Red Hat OpenStack Platform 9
Red Hat OpenStack Platform のアップグレード

Red Hat OpenStack Platform 環境のアップグレード

OpenStack Team

Red Hat OpenStack Platform 9 Red Hat OpenStack Platform のアップグレード

Red Hat OpenStack Platform 環境のアップグレード

OpenStack Team
rhos-docs@redhat.com

法律上の通知

概要

本ドキュメントでは、Red Hat OpenStack Platform 8 (Liberty) から 9 (Mitaka) にアップグレードする複数の異なる方法について説明します。これらの方法は、アップグレード元およびアップグレード先のいずれも Red Hat Enterprise Linux 7 をベースにインストールされたデプロイメントであることを前提としています。

目次

第1章 はじめに

本ガイドは、Red Hat OpenStack Platform を最新の状態に保つためのプロセスについて説明します。アップグレードおよび更新は、**Red Hat OpenStack Platform 9 (Mitaka)** をターゲットとします。

Red Hat は、Red Hat Enterprise Linux 7 をベースとする Red Hat OpenStack Platform 9 へのアップグレードのみをサポートし、以下のいずれかの条件に基づいた異なるシナリオを推奨しています。

- director ベースのオーバークラウドまたは手動で作成した環境を使用している。
- 1 クラスター内で複数のコントローラーノードを管理する高可用性ツールを使用している。

「アップグレードシナリオの比較」には、すべてのアップグレードシナリオについての説明を記載しています。これらのシナリオにより、正常に機能する Red Hat OpenStack Platform 9 へのアップグレードと、バージョン 9 内でのマイナーな更新が可能となります。

1.1. アップグレードシナリオの比較

Red Hat では、Red Hat OpenStack Platform 9 には以下のアップグレードシナリオを推奨しています。以下の表には、各シナリオについての説明をまとめています。

表1.1 アップグレードシナリオ

メソッド	説明
director ベースの環境: マイナーバージョンへの更新の実行	このシナリオでは、Red Hat OpenStack Platform 9 のマイナーバージョン間の更新を行います。これには、director パッケージを更新してから、director を使用してオーバークラウド内の全ノードでパッケージの更新を起動するステップを伴います。
director ベースの環境: メジャーバージョンへのアップグレードの実行	このシナリオでは、Red Hat OpenStack Platform のメジャーバージョン間のアップグレードを行います。この場合は、バージョン 8 から 9 にアップグレードする手順です。これには、director のパッケージを更新してから、director を使用して各ノードにアップグレードスクリプトのセットを提供して、オーバークラウドスタックのアップグレードを実行するステップを伴います。
director を使用しない環境: OpenStack サービスの同時アップグレード	このシナリオでは、director を使用せずに管理している Red Hat OpenStack Platform 環境 (手動で作成した環境) の全パッケージをアップグレードします。このシナリオでは、全パッケージを同時にアップグレードします。

メソッド	説明
director を使用しない環境: 標準的な環境内の個別の OpenStack サービス (稼働中の Compute) のアップグレード	このシナリオでは、director を使用せずに管理している Red Hat OpenStack Platform 環境 (手動で作成した環境) の全パッケージをアップグレードします。このシナリオでは、各 OpenStack サービスを個別に更新します。
director を使用しない環境: 高可用性環境における個別の OpenStack サービス (稼働中の Compute) のアップグレード	このシナリオでは、director を使用せずに管理し、コントローラーベースの OpenStack サービス向けに高可用性ツールを使用している Red Hat OpenStack Platform 環境 (手動で作成した環境) の全パッケージをアップグレードします。このシナリオでは、各 OpenStack サービスを個別に更新します。

すべての方法に共通する注意事項

- 全ホスト上でこのリリースの正しいリポジトリーが有効化されていることを確認します。
- アップグレード時には、一部のサービスを停止する必要があります。
- コンピュートノードを再起動したり、インスタンスを明示的にシャットダウンしたりしない限りは、アップグレードプロセスは、実行中のインスタンスには影響を及ぼしません。

警告

Red Hat は、Red Hat OpenStack Platform のベータリリースからサポート対象リリースへのアップグレードは一切サポートしていません。

1.2. リポジトリーの要件

アンダークラウドおよびオーバークラウドにはいずれも、Red Hat コンテンツ配信ネットワーク (CDN) か Red Hat Satellite 5 または 6 を使用した Red Hat リポジトリーへのアクセスが必要です。Red Hat Satellite サーバーを使用する場合は、必要なリポジトリーをお使いの OpenStack Platform 環境に同期します。以下の CDN チャネル名一覧を参考にしてください。

表1.2 OpenStack Platform リポジトリー

名前	リポジトリー	説明

Red Hat Enterprise Linux 7 Server (RPMS)	**rhel-7-server-rpms**	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 7 Server - Extras (RPMs)	**rhel-7-server-extras-rpms**	Red Hat OpenStack Platform の依存関係が含まれます。
Red Hat Enterprise Linux 7 Server - RH Common (RPMs)	**rhel-7-server-rh-common-rpms**	Red Hat OpenStack Platform のデプロイと設定ツールが含まれます。
Red Hat Satellite Tools for RHEL 7 Server RPMs x86_64	**rhel-7-server-satellite-tools-6.1-rpms**	Red Hat Satellite 6 でのホスト管理ツール
Red Hat Enterprise Linux High Availability (for RHEL 7 Server) (RPMs)	**rhel-ha-for-rhel-7-server-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 OpenStack Platform 9 director for RHEL 7 (RPMs)	**rhel-7-server-openstack-9-director-rpms**	Red Hat OpenStack Platform director のリポジトリー。director でデプロイしたオーバークラウド用のツールも一部提供します。
Red Hat OpenStack Platform 9 for RHEL 7 (RPMs)	**rhel-7-server-openstack-9-rpms**	コアとなる Red Hat OpenStack Platform リポジトリー

Ceph クラスターを使用している場合には、以下のリポジトリーも必要となります。

Red Hat Ceph Storage OSD 1.3 for Red Hat Enterprise Linux 7 Server (RPMs)	**rhel-7-server-rhceph-1.3-osd-rpms**	(Ceph Storage ノード向け) Ceph Storage Object Storage デーモンのリポジトリー。Ceph Storage ノードにインストールします。

Red Hat Ceph Storage MON 1.3 for Red Hat Enterprise Linux 7 Server (RPMs)	**rhel-7-server-rhceph-1.3-mon-rpms**	(Ceph Storage ノード向け) Ceph Storage Monitor デーモンのリポジトリー。Ceph Storage ノードを使用して OpenStack 環境にあるコントローラーノードにインストールします。

注記

ネットワークがオフラインの Red Hat OpenStack Platform 環境向けにリポジトリーを設定するには、「オフライン環境で Red Hat OpenStack Platform Director を設定する」の記事を参照してください。

第2章 DIRECTOR ベースの環境: マイナーバージョンへの更新の実行

本項では、同じバージョンの Red Hat OpenStack Platform 環境の更新方法について考察します。この場合は、Red Hat OpenStack Platform 9 が対象です。これには、アンダークラウドとオーバークラウドの両方の機能の更新が含まれます。

警告

インスタンスの高可用性を実装している場合には、アップグレードやスケールアップはできないので (『コンピュートインスタンスの高可用性』で説明しています)、操作を試みても失敗してしまいます。

また、インスタンスの高可用性を有効にすると、将来、director を使用したアップグレードはできなくなります。これは、メジャーアップグレードとマイナーアップグレードの両方に該当します。詳しくは、https://access.redhat.com/ja/solutions/2898841 を参照してください。

いずれの場合の手順でも、以下のワークフローを実行していきます。

1. Red Hat OpenStack Platform director パッケージの更新
2. Red Hat OpenStack Platform director でのオーバークラウドイメージの更新
3. Red Hat OpenStack Platform director を使用したオーバークラウドパッケージの更新

2.1. DIRECTOR パッケージの更新

director は、標準の RPM メソッドを使用して環境を更新します。このメソッドでは、**yum** コマンドを実行して、director のホストが確実に最新のパッケージを使用するようにします。

1. director に **stack** ユーザーとしてログインします。
2. 主要な OpenStack Platform サービスを停止します。

```
$ sudo systemctl stop 'openstack-*' 'neutron-*' httpd
```

注記

これにより、アンダークラウドで短時間のダウンタイムが生じます。アンダークラウドの更新中もオーバークラウドは引き続き機能します。

3. **python-tripleoclient** パッケージと依存関係を更新し、マイナーバージョンの更新向けの最新のスクリプトを使用できるようにします。

```
$ sudo yum update python-tripleoclient
```

4. director は **openstack undercloud upgrade** コマンドを使用して、アンダークラウドの環境を更新します。以下のコマンドを実行します。

```
$ openstack undercloud upgrade
```

アンダークラウドのオペレーティングシステムが Red Hat Enterprise Linux 7.2 から 7.3 に更新された場合や、Open vSwitch のバージョンが 2.4 から 2.5 に更新された場合などで、カーネルまたは Open vSwitch のメジャー/マイナーバージョンが更新された後には、リブートが必要となります。director ノードの **/var/log/yum.log** ファイルをチェックして、**kernel** または **openvswitch** のパッケージのメジャーバージョンまたはマイナーバージョンが更新されているかどうかを確認し、更新されている場合には各ノードのリブートを実行します。

1. ノードを再起動します。

```
$ sudo reboot
```

2. ノードが起動するまで待ちます。

ノードが起動したら、全サービスのステータスを確認します。

```
$ sudo systemctl list-units "openstack*" "neutron*" "openvswitch*"
```

オーバクラウドとそのノードが存在しているかどうかを確認します。

```
$ source ~/stackrc
$ openstack server list
$ openstack baremetal node list
$ openstack stack list
```

2.2. オーバークラウドイメージの更新

アンダークラウドの更新プロセスにより、**rhosp-director-images** および **rhosp-director-images-ipa** パッケージから新規イメージアーカイブがダウンロードされる可能性があります。**yum** ログをチェックして、新規イメージアーカイブが利用可能かどうかを確認してください。

```
$ sudo grep "rhosp-director-images" /var/log/yum.log
```

新規アーカイブが利用可能な場合には、現在のイメージを新規イメージに置き換えてください。新しいイメージをインストールするには、最初に **stack** ユーザーの **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/*
```

アーカイブを展開します。

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

最新のイメージを director にインポートして、ノードがこれらの新規イメージを使用するように設定します。

```
$ openstack overcloud image upload --update-existing --image-path
~/images/.
$ openstack baremetal configure boot
```

新規イメージの存在をチェックして、イメージの更新を最終確認します。

```
$ openstack image list
$ ls -l /httpboot
```

director が更新され、最新のイメージを使用するようになりました。この更新の後にはサービスを再起動する必要はありません。

2.3. オーバークラウドパッケージの更新

オーバークラウドでは、環境の更新に標準の RPM メソッドを使用します。このメソッドでは、director から **openstack overcloud update** を使用して全ノードを更新します。

-e を使用して、オーバークラウドと関連のある環境ファイルとアップグレードパスを追加します。後で実行される環境ファイルで定義されているパラメーターとリソースが優先されることになるため、環境ファイルの順序は重要となります。以下の一覧は、環境ファイルの順序の例です。

- Heat テンプレートコレクションからの **overcloud-resource-registry-puppet.yaml** ファイル。このファイルは、**openstack overcloud deploy** コマンドを実行すると自動的に追加されますが、 **openstack overcloud update コマンドの実行時にはこのファイルを指定する必要があります**。
- Heat テンプレートコレクションの初期化ファイル (**environments/network-isolation.yaml**) を含むネットワーク分離ファイルと、次にカスタムの NIC 設定ファイル
- 外部のロードバランシングの環境ファイル
- ストレージの環境ファイル
- Red Hat CDN または Satellite 登録用の環境ファイル
- その他のカスタム環境ファイル

全ノードで並行して更新を実行すると問題が発生する可能性があります。たとえば、パッケージの更新には、サービスの再起動が必要となる場合があり、その操作によって他のノードが中断される可能性があります。そのため、この更新プロセスでは、一連のブレークポイントを設けて、ノードごとに更新します。1 つのノードでパッケージの更新が完了すると、更新プロセスは次のノードに移ります。更新のプロセスには、各ブレークポイントで確認が要求される対話モードでコマンドを実行するための **-i** オプションも必要です。**-i** オプションを使用しない場合には、更新は最初のブレークポイントで一時停止の状態のままとなります。

オーバークラウドを更新するコマンドの例を以下に示します。

