpenStack Platform 13 は、Red Hat OpenStack Platform 16.1 にアップグレードすることができます。ただし、完全な製品サポートを受ける場合には、Red Hat OpenStack Platform 13 から Red Hat OpenStack Platform 16.2 アップグレードパスのみがサポートされます。13 から 16.1 にアップグレードする場合は、サポート例外を取得する必要があります。

Red Hat OpenStack Platform 16.2 へのアップグレードについての詳しい情報は、[13 から 16.2 へのアップグレードフレームワーク](#) を参照してください。

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 13	Red Hat OpenStack Platform 16.1

1.2. ロングライフバージョンのライフサイクルサポート

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。

1.3. ロングライフリリースのアップグレードパス

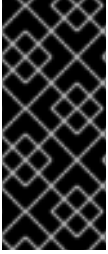
Red Hat では、お使いの環境を次のロングライフリリースにアップグレードするためのオプションを 2 つ提供しています。

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 Red Hat OpenStack Platform 16.1 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。Red Hat OpenStack Platform の並列移行についての詳しい情報は、Red Hat Global Professional Services にお問い合わせください。



重要

以下の表に示す時間は内部テストに基づく最短の推定値であり、すべての実稼働環境には適用されない可能性があります。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。各タスクのアップグレード時間を正確に測定するには、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1.1 アップグレードパスの影響と時間

	インプレースアップグレード	並列移行
アンダークラウドのアップグレード時間	それぞれの主要な操作の推定時間は以下のとおりです。 <ul style="list-style-type: none"> ● Leapp アップグレードコマンド: 30 分間 ● Leapp リブート: 30 分間 ● director のアップグレード: 40 分間 	なし。既存のアンダークラウドに加えて、新しいアンダークラウドを作成します。
オーバークラウドコントロールプレーンのアップグレード時間	コントローラーノードごとの推定時間は以下のとおりです。 <ul style="list-style-type: none"> ● Leapp アップグレードおよびリブート: 60 分間 ● サービスのアップグレード: 60 分間 	なし。既存のコントロールプレーンに加えて、新しいコントロールプレーンを作成します。
コントロールプレーンの機能停止時間	ブートストラップコントローラーノードのサービスアップグレード時間: 約 60 分間	なし。ワークロードの移行中、両方のオーバークラウドは稼働状態にあります。
コントロールプレーンの機能停止による影響	機能停止時間中 OpenStack の操作を行うことはできません。	機能停止時間はありません。
オーバークラウドデータプレーンのアップグレード時間	コンピュートノードおよび Ceph Storage ノードごとの推定時間は以下のとおりです。 <ul style="list-style-type: none"> ● Leapp アップグレードおよびリブート: 60 分間 ● サービスのアップグレード: 30 分間 	なし。既存のデータプレーンに加えて、新しいデータプレーンを作成します。
データプレーンの機能停止時間	ノード間のワークロードの移行により、機能停止時間は最小限に抑えられます。	オーバークラウド間のワークロードの移行により、機能停止時間は最小限に抑えられます。

	インプレースアップグレード	並列移行
追加ハードウェアに関する要件	追加のハードウェアは必要ありません。	新しいアンダークラウドおよびオーバークラウドを作成するために、追加のハードウェアが必要です。

第2章 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。

2.1. RED HAT OPENSTACK PLATFORM 16.1 の理解

アップグレードを実施する前に Red Hat OpenStack Platform 16.1 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 16.1 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。

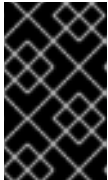
- [Red Hat OpenStack Platform 13](#) (現在のバージョン)
- [Red Hat OpenStack Platform 14](#)
- [Red Hat OpenStack Platform 15](#)
- [Red Hat OpenStack Platform 16.0](#)
- [Red Hat OpenStack Platform 16.1](#) (目的のバージョン)
- バージョン 16.1 の [director のインストールと使用方法](#) を参照し、新たな要件および本ガイドのプロセスについて十分に理解してください。
- 概念実証用の Red Hat OpenStack Platform 16.1 アンダークラウドおよび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。

2.2. RED HAT OPENSTACK PLATFORM 16.1 における変更点の概要

Red Hat OpenStack Platform 16.1 へのアップグレード時に、以下に概要を示す変更が行われます。

- OpenStack Platform director 16.1 では、**config-download** と呼ばれる Ansible による手法を使用してオーバークラウドを設定します。これは、標準の heat ベースの設定手法に代わるものです。director は引き続き heat を使用してプロビジョニング操作のオーケストレーションを行います。
- director のインストールには、オーバークラウドのデプロイメントと同じ手法が使用されません。したがって、アンダークラウドでの各サービスのインストールおよび設定にも、ブループリントとして **openstack-tripleo-heat-templates** が使用されます。
- アンダークラウドでは、OpenStack サービスはコンテナ内で実行されます。
- アンダークラウドは、新たな方法でコンテナイメージをプルして保管します。オーバークラウドのデプロイ前にコンテナイメージをプルする代わりに、アンダークラウドはデプロイメントプロセス中に関連するすべてのコンテナイメージをプルします。

- オーバークラウドのデプロイメントプロセスには、ノードを登録するための Advanced Subscription Management 手法が含まれます。この手法には、OpenStack Platform ノードを登録するための Ansible ロールが組み込まれています。さらに、新しい手法では、必要に応じて異なるノードロールに異なるサブスクリプションを適用します。
- オーバークラウドは、デフォルトの ML2 メカニズムドライバーとして Open Virtual Network (OVN) を使用するようになりました。Open vSwitch (OVS) サービスを OVN に移行することができます。この移行は、アップグレードが正常に完了した後に実施します。
- 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 8 上で動作します。
- **openstack-tripleo-heat-templates** には、**deployment** ディレクトリーに統一されたコンポーザブルサービスプレートコレクションが含まれています。このディレクトリーに含まれるテンプレートのコンテンツは、コンテナ化されたサービスと Puppet ベースのコンポーザブルサービスの両テンプレートからのコンテンツをマージしたものです。
- OpenStack Data Processing サービス (sahara) はサポートされなくなりました。



重要

お使いの Red Hat OpenStack Platform 13 環境で sahara を有効にしている場合には、このアップグレードを続行せず、Red Hat Global Support Services にお問い合わせください。

- Service Telemetry Framework (STF) を優先し、OpenStack Telemetry のコンポーネントは非推奨になりました。

2.3. RED HAT ENTERPRISE LINUX 8 の変更点

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 8 上で動作します。これには、アンダークラウドおよびオーバークラウドに関連する新しいツールおよび機能が含まれます。

- アンダークラウドおよびオーバークラウドは Red Hat Container Toolkit を使用します。コンテナライフサイクルをビルドおよび制御する **docker** に代わって、Red Hat Enterprise Linux 8 には、新しいコンテナイメージをビルドするための **buildah** およびコンテナ管理用の **podman** が含まれます。
- Red Hat Enterprise Linux 8 には **docker-distribution** パッケージが含まれていません。アンダークラウドには、オーバークラウドノードにコンテナイメージを提供するためのプライベート HTTP レジストリーが追加されました。
- Red Hat Enterprise Linux 7 から 8 へのアップグレードプロセスには、**leapp** ツールが使用されます。
- Red Hat Enterprise Linux 8 は、**ntp** サービスを使用しません。その代わりに、Red Hat Enterprise Linux 8 では **chronyd** が使用されます。
- Red Hat Enterprise Linux 8 には、新しいバージョンの高可用性ツールが含まれています。

Red Hat OpenStack Platform 16.1 は、ベースオペレーティングシステムとして Red Hat Enterprise Linux 8.2 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 Red Hat Enterprise Linux 8.2 にアップグレードします。

Red Hat Enterprise Linux 7 と 8 の主な相違点は、[RHEL 8 の導入における検討事項](#) を参照してください。Red Hat Enterprise Linux 8 に関する一般的な情報は、[Product Documentation for Red Hat Enterprise Linux 8](#) を参照してください。

2.4. 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 をアップグレードするには、ベースオペレーティングシステムを Red Hat Enterprise Linux 7 から Red Hat Enterprise Linux 8 にアップグレードする必要があります。Red Hat Enterprise Linux 7 では、Leapp ユーティリティーを使用して Red Hat Enterprise Linux 8 へのアップグレードを実施します。Leapp およびその依存関係を利用できるようにするには、以下の Red Hat Enterprise Linux 7 リポジトリが有効であることを確認します。

- Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server または Red Hat Enterprise Linux 7 Server RPMs x86_64 7.9

```
rhel-7-server-rpms
x86_64 7Server
or:
rhel-7-server-rpms
x86_64 7.9
```

- Red Hat Enterprise Linux 7 Server - Extras RPMs x86_64

```
rhel-7-server-extras-rpms
x86_64
```

詳細は、[アップグレードに向けて RHEL 7 システムの準備](#) を参照してください。

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際、アンダークラウドとオーバークラウドでは別個のプロセスが使用されます。

アンダークラウドのプロセス

openstack undercloud upgrade コマンドを実行する前に、手動で **leapp** によるアップグレードを行います。アンダークラウドのアップグレードには、**leapp** によるアップグレードを実施する手順が含まれます。

オーバークラウドのプロセス

オーバークラウドのアップグレードフレームワークでは、**leapp** によるアップグレードが自動的に実施されます。

制限

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 7 から RHEL 8 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](#)

特に、ディスク全体またはパーティションで暗号化が使用される (LUKS 暗号化、ファイルシステムの暗号化など) ノードで Leapp によるアップグレードを行うことはできません。この制限は、**dmccrypt: true** パラメーターで設定した CephOSD ノードに影響します。

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 7 から RHEL 8 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。

2.5. 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
- **CephConfigOverrides** や **CephAnsibleExtraConfig**などの Ceph Storage カスタム設定を含む 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。コンピュータ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 によって自動的に回避されません。その代わりに、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タノード (DCN) のシナリオ

2.6. 外部の CEPH デプロイメントと組み合わせたアップグレードに関する考慮事項

別途 Red Hat Ceph Storage システムをデプロイしていて、director を使用して OpenStack をデプロイおよび設定している場合は、Red Hat OpenStack Platform のアップグレードフレームワークを使用して、外部の Ceph デプロイメントと共にインプレースアップグレードを行うことができます。このシナリオは、director を使用してデプロイされた Ceph クラスターのアップグレードとは異なります。

外部の Ceph デプロイメントと組み合わせたインプレースアップグレードのプランニングおよび準備を行う際に考慮すべき違いは以下のとおりです。

1. Red Hat OpenStack Platform デプロイメントをバージョン 13 からバージョン 16.1 にアップグレードする前に、Red Hat Ceph Storage クラスターをバージョン 3 からバージョン 4 にアップグレードする必要があります。詳細は、[Red Hat Ceph Storage 4 インストールガイドの Red Hat Ceph Storage クラスターのアップグレード](#) を参照してください。
2. Red Hat Ceph Storage クラスターをバージョン 3 からバージョン 4 にアップグレードした後も、Red Hat OpenStack Platform 13 では引き続き RHCSv3 クライアントコンポーネントが実行されている可能性があります。これらは RHCSv4 クラスターに対して互換性があります。
3. 13 から 16.1 へのアップグレードフレームワークに記載のアップグレードパスに従うことができます。該当する場合は、この特定のシナリオをサポートする条件ステップを実行する必要があります。条件ステップは、外部の Ceph デプロイメントと共にアップグレードする場合はで始まる箇所です。
4. 外部の Ceph デプロイメントと共にアップグレードする場合は、オーバークラウドのアップグレードプロセスの一部として RHCSv4 **ceph-ansible** をインストールします。director を使用してデプロイされた Ceph クラスターをアップグレードする場合は、オーバークラウドのアップグレードプロセスの完了後に RHCSv4 **ceph-ansible** をインストールします。

重要

Red Hat Ceph Storage クラスターを、以前のサポート対象バージョンからバージョン 4.2z2 にアップグレードする場合、**モニターがセキュアでない global_id 再要求を許可すること**を示す警告メッセージと共に、ストレージクラスターが **HEALTH_WARN** の状態でアップグレードが完了します。これは、パッチが適用された CVE (CVE-2021-20288) が原因です。[Ceph HEALTH_WARN with 'mons are allowing insecure global_id reclaim' after install/upgrade to RHCS 4.2z2 \(or newer\)](#) を参照してください。

CVE により **HEALTH_WARN** の状態が表示されるため、一時的に健全性についての警告をミュートすることができます。ただし、警告をミュートすると、古いクライアントおよびパッチが適用されていないクライアントがクラスターに接続されても表示されないリスクがあります。健全性についての警告のミュートについての詳細は、[Red Hat Ceph Storage のドキュメントの Upgrading a Red Hat Ceph Storage cluster](#) を参照してください。

重要

すべてのクライアントがアップグレードしてパッチを当てたまで **global_id_reclaim** は、無効にしないでください。そうでないと、接続できません。**root** ユーザーで以下のコマンドを実行して、クラスターに接続されているクライアントでパッチが当てられていないものをリストで表示できます。

```
# ceph health detail
```

2.7. 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#1997351 - (13→16.1)ブートストラップコントローラーのアップグレード後にインスタンスにアクセスできない

ML2-OVN でデプロイされた Red Hat OpenStack Platform (RHOSP) 13 環境をアップグレードすると、コントローラーノードのアップグレードプロセスに失敗する可能性があります。Leapp のリブート後に、SELinux パーミッションが拒否されることが原因で **ovn-dbs** コンテナが起動しないことがあります。BZ#1997351 のバグを回避する方法は、Red Hat ナレッジベースのソリューション [OVN fails to configure after reboot during OSP-13 → OSP-16.1 FFU](#) を参照してください。

BZ#1902849 - 以前に osp8、osp10 からアップグレードしたクラスターで osp13-osp16.1 ffu が失敗する

以前にバージョン RHOSP 10 からアップグレードした Red Hat OpenStack Platform (RHOSP) 環境には、BZ#1902849 を避けるために **python-docker** パッケージが必要です。詳細は、Red Hat ナレッジベースのソリューション [osp13-osp16.1 ffu fails on older environments missing python-docker package](#) を参照してください。

BZ#1925078 - RHOSP13-16.1 FFU: 間違った ceph イメージを参照して失敗した後、オーバークラウドのアップグレードがコントローラーで停止する

OSP13 で UEFI ブートおよび UEFI ブートローダーを使用するシステムは、UEFI の問題が発生し以下のような状況に陥る可能性があります。

- **/etc/fstab** が更新されない
- EFI システムで grub-install が誤って使用される

詳細は、Red Hat ナレッジベースのソリューション [FFU 13 to 16.1: Leapp fails to update the kernel on UEFI based systems and /etc/fstab does not contain the EFI partition](#) を参照してください。

システムで UEFI が使用されている場合は、Red Hat Technical Support にお問い合わせください。

BZ#1895887 - [FFWD] ovs+dpdk fail to attach device OvsDpdkHCI

Leapp ユーティリティーを使用してアップグレードした後、OVS-DPDK 負荷を持つコンピュータノードが正しく機能しません。この問題を解決するには、次のいずれかの操作を行います。コンピュータノードをアップグレードする前に、**/etc/modules-load.d/vfio-pci.conf** ファイルを削除します。

または

コンピュータノードのアップグレード後に、コンピュータノードで **ovs-vswitchd** サービスを再起動します。

この問題は RHOSP 16.1.3 に影響を及ぼします。詳細については、Red Hat ナレッジベースのソリューション [OVS-DPDK errors after Framework Upgrade from OSP 13 to 16.1 on HCI compute node](#) を参照してください。

BZ#1936419 - FFU 13-16.1 アップグレード: ceph ノードでの Leapp アップグレードは、leap パラメーターが Fast datapath リポジトリを有効にしようとして失敗する

Ceph のサブスクリプションを使用し、Ceph ストレージノード用に **overcloud-minimal** イメージを使用するように director を設定している場合、Leapp の制限により Ceph ストレージノードのオペレーティングシステムのアップグレードに失敗する可能性があります。この問題を回避するには、**system_upgrade** の実行ステップの後に、Ceph ノードにログインして RHEL のマイナーリリースバージョンの設定を解除し、利用可能な最新の RHEL マイナーリリースバージョンに更新し、ノードをリブートする必要があります。

Leapp アップグレード用の RPM コンテンツをホストするのに Red Hat Satellite Server を使用している場合、使用するコンテンツビューに以下の 8.2 リポジトリを追加する必要があります。

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

```
rhel-8-for-x86_64-appstream-rpms
x86_64 8.2
```

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

```
rhel-8-for-x86_64-baseos-rpms
x86_64 8.2
```

本ガイドには、この問題を避けるための回避策が含まれています。

BZ#2016144 - FFU 13-16.1: Leapp アップグレードの再起動時に `openvswitch` が `Starting ovssdb-server ovssdb-server: /var/run/openvswitch/ovssdb-server.pid.tmp: create failed (Permission denied)` とのエラーで起動に失敗する

以前のバージョンからアップグレードされた Red Hat Open Stack Platform (RHOSP) 環境には、`/etc/systemd/system/ovs*` に不要なファイルが含まれている可能性があります。RHOSP 13 から RHOSP 16.1 へのオーバークラウドのアップグレードプロセスを開始する前に、これらのファイルを削除する必要があります。

BZ#2008976 - Leapp の依存関係で失敗した Leapp アップグレード後の Python2 パッケージがクリーンアップする

Leapp バージョン 5.0.8-100.202109241452Z.1332835 では、Leapp パッケージを保持する DNF の `exclude` オプションのため、`python2` Leapp パッケージの自動削除は失敗します。

`UpgradeInitCommand` パラメーターを環境ファイルに追加し、DNF の `exclude` ステートメントを削除します。

```
parameter defaults:
UpgradeInitCommand: "sudo dnf config-manager --save --setopt exclude=""
```

詳しくは、[アップグレード環境ファイルの作成](#)をご覧ください。

BZ#1978228 - OSP13→16.2 Leapp アップグレードが `TLSEverywhere` で失敗する

お使いの環境で `TLS-Everywhere` を使用していて、`authconfig` から `authselect` に移行したい場合は、`authselect_check.confirm` パラメーターを `True` に設定してください。それ以外の場合は、この値を `False` に設定します。詳しくは、[アップグレード環境ファイルの作成](#)をご覧ください。

BZ#2021525 - openstack オーバークラウドアップグレード実行がタイムアウトする / HAProxy コンテナーが起動しない

Red Hat Open Stack Platform (RHOSP) 13 から RHOSP 16.1 へのアップグレードは、無効な SELinux ラベルが原因でデプロイメントステップ中に失敗することがあります。解決策や詳細については、Red Hat ナレッジベースのソリューション [Pacemaker managed services might not restart during an OSP13 - OSP16.x FFU](#) を参照してください。

BZ#2015325 - FFU: upgrade failed during "Upgrade Mysql database from a temporary container" step

Red Hat Enterprise Linux には `mariadb-server` 用のアップグレード可能な RPM が含まれていますが、これが Red Hat Open Stack Platform (RHOSP) のコンテナ化された `mariadb` のアップグレードを妨害します。RHOSP のアップグレードを行う前に、コントローラーホストから `mariadb-server` パッケージを削除します。詳しくは、[アップグレード環境ファイルの作成](#)をご覧ください。

BZ#2024447 - FFU RHOSP 13 から 16 の FFU 実行中に、配置ユーザーの ID サービス (keystone) パスワードが NovaPassword によって上書きされていた

Red Hat OpenStack Platform 13 から 16.1 へのアップグレード中に、**NovaPassword** パラメーターの値を定義し、**PlacementPassword** パラメーターの値を定義しない場合、**NovaPassword** パラメーターは配置ユーザーの OpenStack Identity サービス (keystone) パスワードをオーバーライドします。Identity サービスのパスワードを保存するには、**parameter_defaults** セクションに **NovaPassword** または **PlacementPassword** を設定しないでください。

parameter_defaults セクションで両方のパスワードを設定すると、コンピュータノードはアップグレードされるまでコントロールプレーンと通信できなくなる可能性があります。コンピュータノードのアップグレードの詳細は、[コンピュータノードのアップグレード](#) を参照してください。

また、**NovaPassword**、**PlacementPassword** のいずれか、またはその両方を使用してオーバークラウドを RHOSP 13 にデプロイしている場合、これらのパスワードをテンプレートから削除し、RHOSP 16.1 にアップグレードする前に RHOSP 13 で **openstack overcloud deploy** コマンドを実行する必要があります。

BZ#2164396 - FFU: FFU に対して有効にする Redhat Satellite ツールリポジトリ(13 から 16.2)

Satellite バージョン 6.7 を使用している場合は、Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64 リポジトリを有効にするとアップグレードに失敗します。適切なパッケージをインストールできないため、エラーが発生します。Red Hat エンジニアリングチームは現在、この問題を調査しています。

BZ#2245602 - アップグレード(OSP16.2 →OSP17.1) controller-0 は、パッケージがない ovn2.15 openvswitch2.15 により leapp のアップグレードを実行しません

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、**--answers-file answer-upgrade.yaml** ファイルに **system_upgrade.yaml** ファイルを含めないでください。**system_upgrade.yaml** ファイルがこのファイルに含まれている場合、**environments/lifecycle/upgrade-prepare.yaml** ファイルは **system_upgrade.yaml** ファイル内のパラメーターを上書きします。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 /
```

この回避策により、**system_upgrade.yaml** ファイルで設定されるパラメーターは、**environments/lifecycle/upgrade-prepare.yaml** ファイルのデフォルトのパラメーターを上書きします。

Red Hat Ceph Storage の問題

BZ#1855813 - Ceph ツールのリポジトリは外部アップグレードを実行する前かつコンバート後のみに RHCS3 から RHCS4 に切り替える必要がある

アンダークラウド上の **ceph-ansible** Playbook コレクションにより、オーバークラウド上に Red Hat Ceph Storage コンテナがデプロイされます。環境をアップグレードするには、アップグレードプロセス中 Ceph Storage 3 コンテナを維持するために、Red Hat Ceph Storage 3 バージョンの **ceph-ansible** が必要です。本ガイドには、Ceph Storage 4 へのアップグレード準備が整うまで、アップグレードプロセス中 **ceph-ansible** バージョン 3 を維持する手順が含まれています。13 から 16.1 へのアップグレードを実施する前に、Red Hat OpenStack Platform 13 環境のマイナーバージョン更新を実施し、**ceph-ansible** のバージョンが 3.2.46 以降になるようにしてください。

2.8. バックアップと復元

Red Hat OpenStack Platform 13 環境をアップグレードする前に、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。Relax-and-recover (ReaR) ユーティリティーを使用したノードのバックアップに関する詳細は、[アンダークラウドとコントロールプレーンのバックアップおよびリストア](#)を参照してください。

- 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バックアップの詳細については、[Red Hat OpenStack Platform 13 Undercloud and Control Plane Back Up and Restore](#) を参照してください。
- アップグレードした後に各ノードをバックアップすることができます。アップグレードされたノードのバックアップの詳細については、[Red Hat OpenStack Platform 16.1 アンダークラウドとコントロールプレーンのバックアップおよび復元](#) 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を実施してからオーバークラウドのアップグレードを実施する前に、アンダークラウドノードで実行されるデータベースをバックアップすることができます。アンダークラウドデータベースのバックアップの詳細については、[Red Hat OpenStack Platform 16.1 アンダークラウドとコントロールプレーンのバックアップおよび復元](#) の [アンダークラウドノードのデータベースバックアップ作成](#) を参照してください。

2.9. マイナーバージョンの更新

Red Hat OpenStack Platform 環境をアップグレードする前に、環境を現在のリリースの最新マイナーバージョンに更新します。たとえば、Red Hat OpenStack Platform 16.1 へのアップグレードを実施する前に、お使いの Red Hat OpenStack Platform 13 環境を最新の 13 に更新します。

Red Hat OpenStack Platform 13 のマイナーバージョンの更新を実施する手順は、[Red Hat OpenStack Platform の最新状態の維持](#) を参照してください。

2.10. プロキシ設定

Red Hat OpenStack Platform 13 環境でプロキシを使用している場合、`/etc/environment` ファイルのプロキシ設定は、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 16.1 へのアップグレード後も維持されます。

- Red Hat OpenStack Platform 13 のプロキシ設定についての詳しい情報は、[プロキシ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 16.1 のプロキシ設定についての詳しい情報は、[プロキシ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。

2.11. アップグレード前の RED HAT OPENSTACK PLATFORM 13 の検証

Red Hat OpenStack Platform 16.1 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およびオーバークラウドを検証します。Red Hat OpenStack Platform 13 において、これらの Playbook を OpenStack Workflow サービス (mistral) を使用して実行します。



注記

CDN または Satellite をリポジトリソースとして使用すると、検証に失敗します。この問題を解決するには、Red Hat ナレッジベースのソリューション [Repository validation fails because of SSL certificate error](#) 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. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

3. 以下の内容で、**pre-upgrade-validations.sh** という名前の bash スクリプトを作成します。

```
#!/bin/bash
for VALIDATION in $(openstack action execution run tripleo.validations.list_validations
'{"groups": ["pre-upgrade"]}' | jq ".result[] | .id")
do
  echo "=== Running validation: $VALIDATION ==="
  STACK_NAME=$(openstack stack list -f value -c 'Stack Name')
  ID=$(openstack workflow execution create -f value -c ID tripleo.validations.v1.run_validation
"{\"validation_name\": \"$VALIDATION, \"plan\": \"${STACK_NAME}\"}")
  while [ $(openstack workflow execution show $ID -f value -c State) == "RUNNING" ]
  do
    sleep 1
  done
  echo ""
  openstack workflow execution output show $ID | jq -r ".stdout"
  echo ""
done
```

4. スクリプトを実行する権限を追加します。

```
$ chmod +x pre-upgrade-validations.sh
```

5. スクリプトを実行します。

```
$ ./pre-upgrade-validations.sh
```

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
```


第3章 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。

3.1. アンダークラウドのリポジトリー

Red Hat OpenStack Platform (RHOSP) 16.1 は、Red Hat Enterprise Linux 8.2 上で動作します。したがって、これらのリポジトリーからのコンテンツを該当する Red Hat Enterprise Linux バージョンにロックする必要があります。



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、リポジトリーラベルは同じままです。たとえば、BaseOS リポジトリーの 8.4 バージョンを有効にした場合、リポジトリー名には有効にした特定のバージョンが含まれますが、リポジトリーラベルは依然として **rhel-8-for-x86_64-baseos-eus-rpms** です。



警告

ここで指定する以外のリポジトリーは、サポートされません。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 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 9 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。

名前	リポジトリ	要件の説明
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。コントローラーノードの高可用性に使用します。
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64 (RPMs)	ansible-2.9-for-rhel-8-x86_64-rpms	Red Hat Enterprise Linux 用 Ansible エンジン。最新バージョンの Ansible を提供するために使用されます。
Advanced Virtualization for RHEL 8 x86_64 (RPMs)	advanced-virt-for-rhel-8-x86_64-eus-rpms	OpenStack Platform 用仮想化パッケージを提供します。
Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64	satellite-tools-6.5-for-rhel-8-x86_64-rpms	Red Hat Satellite 6 でのホスト管理ツール
Red Hat OpenStack Platform 16.1 for RHEL 8 (RPMs)	openstack-16.1-for-rhel-8-x86_64-rpms	Red Hat OpenStack Platform のコアリポジトリ。Red Hat OpenStack Platform director のパッケージが含まれます。
Red Hat Fast Datapath for RHEL 8 (RPMS)	fast-datapath-for-rhel-8-x86_64-rpms	OpenStack Platform 用 Open vSwitch (OVS) パッケージを提供します。

Ceph リポジトリ

以下の表には、アンダークラウド用の Ceph Storage 関連のリポジトリをまとめています。

名前	リポジトリ	要件の説明
Red Hat Ceph Storage Tools 4 for RHEL 8 x86_64 (RPMs)	rhceph-4-tools-for-rhel-8-x86_64-rpms	Ceph Storage クラスタと通信するためのノード用のツールを提供します。オーバークラウドで Ceph Storage を使用する場合、または既存の Ceph Storage クラスタと統合する場合、アンダークラウドにはこのリポジトリからの ceph-ansible パッケージが必要です。

IBM POWER 用リポジトリ

次の表には、POWER PC アーキテクチャー上の RHOSP のリポジトリのリストが含まれています。コアリポジトリの該当リポジトリの代わりに、これらのリポジトリを使用してください。

