Red Hat OpenStack Platform 10

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

Last Updated: 2018-01-23

Red Hat OpenStack Platform 10 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 9 (Mitaka) から 10 (Newton) 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 10 (Newton)** をターゲットとします。

Red Hat は、Red Hat Enterprise Linux 7.3 をベースとする Red Hat OpenStack Platform 11 へのアップグレードのみをサポートしており、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 10 へのアップグレードと、バージョン 10 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 10 には以下のアップグレードシナリオを推奨しています。次の表には、各シナリオについての説明をまとめています。

> **警告**
>
> Open vSwitch (OVS) 2.4.0 を OVS 2.5.0 にアップグレードせずに Red Hat Enterprise Linux 7.3 カーネルへのアップグレードを実行しないようにしてください。カーネルのみがアップグレードされると、OVS は機能しなくなります。

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 10 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 9 から 10 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。

メソッド	説明
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。
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。

警告

Open vSwitch (OVS) 2.4.0 を OVS 2.5.0 にアップグレードせずに Red Hat Enterprise Linux 7.3 カーネルへのアップグレードを実行しないようにしてください。カーネルのみがアップグレードされると、OVS は機能しなくなります。

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-6.2-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 10 for RHEL 7 (RPMs)	**`rhel-7-server-openstack-10-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 2.0 for Red Hat Enterprise Linux 7 Server (RPMs)	**`rhel-7-server-rhceph-2-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 2.0 for Red Hat Enterprise Linux 7 Server (RPMs)	**`rhel-7-server-rhceph-2-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。

Red Hat Ceph Storage Tools 2 for Red Hat Enterprise Linux 7 Workstation (RPMs)	**`rhel-7-server-rhceph-2-tools-rpms`**	(Ceph Storage ノード向け) Ceph オブジェクトストレージに必要な Rados REST ゲートウェイを提供します。

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 10 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**
>
> With High Availaibility for Compute instances (or Instance HA, as described in High Availability for Compute Instances), upgrades or scale-up operations are not possible. Any attempts to do so will fail.
>
> If you have Instance HA enabled, disable it before performing an upgrade or scale-up. To do so, perform a **rollback** as described in Rollback.

この手順は、いずれの場合も以下のワークフローに従って作業を行います。

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

Red Hat は、更新を実行する前に以下の作業を済ませておくことを推奨します。

- アップグレード手順のステップを開始する前に、アンダークラウドノードのバックアップを実行してください。バックアップの手順については、『Back Up and Restore the Director Undercloud』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更新の手順を実行します。
- 更新を実行するにあたってアドバイスやサポートが必要な場合には、Red Hat までご連絡ください。

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*"
```

注記

再起動後に **openstack-nova-compute** が有効になるまでに約 10 分かかる場合があります。

オーバー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-10.0.tar
/usr/share/rhosp-director-images/ironic-python-agent-latest-10.0.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。これには、以下の 2 つのステップを実行する必要があります。

1. 元の **openstack overcloud deploy** コマンドに **--update-plan-only** オプションを追加して、現在のプランを更新します。以下に例を示します。

```
$ openstack overcloud deploy --update-plan-only \
  --templates  \
  -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>|...]
```

--update-plan-only のオプションを指定すると、director に保管されているオーバークラウドのプランのみが更新されます。**-e** オプションを使用して、オーバークラウドと関連のある環境ファイルとその更新パスを追加します。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

2. **openstack overcloud update** コマンドを使用して、全ノードでパッケージの更新を実行します。以下に例を示します。