```
$ openstack overcloud update stack overcloud -i \
  --templates  \
  -e /usr/share/openstack-tripleo-heat-templates/overcloud-resource-
registry-puppet.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /home/stack/templates/network-environment.yaml \
```

```
  -e /home/stack/templates/storage-environment.yaml \
  -e /home/stack/templates/rhel-registration/environment-rhel-
registration.yaml \
  [-e <environment_file>|...]
```

このコマンドを実行すると更新のプロセスが開始します。このプロセス中に、director は **IN_PROGRESS** のステータスを報告して、ブレークポイントを通過するように定期的に要求します。以下に例を示します。

```
not_started: [u'overcloud-controller-0', u'overcloud-controller-1',
u'overcloud-controller-2']
on_breakpoint: [u'overcloud-compute-0']
Breakpoint reached, continue? Regexp or Enter=proceed, no=cancel
update, C-c=quit interactive mode:
```

Enter を押すと、**on_breakpoint** 一覧の最後のノードからブレークポイントを通過します。これで、そのノードの更新が開始します。また、ノード名を入力して特定のノードでブレークポイントを通過したり、複数のノードで一度にブレークポイントを通過するための Python ベースの正規表現を入力することも可能です。ただし、複数のコントローラーノードで同時にブレークポイントを通過することはお勧めしません。全ノードが更新を完了するまで、このプロセスを継続します。

更新が完了すると、コマンドにより **COMPLETE** のステータスが報告されます。

```
...
IN_PROGRESS
IN_PROGRESS
IN_PROGRESS
COMPLETE
update finished with status COMPLETE
```

コントローラーノードにフェンシングを設定している場合には、更新プロセスによってその設定が無効になる場合があります。更新プロセスの完了時には、コントローラーノードの 1 つで以下のコマンドを実行してフェンシングを再度有効にします。

```
$ sudo pcs property set stonith-enabled=true
```

更新プロセスを実行しても、オーバークラウド内のノードは自動的には再起動しません。カーネルまたは Open vSwitch のメジャー/マイナーバージョンが更新された場合 (例: オーバークラウドのオペレーティングシステムが Red Hat Enterprise Linux 7.2 から 7.3 に更新された場合や、Open vSwitch がバージョン 2.4 から 2.5 に更新された場合など) には再起動が必要です。各ノードの **/var/log/yum.log** ファイルをチェックして、**kernel** または **openvswitch** のパッケージのメジャー/マイナーバージョンが更新されているかどうかを確認します。更新されている場合には、**『Director Installation and Usage』**ガイドの「Rebooting the Overcloud」の手順に従って各ノードを再起動します。

第3章 DIRECTOR ベースの環境: メジャーバージョンへのアップグレードの実行

警告

最新のメジャーバージョンにアップグレードする前に、アンダークラウドとオーバークラウドが最新のマイナーバージョンに更新されていることを確認してください。これには、OpenStack Platform サービスとベースオペレーティングシステムの両方が含まれます。マイナーバージョンを更新するプロセスについては、Red Hat OpenStack Platform 8『**Red Hat OpenStack Platform のアップグレード**』の「director ベースの環境: マイナーバージョンへの更新の実行」の章を参照してください。先にマイナーバージョンを更新せずに、メジャーバージョンをアップグレードすると、アップグレードプロセスが失敗してしまう可能性があります。

警告

インスタンスの高可用性を実装している場合には、アップグレードやスケールアップはできないので (『コンピュートインスタンスの高可用性』で説明しています)、操作を試みても失敗してしまいます。

また、インスタンスの高可用性を有効にすると、将来、director を使用したアップグレードはできなくなります。これは、メジャーアップグレードとマイナーアップグレードの両方に該当します。詳しくは、https://access.redhat.com/ja/solutions/2898841 を参照してください。

本章では、環境のアップグレードの方法を見て行きます。ここでは、アンダークラウドとオーバークラウド両方のアップグレードプロセスを説明します。このアップグレードプロセスでは、次のメジャーバージョンに移行する手段を提供します。今回は、Red Hat OpenStack Platform 8 から Red Hat OpenStack Platform 9 へのアップグレードです。

いずれの場合の手順でも、以下のワークフローを実行していきます。

1. Red Hat OpenStack Platform director パッケージの更新
2. Red Hat OpenStack Platform director でのオーバークラウドイメージの更新
3. Red Hat OpenStack Platform director を使用したオーバークラウドスタックとパッケージの更新

重要

バージョンのアップグレードを行う前に、「アップグレード前の重要な注記」の情報を必ず一読してください。

3.1. アップグレード前の重要な注記

環境をアップグレードする前に、以下の注記を必ず一読してください。

- Red Hat OpenStack Platform director でのアップグレードは、ライブの実稼働環境で実行する前に、その環境固有の設定で全面的にテストする必要があります。Red Hat では、director を使用する場合の標準のオプションとして提供されているユースケースの大半とそれらの組み合わせを検証済みですが、可能な組み合わせの数が多いため、それらを完全に網羅することはできません。さらに、標準デプロイメントの設定が変更された場合には、手動または設定後のフックを使用して、実稼働用以外の環境でアップグレード機能をテストすることが極めて重要です。そのため、以下を実行することを推奨します。
  - アンダークラウドノードのバックアップを実行してから、アップグレードの手順のステップを開始します。バックアップの手順は、『director のアンダークラウドのバックアップと復元』ガイドを参照してください。
  - 実稼働環境で手順を実行する前に、すべての変更が加えられたテスト環境でアップグレードの手順を実行します。
  - このアップグレードの実行に関して何らかの懸念がある場合は、作業を開始する前に、Red Hat までご連絡いただき、アップグレードプロセスについてのアドバイスおよびサポートを依頼してください。
- 本項で説明するアップグレードプロセスは、director を使ったカスタマイズにのみ対応しています。director 以外を使用してオーバークラウドの機能をカスタマイズした場合は、オーバークラウドをアップグレードして、アップグレードの完了後にその機能を再度有効化してください。つまり、カスタマイズした機能は、アップグレードがすべて完了するまで利用できないということになります。
- Red Hat OpenStack Platform director 9 は、Red Hat OpenStack Platform 8 の特定のオーバークラウドバージョンを管理できます。詳しい情報は、以下のサポートマトリックスを参照してください。

表3.1 Red Hat OpenStack Platform director 9 のサポートマトリックス

バージョン	オーバークラウドの更新	オーバークラウドのデプロイ	オーバークラウドのスケーリング
Red Hat OpenStack Platform 9	8.0	すべてのバージョン	すべてのバージョン

- 旧バージョンのオーバークラウドを管理している場合には、以下の Heat テンプレートコレクションを使用してください。
  - Red Hat OpenStack Platform 8: **`/usr/share/openstack-tripleo-heat-templates/liberty/`**

    以下に例を示します。

    ```
    $ openstack overcloud deploy -templates /usr/share/openstack-tripleo-heat-templates/liberty/ [OTHER_OPTIONS]
    ```
- Red Hat OpenStack Platform 9 へのメジャーアップグレードを試みる前には、アンダークラウドとオーバークラウドを Red Hat OpenStack Platform 8 および Red Hat Enterprise Linux 7 の最新のマイナーリリースに必ずアップグレードしておいてください。「director ベースの環境: マイナーバージョンへの更新の実行」で、アンダークラウドとオーバークラウドにパッケージの

更新を実行する手順を参照してください。カーネルが最新バージョンに更新された場合には、再起動して新規カーネルパラメーターを有効にしてください。

3.2. DIRECTOR のアップグレード

重要

以下の手順を試みる前に、「アップグレード前の重要な注記」に記載の情報を一読してください。

Red Hat OpenStack Platform director をアップグレードするには、以下の手順に従ってください。

1. director に **stack** ユーザーとしてログインします。
2. OpenStack Platform のリポジトリーを更新します。

```
$ sudo subscription-manager repos --disable=rhel-7-server-
openstack-8-rpms --disable=rhel-7-server-openstack-8-director-
rpms
$ sudo subscription-manager repos --enable=rhel-7-server-
openstack-9-rpms --enable=rhel-7-server-openstack-9-director-rpms
```

   上記のコマンドは、**yum** が最新のリポジトリーを使用するように設定します。

3. 主要な OpenStack Platform サービスを停止します。

```
$ sudo systemctl stop 'openstack-*' 'neutron-*' httpd
```

注記

これにより、アンダークラウドで短時間のダウンタイムが生じます。アンダークラウドの更新中もオーバークラウドは引き続き機能します。

4. **yum** を使用して director をアップグレードします。

```
$ sudo yum update python-tripleoclient
```

5. 以下のコマンドでアンダークラウドをアップグレードします。

```
$ openstack undercloud upgrade
```

   このコマンドにより、director のパッケージがアップグレードされ、director の設定が最新の状態に更新されて、バージョンの変更後に指定されていない設定内容が追加されます。このコマンドによって、オーバークラウドのスタックデータや環境内の既存のノードのデータなど、保存されたデータは削除されません。

各ノードで **/var/log/yum.log** ファイルをチェックして、**kernel** または **openvswitch** のパッケージのメジャー/マイナーバージョンが更新されているかどうかを確認します。更新されている場合には、各ノードを再起動します。

1. ノードを再起動します。

```
$ sudo reboot
```

2. ノードが起動するまで待ちます。

ノードが起動したら、全サービスのステータスを確認します。

```
$ sudo systemctl list-units "openstack*" "neutron*" "openvswitch*"
```

オーバクラウドとそのノードが存在しているかどうかを確認します。

```
$ source ~/stackrc
$ openstack server list
$ openstack baremetal node list
$ openstack stack list
```

オーバークラウドを Red Hat OpenStack Platform 9 にアップグレードした後には以下の点に注意してください。

- アンダークラウドの **admin** ユーザーは、Red Hat OpenStack Platform 9 に含まれていない追加のロール (**_member_**) が必要な場合があります。このロールは、オーバークラウドの通信の際に重要です。このロールがあるかどうか確認します。

```
$ openstack role list
```

このロールが存在しない場合は、作成します。

```
$ openstack role create _member_
```

admin テナントで **admin** ユーザーにこのロールを追加します。

```
$ openstack role add --user admin --project admin _member_
```

重要

カスタマイズされたコアの Heat テンプレートを使用する場合には、更新されたコアの Heat テンプレートと現在の設定の相違点を確認するようにしてください。Red Hat では、今後のリリースで Heat テンプレートコレクションへの更新を提供します。変更されたテンプレートコレクションを使用すると、カスタムのコピーと **/usr/share/openstack-tripleo-heat-templates** にあるオリジナルのコピーとの間に相違が生じる可能性があります。以下のコマンドを実行して、カスタムの Heat テンプレートと更新されるオリジナルのバージョンとの差分を確認します。

```
# diff -Nary /usr/share/openstack-tripleo-heat-templates/
~/templates/my-overcloud/
```

これらの更新をカスタムの Heat テンプレートコレクションに適用するか、**/usr/share/openstack-tripleo-heat-templates/** でテンプレートの新しいコピーを作成して、カスタマイズを適用します。

3.3. DIRECTOR 上のオーバークラウドイメージのアップグレード

重要

以下の手順のステップを試す前に、「アップグレード前の重要な注記」の情報を必ず一読してください。

以下の手順では、ノードの検出とオーバークラウドのデプロイメントに向け最新のイメージが確保されるようにします。**rhosp-director-images** および **rhosp-director-images-ipa** のパッケージからの新規イメージは、アンダークラウドのアップグレードですでに更新されています。

stack ユーザーの **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/*
```