名前	リポジトリ	要件の説明
Red Hat Enterprise Linux for IBM Power, little endian - BaseOS (RPMs)	rhel-8-for-ppc64le-baseos-rpms	ppc64le 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
Red Hat Enterprise Linux 8 for IBM Power, little endian - AppStream (RPMs)	rhel-8-for-ppc64le-appstream-rpms	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat Enterprise Linux 8 for IBM Power, little endian - High Availability (RPMs)	rhel-8-for-ppc64le-highavailability-rpms	Red Hat Enterprise Linux の高可用性ツール。コントローラーノードの高可用性に使用します。
Red Hat Fast Datapath for RHEL 8 IBM Power, little endian (RPMs)	fast-datapath-for-rhel-8-ppc64le-rpms	OpenStack Platform 用 Open vSwitch (OVS) パッケージを提供します。
Red Hat Ansible Engine 2.8 for RHEL 8 IBM Power, little endian (RPMs)	ansible-2.8-for-rhel-8-ppc64le-rpms	Red Hat Enterprise Linux 用 Ansible エンジン。最新バージョンの Ansible を提供します。
Red Hat OpenStack Platform 16.1 for RHEL 8 (RPMs)	openstack-16.1-for-rhel-8-ppc64le-rpms	ppc64le システム用 Red Hat OpenStack Platform のコアリポジトリ

3.2. オーバークラウドのリポジトリ

Red Hat OpenStack Platform (RHOSP) 16.1 は、Red Hat Enterprise Linux 8.2 上で動作します。したがって、これらのリポジトリからのコンテンツを該当する Red Hat Enterprise Linux バージョンにロックする必要があります。



注記

リポジトリを 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、リポジトリラベルは同じままです。たとえば、BaseOS リポジトリの 8.4 バージョンを有効にした場合、リポジトリ名には有効にした特定のバージョンが含まれますが、リポジトリラベルは依然として **rhel-8-for-x86_64-baseos-eus-rpms** です。



警告

ここで指定する以外のリポジトリは、サポートされません。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。

コントローラーノード用リポジトリ

以下の表には、オーバークラウドのコントローラーノード用コアリポジトリをまとめています。

名前	リポジトリ	要件の説明
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
Red Hat Enterprise Linux 9 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。
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64 (RPMs)	ansible-2.9-for-rhel-8-x86_64-rpms	Red Hat Enterprise Linux 用 Ansible エンジン。最新バージョンの Ansible を提供するために使用されます。
Advanced Virtualization for RHEL 8 x86_64 (RPMs)	advanced-virt-for-rhel-8-x86_64-eus-rpms	OpenStack Platform 用仮想化パッケージを提供します。
Red Hat OpenStack Platform 16.1 for RHEL 8 (RPMs)	openstack-16.1-for-rhel-8-x86_64-rpms	Red Hat OpenStack Platform のコアリポジトリ
Red Hat Fast Datapath for RHEL 8 (RPMs)	fast-datapath-for-rhel-8-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 4 for RHEL 8 x86_64 (RPMs)	rhceph-4-tools-for-rhel-8-x86_64-rpms	Red Hat Enterprise Linux 8 での Red Hat Ceph Storage 4 用ツール
Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64	satellite-tools-6.5-for-rhel-8-x86_64-rpms	Red Hat Satellite 6 でのホスト管理ツール

Compute ノードおよび ComputeHCI ノードのリポジトリ

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を示します。

名前	リポジトリ	要件の説明
----	-------	-------

名前	リポジトリ	要件の説明
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
Red Hat Enterprise Linux 9 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。
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64 (RPMs)	ansible-2.9-for-rhel-8-x86_64-rpms	Red Hat Enterprise Linux 用 Ansible エンジン。最新バージョンの Ansible を提供するために使用されます。
Advanced Virtualization for RHEL 8 x86_64 (RPMs)	advanced-virt-for-rhel-8-x86_64-eus-rpms	OpenStack Platform 用仮想化パッケージを提供します。
Red Hat OpenStack Platform 16.1 for RHEL 8 (RPMs)	openstack-16.1-for-rhel-8-x86_64-rpms	Red Hat OpenStack Platform のコアリポジトリ
Red Hat Fast Datapath for RHEL 8 (RPMs)	fast-datapath-for-rhel-8-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 4 for RHEL 8 x86_64 (RPMs)	rhceph-4-tools-for-rhel-8-x86_64-rpms	Red Hat Enterprise Linux 8 での Red Hat Ceph Storage 4 用ツール
Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64	satellite-tools-6.5-for-rhel-8-x86_64-rpms	Red Hat Satellite 6 でのホスト管理ツール

リアルタイムコンピュータ用リポジトリ

以下の表には、リアルタイムコンピュータ (RTC) 機能用リポジトリをまとめています。

名前	リポジトリ	要件の説明
----	-------	-------

名前	リポジトリ	要件の説明
Red Hat Enterprise Linux 8 for x86_64 - Real Time (RPMs)	rhel-8-for-x86_64-rt-rpms	リアルタイム KVM (RT-KVM) のリポジトリ。リアルタイムカーネルを有効化するためのパッケージが含まれています。RT-KVM 対象の全コンピュータノードで、このリポジトリを有効にします。注記: このリポジトリにアクセスするには、別途 Red Hat OpenStack Platform for Real Time SKU のサブスクリプションが必要です。
Red Hat Enterprise Linux 8 for x86_64 - Real Time for NFV (RPMs)	rhel-8-for-x86_64-nfv-rpms	NFV 向けのリアルタイム KVM (RT-KVM) のリポジトリ。リアルタイムカーネルを有効化するためのパッケージが含まれています。RT-KVM 対象の全 NFV コンピュータノードで、このリポジトリを有効にします。注記: このリポジトリにアクセスするには、別途 Red Hat OpenStack Platform for Real Time SKU のサブスクリプションが必要です。

Ceph Storage ノード用リポジトリ

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

名前	リポジトリ	要件の説明
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
Red Hat Enterprise Linux 9 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。注記: Ceph Storage ロールに overcloud-full イメージを使用した場合は、このリポジトリを有効にする必要があります。Ceph Storage ロールは、このリポジトリを必要としない overcloud-minimal イメージを使用する必要があります。

名前	リポジトリ	要件の説明
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64 (RPMs)	ansible-2.9-for-rhel-8-x86_64-rpms	Red Hat Enterprise Linux 用 Ansible エンジン。最新バージョンの Ansible を提供するために使用されます。
Red Hat OpenStack Platform 16.1 Director Deployment Tools for RHEL 8 x86_64 (RPMs)	openstack-16.1-deployment-tools-for-rhel-8-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 を組み合わせたサブスクリプションを使用する場合は、 openstack-16.1-for-rhel-8-x86_64-rpms リポジトリを使用します。
Red Hat OpenStack Platform 16.1 for RHEL 8 (RPMs)	openstack-16.1-for-rhel-8-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。このリポジトリは、OpenStack Platform と 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 Ceph Storage サブスクリプションを使用する場合は、 openstack-16.1-deployment-tools-for-rhel-8-x86_64-rpms リポジトリを使用します。
Red Hat Ceph Storage Tools 4 for RHEL 8 x86_64 (RPMs)	rhceph-4-tools-for-rhel-8-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 8 (RPMs)	fast-datapath-for-rhel-8-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。

IBM POWER 用リポジトリ

次の表に、POWER PC アーキテクチャー上の RHOSP のリポジトリをまとめています。コアリポジトリの該当リポジトリの代わりに、これらのリポジトリを使用してください。

名前	リポジトリ	要件の説明
Red Hat Enterprise Linux for IBM Power, little endian - BaseOS (RPMs)	rhel-8-for-ppc64le-baseos-rpms	ppc64le 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
Red Hat Enterprise Linux 8 for IBM Power, little endian - AppStream (RPMs)	rhel-8-for-ppc64le-appstream-rpms	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat Enterprise Linux 8 for IBM Power, little endian - High Availability (RPMs)	rhel-8-for-ppc64le-highavailability-rpms	Red Hat Enterprise Linux の高可用性ツール。コントローラーノードの高可用性に使用します。
Red Hat Fast Datapath for RHEL 8 IBM Power, little endian (RPMs)	fast-datapath-for-rhel-8-ppc64le-rpms	OpenStack Platform 用 Open vSwitch (OVS) パッケージを提供します。
Red Hat Ansible Engine 2.8 for RHEL 8 IBM Power, little endian (RPMs)	ansible-2.8-for-rhel-8-ppc64le-rpms	Red Hat Enterprise Linux 用 Ansible エンジン。最新バージョンの Ansible を提供するために使用されます。
Red Hat OpenStack Platform 16.1 for RHEL 8 (RPMs)	openstack-16.1-for-rhel-8-ppc64le-rpms	ppc64le システム用 Red Hat OpenStack Platform のコアリポジトリ

3.3. 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 環境用の RPM およびコンテナイメージをホストする場合、Red Hat OpenStack Platform 16.1 のアップグレード中に Satellite Server 6 を使用してコンテンツを提供する際には、特定の考慮事項があります。

現在の環境についての仮定

- Satellite Server がすでに Red Hat OpenStack Platform 13 の RPM およびコンテナイメージをホストしている。
- Red Hat OpenStack Platform 13 環境内の全ノードを、ご自分の Satellite Server サーバーにすでに登録している。たとえば、以前に Red Hat OpenStack Platform 13 の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 OpenStack Platform 13 のコンテンツに登録した。

Red Hat OpenStack Platform のアップグレードに関する推奨事項

- Red Hat OpenStack Platform 13 のアンダークラウドおよびオーバークラウドの両方に必要な RPM リポジトリを有効にして同期します。これには、必要な Red Hat Enterprise Linux 8.2 リポジトリが含まれます。
- ご自分の Satellite Server にカスタム製品を作成し、以下の Red Hat OpenStack Platform バージョン用のコンテナイメージをホストします。

- Red Hat OpenStack Platform 16.1
- Red Hat OpenStack Platform 15
- Red Hat OpenStack Platform 16.1 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、以下のコンテンツをコンテンツビューに含めます。
 - 以下の Red Hat Enterprise Linux 7 リポジトリ
 - Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server または Red Hat Enterprise Linux 7 Server RPMs x86_64 7.9

```
rhel-7-server-rpms
x86_64 7Server
or:
rhel-7-server-rpms
x86_64 7.9
```

- Red Hat Enterprise Linux 7 Server - Extras RPMs x86_64

```
rhel-7-server-extras-rpms
x86_64
```

- Red Hat Enterprise Linux 8.2 リポジトリを含む、アンダークラウドおよびオーバークラウドの全 RPM リポジトリ Red Hat Enterprise Linux リポジトリの正しいバージョン (8.2) が含まれていることを確認してください。正しいバージョンが含まれていないと、RHEL 8 のリポジトリを有効にする際に問題が発生する可能性があります。詳細については、Red Hat ナレッジベースのソリューション [RHEL 7 to RHEL 8 LEAPP Upgrade Failing When Using Red Hat Satellite](#) を参照してください。
- Red Hat OpenStack Platform 16.1 コンテナイメージ
- Red Hat OpenStack Platform 15 コンテナイメージ
- アクティベーションキーを、Red Hat OpenStack Platform 16.1 のアップグレード用に作成した Red Hat OpenStack Platform 16.1 の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 によるアップグレードではこのパッケージがアップグレードされず、パッケージが Red Hat Enterprise Linux 8.2 システムに残されるので、**subscription-manager** がエラーを報告します。
- Red Hat OpenStack Platform 16.1 コンテナイメージをホストするように Satellite Server を設定することができます。詳細は、**director のインストールと使用方法の [コンテナイメージ管理用 Satellite サーバーの準備](#)** を参照してください。
- taCeph のサブスクリプションを使用し、Ceph ストレージノード用に **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 Red Hat Enterprise Linux (RHEL) 8.2 リポジトリを追加する必要があります。
 - Red Hat Enterprise Linux 8 for x86_64 - AppStream (RPMs)

```
rhel-8-for-x86_64-appstream-rpms
x86_64 8.2
```

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

```
rhel-8-for-x86_64-baseos-rpms  
x86_64 8.2
```

詳細は、[Red Hat Satellite コンテンツ管理ガイド](#)の [Red Hat コンテンツのインポート](#) および [コンテンツビューの管理](#) を参照してください。

第4章 アンダークラウドアップグレードの準備

アンダークラウドのアップグレードを実施する前に、アンダークラウドのアップグレードが正常に実行されるように準備の手順を完了する必要があります。

4.1. 外部の CEPH と組み合わせたアップグレードの前提条件

外部の Ceph デプロイメントと共にアップグレードする場合、Red Hat OpenStack Platform デプロイメントをアップグレードする前に、Red Hat Ceph Storage クラスタをバージョン 3 からバージョン 4 にアップグレードする必要があります。詳細は、[Red Hat Ceph Storage 4 インストールガイドの Red Hat Ceph Storage クラスタのアップグレード](#) を参照してください。

4.2. 新たなメモリー要件

Red Hat OpenStack Platform 16.1 では、アンダークラウドに新たなメモリー要件が適用されます。

Red Hat OpenStack Platform 13	Red Hat OpenStack Platform 16.1
16 GB RAM	24 GB RAM

アップグレードを続行する前に、アンダークラウドがこれらの新たな要件を満たすことを確認してください。

4.3. 予測可能なアンダークラウドノード NIC 名の使用

アンダークラウドノードで Leapp によるアップグレードを実施する前に、カーネルベースの NIC 名が使用されているかどうかを確認する必要があります。この NIC 名には、通常 **eth** の接頭辞が含まれます。NIC の割り当てに関して、通常これらの NIC 名は予測可能ではありません。

playbook-nics.yaml Playbook を実行して NIC の名前を変更し、インターフェイスの **NIC** 接頭辞を使用することができます。また、Playbook の実行時に **prefix** 変数を変更することで、別の NIC 接頭辞を設定することもできます。ただし、NIC の変更は、Leapp によるアップグレードプロセスが完了し、ノードがリブートされた後のみ適用されます。

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. **playbook-nics.yaml** という名前の Ansible Playbook を作成し、以下のコンテンツを Playbook にコピーします。

```
---
- name: Rename eth devices
  hosts: all
  become: yes
  vars:
    prefix: "em"
    undercloud_conf: "/home/stack/undercloud.conf"
    osnet_conf: "/etc/os-net-config/config.json"
  tasks:
    - set_fact:
        eth_interfaces: "{{ ansible_interfaces | select('match','eth.*') | list }}"
```

```

- debug:
  msg: "{{ eth_interfaces }}"
- name: Update udev rules
  lineinfile:
    line: "SUBSYSTEM==\"net\", ACTION==\"add\", DRIVERS==\"?*\", ATTR{address}==\"
{{ ansible_facts[item]['perm_macaddress'] | default(ansible_facts[item]['macaddress']) }}\",
NAME=\"{{ item|replace('eth',prefix) }}\"
    path: /etc/udev/rules.d/70-rhosp-persistent-net.rules
    create: True
  with_items: "{{ eth_interfaces }}"
- name: Rename eth files
  block:
    - name: Check that eth files exists
      stat:
        path: /etc/sysconfig/network-scripts/ifcfg-{{ item }}
      register: nic_result
      with_items: "{{ eth_interfaces }}"
    - name: Copy nic files using the new prefix
      copy:
        remote_src: True
        src: "{{ item.stat.path }}"
        dest: "{{ item.stat.path|replace('eth',prefix) }}"
        with_items: "{{ nic_result.results }}"
        when: item.stat.exists
    - name: Edit NAME in new network-script files
      lineinfile:
        regexp: "^NAME=.*"
        line: "NAME={{ item.item|replace('eth',prefix) }}"
        path: "{{ item.stat.path|replace('eth',prefix) }}"
        with_items: "{{ nic_result.results }}"
        when: item.stat.exists
    - name: Edit DEVICE in new network-script files
      lineinfile:
        regexp: "^DEVICE=.*"
        line: "DEVICE={{ item.item|replace('eth',prefix) }}"
        path: "{{ item.stat.path|replace('eth',prefix) }}"
        with_items: "{{ nic_result.results }}"
        when: item.stat.exists
    - name: Backup old eth network-script files
      copy:
        remote_src: True
        src: "{{ item.stat.path }}"
        dest: "{{ item.stat.path }}.bak"
        with_items: "{{ nic_result.results }}"
        when: item.stat.exists
    - name: Remove old eth network-script files
      file:
        path: "{{ item.stat.path }}"
        state: absent
        with_items: "{{ nic_result.results }}"
        when: item.stat.exists
    - name: Rename route files
      block:
        - name: Check that route files exists
          stat:
            path: /etc/sysconfig/network-scripts/route-{{ item }}

```

```

register: route_result
with_items: "{{ eth_interfaces }}"
- name: Copy route files using the new prefix
copy:
  remote_src: True
  src: "{{ item.stat.path }}"
  dest: "{{ item.stat.path|replace('eth',prefix) }}"
with_items: "{{ route_result.results }}"
when: item.stat.exists
- name: Update prefix in route files that use IP command arguments format
replace:
  regexp: "eth"
  replace: "{{ prefix }}"
  path: "{{ item.stat.path|replace('eth',prefix) }}"
with_items: "{{ route_result.results }}"
when: item.stat.exists
- name: Backup old route files
copy:
  remote_src: True
  src: "{{ item.stat.path }}"
  dest: "{{ item.stat.path }}.bak"
with_items: "{{ route_result.results }}"
when: item.stat.exists
- name: Remove old route files
file:
  path: "{{ item.stat.path }}"
  state: absent
with_items: "{{ route_result.results }}"
when: item.stat.exists
- name: Perform a final regex for any remaining eth prefixes in ifcfg files
block:
  - name: Get a list of all ifcfg files
  find:
    paths: /etc/sysconfig/network-scripts/
    patterns: 'ifcfg-*'
    excludes: '*.bak'
  register: ifcfg_files
  - name: Perform final regex on ifcfg files
  replace:
    path: "{{ item[0].path }}"
    regexp: "{{ item[1] }}"
    replace: "{{ item[1]|replace('eth',prefix) }}"
  with_nested:
    - "{{ ifcfg_files.files }}"
    - "{{ eth_interfaces }}"
- name: Replace interface name in files referencing old eth interface
block:
  - name: Check if undercloud.conf exists
  stat:
    path: "{{ undercloud_conf }}"
  register: undercloud_conf_stat
  - name: Replace interface name in undercloud.conf
  replace:
    path: "{{ undercloud_conf }}"
    regexp: 'eth(d+)'
    replace: "{{ prefix }}\1"

```

```

when: undercloud_conf_stat.stat.exists
- name: Check if os-net-config's config.json exists
  stat:
    path: "{{ osnet_conf }}"
    register: osnet_conf_stat
- name: Replace interface name in config.json
  replace:
    path: "{{ osnet_conf }}"
    regexp: 'eth(d+)'
    replace: "{{ prefix }}\1"
  when: osnet_conf_stat.stat.exists
- name: Patch vlan devices
  block:
  - name: Check that vlan files exists
    stat:
      path: /etc/sysconfig/network-scripts/ifcfg-{{ item }}
    register: nic_result
    when: item.startswith("vlan")
    with_items: "{{ ansible_interfaces }}"
  - name: Backup old vlan network-script files
    copy:
      remote_src: True
      src: "{{ item.stat.path }}"
      dest: "{{ item.stat.path }}.bak"
    when: item.item.startswith("vlan") and item.stat.exists
    with_items: "{{ nic_result.results }}"
  - name: Edit PHYSDEV in new network-script files
    replace:
      path: "{{ item.stat.path }}"
      regexp: "^PHYSDEV=eth"
      replace: "PHYSDEV={{ prefix }}"
    when: item.item.startswith("vlan") and item.stat.exists
    with_items: "{{ nic_result.results }}"

```



注記

この Playbook を使用して、アップグレードプロセスの後半でオーバークラウドの NIC の名前を変更します。

- アンダークラウドで **playbook-nics.yaml** Playbook を実行します。

```
$ ansible-playbook -c local -i localhost, playbook-nics.yaml
```

この Playbook により、新しい NIC の接頭辞が **em** に設定されます。別の NIC 接頭辞を設定するには、Playbook の実行時に **prefix** 変数を設定します。

```
$ ansible-playbook -c local -i localhost, -e prefix="mynic" ~/playbook-nics.yaml
```

NIC の変更は、Leapp によるアップグレードプロセスが完了し、ノードがリブートされた後のみ適用されます。

関連情報

- RHEL 7でカーネルのNIC名を使用している場合にRHEL 8へのインプレースアップグレードを実行する方法
- ネットワークデバイス命名における一貫性
- 予想可能なネットワークインターフェイスデバイスの命名について

4.4. 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。

4.5. 次世代電源管理ドライバーへの移行

Red Hat OpenStack Platform では **ハードウェアタイプ** と呼ばれる次世代ドライバーが使用され、従来のドライバーがこれに置き換えられています。

従来のドライバーとそれと等価な次世代ハードウェアタイプの対比を、以下の表に示します。

従来のドライバー	新しいハードウェアタイプ
pxe_ipmitool	ipmi
pxe_drac	idrac
pxe_ilo	ilo
pxe_irmc	irmc
VBMC (pxe_ipmitool)	ipmi

従来のドライバー	新しいハードウェアタイプ
fake_pxe	fake-hardware

OpenStack Platform 15 では、これらの従来ドライバーは削除され、使用できなくなっています。OpenStack Platform 15 に **アップグレードする前に** ハードウェアタイプに変更する必要があります。

手順

1. 有効なハードウェアタイプの最新のリストを確認します。

```
$ source ~/stackrc
$ openstack baremetal driver list --type dynamic
```

2. 有効ではないハードウェアタイプのドライバーを使用する場合には、**undercloud.conf** ファイルの **enabled_hardware_types** パラメーターを使用してそのドライバーを有効にします。

```
enabled_hardware_types = ipmi,redfish,idrac
```

3. ファイルを保存し、アンダークラウドをリフレッシュします。

```
$ openstack undercloud install
```

4. 以下のコマンドを実行します。**OLDDRIVER** および **NEWDRIVER** 変数を、実際の電源管理タイプに置き換えてください。

```
$ source ~/stackrc
$ OLDDRIVER="pxe_ipmitool"
$ NEWDRIVER="ipmi"
$ for NODE in $(openstack baremetal node list --driver $OLDDRIVER -c UUID -f value) ; do
openstack baremetal node set $NODE --driver $NEWDRIVER; done
```


第5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

director をアップグレードする前に、アンダークラウドのオペレーティングシステムを Red Hat Enterprise Linux 7 から Red Hat Enterprise Linux 8 にアップグレードする必要があります。このオペレーティングシステムのアップグレードの一環として、Red Hat OpenStack Platform 13 のパッケージを削除し、続いて Leapp ユーティリティーを実行してシステムパッケージをアップグレードする必要があります。このパッケージの削除およびオペレーティングシステムのアップグレードは、アンダークラウドのデータベースには影響を及ぼしません。オペレーティングシステムのアップグレードが完了したら、Red Hat OpenStack Platform 16.1 バージョンの director パッケージを再インストールします。

5.1. RED HAT OPENSTACK PLATFORM DIRECTOR パッケージの削除

Leapp ユーティリティーを実行する前に、Red Hat Enterprise Linux 7 に関連付けられた Red Hat OpenStack Platform 13 パッケージを削除します。これらのパッケージ名には、リリースに関する接尾辞 **el7ost** が使用されます。一部の **el7ost** は、**subscription-manager** および Leapp ユーティリティーの依存関係としてシステム上に残ります。

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. アンダークラウド上の主要 OpenStack サービスを無効にします。

```
$ sudo systemctl stop 'openstack-*' httpd haproxy mariadb 'rabbitmq*' docker xinetd
```

3. OpenvSwitch およびアップグレードに必要な特定の Python 2 パッケージは除き、アンダークラウドから主要 OpenStack サービスを削除します。

```
$ sudo yum -y remove '*el7ost*' 'galera*' 'haproxy*' \
  httpd 'mysql*' 'pacemaker*' xinetd python-jsonpointer \
  qemu-kvm-common-rhev qemu-img-rhev 'rabbit*' \
  'redis*' \
  -- \
  -*openvswitch* -python-docker -python-PyMySQL \
  -python-pysocks -python2-asn1crypto -python2-babel \
  -python2-cffi -python2-cryptography -python2-dateutil \
  -python2-idna -python2-ipaddress -python2-jinja2 \
  -python2-jsonpatch -python2-markupsafe -python2-pyOpenSSL \
  -python2-requests -python2-six -python2-urllib3 \
  -python-httpplib2 -python-passlib -python2-netaddr -ceph-ansible
```

4. **/etc/httpd** および **/var/lib/docker** ディレクトリーからコンテンツを削除します。

```
$ sudo rm -rf /etc/httpd /var/lib/docker
```

5.2. アンダークラウドでの LEAPP アップグレードの実施

Leapp ユーティリティーをインストールして実行し、オペレーティングシステムを Red Hat Enterprise Linux (RHEL) 8 にアップグレードします。

前提条件

- 「Red Hat OpenStack Platform での Leapp アップグレードの使用」 セクションを十分に理解してから Leapp をインストールして実行する。
- 「予測可能なアンダークラウドノード NIC 名の使用」 セクションを完了してから Leapp によるアップグレードを実施する。Leapp によるアップグレードプロセスを実行する前にネットワークインターフェイス名を変更しない場合、RHEL 8.2 へのアップグレードの完了後にインターフェイス名が変わる可能性があります。

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. Leapp ユーティリティーと jq をインストールします。

```
$ sudo yum install leapp
$ sudo yum install jq
```

3. ナレッジベースのアーティクル [RHEL 7 から RHEL 8 へのインプレースアップグレード時に Leapp ユーティリティーで必要なデータ](#) に添付されている追加の必須データファイル (RPM パッケージの変更および RPM リポジトリマッピング) をダウンロードし、それらのファイルを **/etc/leapp/files/** ディレクトリーに置きます。
4. Red Hat サブスクリプションを更新します。

- アンダークラウドの登録に Red Hat カスタマーポータルを使用している場合、現在のサブスクリプションをリフレッシュし、Red Hat Enterprise Linux 8.2 コンテンツへのアクセス権限を取得します。

```
$ sudo subscription-manager refresh
```

- アンダークラウドの登録に Red Hat Satellite Server を使用している場合は、アンダークラウドを Red Hat OpenStack Platform (RHOSP) 16.1 のアクティベーションキーに関連付けられたコンテンツビューに再登録します。

```
$ sudo subscription-manager register --force --org ORG --activationkey
ACTIVATION_KEY
```



注記

Red Hat OpenStack Platform 16.1 用に作成するコンテンツビューには、Red Hat Enterprise Linux 8.2 のコンテンツが含まれている必要があります。

5. Red Hat OpenStack Platform 16.1 では、新しいバージョンの Open vSwitch が使用されます。**to_remove** および **to_install** トランザクションファイルにより、Open vSwitch のバージョンを置き換えます。

```
$ echo 'openvswitch2.11' | sudo tee -a /etc/leapp/transaction/to_remove
$ echo 'openvswitch2.13' | sudo tee -a /etc/leapp/transaction/to_install
```

6. **to_keep** トランザクションファイルを使用して、アップグレードプロセス中に Red Hat Ceph Storage 3 バージョンの **ceph-ansible** を維持します。

```
$ echo 'ceph-ansible' | sudo tee -a /etc/leapp/transaction/to_keep
```

7. RHEL 8 ではサポートされなくなったカーネルモジュールを調整します。

```
$ if [ -f /usr/share/leapp-
repository/repositories/system_upgrade/el7toel8/actors/kernel/checkkerneldrivers/files/removed
_drivers.txt ]; then
    for module in pata_acpi floppy; do
        sudo sed -i "/^${module}$/d" /usr/share/leapp-
repository/repositories/system_upgrade/el7toel8/actors/kernel/checkkerneldrivers/files/removed
_drivers.txt
    done
else
    for module in pata_acpi floppy; do
        jq ". | del(.data[] | select(.driver_name == \"${module}\"))"
/etc/leapp/files/device_driver_deprecation_data.json | sudo tee
/etc/leapp/files/device_driver_deprecation_data.json_modified
        mv /etc/leapp/files/device_driver_deprecation_data.json_modified
/etc/leapp/files/device_driver_deprecation_data.json
    done
fi
```

8. **leapp answer** コマンドを実行し、**pam_pkcs11** モジュールを削除する Leapp の回答を指定します。

```
$ sudo leapp answer --add --section remove_pam_pkcs11_module_check.confirm=True
```

9. (オプション) TLS-Everywhere アーキテクチャーで環境がデプロイされ、システムでの認証を設定するのに非推奨の **authconfig** ユーティリティーが使用される場合は、**authselect** ユーティリティーを使用して RHEL 8 システムを設定します。

```
$ sudo leapp answer --add --section authselect_check.confirm=True
```

Leapp によるアップグレードプロセス時の認証設定の詳細は、**Upgrading from RHEL 7 to RHEL 8** の [Known issues](#) を参照してください。

10. **LEAPP_DEVEL_TARGET_RELEASE** および **LEAPP_UNSUPPORTED** 環境変数を設定して、アップグレードする RHEL 8 マイナーバージョンを指定します。RHOSP 16.1 では、RHEL 8 マイナーバージョンを **8.2** に設定する必要があります。

```
$ export LEAPP_UNSUPPORTED=1
$ export LEAPP_DEVEL_TARGET_RELEASE=8.2
```