```
$ openstack overcloud update stack -i overcloud
```

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。

これで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 Nodes」の手順に従って各ノードを再起動します。

2.4. アップグレード後の注意事項

Red Hat OpenStack Platform 10 の最新の更新には、ライブマイグレーションの機能に必要な **OS::TripleO::Services::Sshd** コンポーザブルサービスが含まれています。初期リリースでは、director のコアテンプレートコレクションにこのサービスは含まれていませんでしたが、**openstack-tripleo-heat-templates-5.2.0-12** パッケージ以降のバージョンで含まれるようになりました。

デフォルトロールデータにはこのサービスが含まれており、director は更新時にオーバークラウド上にこのサービスをインストールします。

カスタムのロールデータファイルを使用する場合には、各オーバークラウドロールに **OS::TripleO::Services::Sshd** サービスを追加してからオーバークラウドのスタックを更新し、この新規サービスが含まれるようにします。

詳しくは、「Red Hat OpenStack Platform director (TripleO) CVE-2017-2637 bug and Red Hat OpenStack Platform」を参照してください。

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 9『**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**
>
> With High Availaibility for Compute instances (or Instance HA, as described in High Availability for Compute Instances), upgrades or scale-up operations are not possible. Any attempts to do so will fail.
>
> If you have Instance HA enabled, disable it before performing an upgrade or scale-up. To do so, perform a **rollback** as described in Rollback.

本章では、環境のアップグレードの方法を考察します。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。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 9 から Red Hat OpenStack Platform 10 へのアップグレードです。

この手順は、いずれの場合も以下のワークフローに従って作業を行います。

1. Red Hat OpenStack Platform director パッケージのアップグレード
2. Red Hat OpenStack Platform director でのオーバークラウドイメージのアップグレード
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アップグレード

3.1. アップグレードのサポートステートメント

アップグレードプロセスを成功させるには、メジャーバージョン間の変更に対応するための準備が必要です。以下のサポートステートメントを確認して、Red Hat OpenStack Platform のアップグレードのプランニングに役立ててください。

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、それらは完全に網羅されません。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、『Back Up and Restore the Director Undercloud』ガイドを参照してください。
- カスタマイズされた設定を使用するアップグレード手順は、実稼働環境で実行する前にテスト環境で実行してください。
- このアップグレードの実行するにあたって懸念がある場合には、作業を開始する前に Red Hat のサポートチームにご連絡いただき、アップグレードの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 を使用せずにオーバークラウドの機能をカスタマイズした場合は、以下のステップを実行してください。

- その機能を無効にします。
- オーバークラウドをアップグレードします。
- アップグレードの完了後に機能を再度有効にします。

これは、アップグレードがすべて完了するまで、カスタマイズされた機能が使用できないことを意味します。

Red Hat OpenStack Platform director 10 は、Red Hat OpenStack Platform の以前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 10 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 10	Red Hat OpenStack Platform 9 および 10	Red Hat OpenStack Platform 9 および 10	Red Hat OpenStack Platform 9 および 10

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 9 の場合: **`/usr/share/openstack-tripleo-heat-templates/mitaka/`**

以下に例を示します。

```
$ openstack overcloud deploy --templates /usr/share/openstack-tripleo-heat-templates/mitaka/ [OTHER_OPTIONS]
```

アップグレードの一般的なヒントは以下のとおりです。

- 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。

- 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 に連絡して、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。

3.2. アップグレード前の重要な注記

- Red Hat OpenStack Platform 10 では、Red Hat Enterprise Linux 7.3 で利用可能となった、一部の新しいカーネルパラメーターを使用します。アンダークラウドとオーバークラウドを Red Hat Enterprise Linux 7.3 と Open vSwitch 2.5 にアップグレードしてください。アンダークラウドとオーバークラウドに対してパッケージの更新を実行する手順は、「director ベースの環境: マイナーバージョンへの更新の実行」を参照してください。カーネルを最新バージョンに更新した場合には、システムを再起動して、新しいカーネルパラメーターが有効になるようにしてください。

- OpenStack Platform 10 のアップグレード手順を実行すると、新しいコンポーザブルアーキテクチャーに移行されます。これは、以前のバージョンでは Pacemaker によって管理されていたサービスの多くが **systemd** の管理機能を使用することになります。このため、Pacemaker によって管理されるリソースの数が削減されます。

- 以前のバージョンの Red Hat OpenStack Platform では、OpenStack Telemetry (ceilometer) はメトリックのストレージに自らのデータベースを使用していました。Red Hat OpenStack Platform 10 では、OpenStack Telemetry のデフォルトのバックエンドとして、OpenStack Telemetry Metrics (gnocchi) を使用します。メトリックデータの移行プランはない点に注意してください。

- OpenStack Telemetry Alarms (aodh) では、コンポジットアラームの方が支持されるようになったため、コンビネーションアラームは非推奨となりました。次の点に注意してください。
  - Aodh は、デフォルトでは、コンビネーションアラームは公開しません。
  - 新規パラメーター **EnableCombinationAlarms** は、オーバークラウドでコンビネーションアラームを有効にします。これは、デフォルト値は **false** です。OpenStack Platform 10 で引き続きコンビネーションアラームを使用するには、**true** に設定してください。
  - OpenStack Platform 10 には、コンポジットアラームに移行するためのマイグレーションスクリプト (**aodh-data-migration**) が含まれています。このデータの移行手順は、本ガイドの「OpenStack Telemetry Alarming データベースの移行」に記載しています。このスクリプトを実行して、アラームをコンポジットに必ず変換してください。
  - コンビネーションアラームに対するサポートは、次のリリースから廃止されます。

3.3. オーバークラウドの確認

アップグレードを実行する前に、オーバークラウドが安定した状態であることをチェックします。director で以下のステップを実行して、オーバークラウド内の全サービスが稼働していることを確認してください。

1. 高可用性サービスのステータスを確認します。

   ```
   ssh heat-admin@[CONTROLLER_IP] "sudo pcs resource cleanup ; sleep 60
   ; sudo pcs status"
   ```

   [CONTROLLER_IP] はコントローラーノードの IP アドレスに置き換えます。このコマンドは、オーバークラウドの Pacemaker クラスターを最新の状態に更新し、60 秒待ってからそのクラスターのステータスを報告します。

2. オーバークラウドノードで、失敗した OpenStack Platform **systemd** サービスがあるかどうかを確認します。以下のコマンドは、失敗したサービスを全ノードで確認します。