アーカイブを展開します。

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

最新のイメージを director にインポートして、ノードがこれらの新規イメージを使用するように設定します。

```
$ openstack overcloud image upload --update-existing --image-path
~/images/.
$ openstack baremetal configure boot
```

新規イメージの存在をチェックして、イメージの更新を最終確認します。

```
$ openstack image list
$ ls -l /httpboot
```

director は、最新のイメージを使用してアップグレードされました。

重要

オーバークラウドのイメージのバージョンがアンダークラウドのバージョンに対応していることを確認してください。

3.4. オーバークラウドのアップグレード

重要

以下の手順のステップを試す前に、「アップグレード前の重要な注記」の情報を必ず一読してください。

本項には、オーバークラウドのアップグレードに必要な手順を記載します。各セクションを順番に従って進み、お使いの環境に関連するセクションのみを適用します。

このプロセスでは、ステージごとに分けた手法を使用してアップグレードを行うには、元の **openstack overcloud deploy** コマンドを複数回実行する必要があります。コマンドを実行する度に、既存の環境ファイルに、別のアップグレードの環境ファイルが追加されます。これらの新しいアップグレード環境ファイルには、以下のようなファイルがあります。

- **major-upgrade-aodh.yaml**: OpenStack Telemetry Alarming (**aodh**) サービスをインストールします。
- **major-upgrade-keystone-liberty-mitaka.yaml**: OpenStack Identity (**keystone**) をスタンドアロンのサービスから、**httpd** 下の Web Server Gateway Interface (WSGI) サービスに移行します。
- **major-upgrade-pacemaker-init.yaml**: アップグレードの初期設定を行います。これには、オーバークラウドの各ノードにある Red Hat OpenStack Platform のリポジトリーの更新や、特定のノードへの特別なアップグレードスクリプトの提供などが含まれます。
- **major-upgrade-pacemaker.yaml**: コントローラーノードをアップグレードします。
- **major-upgrade-pacemaker-converge.yaml**: オーバークラウドのアップグレードの最終処理。これは、実行されるアップグレードが director の最新の Heat テンプレートコレクションの内容と一致するように合わせます。

これらのデプロイメントのコマンドの間に、さまざまなタイプのノードで **upgrade-non-controller.sh** スクリプトを実行します。このスクリプトにより、コントローラー以外のノードでパッケージが更新されます。

ワークフロー

オーバークラウドのアップグレードプロセスでは、以下のワークフローを使用します。

1. **major-upgrade-aodh.yaml** 環境ファイルを指定してデプロイメントコマンドを実行します。
2. **major-upgrade-keystone-liberty-mitaka.yaml** 環境ファイルを指定してデプロイメントコマンドを実行します。
3. **major-upgrade-pacemaker-init.yaml** 環境ファイルを指定してデプロイメントコマンドを実行します。
4. 各 Object Storage ノードで **upgrade-non-controller.sh** を実行します。
5. **major-upgrade-pacemaker.yaml** 環境ファイルを指定してデプロイメントコマンドを実行します。
6. 各 Compute ノードで **upgrade-non-controller.sh** を実行します。
7. 各 Ceph Storage ノードで **upgrade-non-controller.sh** を実行します。
8. **major-upgrade-pacemaker-converge.yaml** 環境ファイルを指定してデプロイメントコマンドを実行します。

3.4.1. オーバークラウドをアップグレードする前の注意事項

- Satellite の登録に環境ファイルを使用する場合は、その環境ファイルで以下のパラメーターを更新するようにしてください。
  - **rhel_reg_repos**: オーバークラウドを有効化するためのリポジトリー。新しい Red Hat OpenStack Platform 9 リポジトリーを含みます。有効化するリポジトリーについては、「リポジトリーの要件」を参照してください。
  - **rhel_reg_activation_key**: Red Hat OpenStack Platform 9 リポジトリー用の新規アクティベーションキー
  - **rhel_reg_sat_repo**: **katello-agent** など、Red Hat Satellite 6 の管理ツールが含まれるリポジトリーを定義する新たなパラメーター。Red Hat Satellite 6 に登録する場合はこのパラメーターを追加するようにしてください。
- 各ステップの後には、コントローラーノードのクラスターで **pcs status** コマンドを実行して、リソースにエラーが発生していないことを確認します。
- Red Hat OpenStack Platform 9 のデフォルトのタイムゾーンは UTC です。Red Hat OpenStack Platform 7 までは EST でした。必要な場合には、タイムゾーンを指定する環境ファイルを追加してください。
- SSL/TLS を有効化したオーバークラウドをアップグレードする場合には、新規サービス用に最新の **EndpointMap** を指定してください。エンドポイントの最新リストは、アンダークラウドの **/usr/share/openstack-tripleo-heat-templates/environments/enable-tls.yaml** にある環境ファイルの例を参照してください。SSL/TLS 設定に関する詳しい情報は、**『Red Hat OpenStack Platform director のインストールと使用方法』**の「6.11. オーバークラウドの SSL/TLS の有効化」 を参照してください。
- カスタムの **ServiceNetMap** を使用してオーバークラウドをアップグレードする場合には、新規サービス向けに最新の **ServiceNetMap** を指定してください。サービスの最新リストは、**『Red Hat OpenStack Platform director のインストールと使用方法』**の「6.2.3. OpenStack サービスの分離ネットワークへの割り当て」 を参照してください。
- director によって管理されているストレージノードがオーバークラウドに含まれている場合には、ノードに "client.openstack" CephX ユーザー用の新しいキーが必要です。Ceph Storage 用のその他のパラメーターが記載されている環境ファイル (通常は **storage-environment.yaml**) に **CephClientKey** パラメーターを追加してください。オーバークラウドノードで新しいキーを生成します。

```
$ ceph-authtool --gen-print-key
AQAfoaRXSAy/HxAAShHIViinopC2xtPW+RceQA==
```

  生成したキーを環境ファイルの **CephClientKey** に追加します。

```
parameter_defaults:
  CephStorageCount: 3
  CephClusterFSID: '4b5c8c0a-ff60-454b-a1b4-9747aa737d19'
  CephMonKey: 'AQC+Ox1VmEr3BxAALZejqeHj50Nj6wJDvs96OQ==''
  CephAdminKey: 'AQDLOh1VgEp6FRAAFzT7Zw+Y9V6JJExQAsRnRQ==''
  CephClientKey: 'AQAfoaRXSAy/HxAAShHIViinopC2xtPW+RceQA==''
```

- オーバークラウドで TLS/SSL の有効化にカスタムのエンドポイントマップを使用している場合には、以下の新規サービスのエンドポイントとのマッピングを必ず更新してください。
  - OpenStack Telemetry Metrics (gnocchi)
  - OpenStack Telemetry Alarming (aodh)

- OpenStack Clustering (sahara)

コアの Heat テンプレートコレクションから最新の TLS/SSL マッピングを確認して (**/usr/share/openstack-tripleo-heat-templates/environments/enable-tls.yaml** の **EndpointMap** を確認)、カスタムの **enable-tls.yaml** ファイルの **EndpointMap** に不足しているエンドポイントを追加します。詳しい情報は、**『Red Hat OpenStack Platform director のインストールと使用方法』**の「オーバークラウドの SSL/TLS の有効化」を参照してください。

3.4.2. 変更されたオーバークラウドテンプレートの使用

重要

本セクションに記載の内容は、コア Heat テンプレートコレクションを使用している場合にのみ必要です。これはコピーに、**/usr/share/openstack-tripleo-heat-templates/** にある元のコア Heat テンプレートコレクションの静的なスナップショットが使用されるためです。 オーバークラウドに未変更のコア Heat テンプレートコレクションを使用する場合には、本セクションのステップは省略してください。

変更されたテンプレートコレクションを更新するには、以下の作業を行う必要があります。

1. 既存のカスタムテンプレートコレクションをバックアップします。

```
$ mv ~/templates/my-overcloud/ ~/templates/my-overcloud.bak
```

2. /usr/share/openstack-tripleo-heat-templates にあるテンプレートコレクションの新しいバージョンを置き換えます。

```
$ sudo cp -rv /usr/share/openstack-tripleo-heat-templates
~/templates/my-overcloud/
```

3. 古いバージョンと新しいバージョンのカスタムテンプレートコレクションの差異を確認します。これらの 2 つの間の変更を確認するには、以下の diff コマンドを使用します。

```
$ diff -Nary ~/templates/my-overcloud.bak/ ~/templates/my-
overcloud/
```

これは、新規テンプレートコレクションに取り入れることが可能な旧テンプレートコレクションのカスタマイズを特定するのに役立ちます。新規カスタムテンプレートコレクションにカスタマイズの設定を取り入れます。

重要

Red Hat は、今後のリリースで Heat テンプレートコレクションの更新を提供します。変更されたテンプレートコレクションを使用すると、カスタムのコピーと **/usr/share/openstack-tripleo-heat-templates** にあるオリジナルのコピーとの間に相違が生じる可能性があります。オーバークラウドをカスタマイズするには、**『Advanced Overcloud Customization』**ガイドの「Configuration Hooks」のセクションを参照することを推奨します。Heat テンプレートコレクションのコピーを作成する場合には、git などのバージョン管理システムを使用して変更をトラッキングすべきです。

3.4.3. Aodh のインストール

このステップにより、旧 OpenStack Telemetry Alarming (**ceilometer-alarm**) サービスが新しいコンポーネント (**aodh**) に置き換えられます。このプロセスには、新しい環境ファイル (**major-upgrade-aodh.yaml**) を指定して **openstack overcloud deploy** 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、以下の操作が行われます。

- Pacemaker から **openstack-ceilometer-alarm**サービスを自動的に削除し、Controller ノードから関連するパッケージを削除します。
- コントローラーノードに **aodh** サービスをインストールし、Pacemaker がそのサービスを管理するように設定します。

アンダークラウドから、**major-upgrade-aodh.yaml** の環境ファイル を指定して **openstack overcloud deploy** を実行します。ネットワークの分離やストレージなどお使いの環境に関連するカスタムの環境ファイルも含めるようにしてください。

以下は、**openstack overcloud deploy** コマンドで、ファイルも追加した例です。

```
$ openstack overcloud deploy --force-postconfig --templates \
  --control-scale 3 \
  --compute-scale 3 \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/net-
single-nic-with-vlans.yaml  \
  -e /home/stack/templates/network_env.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/major-
upgrade-aodh.yaml
```

重要

最終のデプロイメントコマンドには **--force-postconfig** オプションを必ず指定してください。このオプションを使用しなかった場合には、オーバークラウドでの新規サービスポイントの表示が失敗します。

新しい環境ファイルの設定でオーバークラウドが更新されるまで待ちます。

注記

コントローラーノードにログインして **pcs status** コマンドを実行し、コントローラークラスター内の全リソースがアクティブかどうかを確認します。いずれかのリソースが失敗した場合には、**pcs resource cleanup** を実行してエラーをクリーニングし、各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にヘルプ / サポートをリクエストしてください。

3.4.4. Keystone のアップグレード

このステップでは、OpenStack Identity (**keystone**) サービスがコントローラーノードでスタンドアロンのサービスとして実行される代わりに **httpd** 下で Web Server Gateway Interface (WSGI) アプレットとして実行されるようにアップグレードします。このプロセスにより、スタンドアロンの **openstack-keystone** サービスは自動的に無効化され、WSGI アプレットを有効化するのに必

要な設定がインストールされます。

アンダークラウドから **major-upgrade-keystone-liberty-mitaka.yaml** の環境ファイルを指定して **openstack overcloud deploy** を実行します。ネットワークの分離やストレージなどお使いの環境に関連するカスタムの環境ファイルも含めるようにしてください。