LEAPP_DEVEL 接頭辞を付けて環境変数を使用する場合は、**LEAPP_UNSUPPORTED** 環境変数を使用する必要があります。

11. Leapp プロセスから、永続的なネットワーク名のアクターを削除します。



注記

Leapp によるアップグレードプロセスを実行する前にネットワークインターフェイス名を変更しない場合、RHEL 8.2 へのアップグレードの完了後にインターフェイス名が変わる可能性があります。ネットワークインターフェイス名の名前変更の詳細については、「[予測可能なアンダークラウドノード NIC 名の使用](#)」を参照してください。

```
$ sudo rm -f /usr/share/leapp-  
repository/repositories/system_upgrade/el7toel8/actors/persistentnetnamesdisable/actor.py
```

12. Leapp によるアップグレードプロセスを開始します。

```
$ sudo -E leapp upgrade --debug --enablerepo rhel-8-for-x86_64-baseos-tus-rpms --  
enablerepo rhel-8-for-x86_64-appstream-tus-rpms --enablerepo fast-datapath-for-rhel-8-  
x86_64-rpms --enablerepo ansible-2.9-for-rhel-8-x86_64-rpms
```

--enablerepo オプションを使用して、Leapp によるアップグレードプロセス中に有効にするリポジトリを設定します。特に新しいバージョンの Open vSwitch を使用する場合は、Red Hat OpenStack Platform 16.1 への移行を円滑に行うために、これらのリポジトリを追加する必要があります。

13. **leapp upgrade** コマンドが正常に完了するのを待ちます。

14. ルートディレクトリーに空の **.autorelabel** ファイルを作成します。

```
$ sudo touch /.autorelabel
```

リブート後、SELinux はこのファイルを検出し、自動的にファイルシステムのラベルを変更します。

15. アンダークラウドをリブートします。

```
$ sudo reboot
```

16. DNF の設定で定義されているトランザクションの除外対象から、Leapp のパッケージを削除します。

```
$ sudo dnf config-manager --save --setopt exclude=
```

関連情報

- [RHEL 7 から RHEL 8 へのインプレースアップグレード時に Leapp ユーティリティーに必要なデータ](#)
- [RHEL 7 でカーネルの NIC 名を使用している場合に RHEL 8 へのインプレースアップグレードを実行する方法](#)
- [RHEL 7 から RHEL 8 へのアップグレードの実行](#)

第6章 DIRECTOR のアップグレード

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が完了したら、director をアップグレードします。以前の Red Hat OpenStack Platform 13 のアンダークラウドのデータベースは、オペレーティングシステムのアップグレード後にホスト上に残ります。**openstack undercloud upgrade** コマンドを実行する前に、新しい Red Hat OpenStack Platform 16.1 パッケージをインストールし、Red Hat OpenStack Platform 16.1 コンテナイメージの新しいソースを設定します。

6.1. 環境の RED HAT ENTERPRISE LINUX リリースへのロック

Red Hat OpenStack Platform 16.1 は Red Hat Enterprise Linux 8.2 でサポートされています。更新を実施する前に、オペレーティングシステムをより新しいマイナーリリースにアップグレードしないように、アンダークラウドのリポジトリを Red Hat Enterprise Linux 8.2 リリースにロックする必要があります。

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. **subscription-manager release** コマンドを使用して、アンダークラウドを特定のバージョンにロックします。

```
$ sudo subscription-manager release --set=8.2
```

6.2. アンダークラウド用リポジトリの有効化

アンダークラウドに必要なリポジトリを有効にし、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をすべて無効にしてから、必要な Red Hat Enterprise Linux リポジトリを有効にします。

```
[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=ansible-2.9-for-rhel-8-x86_64-rpms --enable=openstack-16.1-for-rhel-8-x86_64-rpms --enable=fast-datapath-for-rhel-8-x86_64-rpms --enable=advanced-virt-for-rhel-8-x86_64-eus-rpms
```

これらのリポジトリには、director のインストールに必要なパッケージが含まれます。

3. **container-tools** リポジトリモジュールをバージョン **2.0** に設定します。

```
[stack@director ~]$ sudo dnf module reset container-tools  
[stack@director ~]$ sudo dnf module enable -y container-tools:2.0
```

4. オペレーティングシステムを同期し、システムパッケージがオペレーティングシステムのバージョンと一致するようにします。

```
[stack@director ~]$ sudo dnf distro-sync -y
[stack@director ~]$ sudo reboot
```

6.3. DIRECTOR パッケージのインストール

Red Hat OpenStack Platform director に関連するパッケージをインストールします。

手順

1. 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
```

6.4. コンテナイメージの準備

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。この環境ファイルを生成してカスタマイズし、コンテナイメージの準備に使用します。



注記

アンダークラウド用に特定のコンテナ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。詳しい情報は、[Pinning container images for the undercloud](#) 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. デフォルトのコンテナイメージ準備ファイルを生成します。

```
$ sudo 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。

3. 要件に合わせて **containers-prepare-parameter.yaml** を変更します。

6.5. コンテナイメージ準備のパラメーター

コンテナ準備用のデフォルトファイル (**containers-prepare-parameter.yaml**) に
は、**ContainerImagePrepare** heat パラメーターが含まれます。このパラメーターで、イメージのセッ
トを準備するためのさまざまな設定を定義します。

```
parameter_defaults:
  ContainerImagePrepare:
    - (strategy one)
    - (strategy two)
    - (strategy three)
    ...
```

それぞれの設定では、サブパラメーターのセットにより使用するイメージやイメージの使用方法を定義
することができます。以下の表には、**ContainerImagePrepare** ストラテジーの各設定で使用するこ
のできるサブパラメーターの情報をまとめています。

パラメーター	説明
excludes	設定からイメージ名を除外する正規表現のリスト
includes	設定に含める正規表現のリスト。少なくとも1つのイ メージ名が既存のイメージと一致している必要があ ります。 includes パラメーターを指定する と、 excludes の設定はすべて無視されます。
modify_append_tag	対象となるイメージのタグに追加する文字列。たと えば、16.1.3-5.161 のタグが付けられたイメージをプ ルし、 modify_append_tag を -hotfix に設定する と、director は最終イメージを 16.1.3-5.161-hotfix と タグ付けします。
modify_only_with_labels	変更するイメージを絞り込むイメージラベルのディ クショナリー。イメージが定義したラベルと一致す る場合には、director はそのイメージを変更プロセス に含めます。
modify_role	イメージのアップロード中 (ただし目的のレジス トリーにプッシュする前) に実行する Ansible ロール名 の文字列
modify_vars	modify_role に渡す変数のディクショナリー

パラメーター	説明
push_destination	<p>アップロードプロセス中にイメージをプッシュするレジストリーの名前空間を定義します。</p> <ul style="list-style-type: none"> ● true に設定すると、push_destination はホスト名を使用してアンダークラウドレジストリーの名前空間に設定されます。これが推奨される方法です。 ● false に設定すると、ローカルレジストリーへのプッシュは実行されず、ノードはソースから直接イメージをプルします。 ● カスタムの値に設定すると、director はイメージを外部のローカルレジストリーにプッシュします。 <p>実稼働環境でこのパラメーターを false に設定した場合、イメージを Red Hat Container Catalog から直接プルする際に、すべてのオーバークラウドノードが同時に外部接続を通じて Red Hat Container Catalog からイメージをプルするため、帯域幅の問題が発生する可能性があります。コンテナイメージをホストする Red Hat Satellite Server から直接プルする場合にのみ、false を使用します。</p> <p>push_destination パラメーターが false に設定されているか、定義されておらずリモートレジストリーで認証が必要な場合は、ContainerImageRegistryLogin パラメーターを true に設定し、ContainerImageRegistryCredentials パラメーターで認証情報を追加します。</p>
pull_source	元のコンテナイメージをプルするソースレジストリー
set	初期イメージの取得場所を定義する、 キー: 値 定義のディクショナリー
tag_from_label	<p>指定したコンテナイメージのメタデータラベルの値を使用して、すべてのイメージのタグを作成し、そのタグが付けられたイメージをプルします。たとえば、tag_from_label: {version}-{release} を設定すると、director は version および release ラベルを使用して新しいタグを作成します。あるコンテナについて、version を 16.1.3 に設定し、release を 5.161 に設定した場合、タグは 16.1.3-5.161 となります。set ディクショナリーで tag を定義していない場合に限り、director はこのパラメーターを使用します。</p>



重要

イメージをアンダークラウドにプッシュする場合は、**push_destination: UNDERCLOUD_IP:PORT** の代わりに **push_destination: true** を使用します。**push_destination: true** 手法を使用することで、IPv4 アドレスおよび IPv6 アドレスの両方で一貫性が確保されます。

set パラメーターには、複数の **キー: 値** 定義を設定することができます。

キー	説明
ceph_image	Ceph Storage コンテナイメージの名前
ceph_namespace	Ceph Storage コンテナイメージの名前空間
ceph_tag	Ceph Storage コンテナイメージのタグ
ceph_alertmanager_image ceph_alertmanager_namespace ceph_alertmanager_tag	Ceph Storage Alert Manager コンテナイメージの名前、namespace、およびタグ。
ceph_grafana_image ceph_grafana_namespace ceph_grafana_tag	Ceph Storage Grafana コンテナイメージの名前、namespace、およびタグ。
ceph_node_exporter_image ceph_node_exporter_namespace ceph_node_exporter_tag	Ceph Storage Node Exporter コンテナイメージの名前、namespace、およびタグ。
ceph_prometheus_image ceph_prometheus_namespace ceph_prometheus_tag	Ceph Storage Prometheus コンテナイメージの名前、namespace、およびタグ。
name_prefix	各 OpenStack サービスイメージの接頭辞
name_suffix	各 OpenStack サービスイメージの接尾辞
namespace	各 OpenStack サービスイメージの名前空間
neutron_driver	使用する OpenStack Networking (neutron) コンテナを定義するのに使用するドライバー。標準の neutron-server コンテナに設定するには、 null 値を使用します。OVN ベースのコンテナを使用するには、 ovn に設定します。

キー	説明
tag	ソースからの全イメージに特定のタグを設定します。定義されていない場合は、director は Red Hat OpenStack Platform のバージョン番号をデフォルト値として使用します。このパラメーターは、 tag_from_label の値よりも優先されます。



注記

コンテナイメージでは、Red Hat OpenStack Platform のバージョンに基づいたマルチストリームタグが使用されます。したがって、今後 **latest** タグは使用されません。

6.6. コンテナイメージタグ付けのガイドライン

Red Hat コンテナレジストリーでは、すべての Red Hat OpenStack Platform コンテナ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 16.1.3 で、コンテナイメージのリリースが **5.161** の場合、コンテナイメージのタグは 16.1.3-5.161 となります。

Red Hat コンテナレジストリーでは、コンテナ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、16.1 と 16.1.3 の両方が、16.1.3 コンテナストリームの最新 **release** とリンクします。16.1 の新規マイナーリリースが公開されると、16.1 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、16.1.3 タグは、引き続き 16.1.3 ストリーム内の最新 **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、16.1 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: 16.1
      ...
```

- OpenStack Platform コンテナ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、16.1.2 に変更するには、**tag** を 16.1.2 に設定します。

```
parameter_defaults:
  ContainerImagePrepare:
    - set:
      ...
      tag: 16.1.2
      ...
```

- **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: 16.1
      ...
  tag_from_label: '{version}-{release}'
```

- **tag_from_label** パラメーターは、Red Hat コンテナレジストリーから検査する最新コンテナ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": "16.1.3",
  ...
}
```

- **tag_from_label** のデフォルト値は **{version}-{release}** です。これは、各コンテナ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、コンテナイメージの **version** に 16.1.3 が、**release** に 5.161 が、それぞれ設定されている場合、コンテナイメージのタグは 16.1.3-5.161 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、各コンテナイメージのメタデータの検査を行います。

6.7. プライベートレジストリーからのコンテナイメージの取得

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

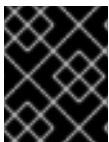
な場合には、**containers-prepare-parameter.yaml** 環境ファイルの **ContainerImageRegistryCredentials** パラメーターを使用します。**ContainerImageRegistryCredentials** パラメーターは、プライベートレジストリーの URL に基づくキーのセットを使用します。それぞれのプライベートレジストリーの URL は、独自のキーと値のペアを使用して、ユーザー名 (キー) およびパスワード (値) を定義します。これにより、複数のプライベートレジストリーに対して認証情報を指定することができます。

```
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** からコンテナイメージをプルします。

詳細は、[Red Hat コンテナ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
```

6.8. アップグレード用の移行コンテナの取得

アップグレードには、以前のバージョンの Red Hat OpenStack Platform および Red Hat Ceph Storage のコンテナが必要です。これらのコンテナは、Red Hat OpenStack Platform 16.1 への移行に役立ちます。

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. **containers-prepare-parameter.yaml** ファイルを編集します。
3. 遷移コンテナパラメーターを **ContainerImagePrepare** パラメーターに **設定**します。デプロイメントの種別に応じて、以下のいずれかの方法でパラメーターを設定します。
 - `director` を使用してデプロイされた Red Hat Ceph Storage クラスターがデプロイメントで使用される場合は、以下のパラメーターを追加します。

```
parameter_defaults:
```

```

ContainerImagePrepare:
- push_destination: true
set:
...
name_prefix_stein: openstack-
name_suffix_stein: "
namespace_stein: registry.redhat.io/rhosp15-rhel8
tag_stein: 15.0
ceph3_namespace: registry.redhat.io/rhceph
ceph3_tag: latest
ceph3_image: rhceph-3-rhel7
...

```

- ***_stein** パラメーターで定義する Red Hat OpenStack Platform 15 用のコンテナイメージが、アップグレードプロセスでデータベースの移行に使用されます。
- **ceph3_*** パラメーターは、オーバークラウドが使用する現在の Red Hat Ceph Storage コンテナイメージを定義します。Red Hat Ceph Storage 3 から 4 への移行には、オーバークラウドに **ceph3_*** および **ceph_*** 両方のパラメーターが必要です。
- コンテナイメージのストレージに Red Hat Satellite Server を使用している場合は、名前空間をご自分の Red Hat Satellite Server 上のイメージの場所に設定します。
- 外部の Ceph Storage クラスターがデプロイメントで使用される場合は、以下のパラメーターを追加します。

```

parameter_defaults:
ContainerImagePrepare:
- push_destination: true
set:
...
name_prefix_stein: openstack-
name_suffix_stein: "
namespace_stein: registry.redhat.io/rhosp15-rhel8
tag_stein: 15.0
ceph_namespace: registry.redhat.io/rhceph
ceph_tag: latest
ceph_image: rhceph-4-rhel8
...

```

- ***_stein** パラメーターで定義する Red Hat OpenStack Platform 15 用のコンテナイメージが、アップグレードプロセスでデータベースの移行に使用されます。
- **ceph_*** パラメーターは、オーバークラウドが使用する現在の Red Hat Ceph Storage 4 コンテナイメージを定義します。
- コンテナイメージのストレージに Red Hat Satellite Server を使用している場合は、名前空間をご自分の Red Hat Satellite Server 上のイメージの場所に設定します。

4. **neutron_driver** パラメーターを **openvswitch** に変更します。

```

parameter_defaults:
ContainerImagePrepare:
- push_destination: true
set:

```

```
...
neutron_driver: openvswitch
...
```

アップグレードプロセスの全期間中、Open vSwitch との互換性が維持されます。Red Hat OpenStack Platform 16.1 へのアップグレードが完了したら、オーバークラウドを Open vSwitch から Open Virtual Network (OVN) に移行します。

5. **containers-prepare-parameter.yaml** ファイルを保存します。

6.9. UNDERCLOUD.CONF ファイルの更新

引き続き Red Hat OpenStack Platform 13 環境からの元の **undercloud.conf** ファイルを使用できますが、Red Hat OpenStack Platform 16.1 との互換性を維持するために、ファイルを変更する必要があります。

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. **undercloud.conf** ファイルを編集します。
3. ファイル内の **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
```

director が正しい場所からアンダークラウドのコンテナイメージをプルできるように、このパラメーターで **containers-prepare-parameter.yaml** 環境ファイルの場所を定義します。

4. **generate_service_certificate** パラメーターを確認します。このパラメーターのデフォルト設定は、**false** から **true** に変更されます。これにより、アップグレード中にアンダークラウドで SSL/TLS が有効になります。
5. 予測可能な NIC 命名規則に移行している場合は、**local_interface** パラメーターを確認してください。
6. Red Hat OpenStack Platform 13 で **masquerade_network** パラメーターを設定した場合には、このパラメーターを削除して、サブネットごとに **masquerade = true** を設定します。
7. 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。
8. ファイルを保存します。

6.10. DIRECTOR の設定パラメーター

以下のリストで、**undercloud.conf** ファイルを設定するパラメーターについて説明します。エラーを避けるために、パラメーターは決して該当するセクションから削除しないでください。



重要

少なくとも、コンテナイメージの設定が含まれる環境ファイルに **container_images_file** パラメーターを設定する必要があります。このパラメーターに適切なファイルを正しく設定しないと、director は **ContainerImagePrepare** パラメーターからコンテナイメージのルールセットを取得することや、**ContainerImageRegistryCredentials** パラメーターからコンテナレジストリーの認証情報を取得することができなくなります。

デフォルト

undercloud.conf ファイルの **[DEFAULT]** セクションで定義されているパラメーターを以下に示します。

additional_architectures

オーバークラウドがサポートする追加の (カーネル) アーキテクチャーのリスト。現在、オーバークラウドは、デフォルトの **x86_64** アーキテクチャーに加えて **ppc64le** アーキテクチャーをサポートしています。

certificate_generation_ca

要求した証明書に署名する CA の **certmonger** のニックネーム。 **generate_service_certificate** パラメーターを設定した場合に限り、このオプションを使用します。 **local** CA を選択する場合は、**certmonger** はローカルの CA 証明書を **/etc/pki/ca-trust/source/anchors/cm-local-ca.pem** に抽出し、証明書をトラストチェーンに追加します。

clean_nodes

デプロイメントを再実行する前とイントロスペクションの後にハードドライブを消去するかどうかを定義します。

cleanup

一時的なファイルを削除します。デプロイメント中に使用される一時ファイルを保持するには、これを **False** に設定します。一時ファイルは、エラーが発生した場合にデプロイメントのデバッグに役立ちます。

container_cli

コンテナ管理用の CLI ツール。このパラメーターは、**podman** に設定したままにしてください。Red Hat Enterprise Linux 8.2 がサポートするのは **podman** だけです。

container_healthcheck_disabled

コンテナ化されたサービスのヘルスチェックを無効にします。Red Hat は、ヘルスチェックを有効にし、このオプションを **false** に設定したままにすることを推奨します。

container_images_file

コンテナイメージ情報が含まれる **heat** 環境ファイル。このファイルには、以下のエントリーを含めることができます。

- 必要なすべてのコンテナイメージのパラメーター
- 必要なイメージの準備を実施する **ContainerImagePrepare** パラメーター。このパラメーターが含まれるファイルの名前は、通常 **containers-prepare-parameter.yaml** です。

container_insecure_registries

podman が使用するセキュアではないレジストリーのリスト。プライベートコンテナレジストリー等の別のソースからイメージをプルする場合には、このパラメーターを使用します。多くの場合、**podman** は Red Hat Container Catalog または Satellite サーバー (アンダークラウドが Satellite に登録されている場合) のいずれかからコンテナイメージをプルするための証明書を持ちます。

container_registry_mirror

設定により **podman** が使用するオプションの **registry-mirror**

custom_env_files

アンダークラウドのインストールに追加する新たな環境ファイル

deployment_user

アンダークラウドをインストールするユーザー。現在のデフォルトユーザー **stack** を使用する場合には、このパラメーターを未設定のままにします。

discovery_default_driver

自動的に登録されたノードのデフォルトドライバーを設定します。**enable_node_discovery** パラメーターを有効にし、**enabled_hardware_types** のリストにドライバーを含める必要があります。

enable_ironic、enable_ironic_inspector、enable_mistral、enable_nova、enable_tempest、enable_validations、enable_zaqar

director で有効にするコアサービスを定義します。これらのパラメーターは **true** に設定されたままにします。

enable_node_discovery

introspection ramdisk を PXE ブートする不明なノードを自動的に登録します。新規ノードは、**fake** ドライバーをデフォルトとして使用しますが、**discovery_default_driver** を設定して上書きすることもできます。また、イントロスペクションルールを使用して、新しく登録したノードにドライバーの情報を指定することもできます。

enable_novajoin

アンダークラウドに **novajoin** メタデータサービスをインストールするかどうかを定義します。

enable_routed_networks

ルーティングされたコントロールプレーンネットワークのサポートを有効にするかどうかを定義します。

enable_swift_encryption

保存データの Swift 暗号化を有効にするかどうかを定義します。

enable_telemetry

アンダークラウドに OpenStack Telemetry サービス (gnocchi、aodh、panko) をインストールするかどうかを定義します。Telemetry サービスを自動的にインストール/設定するには、**enable_telemetry** パラメーターを **true** に設定します。デフォルト値は **false** で、アンダークラウド上の telemetry が無効になります。このパラメーターは、メトリックデータを消費する Red Hat CloudForms などの他の製品を使用する場合に必要です。



警告

RBAC はすべてのコンポーネントでサポートされているわけではありません。Alarming サービス (aodh) と Gnocchi は、安全な RBAC ルールを考慮していません。

enabled_hardware_types

アンダークラウドで有効にするハードウェアタイプのリスト

generate_service_certificate

アンダークラウドのインストール時に SSL/TLS 証明書を生成するかどうかを定義します。これは **undercloud_service_certificate** パラメーターに使用します。アンダークラウドのインストール

で、作成された証明書 `/etc/pki/tls/certs/undercloud-[undercloud_public_vip].pem` を保存します。`certificate_generation_ca` パラメーターで定義される CA はこの証明書に署名します。

heat_container_image

使用する heat コンテナイメージの URL。未設定のままにします。

heat_native

`heat-all` を使用してホストベースのアンダークラウド設定を実行します。`true` のままにします。

hieradata_override

director に Puppet hieradata を設定するための `hieradata` オーバーライドファイルへのパス。これにより、サービスに対して `undercloud.conf` パラメーター以外のカスタム設定を行うことができます。設定すると、アンダークラウドのインストールでこのファイルが `/etc/puppet/hieradata` ディレクトリにコピーされ、階層の最初のファイルに設定されます。この機能の使用についての詳細は、Director Installation and Usage の [Configuring hieradata on the undercloud](#) を参照してください。

inspection_extras

イントロスペクション時に追加のハードウェアコレクションを有効化するかどうかを定義します。このパラメーターを使用するには、イントロスペクションイメージに `python-hardware` または `python-hardware-detect` パッケージが必要です。

inspection_interface

ノードのイントロスペクションに director が使用するブリッジ。これは、director の設定により作成されるカスタムのブリッジです。`LOCAL_INTERFACE` でこのブリッジをアタッチします。これは、デフォルトの `br-ctlplane` のままにします。

inspection_runbench

ノードイントロスペクション時に一連のベンチマークを実行します。ベンチマークを有効にするには、このパラメーターを `true` に設定します。このオプションは、登録ノードのハードウェアを検査する際にベンチマーク分析を実行する場合に必要です。

ipa_otp

IPA サーバーにアンダークラウドノードを登録するためのワンタイムパスワードを定義します。これは、`enable_novajoin` が有効な場合に必要です。

ipv6_address_mode

アンダークラウドのプロビジョニングネットワーク用の IPv6 アドレス設定モード。このパラメーターに設定できる値のリストを以下に示します。

- `dhcpv6-stateless`: ルーター広告 (RA) を使用するアドレス設定と DHCPv6 を使用するオプションの情報
- `dhcpv6-stateful`: DHCPv6 を使用するアドレス設定とオプションの情報

ipxe_enabled

iPXE か標準の PXE のいずれを使用するか定義します。デフォルトは `true` で iPXE を有効化します。標準の PXE を使用するには、このパラメーターを `false` に設定します。

local_interface

director のプロビジョニング NIC 用に選択するインターフェイス。director は、DHCP および PXE ブートサービスにもこのデバイスを使用します。この値を選択したデバイスに変更します。接続されているデバイスを確認するには、`ip addr` コマンドを使用します。`ip addr` コマンドの出力結果の例を、以下に示します。

```
2: em0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 52:54:00:75:24:09 brd ff:ff:ff:ff:ff:ff
```

```

inet 192.168.122.178/24 brd 192.168.122.255 scope global dynamic em0
    valid_lft 3462sec preferred_lft 3462sec
inet6 fe80::5054:ff:fe75:2409/64 scope link
    valid_lft forever preferred_lft forever
3: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noop state DOWN
link/ether 42:0b:c2:a5:c1:26 brd ff:ff:ff:ff:ff:ff

```

この例では、外部 NIC は **em0** を使用し、プロビジョニング NIC は、現在設定されていない **em1** を使用します。この場合は、**local_interface** を **em1** に設定します。この設定スクリプトにより、このインターフェイスが **inspection_interface** パラメーターで定義したカスタムのブリッジにアタッチされます。

local_ip

director のプロビジョニング NIC 用に定義する IP アドレス。director は、DHCP および PXE ブートサービスにもこの IP アドレスを使用します。この IP アドレスが環境内の既存の IP アドレスまたはサブネットと競合するなどの理由により、プロビジョニングネットワークに別のサブネットを使用する場合以外は、この値をデフォルトの **192.168.24.1/24** のままにします。

IPv6 の場合、ステートフル接続とステートレス接続の両方をサポートするには、ローカル IP アドレス接頭辞接頭辞の長さを **/64** にする必要があります。

local_mtu

local_interface に使用する最大伝送単位 (MTU)。アンダークラウドでは 1500 以下にします。

local_subnet

PXE ブートと DHCP インターフェイスに使用するローカルサブネット。**local_ip** アドレスがこのサブネットに含まれている必要があります。デフォルトは **ctlplane-subnet** です。

net_config_override

ネットワーク設定のオーバーライドテンプレートへのパス。このパラメーターを設定すると、アンダークラウドは JSON または YAML 形式のテンプレートを使用して、**os-net-config** でネットワークを設定し、**undercloud.conf** で設定されたネットワークパラメーターを無視します。ボンディングを設定する場合、またはインターフェイスにオプションを追加する場合に、このパラメーターを使用します。アンダークラウドネットワークインターフェイスのカスタマイズの詳細については、[Configuring undercloud network interfaces](#) を参照してください。

networks_file

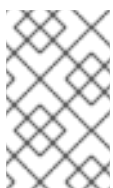
heat をオーバーライドするネットワークファイル

output_dir

状態、処理された heat テンプレート、および Ansible デプロイメントファイルを出力するディレクトリー

overcloud_domain_name

オーバークラウドをデプロイする際に使用する DNS ドメイン名



注記

オーバークラウドを設定する際に、**CloudDomain** にこのパラメーターと同じ値を設定する必要があります。オーバークラウドを設定する際に、環境ファイルでこのパラメーターを設定します。

roles_file

アンダークラウドのインストールで、デフォルトロールファイルを上書きするのに使用するロールファイル。director のインストールにデフォルトのロールファイルが使用されるように、このパラメーターは未設定のままにすることを強く推奨します。

scheduler_max_attempts

スケジューラーがインスタンスのデプロイを試行する最大回数。スケジューリング時に競合状態にならないように、この値を1度にデプロイする予定のベアメタルノードの数以上に指定する必要があります。

service_principal

この証明書を使用するサービスの Kerberos プリンシパル。FreeIPA など CA で Kerberos プリンシパルが必要な場合に限り、このパラメーターを使用します。

subnets

プロビジョニングおよびイントロスペクション用のルーティングネットワークのサブネットのリスト。デフォルト値に含まれるのは、**ctlplane-subnet** サブネットのみです。詳細は、[サブネット](#) を参照してください。

templates

上書きする heat テンプレートファイル

undercloud_admin_host

SSL/TLS 経由の director の管理 API エンドポイントに定義する IP アドレスまたはホスト名。director の設定により、IP アドレスは /32 ネットマスクを使用するルーティングされた IP アドレスとして director のソフトウェアブリッジに接続されます。

undercloud_admin_host が **local_ip** と同じ IP ネットワークにない場合は、Undercloud の管理 API がリッスンするインターフェイスに **ControlVirtualInterface** パラメーターを設定する必要があります。デフォルトでは、管理 API は **br-ctlplane** インターフェイスでリッスンします。**ControlVirtualInterface** パラメーターをカスタム環境ファイルに設定し、**custom_env_files** パラメーターを設定して、**undercloud.conf** ファイルにカスタム環境ファイルを含めます。