```
$ for IP in $(openstack server list -c Networks -f csv | sed '1d' |
sed 's/"//g' | cut -d '=' -f2) ; do echo "Checking systemd services
on $IP" ; ssh heat-admin@$IP "sudo systemctl list-units 'openstack-
*' 'neutron-*' --state=failed --no-legend" ; done
```

3. 各ノードで **os-collect-config** が実行中であることを確認します。以下のコマンドで、このサービスのステータスを各ノードでチェックします。

```
$ for IP in $(openstack server list -c Networks -f csv | sed '1d' |
sed 's/"//g' | cut -d '=' -f2) ; do echo "Checking os-collect-config
on $IP" ; ssh heat-admin@$IP "sudo systemctl list-units 'os-collect-
config.service' --no-legend" ; done
```

3.4. アンダークラウドのアップグレード

3.4.1. director のアップグレード

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-
9-rpms --disable=rhel-7-server-openstack-9-director-rpms
$ sudo subscription-manager repos --enable=rhel-7-server-openstack-
10-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アップグレード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。

アンダークラウドで **/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、アンダークラウドを再起動します。

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*"
```

注記

再起動後に **openstack-nova-compute** が有効になるまでに約 10 分かかる場合があります。

オーバー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
```

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 -Nar /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。

3.4.2. director 上のオーバークラウドイメージのアップグレード

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-10.0.tar
/usr/share/rhosp-director-images/ironic-python-agent-latest-10.0.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。

3.4.3. 以前のテンプレートバージョンの使用と比較

アップグレードプロセスにより、最新のオーバークラウドバージョンに対応したコア Heat テンプレートの新しいセットがインストールされます。Red Hat OpenStack Platform のリポジトリーには、**openstack-tripleo-heat-templates-compat** パッケージ内のコアテンプレートコレクションの以前のバージョンが維持されています。このパッケージは、以下のコマンドを実行してインストールします。

```
$ sudo yum install openstack-tripleo-heat-templates-compat
```

これにより、Heat テンプレートコレクションの **compat** ディレクトリー (**/usr/share/openstack-tripleo-heat-templates/compat**) に以前のテンプレートがインストールされ、以前のバージョン (**mitaka**) から命名された **compat** へのリンクが作成されます。これらのテンプレートは、アップグレードされた director との後方互換性があるので、最新バージョンの director を使用して以前のバージョンのオーバークラウドをインストールすることができます。

以前のバージョンを最新のバージョンと比較すると、アップグレード中にオーバークラウドに加えられる変更内容を確認することができます。現在のテンプレートコレクションを以前のバージョンと比較する必要がある場合は、以下のプロセスを使用してください。

1. コア Heat テンプレートの一時的なコピーを作成します。

```
$ cp -a /usr/share/openstack-tripleo-heat-templates /tmp/osp10
```

2. 以前のバージョンをそれ独自のディレクトリーに移動します。

```
$ mv /tmp/osp10/compat /tmp/osp9
```

3. 両ディレクトリーのコンテンツに対して **diff** を実行します。

```
$ diff -urN /tmp/osp9 /tmp/osp10
```

このコマンドにより、バージョン間におけるコアテンプレートの変更が表示されます。この内容を確認すると、オーバークラウドのアップグレード中にどのような動作が行われるかがわかります。

3.5. オーバークラウドのアップグレード前の設定

3.5.1. Red Hat サブスクリプションの詳細

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 10 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 10 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。

3.5.2. SSL の設定

SSL を使用するオーバークラウドをアップグレードする場合には、以下の点を認識しておいてください。

- ネットワークの設定には、**PublicVirtualFixedIPs** パラメーターを以下の形式で指定する必要があります。