以下は、**openstack overcloud deploy** コマンドで、ファイルも追加した例です。

```
$ openstack overcloud deploy --templates \
  --control-scale 3 \
  --compute-scale 3 \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/net-
single-nic-with-vlans.yaml  \
  -e /home/stack/templates/network_env.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/major-
upgrade-keystone-liberty-mitaka.yaml
```

新しい環境ファイルの設定でオーバークラウドが更新されるまで待ちます。

注記

コントローラーノードにログインして **pcs status** コマンドを実行し、コントローラークラスター内の全リソースがアクティブかどうかを確認します。いずれかのリソースが失敗した場合には、**pcs resource cleanup** を実行してエラーをクリーニングし、各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にヘルプ / サポートをリクエストしてください。

3.4.5. アップグレードスクリプトのインストール

このステップでは、コントローラー以外の各ノードにスクリプトをインストールします。これらのスクリプトは、メジャーバージョンのパッケージアップグレードおよび設定を実行します。各スクリプトは、ノードタイプによって異なります。たとえば、コンピュートノードにインストールされるアップグレードのスクリプトは、Ceph Storage ノードにインストールされるスクリプトとは異なります。

この初期化のステップにより、全オーバークラウド上で有効なリポジトリーの更新も行われるので、手動で古いリポジトリーを無効にして新しいリポジトリーを有効にする必要はありません。

アンダークラウドから **major-upgrade-pacemaker-init.yaml** の環境ファイルを指定して **openstack overcloud deploy** を実行します。ネットワークの分離やストレージなどお使いの環境に関連するカスタムの環境ファイルも含めるようにしてください。

以下は、**openstack overcloud deploy** コマンドで、ファイルも追加した例です。

```
$ openstack overcloud deploy --templates \
  --control-scale 3 \
  --compute-scale 3 \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/net-
```

```
single-nic-with-vlans.yaml  \
  -e /home/stack/templates/network_env.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/major-
upgrade-pacemaker-init.yaml
```

新しい環境ファイルの設定でオーバークラウドが更新されるまで待ちます。

注記

コントローラーノードにログインして **pcs status** コマンドを実行し、コントローラークラスター内の全リソースがアクティブかどうかを確認します。いずれかのリソースが失敗した場合には、**pcs resource cleanup** を実行してエラーをクリーニングし、各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にヘルプ / サポートをリクエストしてください。

3.4.6. Object Storage ノードのアップグレード

director は **upgrade-non-controller.sh** コマンドを使用してアップグレードのスクリプトを実行します。このスクリプトは、**major-upgrade-pacemaker-init.yaml** 環境ファイルから、コントローラー以外の各ノードに渡されます。このステップでは、各 Object Storage ノードを以下のコマンドでアップグレードします。

```
$ for NODE in `nova list | grep swift | awk -F "|" '{ print $2 }'` ; do
upgrade-non-controller.sh --upgrade $NODE ; done
```

各 Object Storage ノードのアップグレードが完了するまで待ちます。

各ノードで **/var/log/yum.log** ファイルをチェックして、**kernel** または **openvswitch** のパッケージのメジャー/マイナーバージョンが更新されているかどうかを確認します。更新されている場合には、各ノードを再起動します。

1. 再起動する Object Storage ノードを選択します。そのノードにログインして再起動します。

   ```
   $ sudo reboot
   ```

2. ノードが起動するまで待ちます。
3. ノードにログインして、ステータスを確認します。

   ```
   $ sudo systemctl list-units "openstack-swift*"
   ```

4. ノードからログアウトして、次の Object Storage ノードでこのプロセスを繰り返します。

注記

コントローラーノードにログインして **pcs status** コマンドを実行し、コントローラークラスター内の全リソースがアクティブかどうかを確認します。いずれかのリソースが失敗した場合には、**pcs resource cleanup** を実行してエラーをクリーニングし、各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にヘルプ / サポートをリクエストしてください。

3.4.7. コントローラーノードのアップグレード

コントローラーノードをアップグレードする際には、高可用性ツールを実行するコントローラーノードへの完全アップグレードを提供する別の環境ファイル (**major-upgrade-pacemaker.yaml**) を指定する必要があります。

アンダークラウドから **major-upgrade-pacemaker.yaml** の環境ファイルを指定して **openstack overcloud deploy** を実行します。ネットワークの分離やストレージなど環境に関連するカスタムの環境ファイルも指定するようにしてください。

以下は、**openstack overcloud deploy** コマンドで、ファイルも追加した例です。

```
$ openstack overcloud deploy --templates \
  --control-scale 3 \
  --compute-scale 3 \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/net-
single-nic-with-vlans.yaml \
  -e network_env.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/major-
upgrade-pacemaker.yaml
```

新しい環境ファイルの設定でオーバークラウドが更新されるまで待ちます。

重要

このステップは、コントローラーのアップグレードの際に Neutron サーバーおよび L3 エージェントを無効化します。これは、Floating IP アドレスがこのステップでは利用できないということを意味します。

各ノードで **/var/log/yum.log** ファイルをチェックして、**kernel** または **openvswitch** のパッケージのメジャー/マイナーバージョンが更新されているかどうかを確認します。更新されている場合には、各ノードを再起動します。

1. 再起動するノードを選択します。そのノードにログインして再起動します。

   ```
   $ sudo reboot
   ```

   クラスター内の残りのコントローラーノードは、再起動中も高可用性サービスが保持されます。

2. ノードが起動するまで待ちます。

3. ノードにログインして、クラスターのステータスを確認します。

   ```
   $ sudo pcs status
   ```

   このノードは、クラスターに再度参加します。

注記

再起動後に失敗するサービスがあった場合には、sudo **pcs resource cleanup** を実行し、エラーを消去して各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にアドバイス/サポートをリクエストしてください。

4. コントローラーノード上の全 **systemd** サービスがアクティブであることを確認します。

```
$ sudo systemctl list-units "openstack*" "neutron*"
"openvswitch*"
```

5. ノードからログアウトして、次に再起動するコントローラーノードを選択し、すべてのコントローラーノードが再起動されるまでこの手順を繰り返します。

3.4.8. Ceph Storage ノードのアップグレード

director は **upgrade-non-controller.sh** コマンドを使用してアップグレードのスクリプトを実行します。このスクリプトは、**major-upgrade-pacemaker-init.yaml** 環境ファイルから、コントローラー以外の各ノードに渡されます。このステップでは、各 Ceph Storage ノードを以下のコマンドでアップグレードします。

各 Ceph Storage ノードをアップグレードします。

```
$ for NODE in `nova list | grep ceph | awk -F "|" '{ print $2 }'` ; do
upgrade-non-controller.sh --upgrade $NODE ; done
```

各ノードで **/var/log/yum.log** ファイルをチェックして、**kernel** または **openvswitch** のパッケージのメジャー/マイナーバージョンが更新されているかどうかを確認します。更新されている場合には、各ノードを再起動します。

1. 再起動する最初の Ceph Storage ノードを選択して、ログインします。
2. Ceph Storage クラスターのリバランシングを一時的に無効にします。

```
$ sudo ceph osd set noout
$ sudo ceph osd set norebalance
```

3. ノードを再起動します。

```
$ sudo reboot
```

4. ノードが起動するまで待ちます。
5. ノードにログインして、クラスターのステータスを確認します。

```
$ sudo ceph -s
```

pgmap により、すべての **pgs** が正常な状態 (**active+clean**) として報告されることを確認します。

6. ノードからログアウトして、次のノードを再起動し、ステータスを確認します。全 Ceph Storage ノードすべてが再起動されるまで、このプロセスを繰り返します。
7. 操作が完了したら、クラスターのリバランシングを再度有効にします。

```
$ sudo ceph osd unset noout
$ sudo ceph osd unset norebalance
```

8. 最終のステータスチェックを実行して、クラスターが **HEALTH_OK** と報告することを確認します。

```
$ sudo ceph status
```

注記

コントローラーノードにログインして **pcs status** コマンドを実行し、コントローラークラスター内の全リソースがアクティブかどうかを確認します。いずれかのリソースが失敗した場合には、**pcs resource cleanup** を実行してエラーをクリーニングし、各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にヘルプ / サポートをリクエストしてください。

3.4.9. コンピュートノードのアップグレード

コンピュートノードを個別にアップグレードして、OpenStack Platform 環境のインスタンスのダウンタイムがゼロになるようにします。この操作は、以下のワークフローに従って実行します。

1. アップグレードするコンピュートノードを選択します。
2. インスタンスを別のコンピュートノードに移行します。
3. 空のコンピュートノードをアップグレードします。

全コンピュートノードとその UUID を一覧表示します。

```
$ nova list | grep "compute"
```

アップグレードするコンピュートノードを選択してから、まず最初に以下の手順に従ってそのノードのインスタンスを移行します。

1. アンダークラウドから、再起動するコンピュートノードを選択し、そのノードを無効にします。

```
$ source ~/overcloudrc
$ openstack compute service list
$ openstack compute service set [hostname] nova-compute --disable
```

2. コンピュートノード上の全インスタンスを一覧表示します。

```
$ openstack server list --host [hostname]
```

3. インスタンスの移行のターゲットホストとして機能する 2 番目のコンピュートノードを選択し、このホストには、移行されるインスタンスをホストするのに十分なリソースが必要です。無効化されたホストからターゲットホストに各インスタンスを移行する操作をアンダークラウドから実行します。

```
$ nova live-migration [instance-name] [target-hostname]
$ nova migration-list
$ nova resize-confirm [instance-name]
```

4. コンピュートノードからすべてのインスタンスが移行されるまで、このステップを繰り返します。

重要

インスタンスの設定と移行の詳しい手順は、**『director のインストールと使用方法』**の「オーバークラウドのコンピュートノードからの仮想マシンの移行」を参照してください。

director は **upgrade-non-controller.sh** コマンドを使用してアップグレードのスクリプトを実行します。このスクリプトは、**major-upgrade-pacemaker-init.yaml** 環境ファイルから、コントローラー以外の各ノードに渡されます。以下のコマンドを実行して、各コンピュートノードをアップグレードします。

```
$ source ~/stackrc
$ upgrade-non-controller.sh --upgrade NODE_UUID
```

NODE_UUID は、選択したコンピュートノードの UUID に置き換えます。コンピュートーノードのアップグレードが完了するまで待ちます。

アップグレードしたコンピュートノード上の **/var/log/yum.log** ファイルをチェックして、**kernel** または **openvswitch** のパッケージがアップグレードされているかどうかを確認します。アップグレードされている場合には、各ノードのリブートを実行します。

1. コンピュートノードのログインしてリブートします。

```
$ sudo reboot
```

2. ノードが起動するまで待ちます。
3. コンピュートノードを再度有効化します。

```
$ source ~/overcloudrc
$ openstack compute service set [hostname] nova-compute --enable
```

4. リブートする次のノードを選択します。

全ノードがリブートされるまで、各ノードで個別にマイグレーションとリブートのプロセスを繰り返します。

注記

コントローラーノードにログインして **pcs status** コマンドを実行し、コントローラークラスター内の全リソースがアクティブかどうかを確認します。いずれかのリソースが失敗した場合には、**pcs resource cleanup** を実行してエラーをクリーニングし、各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にヘルプ / サポートをリクエストしてください。

3.4.10. アップグレードの最終処理

director には、アップグレードの最終処理を最後まで実行して、オーバークラウドスタックが現在の Heat テンプレートコレクションと確実に同期されるようにする必要があります。そのためには、**openstack overcloud deploy** コマンドで環境ファイル (**major-upgrade-pacemaker-converge.yaml**) を指定する必要があります。

アンダークラウドから **major-upgrade-pacemaker-converge.yaml** の環境ファイルを指定して **openstack overcloud deploy** を実行します。ネットワークの分離やストレージなどお使いの環境に関連するカスタムの環境ファイルも含めるようにしてください。