アンダークラウドネットワークインターフェイスのカスタマイズについての詳細は、[Configuring undercloud network interfaces](#) を参照してください。

undercloud_debug

アンダークラウドサービスのログレベルを **DEBUG** に設定します。**DEBUG** ログレベルを有効にするには、この値を **true** に設定します。

undercloud_enable_selinux

デプロイメント時に、SELinux を有効または無効にします。問題をデバッグする場合以外は、この値を **true** に設定したままにすることを強く推奨します。

undercloud_hostname

アンダークラウドの完全修飾ホスト名を定義します。本パラメーターを指定すると、アンダークラウドのインストールでホスト名すべてに設定されます。指定しないと、アンダークラウドは現在のホスト名を使用しますが、システムのホスト名すべてを適切に設定しておく必要があります。

undercloud_log_file

アンダークラウドのインストールログおよびアップグレードログを保管するログファイルへのパス。デフォルトでは、ログファイルはホームディレクトリー内の **install-undercloud.log** です。たとえば、**/home/stack/install-undercloud.log** のようになります。

undercloud_nameservers

アンダークラウドのホスト名解決に使用する DNS ネームサーバーのリスト

undercloud_ntp_servers

アンダークラウドの日付と時刻を同期できるようにする Network Time Protocol サーバーのリスト

undercloud_public_host

SSL/TLS 経由の director のパブリック API エンドポイントに定義する IP アドレスまたはホスト名。director の設定により、IP アドレスは /32 ネットマスクを使用するルーティングされた IP アドレスとして director のソフトウェアブリッジに接続されます。

`undercloud_public_host` が `local_ip` と同じ IP ネットワークにない場合は、`PublicVirtualInterface` パラメーターを、アンダークラウド上のパブリック API がリッスンする公開インターフェイスに設定する必要があります。デフォルトでは、パブリック API は `br-ctlplane` インターフェイスでリッスンします。カスタム環境ファイルに `PublicVirtualInterface` パラメーターを設定し、`custom_env_files` パラメーターを設定して、`undercloud.conf` ファイルにカスタム環境ファイルを含めます。

アンダークラウドネットワークインターフェイスのカスタマイズについての詳細は、[Configuring undercloud network interfaces](#) を参照してください。

`undercloud_service_certificate`

OpenStack SSL/TLS 通信の証明書の場所とファイル名。理想的には、信頼できる認証局から、この証明書を取得します。それ以外の場合は、独自の自己署名の証明書を生成します。

`undercloud_timezone`

アンダークラウド用ホストのタイムゾーン。タイムゾーンを指定しなければ、director は既存のタイムゾーン設定を使用します。

`undercloud_update_packages`

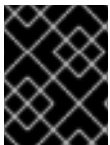
アンダークラウドのインストール時にパッケージを更新するかどうかを定義します。

サブネット

`undercloud.conf` ファイルには、各プロビジョニングサブネットの名前が付いたセクションがあります。たとえば、`ctlplane-subnet` という名前のサブネットを作成するには、`undercloud.conf` ファイルで以下のような設定を使用します。

```
[ctlplane-subnet]
cidr = 192.168.24.0/24
dhcp_start = 192.168.24.5
dhcp_end = 192.168.24.24
inspection_iprange = 192.168.24.100,192.168.24.120
gateway = 192.168.24.1
masquerade = true
```

プロビジョニングネットワークは、環境に応じて、必要なだけ指定することができます。



重要

director がサブネットを作成した後、director はサブネットの IP アドレスを変更することはできません。

`cidr`

オーバークラウドインスタンスの管理に director が使用するネットワーク。これは、アンダークラウドの `neutron` サービスが管理するプロビジョニングネットワークです。プロビジョニングネットワークに別のサブネットを使用しない限り、この値はデフォルト (`192.168.24.0/24`) のままにします。

`masquerade`

外部アクセスのために、`cidr` で定義したネットワークをマスカレードするかどうかを定義します。このパラメーターにより、director 経由で外部アクセスすることができるように、プロビジョニングネットワークにネットワークアドレス変換 (NAT) のメカニズムが提供されます。



注記

director 設定は、適切な **sysctl** カーネルパラメーターを使用して IP フォワーディングも自動的に有効化します。

dhcp_start、dhcp_end

オーバークラウドノードの DHCP 割り当て範囲 (開始アドレスと終了アドレス)。ノードを割り当てるのに十分な IP アドレスがこの範囲に含まれるようにします。

dhcp_exclude

DHCP 割り当て範囲で除外する IP アドレス

dns_nameservers

サブネットに固有の DNS ネームサーバー。サブネットにネームサーバーが定義されていない場合には、サブネットは **undercloud_nameservers** パラメーターで定義されるネームサーバーを使用します。

gateway

オーバークラウドインスタンスのゲートウェイ。External ネットワークにトラフィックを転送するアンダークラウドのホストです。director に別の IP アドレスを使用する場合または直接外部ゲートウェイを使用する場合以外は、この値はデフォルト (**192.168.24.1**) のままにします。

host_routes

このネットワーク上のオーバークラウドインスタンス用の neutron が管理するサブネットのホストルート。このパラメーターにより、アンダークラウド上の **local_subnet** のホストルートも設定されます。

inspection_iprange

検査プロセス中に使用するこのネットワーク上のノードの一時的な IP 範囲。この範囲は、**dhcp_start** と **dhcp_end** で定義された範囲と重複することはできませんが、同じ IP サブネット内になければなりません。

6.11. DIRECTOR のアップグレードの実施

director をアップグレードするには、以下の手順を実施します。

手順

1. 以下のコマンドを実行して、アンダークラウド上の director をアップグレードします。

```
$ openstack undercloud upgrade
```

このコマンドにより、director の設定スクリプトが起動します。director によりパッケージがアップグレードされ、**undercloud.conf** の設定に合わせてサービスが設定されます。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
```

- このスクリプトにより、アンダークラウド上の OpenStack Platform サービスのコンテナも、すべて自動的に起動されます。**systemd** リソースでそれぞれのサービスを管理します。**systemd** リソースを確認します。

```
$ sudo systemctl list-units "tripleo_**"
```

各 **systemd** サービスでコンテナを制御します。以下のコマンドを使用して、有効化されたコンテナを確認してください。

```
$ sudo podman ps
```

- スクリプトにより **stack** ユーザーが **docker** グループに追加され、**stack** ユーザーがコンテナ管理コマンドを使用できるようになります。以下のコマンドで **stack** ユーザーのアクセス権限をリフレッシュします。

```
$ exec su -l stack
```

このコマンドを実行すると、再度ログインが求められます。stack ユーザーのパスワードを入力します。

- stack** ユーザーを初期化してコマンドラインツールを使用するには、以下のコマンドを実行します。

```
$ source ~/stackrc
```

プロンプトには、OpenStack コマンドがアンダークラウドに対して認証および実行されることが表示されるようになります。

```
(undercloud) $
```

director のアップグレードが完了しました。

第7章 オーバークラウド準備の初期手順

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。

7.1. 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。コンピュータ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コンピュータ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

7.2. アップグレードテスト用のコンピュータノードの選択

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
- 個別のノードを別々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、全コンピュータノードをアップグレードする前に、環境内にある個々のコンピュータ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台のコンピュータ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。
- 必要な場合には、選択したテスト用のコンピュータノードから他のコンピュータノードにクリティカルなインスタンスを移行します。

7.3. オーバークラウドインベントリーファイルの作成

tripleo-ansible-inventory コマンドを使用して、環境内の全ノードの Ansible インベントリー ファイルを生成します。

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. source コマンドで **stackrc** ファイルを読み込みます。

```
$ source ~/stackrc
```

3. 全ノードの静的なインベントリーファイルを作成します。

```
$ tripleo-ansible-inventory --static-yaml-inventory ~/inventory.yaml --stack STACK_NAME
```

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

4. お使いの環境で Ansible の Playbook を実行するには、**ansible-playbook** コマンドを実行し、**-i** オプションを使用して動的インベントリーツールの完全パスを追加します。以下に例を示します。

```
(undercloud) $ ansible-playbook -i ~/inventory.yaml PLAYBOOK
```

7.4. アップグレード前の要件の検証

pre-upgrade 検証グループを実行して、アップグレード前の要件を確認します。

Red Hat OpenStack Platform (RHOSP) 検証フレームワークの詳細は、**director** のインストールと使用方法の [検証フレームワークの使用](#) を参照してください。

手順

1. source コマンドで **stackrc** ファイルを読み込みます。

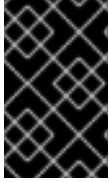
```
$ source ~/stackrc
```

2. **--group pre-upgrade** オプションを指定して **openstack tripleo validator run** コマンドを実行し、**/usr/libexec/platform-python** python ランタイム環境を追加します。

```
$ openstack tripleo validator run --group pre-upgrade --python-interpreter  
/usr/libexec/platform-python -i inventory.yaml
```

3. 検証レポートの結果を確認します。特定の検証からの詳細出力を表示するには、レポートからの特定検証の UUID を指定して **openstack tripleo validator show run --full** コマンドを実行します。

```
$ openstack tripleo validator show run --full <UUID>
```



重要

検証結果が **FAILED** であっても、RHOSP のデプロイや実行が妨げられることはありません。ただし、**FAILED** の検証結果は、実稼働環境で問題が発生する可能性があることを意味します。

7.5. 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。

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 heat-admin@CONTROLLER_IP "sudo pcs property set stonith-enabled=false"
```

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](#)

7.6. 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
```

- 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。
- 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'
```

- 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。

- 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'
```

7.7. アンダークラウドノードデータベースのバックアップ

backup-and-restore Ansible ロールを使用して、アンダークラウドノードで実行するデータベースのバックアップを作成し、そのバックアップを使用して、破損が発生した場合にデータベースの状態を復元することができます。アンダークラウドデータベースのバックアップの詳細については、[Red Hat OpenStack Platform 16.1 アンダークラウドとコントロールプレーンのバックアップおよび復元のアンダークラウドノードのデータベースバックアップ作成](#) を参照してください。

第8章 LEAPP アップグレードのためのオーバークラウド設定

ロングライフバージョンの Red Hat OpenStack Platform (RHOSP) をアップグレードするには、ベースオペレーティングシステムを Red Hat Enterprise Linux 7 から Red Hat Enterprise Linux 8 にアップグレードする必要があります。Red Hat Enterprise Linux 7 では、Leapp ユーティリティーを使用して Red Hat Enterprise Linux 8 へのアップグレードを実施します。Leapp およびその依存関係についての詳細は、[アップグレードに向けて RHEL 7 システムの準備](#) を参照してください。

オーバークラウドのアップグレードフレームワークでは、leapp によるアップグレードが自動的に実施されます。RHOSP が正常にアップグレードされるようにするには、アップグレード前のレポート生成を手動で実行して、発生する可能性のある問題を特定および解決することを推奨します。Compute、Controller、および Ceph Storage ロールのそれぞれについて、少なくとも1つのホストでアップグレード前のレポート生成を実行します。Leapp のアップグレード前のレポートの詳細は、[アップグレード前のレポートの確認](#) を参照してください。

8.1. アップグレード環境ファイルの作成

アップグレードプロセスでは、環境ファイルを使用して、アップグレードプロセスを有効にして特定のアップグレードパラメーターを設定します。

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. **templates** ディレクトリーに **upgrades-environment.yaml** という名前の環境ファイルを作成します。

```
$ touch templates/upgrades-environment.yaml
```

3. ファイルを編集し、以下の必須コンテンツを追加します。

```
parameter_defaults:
  UpgradeLeappDevelSkip: "LEAPP_UNSUPPORTED=1
  LEAPP_DEVEL_TARGET_RELEASE=8.2"
  LeappInitCommand: |
    for module in pata_acpi floppy; do sudo sed -i "/^${module}$/d" /usr/share/leapp-
    repository/repositories/system_upgrade/el7toel8/actors/kernel/checkkerneldrivers/files/removed
    _drivers.txt; done
    sudo rm -f /usr/share/leapp-
    repository/repositories/system_upgrade/el7toel8/actors/persistentnetnamesdisable/actor.py
    sudo yum -y remove mariadb-server* || true
  UpgradelnitCommand: "sudo dnf config-manager --save --setopt exclude=""
```

- **UpgradeLeappDevelSkip**: Leapp によるチェックを省略し、Leapp の実行時に環境変数を設定します。
 - **LeappInitCommand** は、各オーバークラウドノードで実行されるコマンドまたはスクリプトのスニペットを渡し、Leapp によるアップグレードに向けてノードを準備します。
 - **UpgradelnitCommand** は、各オーバークラウドノードで実行されるコマンドまたはスクリプトのスニペットを渡します。**python2** パッケージの自動削除が正常に行われるように、**dnf config-manager --save --setopt exclude=""** コマンドで、Leapp パッケージを DNF の除外対象から外します。
4. (オプション) お使いの環境で TLS-Everywhere を使用していて、**authconfig** から **authselect**

に移行したい場合は、バグ [BZ#1978228 - OSP13→16.2 Leapp upgrade failed with TLSEverywhere](#) を回避するために、`authselect_check.confirm` パラメーターを **True** に設定してください。

```
parameter_defaults:
  LeappInitCommand: |
    sudo leapp answer --section authselect_check.confirm=True --add
```

それ以外の場合は、値を **False** に設定します。



注記

`LeappInitCommand` パラメーターに複数のコマンドを渡すには、`|` 構文を使います。

```
parameter_defaults:
  LeappInitCommand: |
    <command_1>
    <command_2>
```

5. `upgrades-environment.yaml` ファイルを保存します。

8.2. アップグレードのパラメーター

パラメーター	説明
<code>UpgradelnitCommand</code>	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、リポジトリの切り替えなど。
<code>UpgradelnitCommonCommand</code>	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、 <code>major-upgrade-composable-steps.yaml</code> および <code>major-upgrade-converge.yaml</code> 環境ファイルで設定および設定解除されます。
<code>UpgradeLeappCommandOptions</code>	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
<code>UpgradeLeappDebug</code>	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 True です。
<code>UpgradeLeappDevelSkip</code>	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、 <code>LEAPP_DEVEL_SKIP_RHSM=1</code> と設定します。
<code>UpgradeLeappEnabled</code>	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。

パラメーター	説明
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

8.3. オーバークラウドノードでの LEAPP データのコピー

それぞれのオーバークラウドノードには、Leapp データファイルが必要です。アンダークラウドの `/etc/leapp/files` ディレクトリーにあるデータファイルを、各オーバークラウドノードの同じ場所にコピーします。

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. `source` コマンドで **stackrc** ファイルを読み込みます。

```
$ source ~/stackrc
```

3. 環境内の全ノードの静的なインベントリーファイルを作成します。

```
$ tripleo-ansible-inventory --static-yaml-inventory ~/inventory.yaml --stack STACK_NAME
```

デフォルトのスタック名 (**overcloud**) を使用していない場合は、`--stack STACK NAME` オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

4. leapp データをオーバークラウドノードにコピーするには、以下の **synchronize** Ansible コマンドを実行します。

```
$ ansible -i ~/inventory.yaml --become -m synchronize -a "src=/etc/leapp/files  
dest=/etc/leapp/" overcloud
```

8.4. オーバークラウドノードでの予測可能な NIC 名の使用

オーバークラウドノードで Leapp によるアップグレードを実施する前に、カーネルベースの NIC 名が使用されているかどうかを確認する必要があります。この NIC 名には、通常 **eth** の接頭辞が含まれます。NIC の割り当てに関して、通常これらの NIC 名は予測可能ではありません。

playbook-nics.yaml Playbook を実行して NIC の名前を変更し、インターフェイスの **NIC** 接頭辞を使用することができます。また、Playbook の実行時に **prefix** 変数を変更することで、別の NIC 接頭辞を

設定することもできます。ただし、NIC の変更は、Leapp によるアップグレードプロセスが完了し、ノードがリブートされた後にのみ適用されます。

前提条件

- アンダークラウドの準備プロセス中に作成された **playbook-nics.yaml** Playbook **playbook-nics.yaml** Playbook は、イーサネットデバイス、ブリッジ、および Linux ボンディングを使用するほとんどのオーバークラウドネットワーク設定のシナリオに対応します。お使いの環境でこれらのデバイス種別以外の追加設定が必要な場合は、手順を進める前に以下の推奨事項に従ってください。
 - 実際のオーバークラウドノードと類似したネットワーク設定の別システムで Playbook をテストする。
 - Playbook を変更し、他のデバイス種別の設定内の **eth** 接頭辞の名称変更に対応する。
 - 以下の手順を完了したら、オーバークラウドノードのネットワーク設定を確認する。

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. すべてのオーバークラウドノードで **playbook-nics.yaml** Playbook を実行します。

```
$ ansible-playbook -i ~/inventory.yaml playbook-nics.yaml
```

この Playbook により、新しい NIC の接頭辞が **em** に設定されます。別の NIC 接頭辞を設定するには、Playbook の実行時に **prefix** 変数を設定します。

```
$ ansible-playbook -i ~/inventory.yaml -e prefix="mynic" playbook-nics.yaml
```

NIC の変更は、Leapp によるアップグレードプロセスが完了し、ノードがリブートされた後にのみ適用されます。

関連情報

- [RHEL 7 でカーネルの NIC 名を使用している場合に RHEL 8 へのインプレースアップグレードを実行する方法](#)
- [ネットワークデバイス命名における一貫性](#)
- [予想可能なネットワークインターフェイスデバイスの命名について](#)

第9章 コンポーザブルサービスおよびパラメーターの更新

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、いくつかのコンポーザブルサービス設定を完了する必要があります。

9.1. 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。

- デフォルトの **roles_data** ファイルを使用する場合は、これらのサービスが自動的に含まれます。
- カスタムの **roles_data** ファイルを使用する場合は、該当するロールごとに新しいサービスを追加し非推奨のサービスを削除します。



重要

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。

コントローラーノード

コントローラーノードでは、以下のサービスが非推奨になっています。Controller 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::AodhApi OS::TripleO::Services::AodhEvaluator OS::TripleO::Services::AodhListener OS::TripleO::Services::AodhNotifier	<p>メトリック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 16.1 でのみ利用可能で、RHOSP の将来のバージョンでは削除される予定です。</p> <div style="display: flex; align-items: flex-start;">  <div> <p>注記</p> <p>自動スケーリングを使用する場合は、これらのサービスをコントローラーノードから削除しないでください。</p> </div> </div>

サービス	理由
OS::TripleO::Services::PankoApi	<p>メトリック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 16.1 でのみ利用可能で、RHOSP の将来のバージョンでは削除される予定です。</p> <div style="display: flex; align-items: flex-start;">  <div> <p>注記</p> <p>CloudForms を使用する場合は、これらのサービスをコントローラーノードから削除しないでください。</p> </div> </div>
OS::TripleO::Services::CeilometerApi OS::TripleO::Services::CeilometerCollector OS::TripleO::Services::CeilometerExpirer OS::TripleO::Services::MongoDb	<p>これらのサービスはサポートされなくなりました。これらのサービスをコントローラーノードから削除します。</p>
OS::TripleO::Services::Congress	<p>Congress はサポートされなくなったため。</p>
OS::TripleO::Services::GlanceRegistry	<p>OpenStack Platform Image サービス (glance) API v2 により、このサービスはサポートされなくなったため。</p>
OS::TripleO::Services::NeutronCorePluginPiumgrid OS::TripleO::Services::NeutronCorePluginMidonet	<p>これらの OpenStack Networking (neutron) 用プラグインは非推奨になったため。</p>
OS::TripleO::Services::NeutronLbaasv2Agent OS::TripleO::Services::NeutronLbaasv2Api	<p>Octavia を優先し、OpenStack Networking (neutron) Load Balancing as a Service は非推奨になったため。</p>
OS::TripleO::Services::NovaConsoleauth	<p>このサービスは削除されています。</p>
OS::TripleO::Services::NovaPlacement	<p>OS::TripleO::Services::PlacementApi を優先し、非推奨になったため。</p>
OS::TripleO::Services::OpenDaylightApi OS::TripleO::Services::OpenDaylightOvs	<p>OpenDaylight はサポートされなくなったため。</p>

サービス	理由
OS::TripleO::Services::RabbitMQ	このサービスは、2つの新規サービスに置き換えられています。 OS::TripleO::Services::OsloMessagingRpc OS::TripleO::Services::OsloMessagingNotify
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。

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::CephGrafana	Ceph Dashboard サービスを有効にするタスク
OS::TripleO::Services::CinderBackendDellEMC MCPowermax OS::TripleO::Services::CinderBackendDellEMC MCS OS::TripleO::Services::CinderBackendNVMe OF	Block Storage (cinder) の新規バックエンド
OS::TripleO::Services::ContainerImagePrepa re	コマンドを実行して、オーバークラウドのサービスに関連するコンテナイメージを自動的にプルして準備します。
OS::TripleO::Services::DesignateApi OS::TripleO::Services::DesignateCentral OS::TripleO::Services::DesignateProducer OS::TripleO::Services::DesignateWorker OS::TripleO::Services::DesignateMDNS OS::TripleO::Services::DesignateSink	DNS-as-a-Service 用サービス (designate)
OS::TripleO::Services::IronicInspector	オーバークラウドのベアメタルイントロスペクション用サービス
OS::TripleO::Services::IronicNeutronAgent	OpenStack Bare Metal (ironic) 用ネットワークエージェント

サービス	理由
OS::TripleO::Services::NeutronAgentsIBConfig	Mellanox InfiniBand OpenStack Networking (neutron) エージェント用サービス
OS::TripleO::Services::OpenStackClients	Red Hat OpenStack Platform コマンドラインツールをインストールするためのサービス
OS::TripleO::Services::OsloMessagingRpc OS::TripleO::Services::OsloMessagingNotify	OS::TripleO::Services::RabbitMQ の代替サービス
OS::TripleO::Services::PlacementApi	Placement API 用サービス

コンピューターノード

コンピューターノードでは、以下のサービスが非推奨になっています。Compute 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::OpenDaylightOvs	OpenDaylight はサポートされなくなったため。
OS::TripleO::Services::SkydiveAgent	Skydive はサポートされなくなったため。

コンピューターノードの新規サービスは以下のとおりです。Compute 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::NovaAZConfig	OpenStack Compute (nova) でホストアグリゲートおよびアベイラビリティゾーンを設定するためのサービス

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::Docker	Podman に置き換わったため。

サービス	理由
OS::TripleO::Services::Fluentd	OS::TripleO::Services::Rsyslog を優先し、非推奨になったため。
OS::TripleO::Services::Ntp	OS::TripleO::Services::Timesync を優先し、非推奨になったため。
OS::TripleO::Services::SensuClient	このサービスは非推奨になったため。
OS::TripleO::Services::Ptp	OS::TripleO::Services::Timesync を優先し、非推奨になったため。

全ノードでの新規サービスは以下のとおりです。すべての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::BootParams	カーネル引数、Tuned プロファイル、および CPU 分離を設定するためのサービス
OS::TripleO::Services::Collectd	Collectd を設定するためのサービス
OS::TripleO::Services::Multipathd	デフォルトで無効化されている Multipathd サービスを提供します。
OS::TripleO::Services::Podman	Podman をインストールし有効にするためのサービス
OS::TripleO::Services::Rear	Relax-and-Recover (ReaR) バックアップおよびリストアツールをインストールおよび有効にするためのサービス
OS::TripleO::Services::Rsyslog	集中ログコレクションを設定するためのサービス
OS::TripleO::Services::Timesync	時刻同期方法 (デフォルトでは Chronyd) を有効にするためのサービス

9.2. カスタム環境ファイルのコンポーザブルサービスの更新

resource_registry セクションのあるカスタム環境ファイルを使用している場合は、**resource_registry** セクションでコンポーザブルサービステンプレートのマッピングに変更がないか確認してください。Red Hat OpenStack Platform 16.1 では、コンポーザブルサービスのファイルは **/usr/share/openstack-tripleo-heat-templates/** 内の新しい場所にあります。

Red Hat OpenStack Platform 13	Red Hat OpenStack Platform 16.1
docker/services/	deployment

deployment ディレクトリーには、コンポーザブルサービスをグループ化するためのサブディレクトリーセットが含まれます。たとえば、**keystone** サブディレクトリーには、OpenStack Identity (keystone) のコンポーザブルサービステンプレートが含まれます。

カスタム環境ファイルのコンポーザブルサービスを再マッピングするには、現在のサービスマッピングのテンプレートの場所を確認し、マッピングを新しい場所に編集します。以下の手順では、例として **ceph-mgr.yaml** を使用しています。



重要

以下の手順は、コンポーザブルサービスを再マッピングする際のガイドとしてのみ提供されます。いずれかのマッピングが不明であれば、[Red Hat](#) に連絡して正しいマッピングについてアドバイスを求めてください。

手順

1. コンポーザブルサービスを使用するカスタム環境ファイルを検索します。通常、カスタム環境ファイルは **/home/stack/templates** ディレクトリーに保存されます。

```
$ cd ~/templates/
$ grep "OS::TripleO::Services" *
```

このシナリオでは、ファイルの1つに古くなったマッピングがあるとします。

```
OS::TripleO::Services::CephMgr: /usr/share/openstack-tripleo-heat-templates/docker/services/ceph-ansible/ceph-mgr.yaml
```

2. **/usr/share/openstack-tripleo-heat-templates/** の新しい **ceph-mgr.yaml** の場所を特定します。このファイルは、`deployment/ceph-ansible` ディレクトリーに置かれています。

```
$ find /usr/share/openstack-tripleo-heat-templates/ -name ceph-mgr.yaml
/usr/share/openstack-tripleo-heat-templates/deployment/ceph-ansible/ceph-mgr.yaml
```

3. カスタム環境ファイルでサービスを編集します。

```
resource_registry:
  OS::TripleO::Services::CephMgr: /usr/share/openstack-tripleo-heat-templates/deployment/ceph-ansible/ceph-mgr.yaml
```

ファイルを保存します。

9.3. アンダークラウドレジストリーへのアクセスの設定

アンダークラウドレジストリーへのアクセスを設定するには、アンダークラウドのコントロールプレーンのホスト名およびプロビジョニングネットワーク上のアンダークラウドの IP アドレスを **DockerInsecureRegistryAddress** パラメーターに追加します。このパラメーターを **containers-prepare-parameter.yaml** ファイルに置き、パラメーターがこれ以降のオーバークラウドデプロイメントに含まれるようにします。

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. アンダークラウドのコントロールプレーンのホスト名を取得します。

```
$ sudo hiera container_image_prepare_node_names
["undercloud.ctlplane.localdomain"]
```

3. **containers-prepare-parameter.yaml** ファイルを編集し、**DockerInsecureRegistryAddress** パラメーターを追加して、アンダークラウドのコントロールプレーンのホスト名およびプロビジョニングネットワーク上のアンダークラウドの IP アドレスが含まれる YAML のリストを指定します。

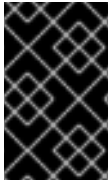
```
parameter_defaults:
  DockerInsecureRegistryAddress:
    - undercloud.ctlplane.localdomain:8787
    - 192.168.24.1:8787
  ...
```

ホスト名および IP アドレスの値に、オーバークラウドレジストリーのポート番号も追加する必要があります。ポート番号は **8787** です。

9.4. 非推奨化および廃止された NOVASCHEDULERDEFAULTFILTERS パラメーターのフィルター

お使いの環境でカスタムの **NovaSchedulerDefaultFilters** パラメーターが使用されている場合は、パラメーターを編集して以下に示す非推奨および廃止されたフィルターを削除します。

Filter	ステータス
AggregateCoreFilter	非推奨
AggregateRamFilter	非推奨
AggregateDiskFilter	非推奨
ExactCoreFilter	削除
ExactRamFilter	削除
ExactDiskFilter	削除
CoreFilter	削除
RamFilter	削除
DiskFilter	削除
RetryFilter	非推奨



重要

非推奨 と識別されたフィルターを使用しないでください。Red Hat OpenStack Platform 16.1 には引き続き非推奨のフィルターが含まれていますが、Red Hat OpenStack Platform の今後のバージョンにはこれらのフィルターは含まれません。

9.5. COMPUTE 名の形式の設定

Red Hat OpenStack Platform 13 では、`%stackname%-compute-%index%` がコンピュートノードのデフォルトの命名形式として使用されます。Red Hat OpenStack Platform 16.1 は `%stackname%-novacompute-%index%` をコンピュートノードのデフォルトの命名形式として使用します。デフォルトの命名形式を変更して、元の Red Hat OpenStack Platform 13 の命名形式を維持します。元の命名形式を使用しない場合には、director は新しい命名形式の新たな OpenStack Compute (nova) エージェントを設定し、古い命名形式の既存の OpenStack Compute (nova) エージェントを孤立サービスとして維持します。

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 の命名形式を設定します。
 - カスタムの **roles_data** ファイルを使用する場合には、カスタムの **roles_data** ファイルを編集し、**Compute** ロールの **HostnameFormatDefault** パラメーターを設定します。

```
- name: Compute
...
  HostnameFormatDefault: '%stackname%-compute-%index%'
...
```

カスタム **roles_data** ファイルを編集します。

- **openstack-tripleo-heat-templates** でデフォルトの **roles_data** ファイルを使用する場合には、環境ファイルで命名形式を設定します。実際のノード数およびフレーバーに合わせて環境ファイルを編集します。このファイルは、通常 **node-info.yaml** という名前です。**parameter_defaults** セクションに **ComputeHostnameFormat** パラメーターを追加します。