  ```
    PublicVirtualFixedIPs: [{'ip_address':'192.168.200.180'}]
  ```

  この行を、ネットワーク環境ファイルの **parameter_defaults** のセクション下に追加してください。

- SSL エンドポイントを記載した新規環境ファイル。このファイルは、オーバークラウドへのアクセスに IP アドレスまたは DNS を使用するかによって異なります。
  - IP アドレスを使用する場合には、**/usr/share/openstack-tripleo-heat-templates/environments/tls-endpoints-public-ip.yaml** を使用してください。
  - DNS を使用する場合には、**/usr/share/openstack-tripleo-heat-templates/environments/tls-endpoints-public-dns.yaml** を使用してください。

- SSL/TLS の設定に関する詳しい情報は、 **『Red Hat OpenStack Platform Advanced Overcloud Customization』**ガイドの「Enabling SSL/TLS on the Overcloud」の章を参照してください。

3.5.3. Ceph Storage

カスタムの **storage-environment.yaml** ファイルを使用する場合には、**resource_registry** のセクションに以下の新規リソースが含まれていることを確認してください。

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

これらのリソースにより、Ceph Storage コンポーザブルサービスが Red Hat OpenStack Platform 10 で確実に有効化されます。Red Hat OpenStack Platform 10 のデフォルトの **storage-environment.yaml** ファイルが更新され、これらのリソースが含まれるようになりました。

3.5.4. オーバークラウドのパラメーター

アップグレードを実行するにあたっては、オーバークラウドパラメーターについての以下の情報に注意してください。

- Red Hat OpenStack Platform 10 のデフォルトのタイムゾーンは UTC です。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。

- カスタムの **ServiceNetMap** を指定してオーバークラウドをアップグレードする場合には、新規サービスに最新の **ServiceNetMap** を追加するようにしてください。**ServiceNetMapDefaults** パラメーターで定義されるデフォルトのサービス一覧は、**network/service_net_map.j2.yaml** ファイルにあります。カスタムの **ServiceNetMap** の使用方法については、 **『Advanced Overcloud Customization』**の「Isolating Networks」のセクションを参照してください。

- 新たなコンポーザブルサービスアーキテクチャーにより、OpenStack Image サービス (Glance) 向けに NFS バックエンドを設定するためのパラメーターが変更されました。新規パラメーターは以下のとおりです。

  GlanceNfsEnabled

  イメージストレージ用の共有を管理するための Pacemaker を有効にするパラメーター。無効に設定されている場合には、オーバークラウドはコントローラーノードのファイルシステムにイメージを保管します。**true** に設定してください。

  GlanceNfsShare

  イメージストレージをマウントするための NFS 共有 (例: 192.168.122.1:/export/glance)

  GlanceNfsOptions

  イメージストレージ用の NFS マウントオプション

- 新しいコンポーザブルサービスアーキテクチャーにより、一部の設定フックの構文が変更されました。設定前または設定後のフックを使用してカスタムのスクリプトを環境に提供する場合には、カスタム環境ファイルの構文をチェックします。 **『Advanced Overcloud Customization』** ガイドで以下のセクションを参照してください。

- 「Customizing Overcloud Pre-Configuration」
- 「Customizing Overcloud Post-Configuration」

3.5.5. カスタムのコアテンプレート

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 -Nar ~/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。

3.6. オーバークラウドのアップグレード

3.6.1. 概要とワークフロー

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。各セクションを順を追って進み、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-ceilometer-wsgi-mitaka-newton.yaml**: OpenStack Telemetry (‘Ceilometer’) を WSGI サービスに変換します。
- **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。
- (オプション) **major-upgrade-remove-sahara.yaml**: オーバークラウドから OpenStack Clustering (**sahara**) を削除します。このステップは、OpenStack Platform 9 と 10 の間の違いを調整します。詳しい情報は、「コントローラーノードのアップグレード」 を参照してください。
- **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。
- **major-upgrade-aodh-migration.yaml**: OpenStack Telemetry Alarming (**aodh**) サービスのデータベースを MongoDB から MariaDB に移行します。

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アップグレードされます。

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-ceilometer-wsgi-mitaka-newton.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-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。
3. 各 Object Storage ノードで **upgrade-non-controller.sh** を実行します。
4. **major-upgrade-pacemaker.yaml** と、オプションの **major-upgrade-remove-sahara.yaml** の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。
5. 各 Ceph Storage ノードで **upgrade-non-controller.sh** を実行します。
6. 各 Compute ノードで **upgrade-non-controller.sh** を実行します。
7. **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。
8. **major-upgrade-aodh-migration.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。

3.6.2. OpenStack Telemetry の WSGI サービスへのアップグレード

このステップでは、OpenStack Telemetry (**ceilometer**) サービスをスタンドアロンのサービスとしてではなく **httpd** 下で Web Server Gateway Interface (WSGI) アプレットとして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-ceilometer-api** 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-ceilometer-wsgi-mitaka-newton.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すべてのオプションとカスタムの環境ファイルも必ず指定してください。

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-ceilometer-wsgi-mitaka-newton.yaml \
  --ntp-server pool.ntp.org
```

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。

3.6.3. 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すべてのオプションとカスタムの環境ファイルも必ず指定してください。

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 \
  --ntp-server pool.ntp.org
```

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。

3.6.4. 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 `openstack server list -c Name -f value --name objectstorage` ; 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。

各オブジェクトストレージノードで **/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 Object Storage 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。

3.6.5. 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すべてのオプションとカスタムの環境ファイルも必ず指定してください。

コントローラーノードでは、OpenStack Data Processing (**sahara**) を有効な状態のままに保持するかどうかによって、追加のファイルが必要な場合があります。OpenStack Platform 9 では、OpenStack Data Processing がデフォルトのオーバークラウド向けに自動的にインストールされていました。OpenStack Platform 10 では、ユーザーは環境ファイルを明示的に指定して、OpenStack Data Processing を有効化する必要があります。これには、以下のような意味があります。

- OpenStack Data Processing が必要なくなった場合には、デプロイメントに **major-upgrade-remove-sahara.yaml** ファイルを指定します。

- OpenStack Data Processing を保持する場合には、デプロイメントで **major-upgrade-remove-sahara.yaml** ファイルを追加しないでください。オーバークラウドのアップグレードが完了したら、**/usr/share/openstack-tripleo-heat-templates/environments/services/sahara.yaml** を追加して、サービスを有効かつ設定済みの状態に維持します。

openstack overcloud deploy コマンドで必須およびオプションのファイルを指定する例を以下に示します。