以下は、**openstack overcloud deploy** コマンドで、ファイルも追加した例です。

```
$ openstack overcloud deploy --force-postconfig --templates \
  --control-scale 3 \
  --compute-scale 3 \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/net-
single-nic-with-vlans.yaml \
  -e network_env.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/major-
upgrade-pacemaker-converge.yaml
```

重要

最終のデプロイメントコマンドには **--force-postconfig** オプションを必ず指定してください。このオプションを使用しなかった場合には、オーバークラウドでの新規サービスポイントの表示が失敗します。

新しい環境ファイルの設定でオーバークラウドが更新されるまで待ちます。

これで、オーバークラウドのアップグレード手順が完了しました。

注記

コントローラーノードにログインして **pcs status** コマンドを実行し、コントローラークラスター内の全リソースがアクティブかどうかを確認します。いずれかのリソースが失敗した場合には、**pcs resource cleanup** を実行してエラーをクリーニングし、各リソースの状態を **Started** に設定します。エラーが引き続き発生する場合には、Red Hat にヘルプ / サポートをリクエストしてください。

3.4.11. オーバークラウドのアップグレード後の注意事項

オーバークラウドを Red Hat OpenStack Platform 9 にアップグレードした後には以下の点に注意してください。

- コンピュートノードが **neutron-openvswitch-agent** の問題をレポートする可能性があります。これが発生した場合には、各コンピュートノードにログインして、このサービスを再起動します。以下のようなコマンドを実行します。

```
$ sudo systemctl restart neutron-openvswitch-agent
```

- 更新プロセスを実行しても、オーバークラウド内のノードは自動的には再起動しません。必要な場合には、更新コマンドが完了した後に手動で再起動を実行してください。クラスターベースのノード (Ceph Storage ノードやコントローラーノード) を個別に再起動して、ノードがクラスターに再度参加するまで待ちます。Ceph Storage ノードの場合は、**ceph health** で確認して、クラスターのステータスが **HEALTH OK** であることを確認します。コントローラーノードの場合は、**pcs resource** で確認して、各ノードですべてのリソースが実行されていることを確認してください。

- 状況によっては、コントローラーノードの再起動後に IPv6 環境で **corosync** サービスの起動に失敗する可能性があります。これは、コントローラーノードが静的な IPv6 アドレスを設定する前に Corosync が起動してしまうことが原因です。このような場合は、コントローラーノードで Corosync を手動で再起動してください。

```
$ sudo systemctl restart corosync
```

- コントローラーノードにフェンシングを設定している場合には、更新プロセスによってその設定が無効になる場合があります。更新プロセスの完了時には、コントローラーノードの 1 つで以下のコマンドを実行してフェンシングを再度有効にします。

```
$ sudo pcs property set stonith-enabled=true
```

- 次回に (**openstack overcloud deploy** コマンドを実行して) オーバークラウドのスタックを更新またはスケーリングする際には、オーバークラウドでのパッケージの更新をトリガーする ID をリセットする必要があります。環境ファイルに空の **UpdateIdentifier** パラメーターを追加して、**openstack overcloud deploy** コマンドの実行時にそのファイルを指定します。環境ファイルの内容の例を以下に示します。

```
parameter_defaults:
  UpdateIdentifier:
```

第4章 DIRECTOR を使用しない環境: OPENSTACK サービスの同時アップグレード

このシナリオでは、**director を使用しない**環境で、Red Hat OpenStack Platform 8 から Red Hat OpenStack Platform 9 にアップグレードします。この手順では、全ノード上の全サービスをアップグレードします。この作業は、以下のワークフローに従って実行します。

1. 全 OpenStack サービスの無効化
2. パッケージアップグレードの実行
3. 全データベースの同期の実行
4. 全 OpenStack サービスの有効化

注記

本章に記載する手順は、すべての Red Hat OpenStack Platform ドキュメントで順守しているアーキテクチャー命名規則に従います。この規則に精通していない場合には、手順を開始する前に Red Hat OpenStack Platform ドキュメントスイート で『アーキテクチャーガイド』を参照してください。

4.1. 全 OPENSTACKサービスの無効化

ノードに対して完全なアップグレードを実行するには、最初のステップとして、全 OpenStack サービスをシャットダウンします。このステップは、ノードが管理を行うために高可用性ツールを使用しているかどうか (例: コントローラーノード上で Pacemaker を使用している) によって異なります。このステップには、両ノードタイプの手順を記載しています。

標準ノード

全標準ノードに **openstack-utils** パッケージをインストールします。

```
# yum install openstack-utils
```

全標準ノードですべての OpenStack サービスを無効にします。

```
# openstack-service stop
```

高可用性ノード

OpenStack のサービスはすべて無効にする必要がありますが、データベースとロードバランシングはそのままにしてください。たとえば、Pacemaker で HAProxy、Galera、および MongoDB のサービスを管理対象外に切り替えます。

```
# pcs resource unmanage haproxy
# pcs resource unmanage galera
# pcs resource unmanage mongod
```

クラスター上で **stop-all-resources** を設定して、Pacemaker で管理されている残りのリソースを無効にします。Pacemaker クラスターの 1 つのメンバーで以下のコマンドを実行します。

```
# pcs property set stop-all-resources=true
```

Pacemaker で管理されている全リソースが停止するまで待ってから、**pcs status** コマンドを実行して各リソースのステータスを確認します。

```
# pcs status
```

重要

HAProxy では、利用できないサービスのブロードキャストメッセージが表示される場合があります。これは正常な動作です。

4.2. パッケージアップグレードの実行

次のステップでは、ノード上の全パッケージをアップグレードします。このステップは、OpenStack サービスを使用する各ノードで実行します。

subscription-manager コマンドを使用して、Red Hat OpenStack Platform 9 リポジトリーに変更します。

```
# subscription-manager repos --disable=rhel-7-server-openstack-8-rpms
# subscription-manager repos --enable=rhel-7-server-openstack-9-rpms
```

ノードで **yum update** コマンドを実行します。

```
# yum update
```

パッケージの更新が完了するまで待ちます。

更新後の設定ファイルを確認します。アップグレードパッケージによって、Red Hat OpenStack Platform 9 のサービスに適した **.rpmnew** ファイルがインストールされているはずです。新しいバージョンの OpenStack サービスでは、特定の設定オプションが非推奨になっている可能性があります。このような非推奨の設定オプションが原因で今後のアップグレードの際に問題が発生する可能性があるため、非推奨の警告については OpenStack のログも参照してください。各サービスで新規追加/更新された設定オプションや非推奨となった設定オプションについての詳しい説明は、Red Hat OpenStack Platform ドキュメントスイート で『Configuration Reference』を参照してください。

環境内の各ノードでパッケージのアップグレードを実行します。

4.3. 全データベースの同期の実行

次のステップでは、各サービスのデータベースをアップグレードします。

注記

データベースの同期の所要時間が短縮するために、Identity サービス内の期限切れトークンをフラッシュします。

```
# keystone-manage token_flush
```

データベースを使用する各サービスのデータベーススキーマをアップグレードします。サービスのデータベースをホストしているノードで以下のコマンドを実行します。

```
# openstack-db --service SERVICENAME --update
```

SERVICENAME には、サービスのプロジェクト名を使用します。たとえば、Identity サービス内のデータベーススキーマをアップグレードするには、以下のコマンドを実行します。

```
# openstack-db --service keystone --update
```

表4.1 OpenStack サービスのプロジェクト名

サービス	プロジェクト名
Identity	keystone
Image サービス	glance
Block Storage	cinder
Orchestration	heat
Compute	nova
ネットワーク	neutron

Telemetry サービスは、データベースのアップグレードに別のコマンドを使用します。

```
# ceilometer-dbsync
```

4.4. 全 OPENSTACK サービスの有効化

最後のステップでは、ノード上で OpenStack サービスを有効化します。このステップは、ノードが管理を行うために高可用性ツールを使用しているかどうか (例: コントローラーノード上で Pacemaker を使用している) によって異なります。このステップには、両ノードタイプの手順を記載しています。

標準ノード

全 OpenStack サービスを有効化します。

```
# openstack-service stop
```

高可用性ノード

Pacemaker を介してリソースを再起動します。Pacemaker クラスター内の 1 つのメンバーで **`stop-all-resources`** プロパティーをリセットします。以下に例を示します。

```
# pcs property set stop-all-resources=false
```

全リソースが開始するまで待ってから、**`pcs status`** コマンドを実行して各リソースのステータスを確認します。

```
# pcs status
```

データベースやロードバランサーなど、Pacemaker で管理対象外にしていたリソースを有効化して管理対象にします。

```
# pcs resource manage haproxy
# pcs resource manage galera
# pcs resource manage mongod
```

4.5. アップグレード後の注意事項

新しいバージョンの OpenStack サービスでは、特定の設定オプションが非推奨になっている可能性があります。このような非推奨の設定オプションが原因で今後のアップグレードの際に問題が発生する可能性があるため、非推奨の警告については OpenStack のログも参照してください。各サービスで新規追加/更新された設定オプションや非推奨となった設定オプションについての詳しい説明は、Red Hat OpenStack Platform ドキュメントスイート で『Configuration Reference』を参照してください。

第5章 DIRECTOR を使用しない環境: 標準的な環境内の個別の OPENSTACK サービス (稼働中の COMPUTE) のアップグレード

以下の項では、非高可用性の環境において Compute を並行稼働させて個別のサービスを更新する方法でクラウドデプロイメントをアップグレードする場合に従う必要のある手順を記載します。このシナリオは、**director を使用しない環境で** Red Hat OpenStack Platform 8 から Red Hat OpenStack Platform 9 にアップグレードします。

稼働中の Compute を使用してアップグレードを行うと、Compute サービスの中断が最小限に抑えられます。小規模なサービスの場合はわずか数分ですが、新たにアップグレードされたコンピュートホストにワークロードを移動する場合は、移行の所要時間がより長くなります。既存のワークロードは無期限で稼働させることが可能です。また、データベース移行を待つ必要はありません。

重要

特定のパッケージの依存関係が原因で、OpenStack サービスのパッケージのアップグレードを試みると、OpenStack のサービスがアップグレードされる前に、Python ライブラリーがアップグレードされてしまう場合があります。そのために、特定のサービスが完了前に失敗する可能性があります。このような状況が発生した場合には、残りのサービスのアップグレードを継続します。このシナリオが完了すると、全サービスが稼動状態になるはずです。

注記

この方法では、コンピュートノードを稼働させるのに追加のハードウェアリソースが必要となる場合があります。

注記

本章に記載する手順は、すべての Red Hat OpenStack Platform ドキュメントで順守しているアーキテクチャー命名規則に従います。この規則に精通していない場合には、手順を開始する前に Red Hat OpenStack Platform ドキュメントスイート で『アーキテクチャーガイド』を参照してください。

5.1. アップグレード前のタスク

各ノードで **subscription-manager** コマンドを使用して、Red Hat OpenStack Platform 9 リポジトリーに変更します。

```
# subscription-manager repos --disable=rhel-7-server-openstack-8-rpms
# subscription-manager repos --enable=rhel-7-server-openstack-9-rpms
```

openstack-selinux パッケージをアップグレードします。

```
# yum upgrade openstack-selinux
```

これは、アップグレードしたサービスが SELinux が有効なシステムで正しく実行されるようにするために必要です。

5.2. IDENTITY (KEYSTONE) および DASHBOARD (HORIZON) のアップグレード

Identity サービスと Dashboard サービスを無効にします。設定に応じて、以下のいずれかの作業を実行する必要があります。

1. Dashboard と Identity の両サービスが WSGI アプレットとして実行されている場合には、**httpd** を無効にします。

```
# systemctl stop httpd
```

2. Identity サービスが別個のサービスとして実行されている場合には、その **openstack-keystone**を無効にしてから、Dashboard 用の **httpd** を無効にします。

```
# openstack-service stop keystone
# systemctl stop httpd
```