```
parameter_defaults:
...
  ComputeHostnameFormat: '%stackname%-compute-%index%'
...
```

node-info.yaml ファイルを保存します。

9.6. インスタンスのシリアル番号の設定

Red Hat OpenStack Platform 13 では、ホストマシンの仮想 BIOS に保存されるインスタンスのシリアル番号は、ホストのシリアル番号に基づいています。

Red Hat OpenStack Platform 16.1 では、ホストマシンの仮想 BIOS に格納されるインスタンスのシリアル番号は、デフォルトでインスタンスの UUID に基づいています。

Red Hat OpenStack Platform 16.1 にアップグレードするときに、Red Hat OpenStack Platform 13 デプロイメントの動作を維持したい場合は、**[libvirt]sysinfo_serial** を設定する必要があります。

手順

1. アンダークラウドに **stack** ユーザーとしてログインします。
2. **stackrc** ファイルを取得します。

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

3. 環境ファイルを開きます。
4. 次の設定を環境ファイルに追加して、インスタンスのシリアル番号がホストのシリアル番号に基づくように指定します。

```
parameter_defaults:
  <Role>ExtraConfig:
    nova::config::nova_config:
      libvirt/sysinfo_serial:
        value: auto
```

5. 更新を環境ファイルに保存し、そのファイルをオーバークラウドのアップグレードおよびデプロイメントコマンドに追加します。

9.7. SSL/TLS 設定の更新

resource_registry から **NodeTLSData** リソースを削除して、SSL/TLS 設定を更新します。

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. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

3. カスタムのオーバークラウド SSL/TLS パブリックエンドポイントファイルを編集します。通常、このファイルの名前は **~/templates/enable-tls.yaml** です。
4. **resource_registry** から **NodeTLSData** リソースを削除します。

```
resource_registry:
  OS::TripleO::NodeTLSData: /usr/share/openstack-tripleo-heat-
    templates/puppet/extraconfig/tls/tls-cert-inject.yaml
  ...
```

オーバークラウドデプロイメントは、HAProxy の新しいサービスを使用して SSL/TLS が有効かどうかを判断します。



注記

enable-tls.yaml ファイルの **resource_registry** セクションにある唯一のリソースである場合は、**resource_registry** セクションをすべて削除します。

5. SSL/TLS パブリックエンドポイントファイルを保存します。

9.8. 設定後テンプレートの更新

OS::TripleO::NodeExtraConfigPost リソースを使用して設定後テンプレートを登録して実行する場合は、テンプレートに **EndpointMap** パラメーターを追加する必要があります。

手順

1. 設定後テンプレートを編集します。
2. **parameters** セクションに、**EndpointMap** パラメーターとそのサブパラメーターを追加します。

```
parameters:
  servers:
    type: json
  nameserver_ip:
    type: string
  DeployIdentifier:
    type: string
  EndpointMap:
    default: {}
    type: json
```

3. テンプレートを保存します。

9.9. 従来のテレメトリサービスを維持する場合の考慮事項

Red Hat OpenStack Platform (RHOSP) 16.1 では、Service Telemetry Framework (STF) を優先し、OpenStack Telemetry のコンポーネントは非推奨になりました。したがって、アップグレード後は従来のテレメトリコンポーネントは有効ではありません。

自動スケーリングまたは CloudForms サービスを使用する場合は、従来のテレメトリサービスを維持する必要があります。

従来の RHOSP 13 Telemetry サービスをそのまま使用するには、**openstack overcloud upgrade prepare** および **openstack overcloud upgrade converge** コマンドを実行する時に、`/usr/share/openstack-tripleo-heat-templates/environments/enable-legacy-telemetry.yaml` 環境ファイルを追加します。

アップグレード後にオーバークラウドを更新するたびに、**enable-legacy-telemetry.yaml** 環境ファイルも追加する必要があります。

従来のテレメトリサービスは、STF への移行を促進するためにのみ利用可能で、RHOSP の将来のバージョンでは削除される予定です。

第10章 RED HAT カスタマーポータルへのオーバークラウド登録の更新

Red Hat OpenStack Platform 16.1 では、Ansible ベースの手法を使用してオーバークラウドノードを Red Hat カスタマーポータルに登録します。

10.1. RED HAT SUBSCRIPTION MANAGER (RHSM) コンポーザブルサービス

rhsm コンポーザブルサービスを使用して、Ansible を介してオーバークラウドノードに登録することができます。デフォルトの **roles_data** ファイルの各ロールには、**OS::TripleO::Services::Rhsm** リソースが含まれており、これはデフォルトで無効になっています。サービスを有効にするには、このリソースを **rhsm** コンポーザブルサービスのファイルに登録します。

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
```

rhsm コンポーザブルサービスは **RhsmVars** パラメーターを受け入れます。これを使用して、登録に必要な複数のサブパラメーターを定義することができます。

```
parameter_defaults:
  RhsmVars:
    rhsm_repos:
      - rhel-8-for-x86_64-baseos-tus-rpms
      - rhel-8-for-x86_64-appstream-tus-rpms
      - rhel-8-for-x86_64-highavailability-tus-rpms
      ...
    rhsm_username: "myusername"
    rhsm_password: "p@55w0rd!"
    rhsm_org_id: "1234567"
    rhsm_release: 8.2
```

RhsmVars パラメーターをロール固有のパラメーター (例: **ControllerParameters**) と共に使用することにより、異なるノードタイプ用の特定のリポジトリを有効化する場合に柔軟性を提供することもできます。

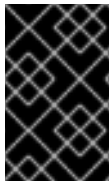
10.2. RHSMVARS サブパラメーター

rhsm コンポーザブルサービスを設定する際に、以下のサブパラメーターを **RhsmVars** パラメーターの一部として使用します。利用可能な Ansible パラメーターについての詳細は、[ロールに関するドキュメント](#) を参照してください。

rhsm	説明
rhsm_method	登録の方法を選択します。 portal 、 satellite 、または disable のいずれかです。

rhsm	説明
rhsm_org_id	登録に使用する組織。この ID を特定するには、アンダークラウドノードから sudo subscription-manager orgs を実行します。プロンプトが表示されたら Red Hat の認証情報を入力して、出力される Key の値を使用します。組織の ID についての詳細は、 Red Hat Subscription Management における組織 ID について理解する を参照してください。
rhsm_pool_ids	使用するサブスクリプションプール ID。サブスクリプションを自動でアタッチしない場合は、このパラメーターを使用します。この ID を特定するには、アンダークラウドノードから sudo subscription-manager list --available --all --matches="Red Hat OpenStack" を実行して、出力される Pool ID 値を使用します。
rhsm_activation_key	登録に使用するアクティベーションキー
rhsm_autosubscribe	このパラメーターを使用して、互換性のあるサブスクリプションを自動的にこのシステムにアタッチします。この機能を有効にするには、値を true に設定します。
rhsm_baseurl	コンテンツを取得するためのベース URL。デフォルトの URL は Red Hat コンテンツ配信ネットワークです。Satellite サーバーを使用している場合は、この値を Satellite サーバーコンテンツリポジトリーのベース URL に変更します。
rhsm_server_hostname	登録用のサブスクリプション管理サービスのホスト名。デフォルトは Red Hat Subscription Management のホスト名です。Satellite サーバーを使用している場合は、この値を Satellite サーバーのホスト名に変更します。
rhsm_repos	有効にするリポジトリーのリスト
rhsm_username	登録用のユーザー名。可能な場合には、登録にアクティベーションキーを使用します。
rhsm_password	登録用のパスワード。可能な場合には、登録にアクティベーションキーを使用します。
rhsm_release	リポジトリー固定用の Red Hat Enterprise Linux リリース。Red Hat OpenStack Platform の場合、このパラメーターは 8.2 に設定されます。
rhsm_rhsm_proxy_host name	HTTP プロキシのホスト名。たとえば、 proxy.example.com 。
rhsm_rhsm_proxy_port	HTTP プロキシ通信のポート。たとえば、 8080 。
rhsm_rhsm_proxy_user	HTTP プロキシにアクセスするためのユーザー名

rhsm	説明
rhsm_rhsm_proxy_pass word	HTTP プロキシにアクセスするためのパスワード



重要

rhsm_method が **portal** に設定されている場合に限り、**rhsm_activation_key** と **rhsm_repos** を使用できます。**rhsm_method** を **satellite** に設定すると、**rhsm_activation_key** または **rhsm_repos** のいずれかを使用できます。

10.3. RHSM コンポーザブルサービスへの切り替え

従来の **rhel-registration** メソッドは、bash スクリプトを実行してオーバークラウドの登録を処理します。このメソッド用のスクリプトと環境ファイルは、**/usr/share/openstack-tripleo-heat-templates/extraconfig/pre_deploy/rhel-registration/** のコア Heat テンプレートコレクションにあります。

rhel-registration メソッドを **rhsm** コンポーザブルサービスに切り替えるには、以下の手順を実施します。

手順

1. **rhel-registration** 環境ファイルは、今後のデプロイメント操作から除外します。通常は、以下のファイルを除外します。
 - **rhel-registration/environment-rhel-registration.yaml**
 - **rhel-registration/rhel-registration-resource-registry.yaml**
2. カスタムの **roles_data** ファイルを使用する場合には、**roles_data** ファイルの各ロールに必ず **OS::TripleO::Services::Rhsm** コンポーザブルサービスを含めてください。以下に例を示します。

```
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  ...
  ServicesDefault:
    ...
    - OS::TripleO::Services::Rhsm
    ...
```

3. **rhsm** コンポーザブルサービスのパラメーター用の環境ファイルを今後のデプロイメント操作に追加します。

このメソッドは、**rhel-registration** パラメーターを **rhsm** サービスのパラメーターに置き換えて、サービスを有効化する Heat リソースを変更します。

```
resource_registry:
  OS::TripleO::NodeExtraConfig: rhel-registration.yaml
```

必要に応じて、以下を行ってください。

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-
  baremetal-ansible.yaml
```

デプロイメントに `/usr/share/openstack-tripleo-heat-templates/environments/rhsm.yaml` 環境ファイルを追加して、サービスを有効にすることもできます。

10.4. RHEL-REGISTRATION から RHSM へのマッピング

`rhel-registration` メソッドから `rhsm` メソッドへの情報の移行を容易に行うには、以下の表を使用してパラメーターとその値をマッピングします。

<code>rhel-registration</code>	<code>rhsm / RhsmVars</code>
<code>rhel_reg_method</code>	<code>rhsm_method</code>
<code>rhel_reg_org</code>	<code>rhsm_org_id</code>
<code>rhel_reg_pool_id</code>	<code>rhsm_pool_ids</code>
<code>rhel_reg_activation_key</code>	<code>rhsm_activation_key</code>
<code>rhel_reg_auto_attach</code>	<code>rhsm_autosubscribe</code>
<code>rhel_reg_sat_url</code>	<code>rhsm_satellite_url</code>
<code>rhel_reg_repos</code>	<code>rhsm_repos</code>
<code>rhel_reg_user</code>	<code>rhsm_username</code>
<code>rhel_reg_password</code>	<code>rhsm_password</code>
<code>rhel_reg_release</code>	<code>rhsm_release</code>
<code>rhel_reg_http_proxy_host</code>	<code>rhsm_rhsm_proxy_hostname</code>
<code>rhel_reg_http_proxy_port</code>	<code>rhsm_rhsm_proxy_port</code>
<code>rhel_reg_http_proxy_username</code>	<code>rhsm_rhsm_proxy_user</code>
<code>rhel_reg_http_proxy_password</code>	<code>rhsm_rhsm_proxy_password</code>

10.5. RHSM コンポーザブルサービスを使用したオーバークラウドの登録

`rhsm` コンポーザブルサービスを有効にして設定する環境ファイルを作成します。director はこの環境ファイルを使用して、ノードを登録し、サブスクライブします。

手順

1. 設定を保存するための環境ファイル (**templates/rhsm.yml**) を作成します。
2. 環境ファイルに設定を追加します。以下に例を示します。

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-
  templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
parameter_defaults:
  RhsmVars:
    rhsm_repos:
      - rhel-8-for-x86_64-baseos-tus-rpms
      - rhel-8-for-x86_64-appstream-tus-rpms
      - rhel-8-for-x86_64-highavailability-tus-rpms
      ...
    rhsm_username: "myusername"
    rhsm_password: "p@55w0rd!"
    rhsm_org_id: "1234567"
    rhsm_pool_ids: "1a85f9223e3d5e43013e3d6e8ff506fd"
    rhsm_method: "portal"
    rhsm_release: 8.2
```

- **resource_registry** セクションは、各ロールで利用可能な **OS::TripleO::Services::Rhsm** リソースに **rhsm** コンポーザブルサービスを関連付けます。
- **RhsmVars** の変数は、Red Hat の登録を設定するためにパラメーターを Ansible に渡します。

3. 環境ファイルを保存します。

10.6. 異なるロールに対する RHSM コンポーザブルサービスの適用

rhsm コンポーザブルサービスをロールごとに適用することができます。たとえば、コントローラーノード、コンピューターノード、および Ceph Storage ノードに、異なる設定セットを適用することができます。

手順

1. 設定を保存するための環境ファイル (**templates/rhsm.yml**) を作成します。
2. 環境ファイルに設定を追加します。以下に例を示します。

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-
  templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
parameter_defaults:
  ControllerParameters:
    RhsmVars:
      rhsm_repos:
        - rhel-8-for-x86_64-baseos-tus-rpms
        - rhel-8-for-x86_64-appstream-tus-rpms
        - rhel-8-for-x86_64-highavailability-tus-rpms
        - ansible-2.9-for-rhel-8-x86_64-rpms
        - advanced-virt-for-rhel-8-x86_64-eus-rpms
```

```

- openstack-16.1-for-rhel-8-x86_64-rpms
- fast-datapath-for-rhel-8-x86_64-rpms
rhsm_username: "myusername"
rhsm_password: "p@55w0rd!"
rhsm_org_id: "1234567"
rhsm_pool_ids: "55d251f1490556f3e75aa37e89e10ce5"
rhsm_method: "portal"
rhsm_release: 8.2
ComputeParameters:
RhsmVars:
rhsm_repos:
- rhel-8-for-x86_64-baseos-tus-rpms
- rhel-8-for-x86_64-appstream-tus-rpms
- rhel-8-for-x86_64-highavailability-tus-rpms
- ansible-2.9-for-rhel-8-x86_64-rpms
- advanced-virt-for-rhel-8-x86_64-eus-rpms
- openstack-16.1-for-rhel-8-x86_64-rpms
- fast-datapath-for-rhel-8-x86_64-rpms
rhsm_username: "myusername"
rhsm_password: "p@55w0rd!"
rhsm_org_id: "1234567"
rhsm_pool_ids: "55d251f1490556f3e75aa37e89e10ce5"
rhsm_method: "portal"
rhsm_release: 8.2
CephStorageParameters:
RhsmVars:
rhsm_repos:
- rhel-8-for-x86_64-baseos-tus-rpms
- rhel-8-for-x86_64-appstream-tus-rpms
- rhel-8-for-x86_64-highavailability-tus-rpms
- ansible-2.9-for-rhel-8-x86_64-rpms
- openstack-16.1-deployment-tools-for-rhel-8-x86_64-rpms
rhsm_username: "myusername"
rhsm_password: "p@55w0rd!"
rhsm_org_id: "1234567"
rhsm_pool_ids: "68790a7aa2dc9dc50a9bc39fab55e0d"
rhsm_method: "portal"
rhsm_release: 8.2

```

resource_registry は、各ロールで利用可能な **OS::TripleO::Services::Rhsm** リソースに **rhsm** コンポーザブルサービスを関連付けます。

ControllerParameters、**ComputeParameters**、および **CephStorageParameters** パラメーターはいずれも、個別の **RhsmVars** パラメーターを使用してサブスクリプションの情報をそれぞれのロールに渡します。



注記

Red Hat Ceph Storage のサブスクリプションおよび Ceph Storage 固有のリポジトリを使用するように、**CephStorageParameters** パラメーター内の **RhsmVars** パラメーターを設定します。 **rhsm_repos** パラメーターに、コントローラーノードおよびコンピューターノードに必要な Extended Update Support (EUS) リポジトリではなく、標準の Red Hat Enterprise Linux リポジトリが含まれるようにします。

3. 環境ファイルを保存します。

第11章 RED HAT SATELLITE SERVER へのオーバークラウド登録の更新

Red Hat OpenStack Platform 16.1 では、Ansible ベースの手法を使用してオーバークラウドノードを Red Hat Satellite Server 6 に登録します。

11.1. RED HAT SUBSCRIPTION MANAGER (RHSM) コンポーザブルサービス

rhsm コンポーザブルサービスを使用して、Ansible を介してオーバークラウドノードを登録することができます。デフォルトの **roles_data** ファイルの各ロールには、**OS::TripleO::Services::Rhsm** リソースが含まれており、これはデフォルトで無効になっています。サービスを有効にするには、このリソースを **rhsm** コンポーザブルサービスのファイルに登録します。

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
```

rhsm コンポーザブルサービスは **RhsmVars** パラメーターを受け入れます。これを使用して、登録に必要な複数のサブパラメーターを定義することができます。

```
parameter_defaults:
  RhsmVars:
    rhsm_repos:
      - rhel-8-for-x86_64-baseos-tus-rpms
      - rhel-8-for-x86_64-appstream-tus-rpms
      - rhel-8-for-x86_64-highavailability-tus-rpms
      ...
    rhsm_username: "myusername"
    rhsm_password: "p@55w0rd!"
    rhsm_org_id: "1234567"
    rhsm_release: 8.2
```

RhsmVars パラメーターをロール固有のパラメーター (例: **ControllerParameters**) と共に使用することにより、異なるノードタイプ用の特定のリポジトリを有効化する場合に柔軟性を提供することもできます。

11.2. RHSMVARS サブパラメーター

rhsm コンポーザブルサービスを設定する際に、以下のサブパラメーターを **RhsmVars** パラメーターの一部として使用します。利用可能な Ansible パラメーターについての詳細は、[ロールに関するドキュメント](#) を参照してください。

rhsm	説明
rhsm_method	登録の方法を選択します。 portal 、 satellite 、 または disable のいずれかです。

rhsm	説明
rhsm_org_id	登録に使用する組織。この ID を特定するには、アンダークラウドノードから sudo subscription-manager orgs を実行します。プロンプトが表示されたら Red Hat の認証情報を入力して、出力される Key の値を使用します。組織の ID についての詳細は、 Red Hat Subscription Management における組織 ID について理解する を参照してください。
rhsm_pool_ids	使用するサブスクリプションプール ID。サブスクリプションを自動でアタッチしない場合は、このパラメーターを使用します。この ID を特定するには、アンダークラウドノードから sudo subscription-manager list --available --all --matches="Red Hat OpenStack" を実行して、出力される Pool ID 値を使用します。
rhsm_activation_key	登録に使用するアクティベーションキー
rhsm_autosubscribe	このパラメーターを使用して、互換性のあるサブスクリプションを自動的にこのシステムにアタッチします。この機能を有効にするには、値を true に設定します。
rhsm_baseurl	コンテンツを取得するためのベース URL。デフォルトの URL は Red Hat コンテンツ配信ネットワークです。Satellite サーバーを使用している場合は、この値を Satellite サーバーコンテンツリポジトリーのベース URL に変更します。
rhsm_server_hostname	登録用のサブスクリプション管理サービスのホスト名。デフォルトは Red Hat Subscription Management のホスト名です。Satellite サーバーを使用している場合は、この値を Satellite サーバーのホスト名に変更します。
rhsm_repos	有効にするリポジトリーのリスト
rhsm_username	登録用のユーザー名。可能な場合には、登録にアクティベーションキーを使用します。
rhsm_password	登録用のパスワード。可能な場合には、登録にアクティベーションキーを使用します。
rhsm_release	リポジトリー固定用の Red Hat Enterprise Linux リリース。Red Hat OpenStack Platform の場合、このパラメーターは 8.2 に設定されます。
rhsm_rhsm_proxy_host name	HTTP プロキシのホスト名。たとえば、 proxy.example.com 。
rhsm_rhsm_proxy_port	HTTP プロキシ通信のポート。たとえば、 8080 。
rhsm_rhsm_proxy_user	HTTP プロキシにアクセスするためのユーザー名

rhsm	説明
rhsm_rhsm_proxy_pass word	HTTP プロキシにアクセスするためのパスワード



重要

rhsm_method が **portal** に設定されている場合に限り、**rhsm_activation_key** と **rhsm_repos** を使用できます。**rhsm_method** を **satellite** に設定すると、**rhsm_activation_key** または **rhsm_repos** のいずれかを使用できます。

11.3. RHSM コンポーザブルサービスへの切り替え

従来の **rhel-registration** メソッドは、bash スクリプトを実行してオーバークラウドの登録を処理します。このメソッド用のスクリプトと環境ファイルは、**/usr/share/openstack-tripleo-heat-templates/extraconfig/pre_deploy/rhel-registration/** のコア Heat テンプレートコレクションにあります。

rhel-registration メソッドを **rhsm** コンポーザブルサービスに切り替えるには、以下の手順を実施します。

手順

1. **rhel-registration** 環境ファイルは、今後のデプロイメント操作から除外します。通常は、以下のファイルを除外します。
 - **rhel-registration/environment-rhel-registration.yaml**
 - **rhel-registration/rhel-registration-resource-registry.yaml**
2. カスタムの **roles_data** ファイルを使用する場合には、**roles_data** ファイルの各ロールに必ず **OS::TripleO::Services::Rhsm** コンポーザブルサービスを含めてください。以下に例を示します。

```
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  ...
  ServicesDefault:
  ...
  - OS::TripleO::Services::Rhsm
  ...
```

3. **rhsm** コンポーザブルサービスのパラメーター用の環境ファイルを今後のデプロイメント操作に追加します。

このメソッドは、**rhel-registration** パラメーターを **rhsm** サービスのパラメーターに置き換えて、サービスを有効化する Heat リソースを変更します。

```
resource_registry:
  OS::TripleO::NodeExtraConfig: rhel-registration.yaml
```

必要に応じて、以下を行ってください。

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-
  baremetal-ansible.yaml
```

デプロイメントに `/usr/share/openstack-tripleo-heat-templates/environments/rhsm.yaml` 環境ファイルを追加して、サービスを有効にすることもできます。

11.4. RHEL-REGISTRATION から RHSM へのマッピング

`rhel-registration` メソッドから `rhsm` メソッドへの情報の移行を容易に行うには、以下の表を使用してパラメーターとその値をマッピングします。

<code>rhel-registration</code>	<code>rhsm / RhsmVars</code>
<code>rhel_reg_method</code>	<code>rhsm_method</code>
<code>rhel_reg_org</code>	<code>rhsm_org_id</code>
<code>rhel_reg_pool_id</code>	<code>rhsm_pool_ids</code>
<code>rhel_reg_activation_key</code>	<code>rhsm_activation_key</code>
<code>rhel_reg_auto_attach</code>	<code>rhsm_autosubscribe</code>
<code>rhel_reg_sat_url</code>	<code>rhsm_satellite_url</code>
<code>rhel_reg_repos</code>	<code>rhsm_repos</code>
<code>rhel_reg_user</code>	<code>rhsm_username</code>
<code>rhel_reg_password</code>	<code>rhsm_password</code>
<code>rhel_reg_release</code>	<code>rhsm_release</code>
<code>rhel_reg_http_proxy_host</code>	<code>rhsm_rhsm_proxy_hostname</code>
<code>rhel_reg_http_proxy_port</code>	<code>rhsm_rhsm_proxy_port</code>
<code>rhel_reg_http_proxy_username</code>	<code>rhsm_rhsm_proxy_user</code>
<code>rhel_reg_http_proxy_password</code>	<code>rhsm_rhsm_proxy_password</code>

11.5. RED HAT SATELLITE SERVER へのオーバークラウドの登録

ノードを Red Hat カスタマーポータルではなく Red Hat Satellite に登録するには、`rhsm` コンポーザブルサービスを有効にして設定する環境ファイルを作成します。

手順

1. 設定を保存するための環境ファイル (**templates/rhsm.yml**) を作成します。
2. 環境ファイルに設定を追加します。以下に例を示します。

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-
  templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
parameter_defaults:
  RhsmVars:
    rhsm_activation_key: "myactivationkey"
    rhsm_method: "satellite"
    rhsm_org_id: "ACME"
    rhsm_server_hostname: "satellite.example.com"
    rhsm_baseurl: "https://satellite.example.com/pulp/repos"
    rhsm_release: 8.2
```

resource_registry は、各ロールで利用可能な **OS::TripleO::Services::Rhsm** リソースに **rhsm** コンポーザブルサービスを関連付けます。

RhsmVars の変数は、Red Hat の登録を設定するためにパラメーターを Ansible に渡します。

3. 環境ファイルを保存します。

11.6. SATELLITE SERVER を使用するための LEAPP の準備

Satellite Server 6 を使用して RPM コンテンツをホストする場合は、以下の準備手順を実施して、Satellite を使用して Leapp によるアップグレードが正常に完了するようにします。

前提条件

- Red Hat OpenStack Platform 16.1 および Red Hat Enterprise Linux 8.2 のリポジトリにリンクされた Satellite Server のアクティベーションキーを作成している。
- オーバークラウドノードの Ansible インベントリーファイルを生成している。
- Satellite Server で Red Hat OpenStack Platform 16.1 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、アップグレードに必要なリポジトリを追加している。詳細は、[Red Hat Satellite Server 6 に関する考慮事項](#) を参照してください。
- taCeph のサブスクリプションを使用し、Ceph ストレージノード用に **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 Red Hat Enterprise Linux (RHEL) 8.2 リポジトリを追加する必要があります。

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

```
rhel-8-for-x86_64-appstream-rpms
x86_64 8.2
```

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

```
rhel-8-for-x86_64-baseos-rpms
x86_64 8.2
```

詳細は、[Red Hat Satellite コンテンツ管理ガイドの Red Hat コンテンツのインポート](#) および [コンテンツビューの管理](#) 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. **playbook-satellite.yaml** という名前のファイルを作成し、以下のコンテンツをファイルに貼り付けます。

```
- name: Pre-install leapp
  hosts: overcloud
  become: yes
  tasks:
    - name: Pre-install leapp
      yum:
        name:
          - leapp
          - leapp-repository
        state: installed
    - name: Remove katello-host-tools-fact-plugin
      yum:
        name:
          - katello-host-tools-fact-plugin
        state: removed
    - name: Register system
      redhat_subscription:
        activationkey: "osp16-key"
        org_id: "ACME"
        server_hostname: "satellite.example.com"
        rhsm_baseurl: "https://satellite.example.com/pulp/repos"
        force_register: True
```

お使いの Satellite Server に合わせて **redhat_subscription** 変数を変更します。

3. Playbook を実行します。

```
$ ansible-playbook -i ~/inventory.yaml playbook-satellite.yaml
```

第12章 DIRECTOR でデプロイされた CEPH STORAGE のアップグレードの準備

director のデプロイした Red Hat Ceph Storage クラスターがデプロイメントで使用される場合は、本項に記載する手順を完了する必要があります。



重要

RHOSP 16.1 は RHEL 8.2 でサポートされています。ただし、Ceph Storage ロールにマップされているホストは、最新のメジャー RHEL リリースに更新されます。詳細は、[Red Hat Ceph Storage: サポートされる設定](#) を参照してください。



注記

外部の Ceph デプロイメントと共にアップグレードする場合は、本項に記載する手順を省略し、[13章 外部の Ceph デプロイメントと組み合わせたアップグレードの準備](#)に進む必要があります。

Red Hat OpenStack Platform 16.1 へのアップグレード中、プロセスでは Red Hat Ceph Storage 3 のコンテナ化されたサービスを継続して使用します。Red Hat OpenStack Platform 16.1 へのアップグレードが完了したら、Ceph Storage サービスを Red Hat Ceph Storage 4 にアップグレードします。

Red Hat OpenStack Platform 16.1 のアップグレードと Ceph Storage サービスの Red Hat Ceph Storage 4 へのアップグレードの両方が完了するまで、Shared File Systems サービス (manila) で新しいファイル共有をプロビジョニングすることはできません。

12.1. CEPH STORAGE ノードのアップグレードプロセスの概要

オーバークラウドのアップグレードプロセス中、director でデプロイされた Ceph Storage ノードは Red Hat Ceph Storage 3 コンテナを継続して使用します。アップグレードプロセス中に Ceph Storage ノードおよびサービスがどのように影響を受けるかについて理解するには、Ceph Storage アップグレードプロセスの各要素ごとに以下の概要を参照してください。

ceph-ansible

ceph-ansible は、ロールおよび Playbook のコレクションです。director はこれを使用して Ceph Storage サービスのインストール、維持、およびアップグレードを行います。アンダークラウドをアップグレードする際に実行した特定のコマンドにより、Red Hat Enterprise Linux 8.2 への移行後に **ceph-ansible** は最新のバージョン 3 コレクションの状態を維持します。バージョン 3 の **ceph-ansible** は、オーバークラウドのアップグレード期間中、バージョン 3 のコンテナ化された Ceph Storage サービスを維持します。アップグレードが完了したら、[Red Hat Ceph Storage Tools 4 for RHEL 8](#) リポジトリを有効にし、**ceph-ansible** をバージョン 4 に更新します。