```
$ 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
  -e /usr/share/openstack-tripleo-heat-templates/environments/major-
upgrade-remove-sahara.yaml \
  --ntp-server pool.ntp.org
```

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

以下の点に注意してください。

- 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。

- このステップにより、コントローラークラスターの Pacemaker の設定が変更されるので、特定の高可用性機能はアップグレードによって一時的に無効になります。

各コントローラーで **/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、各コントローラーを再起動します。

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。

重要

OpenStack Platform 10 のアップグレード手順を実行すると、新しいコンポーザブルアーキテクチャーに移行されます。これは、以前のバージョンでは Pacemaker によって管理されていたサービスの多くが **systemd** の管理機能を使用することになります。このため、Pacemaker によって管理されるリソースの数が削減されます。

3.6.6. 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 `openstack server list -c Name -f value --name ceph` ; do
upgrade-non-controller.sh --upgrade $NODE ; done
```

各 Ceph Storage 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 Ceph Storage ノードを再起動します。

1. Ceph MON またはコントローラーノードにログインして、Ceph Storage クラスターのリバランスを一時的に無効にします。

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

2. 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。

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. 完了したら、Ceph MON またはコントローラーノードにログインして、クラスターのリバランス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** を報告してい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。

3.6.7. 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。

```
$ openstack server 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] --all-projects
```

3. 無効にしたホストから各インスタンスを移行します。以下のコマンドの 1 つを使用します。

   a. 選択した特定のホストにインスタンスを移行します。

   ```
   $ openstack server migrate [instance-id] --live [target-host]--wait
   ```

   b. **nova-scheduler** により対象のホストが自動的に選択されるようにします。

   ```
   $ nova live-migration [instance-id]
   ```

   注記

   nova コマンドで非推奨の警告が表示される可能性がありますが、安全に無視することができます。

4. 移行が完了するまで待ちます。

5. インスタンスがコンピュートノードから移行されたことを確認します。

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

6. 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、コンピュートノードを再起動します。

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。

3.6.8. 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。

重要

Red Hat OpenStack Platform 9 環境が、以前のバージョン(Red Hat Ceph Storage 1.3) の外部の Ceph Storage クラスターと統合されている場合には、後方互換性を有効にする必要があります。そのためには、環境ファイル (例: **/home/stack/templates/ceph-backwards-compatibility.yaml**) を作成して、以下の内容を記載します。

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

このファイルの作成が完了したら、次のステップで **openstack overcloud deploy** を実行する際に指定してください。

アンダークラウドから **openstack overcloud deploy** を実行して、**major-upgrade-pacemaker-converge.yaml** 環境ファイルを指定します。Ceph の後方互換性 (該当する場合)、ネットワークの分離、ストレージなど、お使いの環境に関連するすべてのオプションとカスタムの環境ファイルも必ず指定してください。

以下は、**openstack overcloud deploy** コマンドに追加で **major-upgrade-pacemaker-converge.yaml** ファイルを指定した例です。

```
$ 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-converge.yaml \
  --ntp-server pool.ntp.org
```

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。

3.6.9. OpenStack Telemetry Alarming データベースの移行

このステップにより、OpenStack Telemetry Alarming (**aodh**) サービスのデータベースが MongoDB から MariaDB に移行されます。このプロセスは、自動的にマイグレーションを実行します。

アンダークラウドから **major-upgrade-aodh-migration.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すべてのオプションとカスタムの環境ファイルも必ず指定してください。

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-aodh-migration.yaml \
  --ntp-server pool.ntp.org
```

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。

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。

3.7. 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 10 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。

- 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 10 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可能性があります。

- 「コントローラーノードのアップグレード」で準備したオプションの **major-upgrade-remove-sahara.yaml** ファイルを追加しなかった場合には、**/usr/share/openstack-tripleo-heat-templates/environments/services/sahara.yaml** を追加して、OpenStack Clustering (**sahara**) がオーバークラウド内で引き続き有効な状態となるようにしてください。

- 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
  ```

- アップグレード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、アップグレード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、アップグレード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。アップグレード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以下に例を示します。

```
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 9 から Red Hat OpenStack Platform 10 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、OpenStack の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

更新を実行する前に、OpenStack サービスの **systemd** スナップショットを作成します。

```
# systemctl snapshot openstack-services
```

主要な OpenStack Platform サービスを無効にします。

```
# systemctl stop 'openstack-*'
# systemctl stop 'neutron-*'
# systemctl stop 'openvswitch'
```

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 10 リポジトリーに変更します。

```
# subscription-manager repos --disable=rhel-7-server-openstack-9-rpms
# subscription-manager repos --enable=rhel-7-server-openstack-10-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 10 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。