両サービスのパッケージを更新します。

```
# yum -d1 -y upgrade \*keystone\*
# yum -y upgrade \*horizon\* \*openstack-dashboard\*
# yum -d1 -y upgrade \*horizon\* \*python-django\*
```

Identity サービスのトークンテーブルには、多数の期限切れエントリーが含まれている可能性があります。その場合には、データベーススキーマのアップグレードの所要時間が大幅に長くなる可能性があります。期限切れのトークンをデータベースからフラッシュして問題を緩和するには、Identity のデータベースのアップグレードを実行する前に、**keystone-manage** コマンドを使用することができます。

```
# keystone-manage token_flush
# openstack-db --service keystone --update
```

このコマンドにより、データベースから期限切れのトークンがフラッシュされます。**cron** を使用して、このコマンドが定期的に実行されるように設定してください。

サービスを再起動します。これには、設定に応じて以下のいずれかの作業を実行する必要があります。

1. Dashboard と Identity の両サービスが WSGI アプレットとして実行されている場合には、**httpd** を有効にします。

```
# systemctl start httpd
```

2. Identity サービスが別個のサービスとして実行されている場合には、その **openstack-keystone**を無効にしてから、Dashboard 用の **httpd** を有効にします。

```
# openstack-service start keystone
# systemctl start httpd
```

5.3. OBJECT STORAGE (SWIFT) のアップグレード

Object Storage ホストで以下のコマンドを実行します。

```
# openstack-service stop swift
# yum -d1 -y upgrade \*swift\*
# openstack-service start swift
```

5.4. IMAGE サービス (GLANCE) のアップグレード

Image サービスのホストで以下のコマンドを実行します。

```
# openstack-service stop glance
# yum -d1 -y upgrade \*glance\*
# openstack-db --service glance --update
# openstack-service start glance
```

5.5. BLOCK STORAGE (CINDER) のアップグレード

Block Storage ホストで以下のコマンドを実行します。

```
# openstack-service stop cinder
# yum -d1 -y upgrade \*cinder\*
# openstack-db --service cinder --update
# openstack-service start cinder
```

5.6. ORCHESTRATION (HEAT) のアップグレード

Orchestration ホストで以下のコマンドを実行します。

```
# openstack-service stop heat
# yum -d1 -y upgrade \*heat\*
# openstack-db --service heat --update
# openstack-service start heat
```

5.7. TELEMETRY (CEILOMETER) のアップグレード

1. Telemetry コンポーネントサービスをホストする全ノードで以下のコマンドを実行します。

   ```
   # openstack-service stop ceilometer
   # yum -d1 -y upgrade \*ceilometer\*
   ```

2. Image サービスのホストで以下のコマンドを実行します。

   ```
   # ceilometer-dbsync
   ```

3. パッケージのアップグレードが完了した後には、Telemetry コンポーネントサービスをホストする全ノードで以下のコマンドを実行して Telemetry サービスを再起動します。

```
# openstack-service start ceilometer
```

5.8. COMPUTE (NOVA) のアップグレード

1. コンピュートホストのローリングアップグレードを行うには、明示的に API のバージョンの制限を設定して、環境内の互換性を確保する必要があります。

   コントローラーノードまたはコンピュートノードで Compute サービスを起動する前に、**nova.conf** ファイルの **[upgrade_levels]** セクションで、**compute** オプションを以前の Red Hat OpenStack Platform バージョン (**liberty**) に設定します。

   ```
   # crudini --set /etc/nova/nova.conf upgrade_levels compute
   liberty
   ```

   この変更は、コントローラーノードとコンピュートノードの両方で行う必要があります。

   すべてのコンピュートノードをアップグレードしたら、この操作を元に戻す必要があります。

2. コンピュートホストで以下のコマンドを実行します。

   ```
   # openstack-service stop nova
   # yum -d1 -y upgrade \*nova\*
   # openstack-db --service nova --update
   ```

3. 全ホストをアップグレードした後には、以前のステップで設定した API の制限を削除します。全ホスト上で以下のコマンドを実行します。

   ```
   # crudini --del /etc/nova/nova.conf upgrade_levels compute
   ```

4. すべてのコントローラーノードとコンピュートノードで Compute サービスを再起動します。

   ```
   # openstack-service start nova
   ```

5.9. OPENSTACK NETWORKING (NEUTRON) のアップグレード

1. OpenStack Networking ホストで以下のコマンドを実行します。

   ```
   # openstack-service stop neutron
   # yum -d1 -y upgrade \*neutron\*
   # openstack-db --service neutron --update
   ```

2. OpenStack Networking サービスを再起動します。

   ```
   # openstack-service start neutron
   ```

5.10. アップグレード後のタスク

個別サービスのアップグレードをすべて完了した後には、全システムで完全なパッケージアップグレードを行う必要があります。

```
# yum upgrade
```

このコマンドは、すべてのパッケージを最新の状態にします。実行中のプロセスが、配下のバイナリーの更新されたバージョンを使用するようにするには、OpenStack ホストの再起動を後日にスケジューリングする必要があります。

上記の操作によって生成された設定ファイルを確認します。アップグレードされたパッケージで、Red Hat OpenStack Platform 9 バージョンのサービスに適した **.rpmnew** ファイルがインストールされているはずです。

新しいバージョンの OpenStack サービスでは、特定の設定オプションが非推奨になっている可能性があります。このような非推奨の設定オプションが原因で今後のアップグレードの際に問題が発生する可能性があるため、非推奨の警告については OpenStack のログも参照してください。各サービスで新規追加/更新された設定オプションや非推奨となった設定オプションについての詳しい説明は、Red Hat OpenStack Platform ドキュメントスイート で『Configuration Reference』を参照してください。

第6章 DIRECTOR を使用しない環境: 高可用性環境における個別の OPENSTACK サービス (稼働中の COMPUTE) のアップグレード

本章では、高可用性の環境において Compute を並行稼働させて個別のサービスを更新する方法でクラウドデプロイメントをアップグレードする場合に従う必要のある手順を記載します。このシナリオは、**director を使用しない環境で** Red Hat OpenStack Platform 8 から Red Hat OpenStack Platform 9 にアップグレードします。

稼働中の Compute を使用してアップグレードを行うと、Compute サービスの中断が最小限に抑えられます。小規模なサービスの場合はわずか数分ですが、新たにアップグレードされたコンピュートホストにワークロードを移動する場合は、移行の所要時間がより長くなります。既存のワークロードは無期限で稼働させることが可能です。また、データベース移行を待つ必要はありません。

重要

特定のパッケージの依存関係が原因で、OpenStack サービスのパッケージのアップグレードを試みると、OpenStack のサービスがアップグレードされる前に、Python ライブラリーがアップグレードされてしまう場合があります。そのために、特定のサービスが完了前に失敗する可能性があります。このような状況が発生した場合には、残りのサービスのアップグレードを継続します。このシナリオが完了すると、全サービスが稼動状態になるはずです。

注記

この方法では、Red Hat OpenStack Platform 9 のコンピュートノードを稼働させるのに追加のハードウェアリソースが必要となる場合があります。

注記

本章に記載する手順は、すべての Red Hat OpenStack Platform ドキュメントで順守しているアーキテクチャー命名規則に従います。この規則に精通していない場合には、手順を開始する前に Red Hat OpenStack Platform ドキュメントスイート で『アーキテクチャーガイド』を参照してください。

6.1. アップグレード前のタスク

各ノードで **subscription-manager** コマンドを使用して、Red Hat OpenStack Platform 9 リポジトリーに変更します。

```
# subscription-manager repos --disable=rhel-7-server-openstack-8-rpms
# subscription-manager repos --enable=rhel-7-server-openstack-9-rpms
```

openstack-selinux パッケージをアップグレードします。

```
# yum upgrade openstack-selinux
```

これは、アップグレードしたサービスが SELinux が有効なシステムで正しく実行されるようにするために必要です。

6.2. MARIADB のアップグレード

MariaDB を実行する各ホストで、以下の手順を実行します。1 つのホストでの手順がすべて完了してから、次のホストでこのプロセスを開始してください。

1. ローカルノードで実行されないようにサービスを停止します。

```
# pcs resource ban galera-master $(crm_node -n)
```

2. **pcs status** の出力で、ローカルノードで実行中のサービスがなくなるまで待ちます。これは、数分かかる可能性があります。ローカルノードは、slaves モードに切り替わります。

```
Master/Slave Set: galera-master [galera]
Masters: [ overcloud-controller-1 overcloud-controller-2 ]
Slaves: [ overcloud-controller-0 ]
```

ノードは最終的に Stopped に変わります。

```
Master/Slave Set: galera-master [galera]
Masters: [ overcloud-controller-1 overcloud-controller-2 ]
Stopped: [ overcloud-controller-0 ]
```

3. 適切なパッケージをアップグレードします。

```
# yum upgrade '*mariadb*' '*galera*'
```

4. Pacemaker がローカルノードで **galera** リソースのスケジューリングができるようにします。

```
# pcs resource clear galera-master
```

5. **pcs status** の出力で gelara リソースがローカルノードでマスターとして実行されていることが表示されるまで待ちます。**pcs status** コマンドの出力は、以下のように表示されるはずです。

```
Master/Slave Set: galera-master [galera]
Masters: [ overcloud-controller-0 overcloud-controller-1
overcloud-controller-2 ]
```

上記の手順は、MariaDB クラスターの完全なアップグレードが完了するまで各ノードで個別に実行します。

6.3. MONGODB のアップグレード

この手順では、OpenStack Telemetry サービスのバックエンドとして機能する MongoDB をアップグレードします。

1. **mongod** リソースを Pacemaker の制御対象から除外します。

```
# pcs resource unmanage mongod-clone
```

2. このサービスを全コントローラーノードで停止します。各コントローラーノードで以下のコマンドを実行します。

```
# systemctl stop mongod
```

3. 適切なパッケージをアップグレードします。

```
#  yum upgrade 'mongodb*' 'python-pymongo*'
```

4. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

5. 各コントローラーで以下のコマンドを実行して、**mongod** サービスを再起動します。

```
# systemctl start mongod
```

6. リソースをクリーンアップします。

```
# pcs resource cleanup mongod-clone
```

7. リソースを Pacemaker の制御対象に戻します。

```
# pcs resource manage mongod-clone
```

8. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.4. IDENTITY サービス (KEYSTONE) のアップグレード

この手順では、全コントローラーノード上で Identity サービスのパッケージを同時にアップグレードします。

1. Identity サービスを Pacemaker の制御対象から除外します。

```
# pcs resource unmanage openstack-keystone-clone
```

2. 各コントローラーノードで以下のコマンドを実行して Identity サービスを停止します。

```
# systemctl stop openstack-keystone
```

3. 適切なパッケージをアップグレードします。

```
# yum upgrade 'openstack-keystone*' 'python-keystone*'
```

4. 各コントローラーノードで、更新したユニットファイルを有効にするために、**systemd** を再読み込みます。

```
# systemctl daemon-reload
```

5. 初期のバージョンのインストーラーでは、期限切れの keystone トークンが自動的に削除されるようにシステム設定されていない可能性があるため、トークンテーブルに期限切れのエントリーが多数含まれている可能性があります。このような場合には、データベーススキーマのアップグレードの所要時間が大幅に増大する可能性があります。

   問題を緩和するには、データベースから期限切れのトークンをフラッシュします。Identity データベースのアップグレードを実行する前に **keystone-manage** コマンドを実行します。

   ```
   # keystone-manage token_flush
   ```

   これで、期限切れのトークンがデータベースからフラッシュされます。このコマンドは、**cron** を使用して定期的に (例: 毎日) 実行するように設定することが可能です。

6. Identity サービスのデータベーススキーマを更新します。

   ```
   # openstack-db --service keystone --update
   ```

7. 各コントローラーノードで以下のコマンドを実行してサービスを再起動します。

   ```
   # systemctl start openstack-keystone
   ```

8. Pacemaker を使用して Identity サービスをクリーンアップします。

   ```
   # pcs resource cleanup openstack-keystone-clone
   ```

9. リソースを Pacemaker の制御対象に戻します。

   ```
   # pcs resource manage openstack-keystone-clone
   ```

10. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.5. IMAGE サービス (GLANCE) のアップグレード