Podman への移行

オーバークラウドのアップグレード時には、**openstack overcloud external-upgrade run --tags ceph_systemd** コマンドを実行し、Ceph Storage のコンテナ化されたサービスを制御する **systemd** サービスが Docker ではなく Podman を使用するように変更する必要があります。Ceph Storage のコンテナ化されたサービスが含まれるいずれかのノードでオペレーティングシステムをアップグレードする前に、このコマンドを実行します。

systemd サービスがノード上で Podman を使用するように変更したら、オペレーティングシステムのアップグレードおよび OpenStack Platform サービスのアップグレードを実施します。そのノード上の Ceph Storage コンテナは、OpenStack Platform サービスのアップグレード後に再度実行されます。

Ceph Storage のオペレーティングシステムのアップグレード

概略としては、Ceph Storage ノードで、オーバークラウドノードで実施するのと同じワークフローに従います。Ceph Storage ノードに対して **openstack overcloud upgrade run --tags system_upgrade** コマンドを実行すると、director は Ceph Storage ノードで Leapp を実行し、オペレーティングシステムを Red Hat Enterprise Linux 8.2 にアップグレードします。続いて、Ceph Storage ノードに対してタグ付けされていない **openstack overcloud upgrade run** コマンドを実行します。これにより、以下のコンテナが実行されます。

- Red Hat Ceph Storage 3 のコンテナ化されたサービス
- Red Hat OpenStack Platform 16.1 のコンテナ化されたサービス

Red Hat Ceph Storage 4 へのアップグレード

Leapp によるアップグレードおよび Red Hat OpenStack Platform のアップグレードを完了した後も、Ceph Storage のコンテナ化されたサービスは引き続きバージョン 3 のコンテナを使用します。この時点で **ceph-ansible** をバージョン 4 にアップグレードする必要があります。続いて **openstack overcloud external-upgrade run --tags ceph** コマンドを実行し、すべてのノード上の Red Hat Ceph Storage サービスをすべてバージョン 4 にアップグレードします。

Ceph Storage ワークフローの概要

Red Hat Ceph Storage をアップグレードするワークフローの概要を以下に示します。このワークフローは Red Hat OpenStack Platform の全体的なワークフローに統合されていて、アンダークラウド上でアップグレードフレームワークのコマンドを実行すると、このワークフローの操作が実施されます。

1. アンダークラウドをアップグレードしますが、バージョン 3 の **ceph-ansible** を維持します。
2. オーバークラウドのアップグレードを開始します。
3. Ceph Storage のコンテナ化されたサービスをホストするノードごとに、以下のタスクを実行します。
 - a. Ceph Storage のコンテナ化されたサービスを Podman に移行する。
 - b. オペレーティングシステムをアップグレードする。
 - c. OpenStack Platform サービスをアップグレードする。これにより、Ceph Storage バージョン 3 のコンテナ化されたサービスが再起動されます。
4. オーバークラウドのアップグレードを完了します。
5. アンダークラウドの **ceph-ansible** をバージョン 4 にアップグレードします。
6. オーバークラウドで Red Hat Ceph Storage 4 にアップグレードします。



注記

このリストは Red Hat OpenStack Platform 16.1 アップグレードプロセス全体の全ステップを網羅している訳ではありません。アップグレードプロセス中 Ceph Storage サービスに何が起こるかを説明するために、Red Hat Ceph Storage に該当する要素にのみ焦点を当てています。

12.2. CEPH-ANSIBLE バージョンの確認

アンダークラウドのアップグレード中、Ceph Storage 3 バージョンの **ceph-ansible** パッケージを維持します。これは、Ceph Storage ノード上の Ceph Storage 3 コンテナの互換性を維持するのに役立ちます。このパッケージがアンダークラウドに残っていることを確認します。

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. **dnf** コマンドを実行して、**ceph-ansible** パッケージのバージョンを確認します。

```
$ sudo dnf info ceph-ansible
```

コマンド出力には、**ceph-ansible** パッケージのバージョン 3 が表示されます。

```
Installed Packages
Name      : ceph-ansible
Version   : 3.xx.xx.xx
...
```

重要

ceph-ansible パッケージが見つからない、またはバージョン 3 のパッケージではない場合は、[Red Hat Package Browser](#) から最新のバージョン 3 パッケージをダウンロードし、アンダークラウドにパッケージを手動でインストールします。**ceph-ansible** バージョン 3 パッケージは Red Hat Enterprise Linux 7 リポジトリからのみ利用可能で、Red Hat Enterprise Linux 8 リポジトリでは利用できない点に注意してください。**ceph-ansible** バージョン 3 は、Red Hat OpenStack Platform のアップグレードフレームワークでの使用を除き、Red Hat Enterprise Linux 8 ではサポートされていません。

12.3. CEPH-ANSIBLE リポジトリの設定

director がオーバークラウドを Red Hat Ceph Storage 4 にアップグレードする前に、Red Hat OpenStack Platform 16.1 の検証フレームワークで **ceph-ansible** が適切にインストールされていることを確認します。フレームワークは、**CephAnsibleRepo** パラメーターを使用して、**ceph-ansible** が正しいリポジトリからインストールされていることを確認します。**openstack overcloud upgrade prepare** コマンドの実行後に director はこのテストを無効にし、テストは Red Hat OpenStack Platform 16.1 オーバークラウドのアップグレード期間中は無効なままとなります。**openstack overcloud upgrade converge** コマンドの実行後に、director はこのテストを再度有効にします。ただし、この検証を準備するために、**CephAnsibleRepo** パラメーターを **Red Hat Ceph Storage Tools 4 for RHEL 8** リポジトリに設定する必要があります。

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. オーバークラウドの Ceph Storage 設定が含まれる環境ファイルを編集します。このファイルは、通常 **ceph-config.yaml** という名前で、**templates** ディレクトリにあります。

```
$ vi /home/stack/templates/ceph-config.yaml
```

3. **parameter_defaults** セクションに **CephAnsibleRepo** パラメーターを追加します。

```
parameter_defaults:
```

```
...  
CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms  
...
```

CephAnsibleRepo は、**ceph-ansible** が含まれるリポジトリを設定します。検証フレームワークでは、このパラメーターを使用して、アンダークラウドに **ceph-ansible** がインストールされていることを確認します。

4. **ceph-config.yaml** ファイルを保存します。

12.4. アップグレード前の CEPH クラスタステータスの確認

オーバークラウドのアップグレードを進める前に、Ceph クラスタが想定どおりに機能していることを確認する必要があります。

手順

1. **ceph-mon** サービスを実行しているノードにログインします。このノードは、通常コントローラーノードまたはスタンドアロンの Ceph Monitor ノードです。
2. 以下のコマンドを入力して、Ceph クラスタのステータスを表示します。

```
$ docker exec ceph-mon-$HOSTNAME ceph -s
```

3. クラスタの健全性ステータスが **HEALTH_OK** であること、およびすべての OSD が **up** の状態にあることを確認します。

第13章 外部の CEPH デプロイメントと組み合わせたアップグレードの準備

外部の Ceph デプロイメントと共にアップグレードする場合は、本項に記載する手順を完了する必要があります。



注記

デプロイメントで外部の Ceph Storage クラスターが使用されない場合は、本項に記載する手順を省略し、次のセクションに進む必要があります。

13.1. CEPH-ANSIBLE のインストール

外部の Ceph デプロイメントと共にアップグレードする場合は、以下の手順を完了する必要があります。

Red Hat OpenStack Platform で Ceph Storage を使用する場合は、**ceph-ansible** パッケージが必要です。

手順

1. Ceph Tools リポジトリを有効にします。

```
[stack@director ~]$ sudo subscription-manager repos --enable=rhceph-4-tools-for-rhel-8-x86_64-rpms
```

2. **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
```

13.2. CEPH-ANSIBLE リポジトリの設定

director がオーバークラウドを Red Hat Ceph Storage 4 にアップグレードする前に、Red Hat OpenStack Platform 16.1 の検証フレームワークで **ceph-ansible** が適切にインストールされていることを確認します。フレームワークは、**CephAnsibleRepo** パラメーターを使用して、**ceph-ansible** が正しいリポジトリからインストールされていることを確認します。**openstack overcloud upgrade prepare** コマンドの実行後に director はこのテストを無効にし、テストは Red Hat OpenStack Platform 16.1 オーバークラウドのアップグレード期間中は無効なままとなります。**openstack overcloud upgrade converge** コマンドの実行後に、director はこのテストを再度有効にします。ただし、この検証を準備するために、**CephAnsibleRepo** パラメーターを Red Hat Ceph Storage Tools 4 for RHEL 8 リポジトリに設定する必要があります。

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. オーバークラウドの Ceph Storage 設定が含まれる環境ファイルを編集します。このファイルは、通常 **ceph-config.yaml** という名前で、**templates** ディレクトリにあります。

```
$ vi /home/stack/templates/ceph-config.yaml
```

3. **parameter_defaults** セクションに **CephAnsibleRepo** パラメーターを追加します。

```
parameter_defaults:  
  ...  
  CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms  
  ...
```

CephAnsibleRepo は、**ceph-ansible** が含まれるリポジトリを設定します。検証フレームワークでは、このパラメーターを使用して、アンダークラウドに **ceph-ansible** がインストールされていることを確認します。

4. **ceph-config.yaml** ファイルを保存します。

第14章 ネットワーク設定の更新

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、いくつかのネットワーク設定を完了する必要があります。

14.1. ネットワークインターフェイステンプレートの更新

Red Hat OpenStack Platform には、不足しているパラメーターを NIC テンプレートファイルに自動的に追加するスクリプトが用意されています。

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. `source` コマンドで **stackrc** ファイルを読み込みます。

```
$ source ~/stackrc
```

3. アンダークラウドにおいて、**update-nic-templates.sh** という名前のファイルを作成し、以下の内容を追加します。

```
#!/bin/bash
STACK_NAME="overcloud"
ROLES_DATA="/usr/share/openstack-tripleo-heat-templates/roles_data.yaml"
NETWORK_DATA="/usr/share/openstack-tripleo-heat-templates/network_data.yaml"
NIC_CONFIG_LINES=$(openstack stack environment show $STACK_NAME | grep
"::Net::SoftwareConfig" | sed -E 's/ *OS::TripleO::// ; s/::Net::SoftwareConfig:// ; s/ http.*user-
files/ /')
echo "$NIC_CONFIG_LINES" | while read LINE; do
    ROLE=$(echo "$LINE" | awk '{print $1;}')
    NIC_CONFIG=$(echo "$LINE" | awk '{print $2;}')

    if [ -f "$NIC_CONFIG" ]; then
        echo "Updating template for $ROLE role."
        python3 /usr/share/openstack-tripleo-heat-templates/tools/merge-new-params-nic-
config-script.py \
            --tth-dir /usr/share/openstack-tripleo-heat-templates \
            --roles-data $ROLES_DATA \
            --network-data $NETWORK_DATA \
            --role-name "$ROLE" \
            --discard-comments yes \
            --template "$NIC_CONFIG"
    else
        echo "No NIC template detected for $ROLE role. Skipping $ROLE role."
    fi
done
```

- カスタムのオーバークラウド名を使用している場合には、**STACK_NAME** 変数をそのオーバークラウドの名前に設定します。オーバークラウドスタックのデフォルト名は **overcloud** です。
- カスタムの **roles_data** ファイルを使用する場合は、**ROLES_DATA** 変数をカスタムファイルの場所に設定します。デフォルトの **roles_data** ファイルを使用する場合には、変数を **/usr/share/openstack-tripleo-heat-templates/roles_data.yaml** のままにします。

- カスタムの **network_data** ファイルを使用する場合は、**NETWORK_DATA** 変数をカスタムファイルの場所に設定します。デフォルトの **network_data** ファイルを使用する場合には、変数を **/usr/share/openstack-tripleo-heat-templates/network_data.yaml** のままにします。
 - **/usr/share/openstack-tripleo-heat-templates/tools/merge-new-params-nic-config-script.py -h** を実行して、スクリプトに追加するオプションのリストを確認します。
4. スクリプトに実行権限を追加します。

```
$ chmod +x update-nic-templates.sh
```

5. (オプション) RHOSP 環境にスパイン/リーフ型ネットワークトポロジーを使用する場合には、**roles_data.yaml** ファイルをチェックして、デプロイメントの NIC テンプレートに正しいロール名が使用されていることを確認します。このスクリプトは、**roles_data.yaml** ファイルの **deprecated_nic_config_name** パラメーターの値を使用します。
6. スクリプトを実行します。

```
$. /update-nic-templates.sh
```

スクリプトにより各カスタム NIC テンプレートのコピーが保存され、不足しているパラメーターで各テンプレートが更新されます。このスクリプトは、カスタムテンプレートを持たないロールもスキップします。

```
No NIC template detected for BlockStorage role. Skipping BlockStorage role.
Updating template for CephStorage role.
The original template was saved as: /home/stack/templates/custom-nics/ceph-storage.yaml.20200903144835
The update template was saved as: /home/stack/templates/custom-nics/ceph-storage.yaml
Updating template for Compute role.
The original template was saved as: /home/stack/templates/custom-nics/compute.yaml.20200903144838
The update template was saved as: /home/stack/templates/custom-nics/compute.yaml
Updating template for Controller role.
The original template was saved as: /home/stack/templates/custom-nics/controller.yaml.20200903144841
The update template was saved as: /home/stack/templates/custom-nics/controller.yaml
No NIC template detected for ObjectStorage role. Skipping ObjectStorage role.
```

14.2. アップグレード中の OPEN VSWITCH との互換性の維持

Red Hat OpenStack Platform 13 では、OpenStack Networking (neutron) のデフォルト ML2 バックエンドとして Open vSwitch (OVS) が使用されます。新しいバージョンの Red Hat OpenStack Platform では、OVS 機能を拡張した Open Virtual Network (OVN) が使用されます。ただし、安定したアップグレードを確保するには、アップグレードの期間中 OVS 機能を維持し、アップグレードの完了後に OVN に移行する必要があります。

アップグレード中に OVS との互換性を維持するには、環境ファイルコレクションの一部として以下の環境ファイルを追加します。

- **/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml**

注記

neutron-ovs.yaml 環境ファイルを追加する場合は、**neutron-ovs-dvr.yaml** 環境ファイルが環境ファイルコレクションに含まれているかどうかを確認します。アップグレード中にエラーが発生するのを防ぐためには、**neutron-ovs-dvr.yaml** ファイルの前に **neutron-ovs.yaml** 環境ファイルを追加する必要があります。

OVN への移行が完了するまで、このファイルをデプロイメントの一部として扱います。すべてのオーバークラウドアップグレードコマンドおよびデプロイメントコマンドに、このファイルを追加します。

- **openstack overcloud upgrade prepare**
- **openstack overcloud upgrade converge**
- **openstack overcloud deploy**
- **openstack overcloud update prepare**
- **openstack overcloud update converge**
- 環境ファイルを使用するその他すべてのコマンド

OVS の互換性に関するトラブルシューティング

neutron-ovs.yaml ファイルで定義されたパラメーターが **neutron-ovs-dvr.yaml** で定義されたパラメーターを上書きするためにアップグレードプロセスが失敗する場合は、これらのファイルを追加する順序を変更し、影響を受けるノードで再度 **openstack overcloud upgrade prepare** および **openstack overcloud upgrade run** を実行します。影響を受けるノードの1つがコンピュータノードである場合は、そのノードから **openstack-neutron*** パッケージを削除します。

14.3. アップグレード中のコンポーザブルネットワーク互換性の維持

Red Hat OpenStack Platform 16.1 の **network_data** ファイル形式には、ネットワークの追加サブネットおよびルートを定義するのに使用できる新しいセクションが含まれています。ただし、カスタムの **network_data** ファイルを使用する場合には、引き続き Red Hat OpenStack Platform 13 の **network_data** ファイル形式を使用できます。

- Red Hat OpenStack Platform 13 から 16.1 にアップグレードする場合は、アップグレード中およびアップグレード後に Red Hat OpenStack Platform 13 の **network_data** ファイル形式を使用します。Red Hat OpenStack Platform 13 コンポーザブルネットワークの構文についての詳しい情報は、[Custom composable networks](#) を参照してください。
- Red Hat OpenStack Platform 16.1 に新しいオーバークラウドを作成する場合には、Red Hat OpenStack Platform 16.1 の **network_data** ファイル形式を使用します。Red Hat OpenStack Platform 16.1 コンポーザブルネットワークの構文についての詳しい情報は、[カスタムコンポーザブルネットワーク](#) を参照してください。

第15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。

15.1. 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 16.1 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 16.1 **openstack-tripleo-heat-templates** コレクションの **environments/services** ディレクトリにあります。Red Hat OpenStack Platform 13 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 16.1 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 13 から Red Hat OpenStack Platform 16.1 へのアップグレード中に OVS との互換性を維持するには、**/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml** 環境ファイルを追加する必要があります。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

RHOSP 13 のコンピューターノードが **hybrid** の状態にある場合に限り、アップグレード中に RHOSP 13 と RHOSP 16.1.x のコンピューターノード間でインスタンスを移行することができます。詳細については、**Configuring the Compute Service for Instance Creation** の [Migration constraints](#) を参照してください。

NFV ワークロードには、移行に関する追加の制約があります。アップグレード中に、OVS-DPDK コンピューター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 コンピューター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16章 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。

16.1. 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。その後、`--roles-file` オプションを使用して、**openstack overcloud upgrade prepare** または **openstack overcloud upgrade converge** コマンドにそのファイルを渡します。

16.2. デプロイメントに追加する新たな環境ファイル

通常のオーバークラウドの環境ファイルに加えて、Red Hat OpenStack Platform (RHOSP) 16.1 へのアップグレードを円滑に行うために、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	備考
<code>/home/stack/templates/upgrades-environment.yaml</code>	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。 openstack overcloud upgrade converge を実行した後は、このファイルを破棄します。
<code>/home/stack/templates/rhsm.yaml</code>	このファイルには、オーバークラウドの登録およびサブスクリプション情報が含まれます。このファイルにより、お使いのシステムを Red Hat カスタマーポータルまたは Red Hat Satellite Server のいずれかに登録します。
<code>/home/stack/containers-prepare-parameter.yaml</code>	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。
<code>/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml</code>	OpenStack Platform 16.1 では、デフォルトのネットワークバックエンドとして Open Virtual Network (OVN) が使用されます。しかし、OpenStack Platform 13 では Open vSwitch (OVS) が使用されていました。アップグレード中に OVS との互換性を維持するために、このファイルを追加します。OpenStack Platform 16.1 のドキュメントには、アップグレード後に OVS から OVN に移行するための手順が記載されています。

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 upgrade converge**
- **openstack overcloud deploy**

16.3. デプロイメントから削除する環境ファイル

OpenStack Platform Red Hat OpenStack Platform 13 に固有の環境ファイルをすべて削除します。

- Red Hat OpenStack Platform 13 のコンテナイメージリスト
- Red Hat OpenStack Platform 13 のカスタマーポータルまたは Satellite 用 **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 upgrade converge**
- **openstack overcloud deploy**

16.4. 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した。詳しくは、 Red Hat OpenStack Platform 13 Undercloud and Control Plane Back Up and Restore 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 16.1 アンダークラウドとコントロールプレーンのバックアップおよび復元のアンダークラウドノードのデータベースバックアップの作成 を参照してください。	はい / いいえ
以下の項目を含め、Leapp に関するすべての準備を実施した。 <ul style="list-style-type: none"> • Leapp 環境ファイル • オーバークラウドノードに Leapp データをコピーした。 • すべてのオーバークラウドノードで予測可能な NIC 名が使用されている。 • すべてのオーバークラウドノードに PermitRootLogin が設定されている。 	はい / いいえ
登録情報を Red Hat OpenStack Platform 16.1 リポジトリに更新し、Ansible ベースの手法を使用するように環境ファイルを変更した。	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
環境ファイルのリストを Red Hat OpenStack Platform 16.1 用の新しい環境ファイルで更新した。	はい / いいえ

確認項目	実施状況
オプション: デプロイメントに専用の Object Storage (swift) ノードが含まれる場合: roles_data.yaml ファイルをコピーし、 deprecated_server_resource_name: 'SwiftStorage' を削除し、そのファイルを openstack overcloud upgrade prepare または openstack overcloud upgrade converge コマンドに渡します。	はい / いいえ
古い Red Hat 登録やコンテナイメージの場所に関するファイルなど、Red Hat OpenStack Platform 13 にしか該当しない古い環境ファイルを削除した。	はい / いいえ

第17章 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では、プロセスの特定の段階で異なるコマンドを実行します。



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。

17.1. 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ステップが実行されます。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 16.1 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せん。

17.2. 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 16.1 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、OpenStack Platform の 13 から 16.1 までの各バージョンで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

ワークロード移行のアップグレードタスクタグ

nova_hybrid_state

アップグレード中のワークロードの移行を円滑に行うために、コンピュートノード上に一時的な OpenStack Platform 16.1 コンテナをセットアップするタスク

17.3. 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 16.1 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

Ceph Storage アップグレードの外部タスクタグ

- director を使用してデプロイされた Red Hat Ceph Storage クラスターがデプロイメントで使用される場合は、以下のタグを使用することができます。

ceph

ceph-ansible Playbook を使用して Red Hat Ceph Storage をインストールするタスク

ceph_systemd

podman 管理を使用するために、Red Hat Ceph Storage の systemd ユニットファイルを変換するタスク

- 外部の Ceph デプロイメントと共にアップグレードする場合は、**ceph** および **ceph_systemd** タグを使用するタスクを省略することができます。

データベース移行の外部タスクタグ

system_upgrade_cleanup

system_upgrade_transfer_data タスクに関連するストレージディレクトリーを消去するタスク

system_upgrade_stop_services

すべてのサービスをシャットダウンするタスク

system_upgrade_transfer_data

すべてのサービスをシャットダウンし、ブートストラップノードへのデータベース移行を実施するタスク

17.4. OPENSTACK OVERCLOUD UPGRADE CONVERGE

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最終ステップが実施されます。この最終ステップでは、オーバークラウドの heat スタックを OpenStack Platform 16.1 の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と同期します。このプロセスにより、作成されるオーバークラウドが新しい OpenStack Platform 16.1 オーバークラウドの設定と一致するようになります。このコマンドは **openstack overcloud deploy** コマンドと類似していて、多くの同一オプションが使用されます。

17.5. オーバークラウドノードのアップグレードワークフロー

各オーバークラウドノードでアップグレードを実施する場合、以下の要素を考慮して、アップグレードの各段階で実行する正しいコマンドを判断する必要があります。

コントローラーサービス

- ノードに **Pacemaker** サービスが含まれているか？データベースの移行を開始し、Red Hat OpenStack 13 から 16.1 へのアップグレード時の移行を円滑に行うために一時コンテナを起動

するためには、まずブートストラップノードをアップグレードする必要があります。ブートストラップコントローラーノードのアップグレードプロセス中、新しい Pacemaker クラスターが作成され、新しい Red Hat OpenStack 16.1 コンテナがノードで起動します。一方、残りのコントローラーノードは Red Hat OpenStack 13 で稼働を続けます。ブートストラップノードをアップグレードしたら、Pacemaker サービスが含まれるその他のノードをアップグレードし、ブートストラップノードで起動した新しい Pacemaker クラスターに各ノードが参加するようにしなければなりません。Pacemaker が含まれない split-service コントローラーノードをアップグレードするプロセスでは、これらの追加手順は必要ありません。

Compute サービス

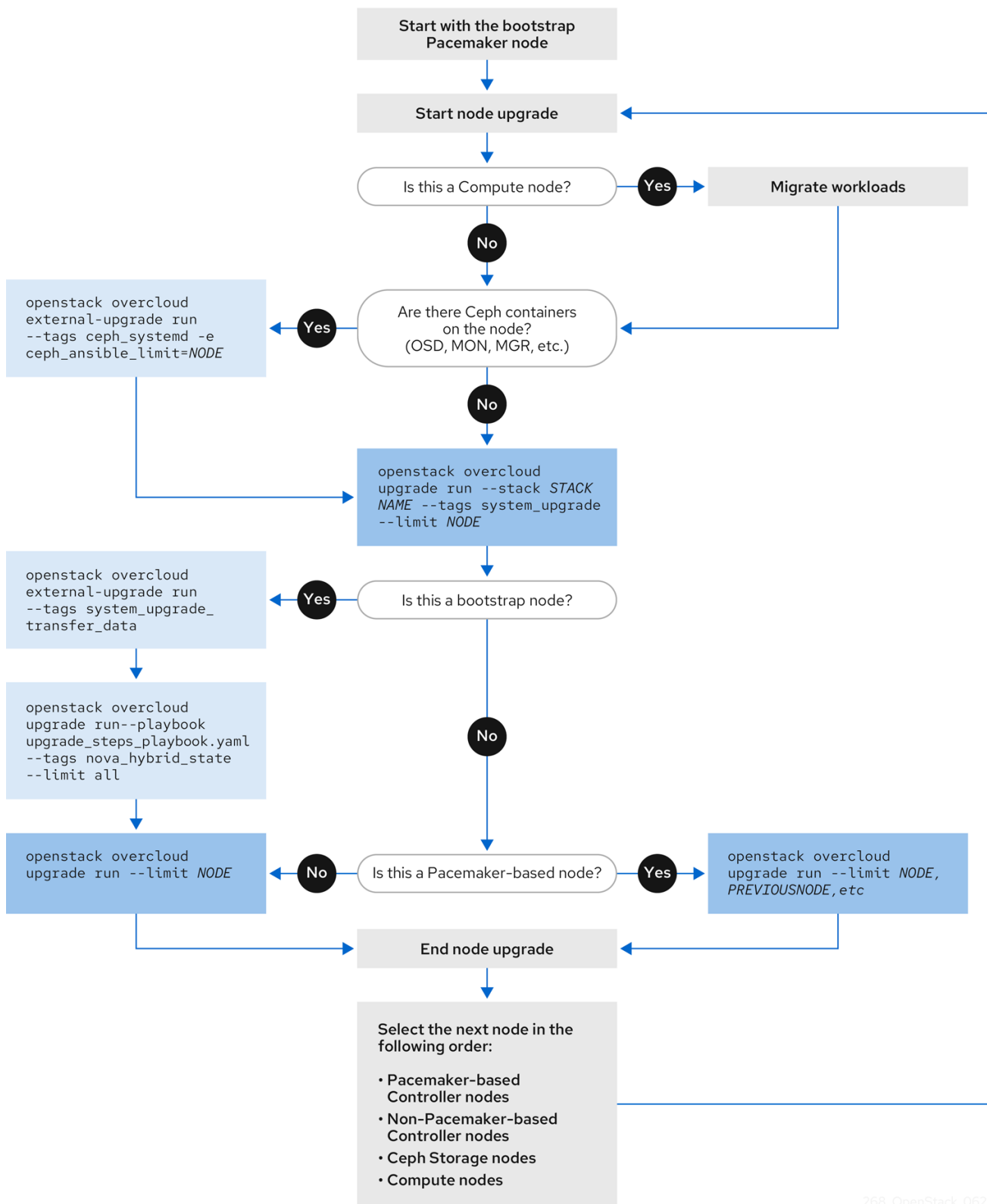
- **ノードがコンピュートノードか？**ノードに Compute サービスが含まれる場合、そのノードから仮想マシンを移行して最大限の可用性を確保する必要があります。ここで言うコンピュートノードには、仮想マシンをホストするためのあらゆるノードが含まれます。この定義には、以下のコンピュートノード種別が含まれます。
 - 通常のコンピュートノード
 - ハイパーコンバージドインフラストラクチャー (HCI) を持つコンピュートノード
 - Data Plane Development Kit (DPDK) または Single Root Input/Output Virtualization (SR-IOV) 等のネットワーク機能仮想化技術を使用するコンピュートノード
 - リアルタイムコンピュートノード

Ceph Storage サービス

- **ノードに Ceph Storage サービスが含まれているか？**`docker` ではなく `podman` を使用するために、ノード上のコンテナ化された Ceph Storage サービスの `systemd` ユニットファイルを交換する必要があります。以下のノード種別がこれに該当します。
 - Ceph Storage OSD ノード
 - Ceph MON サービスが含まれるコントローラーノード
 - Split-Controller Ceph MON 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) を持つコンピュートノード

ワークフロー

以下のワークフロー図を使用して、特定ノードの正しい更新パスを特定します。



268_OpenStack_0622

第18章 標準的なオーバークラウドのアップグレード

このシナリオでは、標準的なオーバークラウド環境のアップグレードプロセスの例を説明します。この環境には、以下のノード種別が含まれます。

- コントローラーノード 3 台
- Ceph Storage ノード 3 台
- 複数のコンピュートノード

18.1. オーバークラウドアップグレード準備タスクの実行

アップグレードを行うには、**openstack overcloud upgrade prepare** コマンドを実行する必要があります。このコマンドにより、以下のタスクが実行されます。

- オーバークラウドのプランを OpenStack Platform 16.1 に更新する
- アップグレードに向けてノードを準備する



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **upgrade prepare** コマンドを実行します。

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE \
  ... \
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  ...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- アップグレード固有のパラメーターが含まれる環境ファイル (**upgrades-environment.yaml**) (-e)
- 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)

- 新しいコンテナ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。
 - OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
 - 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** でカスタムロール (**roles_data**) ファイルを指定します。
 - 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
 - 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。
3. アップグレードの準備が完了するまで待ちます。
 4. コンテナイメージをダウンロードします。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

18.2. DIRECTOR でデプロイされた CEPH STORAGE と組み合わせたコントローラーノードのアップグレード

director を使用してデプロイされた Red Hat Ceph Storage クラスタがデプロイメントで使用される場合は、以下の手順を完了する必要があります。

すべてのコントローラーノードを OpenStack Platform 16.1 にアップグレードする場合は、ブートストラップコントローラーノードからアップグレードを始める必要があります。

ブートストラップコントローラーノードのアップグレードプロセス中、新しい Pacemaker クラスタが作成され、新しい Red Hat OpenStack 16.1 コンテナがノードで起動します。一方、残りのコントローラーノードは Red Hat OpenStack 13 で稼働を続けます。

ブートストラップノードをアップグレードしたら、Pacemaker サービスが含まれるその他のノードをアップグレードし、ブートストラップノードで起動した新しい Pacemaker クラスタに各ノードが参加するようにしなければなりません。詳細は、[オーバークラウドノードのアップグレードワークフロー](#)を参照してください。

以下の例では、コントローラーノードの名前はデフォルトの **overcloud-controller-NODEID** 命名規則を使用して名前を付けています。これには、以下の3つのコントローラーノードが含まれます。

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**

これらの値は、該当する実際のノード名に置き換えてください。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. アンダークラウドノードで以下のコマンドを実行し、ブートストラップコントローラーノードを特定します。

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

3. ブートストラップ Controller ノードをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd  
-e ceph_ansible_limit=overcloud-controller-0
```