表4.1 OpenStack サービスデータベースを同期するためのコマンド

サービス	プロジェクト名	コマンド
Identity	keystone	`# su -s /bin/sh -c "keystone-manage db_sync" keystone`
Image	glance	`# su -s /bin/sh -c "glance-manage db_sync" glance`
Block Storage	cinder	`# su -s /bin/sh -c "cinder-manage db sync" cinder`
Orchestration	heat	`# su -s /bin/sh -c "heat-manage db_sync" heat`
Compute	nova	`# su -s /bin/sh -c "nova-manage api_db sync" nova` `# su -s /bin/sh -c "nova-manage db sync" nova`
Telemetry	ceilometer	`# ceilometer-dbsync`
Telemetry Alarming	aodh	`# aodh-dbsync`
Telemetry Metrics	gnocchi	`# gnocchi-upgrade`
Clustering	sahara	`# su -s /bin/sh -c "sahara-db-manage upgrade heads" sahara`
Networking	neutron	`# su -s /bin/sh -c "neutron-db-manage upgrade heads" neutron`

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、OpenStack の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 サービスを再起動します。

```
# systemctl isolate openstack-services.snapshot
```

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
```

注記

このアップグレードシナリオでは、「*7章director を使用しない環境向けの追加の手順*」に詳述したように、いくつかの追加のアップグレード手順があります。これらの追加手順は、手動の環境ではオプションですが、現在の OpenStack Platform の推奨事項に合わせるのに役立ちます。

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 9 から Red Hat OpenStack Platform 10 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 10 リポジトリーに変更します。

```
# subscription-manager repos --disable=rhel-7-server-openstack-9-rpms
# subscription-manager repos --enable=rhel-7-server-openstack-10-rpms
```

更新を実行する前に、OpenStack サービスの **`systemd`** スナップショットを作成します。

```
# systemctl snapshot openstack-services
```

主要な OpenStack Platform サービスを無効にします。

```
sudo systemctl stop 'openstack-*'
sudo systemctl stop 'neutron-*'
sudo systemctl stop 'openvswitch'
```

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. WSGI サービスのアップグレード

Identity サービスと Dashboard WSGI アプレットを無効にします。

```
# 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
# su -s /bin/sh -c "keystone-manage db_sync" keystone
```

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。

httpd サービスを再起動します。

```
# 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。

```
# systemctl stop '*swift*'
# yum -d1 -y upgrade \*swift\*
# systemctl start openstack-swift-account-auditor \
    openstack-swift-account-reaper \
    openstack-swift-account-replicator \
    openstack-swift-account \
    openstack-swift-container-auditor \
    openstack-swift-container-replicator \
    openstack-swift-container-updater \
    openstack-swift-container \
    openstack-swift-object-auditor \
    openstack-swift-object-replicator \
```

```
    openstack-swift-object-updater \
    openstack-swift-object \
    openstack-swift-proxy
```

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。

```
# systemctl stop '*glance*'
# yum -d1 -y upgrade \*glance\*
# su -s /bin/sh -c "glance-manage db_sync" glance
# systemctl start openstack-glance-api \
    openstack-glance-registry
```

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。

```
# systemctl stop '*cinder*'
# yum -d1 -y upgrade \*cinder\*
# su -s /bin/sh -c "cinder-manage db sync" cinder
# systemctl start openstack-cinder-api \
    openstack-cinder-scheduler \
    openstack-cinder-volume
```

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。

```
# systemctl stop '*heat*'
# yum -d1 -y upgrade \*heat\*
# su -s /bin/sh -c "heat-manage db_sync" heat
# systemctl start openstack-heat-api-cfn \
    openstack-heat-api-cloudwatch \
    openstack-heat-api \
    openstack-heat-engine
```

5.7. TELEMETRY (CEILOMETER) のアップグレード

注記

このコンポーネントには、「*7章director を使用しない環境向けの追加の手順*」に記載したように、追加のアップグレード手順がいくつかあります。これらの追加手順は、手動の環境ではオプションですが、現在の OpenStack の推奨事項に合わせるのに役立ちます。

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。

```
# systemctl stop '*ceilometer*'
# systemctl stop '*aodh*'
# systemctl stop '*gnocchi*'
```

```
# yum -d1 -y upgrade \*ceilometer\* \*aodh\* \*gnocchi\*
```

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
# aodh-dbsync
# gnocchi-upgrade
```

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。

```
# systemctl start openstack-ceilometer-api \
    openstack-ceilometer-central \
    openstack-ceilometer-collector \
    openstack-ceilometer-notification \
    openstack-aodh-evaluator \
    openstack-aodh-listener \
    openstack-aodh-notifier \
    openstack-gnocchi-metricd \
    openstack-gnocchi-statsd
```

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 (**mitaka**) に設定します。

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

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。