この手順では、全コントローラーノード上で Image サービスのパッケージを同時にアップグレードします。

1. Pacemaker で Image サービスのリソースを停止します。

   ```
   # pcs resource disable openstack-glance-registry-clone
   # pcs resource disable openstack-glance-api-clone
   ```

2. **pcs status** の出力で、両サービスが停止されるまで待ちます。

3. 適切なパッケージをアップグレードします。

   ```
   # yum upgrade 'openstack-glance*' 'python-glance*'
   ```

4. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

5. Image サービスのデータベーススキーマを更新します。

```
# openstack-db --service glance --update
```

6. Pacemaker を使用して Image サービスをクリーンアップします。

```
# pcs resource cleanup openstack-glance-api-clone
# pcs resource cleanup openstack-glance-registry-clone
```

7. Pacemaker で Image サービスのリソースを再起動します。

```
# pcs resource enable openstack-glance-api-clone
# pcs resource enable openstack-glance-registry-clone
```

8. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.6. BLOCK STORAGE (CINDER) サービスのアップグレード

この手順では、全コントローラーノード上で Block Storage サービスのパッケージを同時にアップグレードします。

1. Pacemaker で Block Storage サービスのリソースを停止します。

```
# pcs resource disable openstack-cinder-api-clone
# pcs resource disable openstack-cinder-scheduler-clone
# pcs resource disable openstack-cinder-volume
```

2. **pcs status** の出力で、上記のサービスが停止されるまで待ちます。
3. 適切なパッケージをアップグレードします。

```
# yum upgrade 'openstack-cinder*' 'python-cinder*'
```

4. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

5. Block Storage サービスのデータベーススキーマを更新します。

```
# openstack-db --service cinder --update
```

6. Pacemaker を使用して Block Storage サービスをクリーンアップします。

```
# pcs resource cleanup openstack-cinder-volume
# pcs resource cleanup openstack-cinder-scheduler-clone
# pcs resource cleanup openstack-cinder-api-clone
```

7. Pacemaker で Block Storage サービスのリソースを再起動します。

```
# pcs resource enable openstack-cinder-volume
# pcs resource enable openstack-cinder-scheduler-clone
# pcs resource enable openstack-cinder-api-clone
```

8. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.7. ORCHESTRATION (HEAT) のアップグレード

この手順では、全コントローラーノード上で Orchestration サービスのパッケージを同時にアップグレードします。

1. Pacemaker で Orchestration リソースを停止します。

```
# pcs resource disable openstack-heat-api-clone
# pcs resource disable openstack-heat-api-cfn-clone
# pcs resource disable openstack-heat-api-cloudwatch-clone
# pcs resource disable openstack-heat-engine-clone
```

2. **pcs status** の出力で、上記のサービスが停止されるまで待ちます。
3. 適切なパッケージをアップグレードします。

```
# yum upgrade 'openstack-heat*' 'python-heat*'
```

4. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

5. Orchestration のデータベーススキーマを更新します。

```
# openstack-db --service heat --update
```

6. Pacemaker を使用して Orchestration サービスをクリーンアップします。

```
# pcs resource cleanup openstack-heat-clone
# pcs resource cleanup openstack-heat-api-cloudwatch-clone
# pcs resource cleanup openstack-heat-api-cfn-clone
# pcs resource cleanup openstack-heat-api-clone
```

7. Pacemaker で Orchestration リソースを再起動します。

```
# pcs resource enable openstack-heat-clone
# pcs resource enable openstack-heat-api-cloudwatch-clone
# pcs resource enable openstack-heat-api-cfn-clone
# pcs resource enable openstack-heat-api-clone
```

8. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.8. TELEMETRY (CEILOMETER) のアップグレード

この手順では、全コントローラーノード上で Telemetry サービスのパッケージを同時にアップグレードします。

1. Pacemaker で Telemetry リソースをすべて停止します。

```
# pcs resource disable openstack-ceilometer-central
# pcs resource disable openstack-ceilometer-api-clone
# pcs resource disable openstack-ceilometer-alarm-evaluator-clone
# pcs resource disable openstack-ceilometer-collector-clone
# pcs resource disable openstack-ceilometer-notification-clone
# pcs resource disable openstack-ceilometer-alarm-notifier-clone
# pcs resource disable delay-clone
```

2. **pcs status** の出力で、上記のサービスが停止されるまで待ちます。

3. 適切なパッケージをアップグレードします。

```
# yum upgrade 'openstack-ceilometer*' 'python-ceilometer*'
```

4. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

5. 以下のコマンドを使用して Telemetry データベーススキーマを更新します。

```
# ceilometer-dbsync
```

6. Pacemaker を使用して Telemetry サービスをクリーンアップします。

```
# pcs resource cleanup delay-clone
# pcs resource cleanup openstack-ceilometer-alarm-notifier-clone
# pcs resource cleanup openstack-ceilometer-notification-clone
# pcs resource cleanup openstack-ceilometer-collector-clone
# pcs resource cleanup openstack-ceilometer-alarm-evaluator-clone
# pcs resource cleanup openstack-ceilometer-api-clone
# pcs resource cleanup openstack-ceilometer-central
```

7. Pacemaker で Telemetry リソースをすべて再起動します。

```
# pcs resource enable delay-clone
# pcs resource enable openstack-ceilometer-alarm-notifier-clone
# pcs resource enable openstack-ceilometer-notification-clone
# pcs resource enable openstack-ceilometer-collector-clone
# pcs resource enable openstack-ceilometer-alarm-evaluator-clone
# pcs resource enable openstack-ceilometer-api-clone
# pcs resource enable openstack-ceilometer-central
```

8. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

重要

以前のバージョンの Telemetry サービスは、現在のバージョンでは非推奨となっている **rpc_backend** パラメーターの値を使用していました。**/etc/ceilometer/ceilometer.conf** ファイルの **rpc_backend** パラメーターが以下のように設定されていることを確認してください。

```
rpc_backend=rabbit
```

6.9. コントローラーノード上の COMPUTE サービス (NOVA) のアップグレード

この手順では、全コントローラーノード上で Compute サービスのパッケージを同時にアップグレードします。

1. Pacemaker で Compute リソースをすべて停止します。

```
# pcs resource disable openstack-nova-novncproxy-clone
# pcs resource disable openstack-nova-consoleauth-clone
# pcs resource disable openstack-nova-conductor-clone
# pcs resource disable openstack-nova-api-clone
# pcs resource disable openstack-nova-scheduler-clone
```

2. **pcs status** の出力で、上記のサービスが停止されるまで待ちます。

3. 適切なパッケージをアップグレードします。

```
# yum upgrade 'openstack-nova*' 'python-nova*'
```

4. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

5. Compute のデータベーススキーマを更新します。

```
# openstack-db --service nova --update
```

6. コンピュートホストのローリングアップグレードを行うには、明示的に API のバージョンの制限を設定して、Liberty と Mitaka の環境間の互換性を確保する必要があります。

   コントローラーノードまたはコンピュートノードで Compute サービスを起動する前に、**nova.conf** ファイルの **[upgrade_levels]** セクションで、**compute** オプションを以前の Red Hat OpenStack Platform バージョン (**liberty**) に設定します。

```
# crudini --set /etc/nova/nova.conf upgrade_levels compute
liberty
```

   これにより、コントローラーノードは、以前のバージョンを使用しているコンピュートノードとの通信が引き続き可能となります。

まず、コントローラーの 1 つで **pcs resource unmanage** を実行して Compute リソースの管理を解除する必要があります。

```
# pcs resource unmanage openstack-nova-novncproxy-clone
# pcs resource unmanage openstack-nova-consoleauth-clone
# pcs resource unmanage openstack-nova-conductor-clone
# pcs resource unmanage openstack-nova-api-clone
# pcs resource unmanage openstack-nova-scheduler-clone
```

コントローラーすべてで全サービスを再起動します。

```
# openstack-service restart nova
```

コンピュートホストをすべて OpenStack Mitaka にアップグレードした後で、Pacemaker が制御できるように戻す必要があります。

```
# pcs resource manage openstack-nova-scheduler-clone
# pcs resource manage openstack-nova-api-clone
# pcs resource manage openstack-nova-conductor-clone
# pcs resource manage openstack-nova-consoleauth-clone
# pcs resource manage openstack-nova-novncproxy-clone
```

7. Pacemaker で Compute リソースをすべてクリーンアップします。

```
# pcs resource cleanup openstack-nova-scheduler-clone
# pcs resource cleanup openstack-nova-api-clone
# pcs resource cleanup openstack-nova-conductor-clone
# pcs resource cleanup openstack-nova-consoleauth-clone
# pcs resource cleanup openstack-nova-novncproxy-clone
```

8. Pacemaker で Compute リソースをすべて再起動します。

```
# pcs resource enable openstack-nova-scheduler-clone
# pcs resource enable openstack-nova-api-clone
# pcs resource enable openstack-nova-conductor-clone
# pcs resource enable openstack-nova-consoleauth-clone
# pcs resource enable openstack-nova-novncproxy-clone
```

9. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.10. OPENSTACK NETWORKING (NEUTRON) のアップグレード

この手順では、全コントローラーノード上で Networking サービスのパッケージを同時にアップグレードします。

1. Pacemaker による OpenStack Networking クリーンアップスクリプトがトリガーされないようにします。

```
# pcs resource unmanage neutron-ovs-cleanup-clone
# pcs resource unmanage neutron-netns-cleanup-clone
```

2. Pacemaker で OpenStack Networking のリソースを停止します。

```
# pcs resource disable neutron-server-clone
# pcs resource disable neutron-openvswitch-agent-clone
# pcs resource disable neutron-dhcp-agent-clone
# pcs resource disable neutron-l3-agent-clone
# pcs resource disable neutron-metadata-agent-clone
```

3. 適切なパッケージをアップグレードします。

```
# yum upgrade 'openstack-neutron*' 'python-neutron*'
```

4. **neutron.conf** ファイルで有効化された高度な OpenStack Networking サービスのパッケージをインストールします。たとえば、**openstack-neutron-vpnaas**、**openstack-neutron-fwaas**、**openstack-neutron-lbaas** などのサービスをアップグレードするには、以下のコマンドを実行します。

```
# yum install openstack-neutron-vpnaas
# yum install openstack-neutron-fwaas
# yum install openstack-neutron-lbaas
```

これらのパッケージをインストールすると、対応する設定ファイルが作成されます。

5. **neutron.conf** ファイルの VPNaaS および LBaaS サービスのエントリーの場合は、「service_provider」エントリーを **/etc/neutron** 配下の対応する **neutron-*aas.conf** ファイルにコピーし、**neutron.conf** ファイルではこれらのエントリーをコメント化します。

FWaaS サービスのエントリーの場合は、**service_provider** パラメーターを **neutron.conf** ファイルに **残す必要があります**。

6. LBaaS エージェントを実行する全ノードで、**openstack-neutron-lbaas** パッケージをインストールします。

```
# yum install openstack-neutron-lbaas
```

7. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

8. OpenStack Networking のデータベーススキーマを更新します。

```
# openstack-db --service neutron --update
```

9. Pacemaker で OpenStack Networking のリソースをクリーンアップします。

```
# pcs resource cleanup neutron-metadata-agent-clone
# pcs resource cleanup neutron-l3-agent-clone
# pcs resource cleanup neutron-dhcp-agent-clone
# pcs resource cleanup neutron-openvswitch-agent-clone
# pcs resource cleanup neutron-server-clone
```

10. Pacemaker で OpenStack Networking のリソースを再起動します。

```
# pcs resource enable neutron-metadata-agent-clone
# pcs resource enable neutron-l3-agent-clone
```

```
# pcs resource enable neutron-dhcp-agent-clone
# pcs resource enable neutron-openvswitch-agent-clone
# pcs resource enable neutron-server-clone
```