<stack_name> は、実際のスタック名に置き換えます。

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したコントローラーノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit  
overcloud-controller-0
```

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。



重要

次のコマンドにより、コントロールプレーンで機能停止が生じます。これ以降の数ステップを実施している間は、標準的なオーバークラウド操作を行うことはできません。

- c. **system_upgrade_transfer_data** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags  
system_upgrade_transfer_data
```

このコマンドにより、最新バージョンのデータベースが既存のノードからブートストラップノードにコピーされます。

- d. **nova_hybrid_state** タグを指定してアップグレードコマンドを実行し、**upgrade_steps_playbook.yaml** Playbook だけを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --playbook
upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
```

このコマンドにより、コンピュータノード上の一時的な 16.1 コンテナが起動します。これにより、後のステップでコンピュータノードをアップグレードする際に、ワークロードの移行が円滑に行われます。

- e. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。



重要

このコマンドの処理が完了すると、コントロールプレーンがアクティブになります。再び標準的なオーバークラウド操作を実施することができます。

- f. アップグレード後に新しい Pacemaker クラスターが起動していること、および galera、rabbit、haproxy、redis 等のコントロールプレーンサービスが実行中であることを確認します。

```
$ sudo pcs status
```

4. 次のコントローラーノードをアップグレードします。

- a. 古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```

クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
ceph_systemd -e ceph_ansible_limit=overcloud-controller-1
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したコントローラーノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- c. 次のコントローラーノードで、**system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-1
```

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。

- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。このノードに加えて、前のステップでアップグレードしたブートストラップノードを **--limit** オプションに含めます。

5. 最後のコントローラーノードをアップグレードします。

- a. 古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```

クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansiible_limit=overcloud-controller-2
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansiible_limit** 変数を使用して、アクションを選択したコントローラーノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- c. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-2
```

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。
- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1,overcloud-controller-2
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。--limit オプションにすべてのコントローラーノードを含めます。

18.3. CEPH STORAGE ノードのオペレーティングシステムのアップグレード

director を使用してデプロイされた Red Hat Ceph Storage クラスターがデプロイメントで使用される場合は、各 Ceph Storage ノードのオペレーティングシステムをアップグレードする必要があります。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. Ceph Storage ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-0
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-cephstorage-0
```

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。
- c. (オプション) Ceph のサブスクリプションを使用し、Ceph ストレージノード用に **overcloud-minimal** イメージを使用するように director を設定している場合は、以下の手順を完了する必要があります。

- i. ノードにログインし、Red Hat Enterprise Linux (RHEL) のマイナーリリースバージョンの設定を解除します。

```
$ sudo subscription-manager release --unset
```

- ii. ノードでシステムの更新を実行します。

```
$ sudo dnf -y update
```

- iii. ノードをリブートします。

```
$ sudo reboot
```

- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-0
```

このコマンドにより **config-download** Playbook が実行され、Ceph Storage ノードでコンポーザブルサービスが設定されます。このステップでは、Ceph Storage ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

3. 次の Ceph Storage ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags  
ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-1
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit  
overcloud-cephstorage-1
```


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。
- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-
cephstorage-1
```

このコマンドにより **config-download** Playbook が実行され、Ceph Storage ノードでコンポーザブルサービスが設定されます。このステップでは、Ceph Storage ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

4. 最後の Ceph Storage ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-2
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephstorage-2
```

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。
- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-
cephstorage-2
```

このコマンドにより **config-download** Playbook が実行され、Ceph Storage ノードでコンポーザブルサービスが設定されます。このステップでは、Ceph Storage ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

18.4. コンピュートノードのアップグレード

すべてのコンピューターノードを OpenStack Platform 16.1 にアップグレードします。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. インスタンスを移行します。移行計画の詳細は、[コンピューター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。

3. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-compute-0
```

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。

4. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。

5. 複数のコンピューターノードを並行してアップグレードするには、**--limit** オプションをアップグレードするノードのコンマ区切りリストに設定します。まず **system_upgrade** タスクを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

続いて、標準の OpenStack サービスのアップグレードを実施します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

18.5. オーバークラウドスタックの同期

アップグレードにおいては、スタックのリソース構造およびパラメーターが OpenStack Platform 16.1 の新規デプロイメントと整合するように、オーバークラウドスタックを更新する必要があります。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **containers-prepare-parameter.yaml** ファイルを編集し、以下のパラメーターおよびその値を削除します。

- **ceph3_namespace**
- **ceph3_tag**
- **ceph3_image**
- **name_prefix_stein**
- **name_suffix_stein**
- **namespace_stein**
- **tag_stein**

3. オーバークラウドでフェンシングを再度有効にするには、**fencing.yaml** 環境ファイルで **EnableFencing** パラメーターを **true** に設定します。

4. アップグレードの最終処理コマンドを実行します。

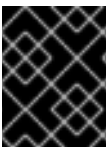
```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- 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**)
- **EnableFencing** パラメーターが **true** に設定された環境ファイル (**fencing.yaml**)。
- 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (**-e**)

- 新しいコンテナ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。
- OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
- 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** でカスタムロール (**roles_data**) ファイルを指定します。
- 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
- 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。

5. スタックの同期が完了するまで待ちます。



重要

これ以降のデプロイメント操作には、**upgrades-environment.yaml** ファイルは必要ありません。

第19章 外部の CEPH デプロイメントと組み合わせたオーバークラウドのアップグレード

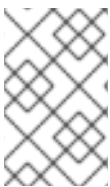
このシナリオでは、外部の Ceph デプロイメントと組み合わせたオーバークラウド環境のアップグレードプロセスの例を説明します。この環境には、以下のノード種別が含まれます。

- コントローラーノード 3 台
- 外部の Ceph Storage クラスター
- 複数のコンピュートノード

19.1. オーバークラウドアップグレード準備タスクの実行

アップグレードを行うには、**openstack overcloud upgrade prepare** コマンドを実行する必要があります。このコマンドにより、以下のタスクが実行されます。

- オーバークラウドのプランを OpenStack Platform 16.1 に更新する
- アップグレードに向けてノードを準備する



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **upgrade prepare** コマンドを実行します。

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE \
  ... \
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  ...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- アップグレード固有のパラメーターが含まれる環境ファイル (**upgrades-environment.yaml**) (-e)
- 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)

- 新しいコンテナ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。
 - OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
 - 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** でカスタムロール (**roles_data**) ファイルを指定します。
 - 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
 - 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。
3. アップグレードの準備が完了するまで待ちます。
 4. コンテナイメージをダウンロードします。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

19.2. 外部の CEPH デプロイメントと組み合わせたコントローラーノードのアップグレード

外部の Ceph デプロイメントと共にアップグレードする場合は、以下の手順を完了する必要があります。

すべてのコントローラーノードを OpenStack Platform 16.1 にアップグレードする場合は、ブートストラップコントローラーノードからアップグレードを始める必要があります。

ブートストラップコントローラーノードのアップグレードプロセス中、新しい Pacemaker クラスターが作成され、新しい Red Hat OpenStack 16.1 コンテナがノードで起動します。一方、残りのコントローラーノードは Red Hat OpenStack 13 で稼働を続けます。

ブートストラップノードをアップグレードしたら、Pacemaker サービスが含まれるその他のノードをアップグレードし、ブートストラップノードで起動した新しい Pacemaker クラスターに各ノードが参加するようにならなければなりません。詳細は、[オーバークラウドノードのアップグレードワークフロー](#)を参照してください。

以下の例では、コントローラーノードの名前はデフォルトの **overcloud-controller-NODEID** 命名規則を使用して名前を付けています。これには、以下の 3 つのコントローラーノードが含まれます。

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**

これらの値は、該当する実際のノード名に置き換えてください。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. アンダークラウドノードで以下のコマンドを実行し、ブートストラップコントローラーノードを特定します。

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

3. ブートストラップ Controller ノードをアップグレードします。

- a. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-0
```

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。



重要

次のコマンドにより、コントロールプレーンで機能停止が生じます。これ以降の数ステップを実施している間は、標準的なオーバークラウド操作を行うことはできません。

- b. **system_upgrade_transfer_data** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags system_upgrade_transfer_data
```

このコマンドにより、最新バージョンのデータベースが既存のノードからブートストラップノードにコピーされます。

- c. **nova_hybrid_state** タグを指定してアップグレードコマンドを実行し、**upgrade_steps_playbook.yaml** Playbook だけを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --playbook upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
```

このコマンドにより、コンピューターノード上の一時的な 16.1 コンテナが起動します。これにより、後のステップでコンピューターノードをアップグレードする際に、ワークロードの移行が円滑に行われます。

- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。



重要

このコマンドの処理が完了すると、コントロールプレーンがアクティブになります。再び標準的なオーバークラウド操作を実施することができます。

- e. アップグレード後に新しい Pacemaker クラスターが起動していること、および galera、rabbit、haproxy、redis 等のコントロールプレーンサービスが実行中であることを確認します。

```
$ sudo pcs status
```

4. 次のコントローラーノードをアップグレードします。

- a. 古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```

クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. 次のコントローラーノードで、**system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-1
```

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。

- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。このノードに加えて、前のステップでアップグレードしたブートストラップノードを **--limit** オプションに含めます。

5. 最後のコントローラーノードをアップグレードします。

- a. 古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```


クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-2
```

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。

- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1,overcloud-controller-2
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。--**limit** オプションにすべてのコントローラーノードを含めます。

19.3. コンピュートノードのアップグレード

すべてのコンピュートノードを OpenStackPlatform 16.1 にアップグレードします。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、--**stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. 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インスタンスの移行](#) を参照してください。
3. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-compute-0
```

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。

4. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。

- 複数のコンピューターノードを並行してアップグレードするには、**--limit** オプションをアップグレードするノードのコンマ区切りリストに設定します。まず **system_upgrade** タスクを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

続いて、標準の OpenStack サービスのアップグレードを実施します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

19.4. オーバークラウドスタックの同期

アップグレードにおいては、スタックのリソース構造およびパラメーターが OpenStack Platform 16.1 の新規デプロイメントと整合するように、オーバークラウドスタックを更新する必要があります。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

- stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

- containers-prepare-parameter.yaml** ファイルを編集し、以下のパラメーターおよびその値を削除します。

- **ceph3_namespace**
- **ceph3_tag**
- **ceph3_image**
- **name_prefix_stein**
- **name_suffix_stein**
- **namespace_stein**
- **tag_stein**

- オーバークラウドでフェンシングを再度有効にするには、**fencing.yaml** 環境ファイルで **EnableFencing** パラメーターを **true** に設定します。
- アップグレードの最終処理コマンドを実行します。

```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- 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)
- **EnableFencing** パラメーターが **true** に設定された環境ファイル (**fencing.yaml**)。
- 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)
- 新しいコンテナ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。
- OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
- 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** でカスタムロール (**roles_data**) ファイルを指定します。
- 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
- 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。

5. スタックの同期が完了するまで待ちます。



重要

これ以降のデプロイメント操作には、**upgrades-environment.yaml** ファイルは必要ありません。

第20章 オーバークラウドアップグレードのスピードアップ

オーバークラウドのアップグレードプロセスを迅速化するために、ブートストラップノードから始めてコントロールプレーンの 1/3 を一度にアップグレードできます。

コントロールプレーンの最初の 1/3 のアップグレードが完了したら、コントロールプレーン API が実行され、クラウドが機能している混在モードに環境を移動できます。高可用性の操作パフォーマンスは、コントロールプレーン全体がアップグレードされた後にのみ再開できます。

多数のコンピューターノードをアップグレードする場合は、パフォーマンスを向上させるために、**--limit Compute** オプションを指定して **openstack overcloud upgrade run** コマンドを実行し、20 ノードのグループを並行して処理することができます。バックグラウンドで複数のアップグレードタスクを実行できます。この場合、各タスクは 20 ノードの個別のグループをアップグレードします。

このシナリオは、コンポーザブルロールによる以下のノード種別が含まれるオーバークラウド環境のアップグレードプロセスの例です。

- コントローラーノード 3 台
- データベースノード 3 台
- ネットワークノード 3 台
- Ceph Storage ノード 3 台
- 複数のコンピューターノード

20.1. オーバークラウドアップグレード準備タスクの実行

アップグレードを行うには、**openstack overcloud upgrade prepare** コマンドを実行する必要があります。このコマンドにより、以下のタスクが実行されます。

- オーバークラウドのプランを OpenStack Platform 16.1 に更新する
- アップグレードに向けてノードを準備する



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **upgrade prepare** コマンドを実行します。

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
...
```

```
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- アップグレード固有のパラメーターが含まれる環境ファイル (**upgrades-environment.yaml**) (-e)
 - 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)
 - 新しいコンテナ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。
 - OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
 - 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** でカスタムロール (**roles_data**) ファイルを指定します。
 - 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
 - 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。
3. アップグレードの準備が完了するまで待ちます。
 4. コンテナイメージをダウンロードします。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

20.2. 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 OpenStack Platform 16.1 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始めてコントロールプレーンノードの 1/3 を一度にアップグレードする必要があります。

ブートストラップコントローラーノードのアップグレードプロセス中、新しい Pacemaker クラスターが作成され、新しい Red Hat OpenStack 16.1 コンテナがノードで起動します。一方、残りのコントローラーノードは Red Hat OpenStack 13 で稼働を続けます。

以下の例では、コントロールプレーンノードの名前はデフォルトの **overcloud-ROLE-NODEID** の命名規則を使用して名前を付けています。これには、コンポーザブルロールによる以下のノード種別が含まれます。

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**

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

これらの値は、該当する実際のノード名に置き換えてください。

コントロールプレーンノードの最初の 1/3 を設定する **overcloud-controller-0**、**overcloud-database-0**、**overcloud-networker-0** および **overcloud-ceph-0** ブートストラップノードをアップグレードした後、Pacemaker サービスが含まれる追加の各 1/3 のノードのアップグレードを行い、各ノードがブートストラップノードで起動した新規 Pacemaker クラスターに参加するようにする必要があります。したがって、**overcloud-controller-2**、**overcloud-database-2**、**overcloud-networker-2** および **overcloud-ceph-2** の前に、**overcloud-controller-1**、**overcloud-database-1**、**overcloud-networker-1** および **overcloud-ceph-1** をアップグレードする必要があります。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

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** ファイルを取得します。

```
$ source ~/stackrc
```

3. アンダークラウドノードで以下のコマンドを実行し、ブートストラップコントローラーノードを特定します。

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

4. **overcloud-controller-0**、**overcloud-database-0**、**overcloud-networker-0**、および **overcloud-ceph-0** コントロールプレーンノードをアップグレードします。
 - a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd
-e ceph_ansible_limit=overcloud-controller-0,overcloud-database-0,overcloud-networker-0,overcloud-ceph-0
```

<stack_name> は、実際のスタック名に置き換えます。

このコマンドにより、以下のアクションが行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-0 &
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-database-0 &
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-networker-0 &
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-ceph-0 &
```

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。



重要

次のコマンドにより、コントロールプレーンで機能停止が生じます。これ以降の数ステップを実施している間は、標準的なオーバークラウド操作を行うことはできません。

- c. **system_upgrade_transfer_data** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
system_upgrade_transfer_data
```

このコマンドにより、最新バージョンのデータベースが既存のノードからブートストラップノードにコピーされます。

- d. **nova_hybrid_state** タグを指定してアップグレードコマンドを実行し、**upgrade_steps_playbook.yaml** Playbook だけを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --playbook
upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
```

このコマンドにより、コンピュータード上の一時的な 16.1 コンテナが起動します。これにより、後のステップでコンピュータードをアップグレードする際に、ワークロードの移行が円滑に行われます。

- e. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-database-0,overcloud-networker-0,overcloud-ceph-0 --playbook all
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。



重要

このコマンドの処理が完了すると、コントロールプレーンがアクティブになります。再び標準的なオーバークラウド操作を実施することができます。

- f. (オプション) ブートストラップコントローラーノードにおいて、アップグレード後に新しい Pacemaker クラスターが起動していること、および galera、rabbit、haproxy、redis 等のコントロールプレーンサービスが実行中であることを確認します。

```
$ sudo pcs status
```

5. **overcloud-controller-1**、**overcloud-database-1**、**overcloud-networker-1**、および **overcloud-ceph-1** コントロールプレーンノードをアップグレードします。

- a. **overcloud-controller-1** ノードにログインし、古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```

クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-controller-1,overcloud-database-1,overcloud-networker-1,overcloud-ceph-1
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- c. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-1,overcloud-database-1,overcloud-networker-1,overcloud-ceph-1
```

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。

- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1,overcloud-database-0,overcloud-database-1,overcloud-networker-0,overcloud-networker-1,overcloud-ceph-0,overcloud-ceph-1
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。このノードに加えて、前のステップでアップグレードしたブートストラップノードを **--limit** オプションに含めます。

6. **overcloud-controller-2**、**overcloud-database-2**、**overcloud-networker-2**、および **overcloud-ceph-2** コントロールプレーンノードをアップグレードします。

- a. **overcloud-controller-2** ノードにログインし、古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```

クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansiible_limit=overcloud-controller-2,overcloud-database-2,overcloud-networker-2,overcloud-ceph-2
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansiible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- c. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-2,overcloud-database-2,overcloud-networker-2,overcloud-ceph-2
```

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。

- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1,overcloud-controller-2,overcloud-database-0,overcloud-database-1,overcloud-database-2,overcloud-networker-0,overcloud-networker-1,overcloud-networker-2,overcloud-ceph-0,overcloud-ceph-1,overcloud-ceph-2
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。--**limit** オプションにすべてのコントロールプレーンノードを含めます。

20.3. コンピュートノードの並列アップグレード

多数のコンピュートノードを OpenStack Platform 16.1 にアップグレードする場合は、--**limit Compute** オプションを指定して **openstack overcloud upgrade run** コマンドを実行し、20 ノードのグループを並行して処理することができます。

バックグラウンドで複数のアップグレードタスクを実行できます。この場合、各タスクは 20 ノードの個別のグループをアップグレードします。この方法を使用してコンピュートノードを並行してアップグレードする場合は、アップグレードするノードを選択することはできません。ノードの選択は、**tripleo-ansible-inventory** コマンドの実行時に生成するインベントリーファイルに基づきます。たとえば、デプロイメントに 80 のコンピュートノードがある場合、次のコマンドを実行して、コンピュートノードを並行して更新できます。

```
$ openstack overcloud upgrade run -y --limit 'Compute[0:19]' > upgrade-compute-00-19.log 2>&1 &
$ openstack overcloud upgrade run -y --limit 'Compute[20:29]' > upgrade-compute-20-29.log 2>&1 &
$ openstack overcloud upgrade run -y --limit 'Compute[40:59]' > update-compute-40-59.log 2>&1 &
$ openstack overcloud upgrade run -y --limit 'Compute[60:79]' > update-compute-60-79.log 2>&1 &
```

特定のコンピュートノードをアップグレードするには、ノードのコンマ区切りリストを使用します。

```
$ openstack overcloud upgrade run --limit <Compute0>,<Compute1>,<Compute2>,<Compute3>
```



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、--**stack STACK NAME** オプションを使用します。**STACK NAME** は実際のスタック名に置き換えます。

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** ファイルを取得します。

```
$ source ~/stackrc
```

3. 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インスタンスの移行](#) を参照してください。
4. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[0:19]' > upgrade-compute-00-19.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[20:29]' > upgrade-compute-20-29.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[40:59]' > update-compute-40-59.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[60:79]' > update-compute-60-79.log 2>&1 &
```

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。
5. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[0:19]' >
upgrade-compute-00-19.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[20:29]' >
upgrade-compute-20-29.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[40:59]' >
update-compute-40-59.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[60:79]' >
update-compute-60-79.log 2>&1 &
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。

6. (オプション) 選択したコンピュータノードをアップグレードするには、アップグレードするノードのコンマ区切りリストと共に **--limit** オプションを使用します。以下の例では、**overcloud-compute-0**、**overcloud-compute-1**、**overcloud-compute-2** ノードを並行してアップグレードします。
- a. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

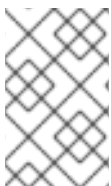
```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

- b. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-
0,overcloud-compute-1,overcloud-compute-2
```

20.4. オーバークラウドスタックの同期

アップグレードにおいては、スタックのリソース構造およびパラメーターが OpenStack Platform 16.1 の新規デプロイメントと整合するように、オーバークラウドスタックを更新する必要があります。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **containers-prepare-parameter.yaml** ファイルを編集し、以下のパラメーターおよびその値を削除します。

- **ceph3_namespace**

- **ceph3_tag**
 - **ceph3_image**
 - **name_prefix_stein**
 - **name_suffix_stein**
 - **namespace_stein**
 - **tag_stein**
3. オーバークラウドでフェンシングを再度有効にするには、**fencing.yaml** 環境ファイルで **EnableFencing** パラメーターを **true** に設定します。
 4. アップグレードの最終処理コマンドを実行します。

```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- 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**)
 - **EnableFencing** パラメーターが **true** に設定された環境ファイル (**fencing.yaml**)。
 - 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (**-e**)
 - 新しいコンテナ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。
 - OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
 - 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** でカスタムロール (**roles_data**) ファイルを指定します。
 - 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
 - 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。
5. スタックの同期が完了するまで待ちます。



重要

これ以降のデプロイメント操作には、**upgrades-environment.yaml** ファイルは必要ありません。

第21章 コントローラーが分割されたオーバークラウドのアップグレード

このシナリオでは、コントローラーノードサービスが複数のノードに分割されているオーバークラウドのアップグレードプロセスの例を説明します。この環境には、以下のノード種別が含まれます。

- Pacemaker を使用する複数に分割された高可用性サービス
- 複数に分割されたコントローラーサービス
- Ceph MON ノード 3 台
- Ceph Storage ノード 3 台
- 複数のコンピュートノード

21.1. オーバークラウドアップグレード準備タスクの実行

アップグレードを行うには、**openstack overcloud upgrade prepare** コマンドを実行する必要があります。このコマンドにより、以下のタスクが実行されます。

- オーバークラウドのプランを OpenStack Platform 16.1 に更新する
- アップグレードに向けてノードを準備する



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **upgrade prepare** コマンドを実行します。

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE \
  ... \
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  ...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- アップグレード固有のパラメーターが含まれる環境ファイル (**upgrades-environment.yaml**) (-e)

- 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)
 - 新しいコンテナ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。
 - OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
 - 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** でカスタムロール (**roles_data**) ファイルを指定します。
 - 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
 - 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。
3. アップグレードの準備が完了するまで待ちます。
 4. コンテナイメージをダウンロードします。

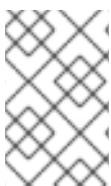
```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

21.2. PACEMAKER ベースのノードのアップグレード

Pacemaker サービスをホストする全ノードを OpenStack Platform 16.1 にアップグレードします。以下のロールに Pacemaker ベースのサービスが含まれます。

- Controller
- Database (MySQL、Galera)
- Messaging (RabbitMQ)
- Load Balancing (HAProxy)
- 以下のサービスが含まれるその他すべてのロール
 - **OS::TripleO::Services::Pacemaker**
 - **OS::TripleO::Services::PacemakerRemote**

このプロセスでは、ブートストラップノードから始めて各ノードをアップグレードします。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

- アンダークラウドノードで以下のコマンドを実行し、ブートストラップノードを特定します。

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

- ブートストラップノードをアップグレードします。

- ノードに Ceph Storage コンテナが含まれていれば、**ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd
-e ceph_ansible_limit=overcloud-controller-0
```

<stack_name> は、実際のスタック名に置き換えます。

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-0
```

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。

- system_upgrade_transfer_data** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
system_upgrade_transfer_data
```

このコマンドにより、最新バージョンのデータベースが既存のノードからブートストラップノードにコピーされます。

- nova_hybrid_state** タグを指定してアップグレードコマンドを実行し、**upgrade_steps_playbook.yaml** Playbook だけを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --playbook
upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
```

このコマンドにより、コンピュータノード上の一時的な 16.1 コンテナが起動します。これにより、後のステップでコンピュータノードをアップグレードする際に、ワークロードの移行が円滑に行われます。

- e. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。

4. Pacemaker ベースの各ノードをアップグレードします。

- a. ノードに Ceph Storage コンテナが含まれていれば、**ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags  
ceph_systemd -e ceph_ansible_limit=overcloud-database-0
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. 次のノードで、**system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit  
overcloud-database-0
```

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。

- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-  
0,overcloud-database-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。このノードに加えて、前のステップでアップグレードしたすべてのノードを **--limit** オプションに含めます。

5. 各 Pacemaker ベースのノードでアップグレードプロセスを繰り返し、すべての Pacemaker ベースのノードをアップグレードします。

21.3. PACEMAKER コントローラーノード以外のアップグレード

Pacemaker ベースのサービスが含まれないノードをすべて OpenStack Platform 16.1 にアップグレードします。これらのノードには、通常特定の OpenStack サービスが含まれます。Pacemaker ベースのサービスが含まれないロールの例は以下のとおりです。

- Networker

- Ironic Conductor
- Object Storage
- 標準のコントローラーノードから分割またはスケーリングしたサービスを持つカスタムロール

このグループには、以下のノードを含めないでください。

- コンピュートノード
- Ceph Storage ノード

このプロセスでは、各ノードのアップグレードを行います。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-networker-0
```

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。
3. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-networker-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。

4. 各ノードでアップグレードプロセスを繰り返し、すべてのコントローラーベースのノードをアップグレードします。

21.4. CEPH MON ノードのオペレーティングシステムのアップグレード

各 Ceph MON ノードのオペレーティングシステムをアップグレードします。ノード間のクォーラムを維持するために、それぞれの Ceph MON ノードを個別にアップグレードすることを推奨します。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. Ceph MON ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags  
ceph_systemd -e ceph_ansible_limit=overcloud-cephmon-0
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit  
overcloud-cephmon-0
```

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。

- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephmon-0
```

このコマンドにより **config-download** Playbook が実行され、Ceph MON ノードでコンポーザブルサービスが設定されます。以下の手順では、Ceph MON ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

3. 次の Ceph MON ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags  
ceph_systemd -e ceph_ansible_limit=overcloud-cephmon-1
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-cephmon-1
```

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。

- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephmon-1
```

このコマンドにより **config-download** Playbook が実行され、Ceph MON ノードでコンポーザブルサービスが設定されます。以下の手順では、Ceph MON ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

4. 最後の Ceph MON ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-cephmon-2
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-cephmon-2
```

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。
- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephmon-2
```

このコマンドにより **config-download** Playbook が実行され、Ceph MON ノードでコンポーザブルサービスが設定されます。以下の手順では、Ceph MON ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

21.5. CEPH STORAGE ノードのオペレーティングシステムのアップグレード

director を使用してデプロイされた Red Hat Ceph Storage クラスターがデプロイメントで使用される場合は、各 Ceph Storage ノードのオペレーティングシステムをアップグレードする必要があります。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. Ceph Storage ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags  
ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-0
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit  
overcloud-cephstorage-0
```