```
# systemctl stop '*nova*'
# yum -d1 -y upgrade \*nova\*
# su -s /bin/sh -c "nova-manage api_db sync" nova
# su -s /bin/sh -c "nova-manage db sync" nova
```

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。

```
# systemctl start openstack-nova-api \
    openstack-nova-conductor \
    openstack-nova-consoleauth \
```

```
    openstack-nova-novncproxy \
    openstack-nova-scheduler
```

5.9. CLUSTERING (SAHARA) のアップグレード

1. Clustering コンポーネントサービスをホストしている全ノードで、以下のコマンドを実行します。

```
# systemctl stop '*sahara*'
# yum -d1 -y upgrade \*sahara\*
```

2. Image サービスのホストで以下のコマンドを実行します。

```
# su -s /bin/sh -c "sahara-db-manage upgrade heads" sahara
```

3. パッケージのアップグレードが完了した後には、Clustering 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 Clustering サービスを再起動します。

```
# systemctl start openstack-sahara-api \
    openstack-sahara-engine
```

5.10. 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。

```
# systemctl stop '*neutron*'
# yum -d1 -y upgrade \*neutron\*
```

2. 同じホストで、OpenStack Networking データベーススキーマを更新します。

```
# su -s /bin/sh -c "neutron-db-manage upgrade heads" neutron
```

3. OpenStack Networking サービスを再起動します。

```
# systemctl start neutron-dhcp-agent \
    neutron-l3-agent \
    neutron-metadata-agent \
    neutron-openvswitch-agent \
    neutron-server
```

注記

お使いの環境で有効化されている OpenStack Networking の追加のサービスを起動します。

5.11. 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 10 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、高可用性の環境でライブのコンピュートを使用して 1 度に 1 サービスずつアップグレードするのに必要な手順について説明します。このシナリオでは、**director を使用しない** Red Hat OpenStack Platform 9 から Red Hat OpenStack Platform 10 へのアップグレードを行います。

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。

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 10 リポジトリーに変更します。

```
# subscription-manager repos --disable=rhel-7-server-openstack-9-rpms
# subscription-manager repos --enable=rhel-7-server-openstack-10-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. WSGI 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の WSGI サービスのパッケージを同時にアップグレードします。これには、OpenStack Identity (keystone) と OpenStack Dashboard (horizon) が含まれます。

1. サービスを Pacemaker の制御下から削除します。

```
# pcs resource unmanage httpd-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 **httpd** サービスを停止します。

```
# systemctl stop httpd
```

3. 適切なパッケージをアップグレードします。

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

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** を再読み込みします。

```
# 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。

```
# su -s /bin/sh -c "keystone-manage db_sync" keystone
```

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 httpd
```

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 httpd-clone
```

9. リソースを Pacemaker の制御対象に戻します。

```
# pcs resource manage httpd-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 '*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。

```
# su -s /bin/sh -c "glance-manage db_sync" glance
```

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 '*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。

```
# su -s /bin/sh -c "cinder-manage db sync" cinder
```

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すべてを再起動します。

```
# 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 '*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。

```
# su -s /bin/sh -c "heat-manage db_sync" heat
```

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。

注記

このコンポーネントには、「*7章director を使用しない環境向けの追加の手順*」に記載したように、追加のアップグレード手順がいくつかあります。これらの追加手順は、手動の環境ではオプションですが、現在の OpenStack の推奨事項に合わせるのに役立ちます。

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

```
# pcs resource disable openstack-ceilometer-api-clone
# pcs resource disable openstack-ceilometer-collector-clone
# pcs resource disable openstack-ceilometer-notification-clone
```

```
# pcs resource disable openstack-ceilometer-central-clone
# pcs resource disable openstack-aodh-evaluator-clone
# pcs resource disable openstack-aodh-listener-clone
# pcs resource disable openstack-aodh-notifier-clone
# pcs resource disable openstack-gnocchi-metricd-clone
# pcs resource disable openstack-gnocchi-statsd-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 '*ceilometer*' '*aodh*' '*gnocchi*'
```

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
# aodh-dbsync
# gnocchi-upgrade
```

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-api-clone
# pcs resource cleanup openstack-ceilometer-collector-clone
# pcs resource cleanup openstack-ceilometer-notification-clone
# pcs resource cleanup openstack-ceilometer-central-clone
# pcs resource cleanup openstack-aodh-evaluator-clone
# pcs resource cleanup openstack-aodh-listener-clone
# pcs resource cleanup openstack-aodh-notifier-clone
# pcs resource cleanup openstack-gnocchi-metricd-clone
# pcs resource cleanup openstack-gnocchi-statsd-clone
```

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-api-clone
# pcs resource enable openstack-ceilometer-collector-clone
# pcs resource enable openstack-ceilometer-notification-clone
# pcs resource enable openstack-ceilometer-central-clone
# pcs resource enable openstack-aodh-evaluator-clone
# pcs resource enable openstack-aodh-listener-clone
# pcs resource enable openstack-aodh-notifier-clone
# pcs resource enable openstack-gnocchi-metricd-clone
# pcs resource enable openstack-gnocchi-statsd-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。