11. クリーンアップエージェントを Pacemaker の制御対象に戻します。

```
# pcs resource manage neutron-ovs-cleanup-clone
# pcs resource manage neutron-netns-cleanup-clone
```

12. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.11. DASHBOARD (HORIZON) のアップグレード

この手順では、全コントローラーノード上で Dashboard のパッケージを同時にアップグレードします。

1. Pacemaker で Dashboard リソースを停止します。

```
# pcs resource disable httpd-clone
```

2. **pcs status** の出力で、このサービスが停止されるまで待ちます。
3. 適切なパッケージをアップグレードします。

```
# yum upgrade httpd 'openstack-dashboard*' 'python-django*'
```

4. 更新したユニットファイルを有効にするために **systemd** を再読み込みします。

```
# systemctl daemon-reload
```

5. 全コントローラーで Web サーバーを再起動してすべての変更を適用します。

```
# service httpd restart
```

6. Pacemaker で Dashboard リソースをクリーンアップします。

```
#  pcs resource cleanup httpd-clone
```

7. Pacemaker で Dashboard リソースを再起動します。

```
#  pcs resource enable httpd-clone
```

8. **pcs status** の出力で、上記のリソースが実行中と表示されるまで待ちます。

6.12. コンピュートノード (NOVA) のアップグレード

この手順では、単一のコンピュートノードのパッケージをアップグレードします。以下のステップを各コンピュートノードで個別に実行してください。

コンピュートホストのローリングアップグレードを行うには、明示的に API のバージョンの制限を設定して、Liberty と Mitaka の環境間の互換性を確保する必要があります。

コントローラーノードまたはコンピュートノードで Compute サービスを起動する前に、**nova.conf** ファイルの **[upgrade_levels]** セクションで、**compute** オプションを以前の Red Hat OpenStack Platform バージョン (**liberty**) に設定します。

```
# crudini --set /etc/nova/nova.conf upgrade_levels compute liberty
```

これにより、コントローラーノードは、以前のバージョンを使用しているコンピュートノードとの通信が引き続き可能となります。

1. ホスト上の OpenStack サービスをすべて停止します。

   ```
   # openstack-service stop
   ```

2. すべてのパッケージをアップグレードします。

   ```
   # yum upgrade
   ```

3. ホスト上の OpenStack サービスをすべて起動します。

   ```
   # openstack-service start
   ```

4. 全ホストをアップグレードした後には、以前のステップで設定した API の制限を削除します。全ホスト上で以下のコマンドを実行します。

   ```
   # crudini --del /etc/nova/nova.conf upgrade_levels compute
   ```

5. ホスト上の OpenStack サービスをすべて再起動します。

   ```
   # openstack-service restart
   ```

6.13. アップグレード後のタスク

個別サービスのアップグレードをすべて完了した後には、全ノードで完全なパッケージアップグレードを行う必要があります。

```
# yum upgrade
```

このコマンドは、すべてのパッケージを最新の状態にします。実行中のプロセスが、配下のバイナリーの更新されたバージョンを使用するようにするには、OpenStack ホストの再起動を後日にスケジューリングする必要があります。

上記の操作によって生成された設定ファイルを確認します。アップグレードされたパッケージで、Red Hat OpenStack Platform 9 バージョンのサービスに適した **.rpmnew** ファイルがインストールされているはずです。

新しいバージョンの OpenStack サービスでは、特定の設定オプションが非推奨になっている可能性があります。このような非推奨の設定オプションが原因で今後のアップグレードの際に問題が発生する可能性があるため、非推奨の警告については OpenStack のログも参照してください。各サービスで新規追加/更新された設定オプションや非推奨となった設定オプションについての詳しい説明は、Red Hat OpenStack Platform ドキュメントスイート で『Configuration Reference』を参照してください。

第7章 DIRECTOR を使用しない環境向けの追加の手順

以下の項では、director で管理されていない Red Hat OpenStack Platform 環境を対象とするいくつかの追加手順について説明します。これらの手順は、OpenStack Platform エコシステム内の変化に対応しており、Red Hat OpenStack Platform 9 にアップグレードした後に実行するのが最適です。

7.1. OPENSTACK IDENTITY API から WSGI SERVICE へのアップグレード

このステップでは、OpenStack Identity (**keystone**) API がスタンドアロンサービスの代わりに **httpd** 下で Web Server Gateway Interface (WSGI) アプレットを実行するようにアップグレードします。このプロセスにより、スタンドアロンの **openstack-keystone** サービスは無効化され、WSGI アプレットを有効化するために必要な設定がインストールされます。

1. OpenStack Identity サービスを無効にします。このステップは、高可用性のコントローラーノードを使用するかどうかによって異なります。
   - 高可用性を使用していない環境の場合:

     ```
     $ sudo systemctl stop openstack-keystone
     ```

   - 高可用性を使用している環境の場合:

     ```
     $ sudo pcs resource disable openstack-keystone
     ```

2. 各コントローラーで、以下の OpenStack Identity サービスの WSGI アプレットを **/var/www/cgi-bin/** 内の新規ディレクトリーにコピーします。
   - Admin WSGI アプレット: **/usr/bin/keystone-wsgi-admin**
   - Public WSGI アプレット: **/usr/bin/keystone-wsgi-public**

   以下に例を示します。

   ```
   $ sudo mkdir /var/www/cgi-bin/keystone
   $ sudo cp /usr/bin/keystone-wsgi-admin /var/www/cgi-
   bin/keystone/keystone-wsgi-admin
   $ sudo cp /usr/bin/keystone-wsgi-public /var/www/cgi-
   bin/keystone/keystone-wsgi-public
   ```

3. 各コントローラーで、Admin WSGI OpenStack Identity サービス向けの仮想ホスト設定ファイル (**10-keystone_wsgi_admin.conf**) を作成します。このファイルを **/etc/httpd/conf.d/** に保存します。仮想ホストファイルの内容は、以下のような形式で記載する必要があります。

   ```
   Listen 35357

   <VirtualHost *:35357>
     DocumentRoot "/var/www/cgi-bin/keystone"

     <Directory "/var/www/cgi-bin/keystone">
       Options Indexes FollowSymLinks MultiViews
       AllowOverride None
   ```

```
    Require all granted
  </Directory>

  ErrorLog "/var/log/httpd/keystone_wsgi_admin_error.log"
  ServerSignature Off
  CustomLog "/var/log/httpd/keystone_wsgi_admin_access.log"
combined

  WSGIApplicationGroup %{GLOBAL}
  WSGIDaemonProcess keystone_admin display-name=keystone-admin
group=keystone processes=1 threads=12 user=keystone
  WSGIProcessGroup keystone_admin
  WSGIScriptAlias / "/var/www/cgi-bin/keystone/keystone-admin"
  WSGIPassAuthorization On
</VirtualHost>
```

4. 各コントローラーで、Admin WSGI OpenStack Identity サービス向けの仮想ホスト設定ファイル (**10-keystone_wsgi_public.conf**) を作成します。このファイルを **/etc/httpd/conf.d/** に保存します。仮想ホストファイルの内容は、以下のような形式で記載する必要があります。

```
Listen 5000

<VirtualHost *:5000>>
  DocumentRoot "/var/www/cgi-bin/keystone"

  <Directory "/var/www/cgi-bin/keystone">
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Require all granted
  </Directory>

  ErrorLog "/var/log/httpd/keystone_wsgi_public_error.log"
  ServerSignature Off
  CustomLog "/var/log/httpd/keystone_wsgi_public_access.log"
combined

  WSGIApplicationGroup %{GLOBAL}
  WSGIDaemonProcess keystone_public display-name=keystone-public
group=keystone processes=1 threads=12 user=keystone
  WSGIProcessGroup keystone_public
  WSGIScriptAlias / "/var/www/cgi-bin/keystone/keystone-public"
  WSGIPassAuthorization On
</VirtualHost>
```

5. **httpd** サービスを再起動します。このステップは、高可用性のコントローラーノードを使用しているかどうかによって異なります。

   - 高可用性を使用していない環境の場合:

     ```
     $ sudo systemctl restart httpd
     ```

   - 高可用性を使用している環境の場合:

     ```
     $ sudo pcs resource restart httpd
     ```

第8章 DIRECTOR ベースのアップグレードのトラブルシューティング

本項では、両シナリオのトラブルシューティングのためのアドバイスを記載します。

8.1. アンダークラウドのアップグレード

アンダークラウドのアップグレードコマンド (**openstack undercloud upgrade**) が失敗した場合には、以下のアドバイスに従って、アップグレードの進捗の妨げとなっている問題を特定してください。

- **openstack undercloud upgrade** コマンドは、実行中に進捗ログを出力します。アップグレードのプロセス中にエラーが発生した場合には、エラーの発生時にこのコマンドは停止します。この情報を使用して、アップグレードの進捗を妨げている問題を特定してください。
- **openstack undercloud upgrade** コマンドは、Puppet を実行してアンダークラウドサービスを設定します。これにより、以下のディレクトリーで便利な Puppet のレポートが生成されます。
  - **/var/lib/puppet/state/last_run_report.yaml**: 最後の Puppet レポートは、アンダークラウド向けに生成されます。このファイルは、問題のある Puppet のアクションの原因を表示します。
  - **/var/lib/puppet/state/last_run_summary.yaml**: **last_run_report.yaml** ファイルのサマリー
  - **/var/lib/puppet/reports**: アンダークラウドの全 Puppet レポート

    この情報を使用して、アップグレードプロセスを妨げている問題を特定します。
- エラーが発生しているサービスがあるかどうかを確認します。

  ```
  $ sudo systemctl -t service
  ```

  エラーが発生しているサービスがある場合は、対応するログを確認します。たとえば、**openstack-ironic-api** でエラーが発生している場合には、以下のコマンドを使用してこのサービスのログを確認します。

  ```
  $ sudo journalctl -xe -u openstack-ironic-api
  $ sudo tail -n 50 /var/log/ironic/ironic-api.log
  ```

アンダークラウドのアップグレードを妨げていた問題を修正した後に、アップグレードのコマンドを再度実行します。

```
$ openstack undercloud upgrade
```

アップグレードのコマンドをもう 1 度開始して、アンダークラウドを設定します。

8.2. オーバークラウドのアップグレード

オーバークラウドのアップグレードプロセスで障害が発生した場合には、以下のアドバイスに従ってアップグレードプロセスを妨げている問題を特定します。

- Heat スタックの一覧を確認して、**UPDATE_FAILED** のステータスがあるスタックを特定します。以下のコマンドで、そのようなスタックを特定します。

```
$ heat stack-list --show-nested | awk -F "|" '{ print $3,$4 }' |
grep "UPDATE_FAILED" | column -t
```

  エラーの発生したスタックとそのスタックのテンプレートを表示して、スタックが失敗した原因を究明します。

```
$ heat stack-show overcloud-Controller-qyoy54dyhrll-1-gtwy5bgta3np
$ heat template-show overcloud-Controller-qyoy54dyhrll-1-
gtwy5bgta3np
```

- 全コントローラーノード上で Pacemaker が正しく実行されていることを確認します。必要な場合は、コントローラーノードにログインして、コントローラークラスターを再起動します。

```
$ sudo pcs cluster start
```

オーバークラウドのアップグレードを妨げていた問題を修正した後に、アップグレードを試みて失敗したステップの **openstack overcloud deploy** コマンドを再度実行します。アップグレードプロセスで、**major-upgrade-pacemaker-init.yaml** を指定して実行する最初の **openstack overcloud deploy** コマンドの例を以下に示します。

```
$ openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/net-
single-nic-with-vlans.yaml  \
  -e network_env.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/major-
upgrade-pacemaker-init.yaml
```

openstack overcloud deploy は、オーバークラウドのスタックの更新を再試行します。