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。
- c. (オプション) Ceph のサブスクリプションを使用し、Ceph ストレージノード用に **overcloud-minimal** イメージを使用するように director を設定している場合は、以下の手順を完了する必要があります。

- i. ノードにログインし、Red Hat Enterprise Linux (RHEL) のマイナーリリースバージョンの設定を解除します。

```
$ sudo subscription-manager release --unset
```

- ii. ノードでシステムの更新を実行します。

```
$ sudo dnf -y update
```

- iii. ノードをリブートします。

```
$ sudo reboot
```

- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-0
```

このコマンドにより **config-download** Playbook が実行され、Ceph Storage ノードでコンポーザブルサービスが設定されます。このステップでは、Ceph Storage ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

3. 次の Ceph Storage ノードを選択し、オペレーティングシステムをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-1
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-cephstorage-1
```

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。
- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-
cephstorage-1
```

このコマンドにより **config-download** Playbook が実行され、Ceph Storage ノードでコンポーザブルサービスが設定されます。このステップでは、Ceph Storage ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

4. 最後の Ceph Storage ノードを選択し、オペレーティングシステムをアップグレードします。
- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-2
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephstorage-2
```

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。
- c. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-
cephstorage-2
```

このコマンドにより **config-download** Playbook が実行され、Ceph Storage ノードでコンポーザブルサービスが設定されます。このステップでは、Ceph Storage ノードは Red Hat Ceph Storage 4 にアップグレードされません。Red Hat Ceph Storage 4 へのアップグレードは、後の手順で実施されます。

21.6. コンピュートノードのアップグレード

すべてのコンピュートノードを OpenStackPlatform 16.1 にアップグレードします。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. インスタンスを移行します。移行計画の詳細は、[コンピュータード間の仮想マシンインスタンスの移行](#) を参照してください。

3. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-compute-0
```

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。

4. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。

5. 複数のコンピュータードを並行してアップグレードするには、**--limit** オプションをアップグレードするノードのコンマ区切りリストに設定します。まず **system_upgrade** タスクを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

続いて、標準の OpenStack サービスのアップグレードを実施します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

21.7. オーバークラウドスタックの同期

アップグレードにおいては、スタックのリソース構造およびパラメーターが OpenStack Platform 16.1 の新規デプロイメントと整合するように、オーバークラウドスタックを更新する必要があります。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **containers-prepare-parameter.yaml** ファイルを編集し、以下のパラメーターおよびその値を削除します。

- **ceph3_namespace**
- **ceph3_tag**
- **ceph3_image**
- **name_prefix_stein**
- **name_suffix_stein**
- **namespace_stein**
- **tag_stein**

3. オーバークラウドでフェンシングを再度有効にするには、**fencing.yaml** 環境ファイルで **EnableFencing** パラメーターを **true** に設定します。

4. アップグレードの最終処理コマンドを実行します。

```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- 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)
- **EnableFencing** パラメーターが **true** に設定された環境ファイル (**fencing.yaml**)。
- 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)
- 新しいコンテナ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。
- OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
- 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** でカスタムロール (**roles_data**) ファイルを指定します。
- 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
- 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。

5. スタックの同期が完了するまで待ちます。



重要

これ以降のデプロイメント操作には、**upgrades-environment.yaml** ファイルは必要ありません。

第22章 ハイパーコンバージドインフラストラクチャーを持つオーバークラウドのアップグレード

このシナリオでは、ハイパーコンバージドインフラストラクチャー (HCI) を持つオーバークラウドのアップグレードプロセスの例を説明します。この環境には、以下のノード種別が含まれます。

- コントローラーノード 3 台
- 複数の HCI コンピュートノード (Compute サービスと Ceph OSD サービスが組み合わされてノードに含まれる)

22.1. オーバークラウドアップグレード準備タスクの実行

アップグレードを行うには、**openstack overcloud upgrade prepare** コマンドを実行する必要があります。このコマンドにより、以下のタスクが実行されます。

- オーバークラウドのプランを OpenStack Platform 16.1 に更新する
- アップグレードに向けてノードを準備する



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **upgrade prepare** コマンドを実行します。

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE \
  ... \
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  ...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- アップグレード固有のパラメーターが含まれる環境ファイル (**upgrades-environment.yaml**) (-e)
- 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)

- 新しいコンテナ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。
 - OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
 - 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** でカスタムロール (**roles_data**) ファイルを指定します。
 - 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
 - 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。
3. アップグレードの準備が完了するまで待ちます。
 4. コンテナイメージをダウンロードします。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

22.2. DIRECTOR でデプロイされた CEPH STORAGE と組み合わせたコントローラーノードのアップグレード

director を使用してデプロイされた Red Hat Ceph Storage クラスタがデプロイメントで使用される場合は、以下の手順を完了する必要があります。

すべてのコントローラーノードを OpenStack Platform 16.1 にアップグレードする場合は、ブートストラップコントローラーノードからアップグレードを始める必要があります。

ブートストラップコントローラーノードのアップグレードプロセス中、新しい Pacemaker クラスタが作成され、新しい Red Hat OpenStack 16.1 コンテナがノードで起動します。一方、残りのコントローラーノードは Red Hat OpenStack 13 で稼働を続けます。

ブートストラップノードをアップグレードしたら、Pacemaker サービスが含まれるその他のノードをアップグレードし、ブートストラップノードで起動した新しい Pacemaker クラスタに各ノードが参加するようにしなければなりません。詳細は、[オーバークラウドノードのアップグレードワークフロー](#)を参照してください。

以下の例では、コントローラーノードの名前はデフォルトの **overcloud-controller-NODEID** 命名規則を使用して名前を付けています。これには、以下の 3 つのコントローラーノードが含まれます。

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**

これらの値は、該当する実際のノード名に置き換えてください。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. アンダークラウドノードで以下のコマンドを実行し、ブートストラップコントローラーノードを特定します。

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

3. ブートストラップ Controller ノードをアップグレードします。

- a. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd  
-e ceph_ansible_limit=overcloud-controller-0
```

<stack_name> は、実際のスタック名に置き換えます。

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したコントローラーノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- b. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit  
overcloud-controller-0
```

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。



重要

次のコマンドにより、コントロールプレーンで機能停止が生じます。これ以降の数ステップを実施している間は、標準的なオーバークラウド操作を行うことはできません。

- c. **system_upgrade_transfer_data** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags  
system_upgrade_transfer_data
```

このコマンドにより、最新バージョンのデータベースが既存のノードからブートストラップノードにコピーされます。

- d. **nova_hybrid_state** タグを指定してアップグレードコマンドを実行し、**upgrade_steps_playbook.yaml** Playbook だけを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --playbook
upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
```

このコマンドにより、コンピュータノード上の一時的な 16.1 コンテナが起動します。これにより、後のステップでコンピュータノードをアップグレードする際に、ワークロードの移行が円滑に行われます。

- e. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。



重要

このコマンドの処理が完了すると、コントロールプレーンがアクティブになります。再び標準的なオーバークラウド操作を実施することができます。

- f. アップグレード後に新しい Pacemaker クラスターが起動していること、および galera、rabbit、haproxy、redis 等のコントロールプレーンサービスが実行中であることを確認します。

```
$ sudo pcs status
```

4. 次のコントローラーノードをアップグレードします。

- a. 古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```

クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
ceph_systemd -e ceph_ansible_limit=overcloud-controller-1
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択したコントローラーノードに制限する。

このコマンドは、[Red Hat OpenStack Platform 16.1 のアップグレードガイド](#)の[アップグレード準備](#)セクションで詳しく説明されています。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- c. 次のコントローラーノードで、**system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-1
```

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。

- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。このノードに加えて、前のステップでアップグレードしたブートストラップノードを **--limit** オプションに含めます。

5. 最後のコントローラーノードをアップグレードします。

- a. 古いクラスターが実行されなくなったことを確認します。

```
$ sudo pcs status
```

クラスターが実行されていない場合、以下のようなエラーが表示されます。

```
Error: cluster is not currently running on this node
```

- b. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansi ble_limit=overcloud-controller-2
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansi ble_limit** 変数を使用して、アクションを選択したコントローラーノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

- c. **system_upgrade** タグを指定してアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-2
```

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。
- d. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0,overcloud-controller-1,overcloud-controller-2
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。--**limit** オプションにすべてのコントローラーノードを含めます。

22.3. ハイパーコンバージドインフラストラクチャー (HCI) を持つコンピュートノードのアップグレード

HCI コンピュートノードを OpenStack Platform 16.1 にアップグレードします。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、--**stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. 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インスタンスの移行](#) を参照してください。
3. Ceph MON サービスが含まれるノードからログアウトし、アンダークラウドに戻ります。
4. **ceph_systemd** タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-computehci-0
```

このコマンドにより、以下の操作が行われます。

- Podman 管理を使用するために、Ceph Storage コンテナを制御する systemd ユニットを変更する。
- **ceph_ansible_limit** 変数を使用して、アクションを選択した Ceph Storage ノードに制限する。

このステップは、**leapp** によるアップグレードに向けて Ceph Storage サービスを準備するための予備的な処置です。

5. **system_upgrade** タグを指定してアップグレードコマンドを実行します。


```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-computehci-0
```

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。

6. タグを指定せずにアップグレードコマンドを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-computehci-0
```

このコマンドにより、Red Hat OpenStack Platform のアップグレードが実施されます。

7. 複数のコンピューターノードを並行してアップグレードするには、`--limit` オプションをアップグレードするノードのコンマ区切りリストに設定します。まず、`ceph_systemd` タグを指定して外部アップグレードコマンドを実行します。

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e
ceph_ansible_limit=overcloud-computehci-0,overcloud-computehci-1,overcloud-computehci-2
```

次に、`system_upgrade` タスクを実行します。

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-computehci-0,overcloud-computehci-1,overcloud-computehci-2
```

続いて、標準の OpenStack サービスのアップグレードを実施します。

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-computehci-0,overcloud-computehci-1,overcloud-computehci-2
```

22.4. オーバークラウドスタックの同期

アップグレードにおいては、スタックのリソース構造およびパラメーターが OpenStack Platform 16.1 の新規デプロイメントと整合するように、オーバークラウドスタックを更新する必要があります。



注記

デフォルトのスタック名 (`overcloud`) を使用していない場合は、`--stack STACK NAME` オプションでスタック名を設定します。`STACK NAME` は実際のスタック名に置き換えます。

手順

1. `stackrc` ファイルを取得します。

```
$ source ~/stackrc
```

2. `containers-prepare-parameter.yaml` ファイルを編集し、以下のパラメーターおよびその値を削除します。

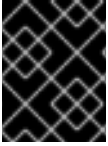
- `ceph? nameosppc`

- ▼ **ceph3_namespace**
 - **ceph3_tag**
 - **ceph3_image**
 - **name_prefix_stein**
 - **name_suffix_stein**
 - **namespace_stein**
 - **tag_stein**
3. オーバークラウドでフェンシングを再度有効にするには、**fencing.yaml** 環境ファイルで **EnableFencing** パラメーターを **true** に設定します。
 4. アップグレードの最終処理コマンドを実行します。

```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
...
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
...
```

以下のオプションの中で、お使いの環境に適切なオプションを追加します。

- 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)
 - **EnableFencing** パラメーターが **true** に設定された環境ファイル (**fencing.yaml**)。
 - 登録およびサブスクリプションに関するパラメーターが含まれる環境ファイル (**rhsm.yaml**) (-e)
 - 新しいコンテナ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。
 - OVS との互換性を維持するための環境ファイル (**neutron-ovs.yaml**)
 - 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** でカスタムロール (**roles_data**) ファイルを指定します。
 - 該当する場合は、**--networks-file** でコンポーザブルネットワーク (**network_data**) ファイルを指定します。
 - 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。
5. スタックの同期が完了するまで待ちます。



重要

これ以降のデプロイメント操作には、**upgrades-environment.yaml** ファイルは必要ありません。

第23章 DIRECTOR でデプロイされた CEPH STORAGE クラスターの RED HAT CEPH STORAGE 4 へのアップグレード

director を使用してデプロイされた Red Hat Ceph Storage クラスターがデプロイメントで使用される場合は、本項に記載する手順を完了する必要があります。

重要

Red Hat Ceph Storage クラスターを、以前のサポート対象バージョンからバージョン 4.2z2 にアップグレードする場合、**モニターがセキュアでない global_id 再要求を許可**することを示す警告メッセージと共に、ストレージクラスターが **HEALTH_WARN** の状態でアップグレードが完了します。これは、パッチが適用された CVE (CVE-2021-20288) が原因です。[Ceph HEALTH_WARN with 'mons are allowing insecure global_id reclaim' after install/upgrade to RHCS 4.2z2 \(or newer\)](#) を参照してください。

CVE により **HEALTH_WARN** の状態が表示されるため、一時的に健全性についての警告をミュートすることができます。ただし、警告をミュートすると、古いクライアントおよびパッチが適用されていないクライアントがクラスターに接続されても表示されないリスクがあります。健全性についての警告のミュートについての詳細は、[Red Hat Ceph Storage](#) のドキュメントの [Upgrading a Red Hat Ceph Storage cluster](#) を参照してください。

重要

すべてのクライアントがアップグレードしてパッチを当てたまで **global_id_reclaim** は、無効にしないでください。そうでないと、接続できません。**root** ユーザーで以下のコマンドを実行して、クラスターに接続されているクライアントでパッチが当てられていないものをリストで表示できます。

```
# ceph health detail
```

オーバークラウドをアップグレードしたら、director でデプロイされた Ceph Storage クラスターを Red Hat Ceph Storage クラスターバージョン 4 にアップグレードします。

23.1. CEPH-ANSIBLE のインストール

director を使用してデプロイされた Red Hat Ceph Storage クラスターがデプロイメントで使用される場合は、以下の手順を完了する必要があります。

Red Hat OpenStack Platform で Ceph Storage を使用する場合、**ceph-ansible** パッケージが必要です。

手順

1. Ceph Tools リポジトリを有効にします。

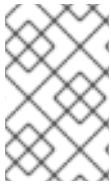
```
[stack@director ~]$ sudo subscription-manager repos --enable=rhceph-4-tools-for-rhel-8-x86_64-rpms
```

2. **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
```

23.2. CEPH STORAGE 4 へのアップグレード

Ceph Storage ノードをバージョン 3 からバージョン 4 にアップグレードします。



注記

デフォルトのスタック名 (**overcloud**) を使用していない場合は、**--stack STACK NAME** オプションでスタック名を設定します。**STACK NAME** は実際のスタック名に置き換えます。

手順

1. **stackrc** ファイルを取得します。

```
$ source ~/stackrc
```

2. **ceph** タグを指定して、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 --stack STACK NAME --tags ceph
```

3. Ceph Storage のアップグレードが完了するまで待ちます。

第24章 FILESTORE から BLUESTORE への OSD の移行

アップグレードプロセスを完了して検証を行ったら、FileStore OSD を BlueStore に移行する必要があります。1台ずつノードの移行作業を完了していく必要があります。以下の手順では、**ceph-ansible** を使用して移行を完了します。以下の手順は、Ceph クラスタが director によりデプロイされている場合にのみ適用されます。

24.1. クラスタが FILESTORE を実行している (したがって移行が必要である) ことの確認

手順

1. コントローラーノードやスタンドアロンの Ceph MON ノードなど、Ceph MON コンテナを持つノードに **heat-admin** ユーザーとしてログインします。たとえば、標準のオーバークラウドデプロイメントでは、overcloud **-controller-1** は Ceph MON コンテナを使用します。
2. OSD が使用しているドライバーを確認するために、Ceph クラスタに対してクエリーを行います。

```
[heat-admin@overcloud-controller-1 ~]$ sudo -i
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph -f json osd metadata" | jq -c 'sort_by(.hostname) | .[] | [{"host", .hostname, "osd_id", .id,
"objectstore", .osd_objectstore}]'
[root@overcloud-controller-1 ~]#
```

3. いずれかの行が **"objectstore": "filestore"** を返す場合、そのノードには OSD の移行が必要です。



警告

移行の時間はクラスタのサイズによって異なります。非常に大きなクラスタの場合、移行時間はそのクラスタ内の OSD の数および保存されるデータ量に比例します。お使いの環境が混合アーキテクチャーのシナリオにならないように、できるだけ早く移行を完了してください。混合アーキテクチャーのシナリオでは、パフォーマンスに影響が及ぶ可能性があります。



警告

Red Hat Ceph Storage (RHCS) 4 バージョンの **ceph-ansible** で FileStore ベースの OSD を管理する設定はサポートされていないため、スタックの更新を実行する前に移行を完了してください。

24.2. FILESTORE から BLUESTORE への OSD の移行

FileStore から BlueStore に移行するのに、director は Ansible を使用してノード上のすべての OSD を削除して再作成します。director は、移行プロセスの前に容量の確認を行います。最後に、director は BlueStore バックエンドを使用して OSD を再デプロイします。

前提条件

- 正常な稼働中の Red Hat Ceph Storage (RHCS) 4 クラスター。コントローラーノードまたはスタンドアロンの Ceph MON ノード上の ceph MON コンテナにおいて、以下のコマンドを入力してクラスターを確認することができます。

```
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c "ceph -s"
```

手順

1. **CephAnsibleDisksConfig** パラメーターの **osd_objectstore** が **filestore** にデフォルト設定されないようにします。いずれかのカスタム heat 環境ファイルに **osd_objectstore** パラメーターが存在する場合は、**bluestore** の値を明示的に定義する必要があります。

```
parameter_defaults:
  CephAnsibleDisksConfig:
    devices:
      - /dev/sdb
      - /dev/sdc
      - /dev/sdd
    osd_scenario: lvm
    osd_objectstore: bluestore
```



注記

ジャーナル等に対する特定の FileStore 設定がある場合は、適切に設定を更新するようにしてください。高度な設定についての詳しい情報は、[コンテナ化された Red Hat Ceph を持つオーバークラウドのデプロイの Ceph Storage ノードのディスクレイアウトのマッピング](#) を参照してください。

2. アンダークラウドに **stack** ユーザーとしてログインします。
3. **ceph_fstobs** タグを指定して **openstack overcloud external-upgrade run** コマンドを入力します。<NODE_NAME> は、アップグレードする Ceph OSD ノードの名前に置き換えてください。 **openstack server list** コマンドを使用してノード名を探することができます。

```
[stack@undercloud ~] $ openstack overcloud external-upgrade run --tags ceph_fstobs -e ceph_ansible_limit=<NODE_NAME> | tee oc-fstobs.log
```

4. Ceph MON サービスが実行されているノードにログインし、Ceph クラスターにクエリーを行いアップグレードしたノードの OSD のステータスを確認します。次の OSD ノードの移行を開始する前に、アップグレードしたノードが正常に移行されていることを確認する必要があります。

```
[heat-admin@overcloud-controller-1 ~]$ sudo -i
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c "ceph -f json osd metadata" | jq -c '.[] | select(.hostname == "<NODE_NAME>") | [{"host",
```

```
.hostname, "osd_id", .id, "objectstore", .osd_objectstore]'
[root@overcloud-controller-1 ~]# exit
```

<NODE_NAME> を移行したノードの名前に置き換えます。出力結果を確認し、OSD が BlueStore を使用することが示されていれば、移行は正常に完了しています。

- (オプション) 特定の OSD に関する追加情報を表示するには、以下のコマンドを入力します。

```
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph osd metadata <ID>"
```

<ID> はクエリーを行う OSD の ID に置き換えてください。

- 次のノードで移行プロセスを開始する前に、クラスターが同期するのを待つ必要があります。

```
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph -s"
```

Review the command output and ensure that the health of the cluster is `HEALTH_OK` and the PGs are in the `active+clean` state.

- 次のノードで移行プロセスを開始する前に、クラスターのリバランスプロセスが完了するまで待つ必要があります。ステータスを確認するには、以下のコマンドを実行します。

```
[heat-admin@overcloud-controller-0 ~]$ sudo podman exec ceph-mon-<NODE_NAME>
ceph -w
```

<NODE_NAME> を移行したノードの名前に置き換えます。

- ストレージクラスター内の各ノードに対して移行プロセスを繰り返します。

FileStore から BlueStore への移行に関する詳細は、Red Hat Ceph Storage の [管理ガイド](#) の [BlueStore](#) を参照してください。

24.3. FILESTORE から BLUESTORE への移行の確認

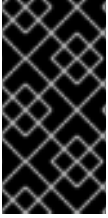
OSD のステータスを確認して、移行が正常に完了していることを確認することができます。

手順

- いずれかのコントローラーノードがホストする ceph-mon コンテナに **heat-admin** ユーザーとしてログインします。
- Ceph クラスターに対してクエリーを行います。

```
[heat-admin@overcloud-controller-1 ~]$ sudo -i
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph -f json osd metadata" | jq -c 'sort_by(.hostname) | .[] | [{"host", .hostname, "osd_id", .id,
"objectstore", .osd_objectstore}]'
[root@overcloud-controller-1 ~]#
```

設定を確認し、クラスター全体のすべての OSD が BlueStore を使用することが示されていれば、移行は正常に完了しています。



重要

推奨されるベストプラクティスは、べき等性を持つスタック更新を実行し、設定の定義と実際の設定が一致するようにすることです。スタック更新の時間はシステムのサイズによって異なります。したがって、ダウンタイムを短縮するため、メンテナンス期間中に移行を完了するよう計画することができます。

第25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。

25.1. アンダークラウドからの不要なパッケージおよびディレクトリーの削除

Leapp によるアップグレード後に、アンダークラウドに残っている不要なパッケージおよびディレクトリーを削除します。

手順

1. 不要なパッケージを削除します。

```
$ sudo dnf -y remove --exclude=python-pycadf-common python2*
```

2. Red Hat OpenStack 13 で使用されていた古いイメージを含む `/httpboot` および `/tftpboot` ディレクトリーからコンテンツを削除します。

```
$ sudo rm -rf /httpboot /tftpboot
```

25.2. アップグレード後の機能検証

`post-upgrade` 検証グループを実行し、アップグレード後の機能を確認します。

手順

1. `source` コマンドで `stackrc` ファイルを読み込みます。

```
$ source ~/stackrc
```

2. `--group post-upgrade` オプションを指定して、`openstack tripleo validator run` コマンドを実行します。

```
$ openstack tripleo validator run --group post-upgrade
```

3. 検証レポートの結果を確認します。特定の検証からの詳細出力を表示するには、レポートからの特定検証の UUID を指定して `openstack tripleo validator show run --full` コマンドを実行します。

```
$ openstack tripleo validator show run --full <UUID>
```



重要

検証結果が **FAILED** であっても、Red Hat OpenStack Platform のデプロイや実行が妨げられることはありません。ただし、**FAILED** の検証結果は、実稼働環境で問題が発生する可能性があることを意味します。

25.3. 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 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ますようになります。

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. 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. オーバークラウドの QCOW2 アーカイブが含まれるパッケージをインストールします。

```
$ sudo dnf install rhosp-director-images rhosp-director-images-ipa
```

4. **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/*
```

5. 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-16.1.tar /usr/share/rhosp-director-images/ironic-python-agent-latest-16.1.tar; do tar -xvf $i; done
$ cd ~
```

6. director に最新のイメージをインポートします。

```
$ openstack overcloud image upload --update-existing --image-path /home/stack/images/
```

7. ノードが新しいイメージを使用するように設定します。

```
$ openstack overcloud node configure $(openstack baremetal node list -c UUID -f value)
```



重要

オーバークラウドノードをデプロイする際には、オーバークラウドイメージのバージョンが該当する heat テンプレートバージョンに対応している状態にします。たとえば、OpenStack Platform 16.1 のイメージは、OpenStack Platform 16.1 の heat テンプレートだけに使用してください。



重要

新しい **overcloud-full** イメージは、古い **overcloud-full** イメージを置き換えます。古いイメージに変更を加えた場合、特に今後新規ノードをデプロイする場合は、新しいイメージで変更を繰り返す必要があります。

25.4. CPU ピニングパラメーターの更新

Red Hat OpenStack Platform 16.1 では、CPU ピニングに新たなパラメーターが使用されます。

NovaComputeCpuDedicatedSet

専用の (ピンングされた) CPU を設定します。

NovaComputeCpuSharedSet

共有の (ピンングされていない) CPU を設定します。

Red Hat OpenStack Platform 16.1 へのアップグレードが完了したら、CPU ピニングの設定を **NovaVcpuPinSet** パラメーターから **NovaComputeCpuDedicatedSet** と **NovaComputeCpuSharedSet** のパラメーターに移行する必要があります。

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. コンピュート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=isolate** 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** スレッドポリシーの設定を解除し、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. フレーバーの **hw:cpu_thread_policy** プロパティの設定を解除します。

```
(overcloud) $ openstack flavor unset --property hw:cpu_thread_policy <flavor>
```

注記

- **hw:cpu_thread_policy** 属性の設定を解除すると、ポリシーがデフォルトの **prefer** ポリシーに設定されます。これにより、インスタンスは SMT 対応のコンピュートノードを使用するように設定されます (利用可能な場合)。**hw:cpu_thread_policy** 属性を **require** に設定することもできます。これにより、SMT 対応のコンピュートノードに対するハード要件が設定されます。
- コンピュートノードに SMT アーキテクチャーがない場合や、スレッドシブリングが利用可能な CPU コアが十分でない場合には、スケジューリングが失敗します。これを回避するには、**hw:cpu_thread_policy** を **require** ではなく **prefer** に設定します。デフォルトの **prefer** ポリシーでは、スレッドシブリングが利用可能な場合には、必ず使用されます。
- **hw:cpu_thread_policy=isolate** を使用する場合は、SMT を無効にするか、SMT をサポートしないプラットフォームを使用する必要があります。

- iii. 新しいスレッドポリシ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>
```

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。

- コンピュートノードからインスタンスを移行して、コンピュートノードの 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. コンピュート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. コンピュート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についての詳細は、[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. コンピュートノードをリブートし、コンピュートノードの BIOS で SMT を無効にします。
- v. コンピュートノードをブートします。
- vi. コンピュート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。
 - これまでピン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。
 - これまでピン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、すべてのコンピュートノードのコアを **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ン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
```

...

設定をピン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: ""
...
```

設定をピン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ンング設定のコンピュータノードが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 ピン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

- [コンピュータノードでの CPU ピンングの設定](#)

25.5. メンバーロールへのユーザーの移行

Red Hat OpenStack Platform 13 では、デフォルトのメンバーロールは **_member_** と呼ばれています。Red Hat OpenStack Platform 16.1 では、デフォルトのメンバーロールは **member** と呼ばれています。

Red Hat OpenStack Platform 13 から Red Hat OpenStack Platform 16.1 へのアップグレードを完了しても、**_member_** ロールに割り当てたユーザーにはそのロールが残っています。以下の手順で、すべてのユーザーを **member** ロールに移行することができます。

前提条件

- オーバ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. **_member_**ロールを持つクラウド上のすべてのユーザーをリストアップします。

```
openstack role assignment list --names --role _member_ --sort-column project
```

2. 各ユーザーに対して、**_member_**ロールを削除し、**member**ロールを適用します。

```
openstack role remove --user <user> --project <project> _member_  
openstack role add --user <user> --project <project> member
```

第26章 アップグレードに関する問題のトラブルシューティング

アップグレードプロセス中に問題が発生した場合は、本セクションのアドバイスを参照してください。

26.1. 環境ファイルの修正

カスタム環境ファイルのパラメーターを誤って設定していた場合は、環境ファイルを修正して、アップグレード中いつでも **openstack overcloud upgrade prepare** コマンドを実行することができます。このコマンドにより、新しいバージョンのオーバークラウドプランが director にアップロードされ、これにより **config-download** Playbook の新しいセットが生成されます。

upgrades-environment.yaml ファイルのリポジトリ名が間違っている例を以下に示します。

```
parameter_defaults:
  UpgradeLeappEnabled: true
  UpgradeLeappCommandOptions: "--enablerepo rhel-7-for-x86_64-baseos-eus-rpms --enablerepo
rhel-8-for-x86_64-appstream-eus-rpms --enablerepo fast-datapath-for-rhel-8-x86_64-rpms"
  CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms
```

この間違いにより、コントローラーノードの Leapp アップグレード時に問題が発生します。この問題を正すには、間違いを修正して **openstack overcloud upgrade prepare** コマンドを実行します。

手順

1. ファイルの間違いを修正します。

```
parameter_defaults:
  UpgradeLeappEnabled: true
  UpgradeLeappCommandOptions: "--enablerepo rhel-8-for-x86_64-baseos-tus-rpms --
enablerepo rhel-8-for-x86_64-appstream-tus-rpms --enablerepo fast-datapath-for-rhel-8-
x86_64-rpms"
  CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms
```

2. 修正したファイルでアップグレードの準備コマンドを実行します。

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
...
-e /home/stack/templates/upgrades-environment.yaml \
...
```

オーバークラウドスタックの更新が完了するまで待ちます。

3. 失敗したアップグレードのステップから操作を続行します。