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 '*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。

```
# su -s /bin/sh -c "nova-manage api_db sync" nova
# su -s /bin/sh -c "nova-manage db sync" nova
```

6. コンピュートホストのローリングアップグレードを実行する場合には、API バージョンの制限を明示的に設定して、Mitaka と Newton 環境間で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 (**mitaka**) に設定します。

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

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
```

全コンピュートホストを Red Hat OpenStack Platform 10 にアップグレードした後には、制御を Pacemaker に戻します。

```
# 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. CLUSTERING サービス (SAHARA) のアップグレード

本手順では、全コントローラーノードで同時に Clustering サービスのパッケージを更新します。

1. Pacemaker で Clustering サービスのリソースをすべて停止します。

```
# pcs resource disable openstack-sahara-api-clone
# pcs resource disable openstack-sahara-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 '*sahara*'
```

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. Clustering サービスのデータベーススキーマを更新します。

```
# su -s /bin/sh -c "sahara-db-manage upgrade heads" sahara
```

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

```
# pcs resource cleanup openstack-sahara-api-clone
# pcs resource cleanup openstack-sahara-engine-clone
```

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

```
# pcs resource enable openstack-sahara-api-clone
# pcs resource enable openstack-sahara-engine-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。

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

```
# su -s /bin/sh -c "neutron-db-manage upgrade heads" neutron
```

5. 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
```

6. 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
```

7. 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
```

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実行する場合には、API バージョンの制限を明示的に設定して、Mitaka と Newton 環境間で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 (**mitaka**) に設定します。

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

更新を実行する前に、OpenStack サービスの **systemd** スナップショットを作成します。

```
# systemctl snapshot openstack-services
```

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。

```
# systemctl stop 'openstack*' '*nova*'
```

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。

```
# systemctl isolate openstack-services.snapshot
```

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 10 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 10 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 TELEMETRY API の WSGI サービスへのアップグレード

このステップでは、OpenStack Telemetry (**ceilometer**) API が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-ceilometer-api** サービスは無効化され、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。

1. OpenStack Telemetry 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 stop openstack-ceilometer-api
     ```

   - 高可用性を使用している環境の場合:

     ```
     $ sudo pcs resource disable openstack-ceilometer-api
     ```

2. 各コントローラー上で、OpenStack Telemetry サービスの WSGI アプレット (**/lib/python2.7/site-packages/ceilometer/api/app.wsgi**) を **/var/www/cgi-bin/** の新しいディレクトリーにコピーします。以下に例を示します。

   ```
   $ sudo mkdir /var/www/cgi-bin/ceilometer
   $ cp /lib/python2.7/site-packages/ceilometer/api/app.wsgi
   /var/www/cgi-bin/ceilometer/app
   ```

3. 各コントローラーで、OpenStack Telemetry サービス向けに仮想ホストの設定ファイル (**10-ceilometer_wsgi.conf**) を作成します。このファイルを **/etc/httpd/conf.d/** に保存します。仮想ホストのファイルの内容は、以下の例のようになります。

   ```
   Listen 8777

   <VirtualHost *:8777>
     DocumentRoot "/var/www/cgi-bin/ceilometer"

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

     ErrorLog "/var/log/httpd/ceilometer_wsgi_error.log"
     ServerSignature Off
     CustomLog "/var/log/httpd/ceilometer_wsgi_access.log" combined

     SetEnvIf X-Forwarded-Proto https HTTPS=1
     WSGIApplicationGroup %{GLOBAL}
     WSGIDaemonProcess ceilometer group=ceilometer processes=1
   ```

```
threads=4 user=ceilometer
  WSGIProcessGroup ceilometer
  WSGIScriptAlias / "/var/www/cgi-bin/ceilometer/app"
</VirtualHost>
```

4. **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
     ```

7.2. OPENSTACK TELEMETRY ALARMING データベースの移行

アップグレードの一環として、**aodh-dbsync** ツールにより新規 MariaDB データベースが作成されますが、以前のデータベースは MongoDB を使用していました。この手順では、旧バージョンの OpenStack Telemetry Alarming (**aodh**) サービスのデータベースを MongoDB から MariaDB に移行します。このステップは、環境をアップグレードした後に実行してください。

/etc/aodh/aodh.conf 設定ファイルを編集して、データベースの接続を MariaDB データベースに変更します。以下に例を示します。

```
[database]
connection = mysql+pymysql://username:password@host/aodh?charset=utf8
```

以下のコマンドを実行して移行します。

```
$ sudo /usr/bin/aodh-data-migration \
  --nosql-conn `crudini --get /etc/ceilometer/ceilometer.conf database
connection` \
  --sql-conn `crudini --get /etc/aodh/aodh.conf database connection`
```

このコマンドにより、MongoDB (through **--nosql-conn**) から MariaDB (through **--sql-conn**) にデータが移行されます。

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、両シナリオで問題が発生した場合のトラブルシューティング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。