Red Hat OpenStack Platform 15

オーバークラウド用の外部ロードバランサー

外部ロードバランサーを使用するように OpenStack Platform 環境を設定する

Last Updated: 2022-09-15

Red Hat OpenStack Platform 15 オーバークラウド用の外部ロードバランサー

外部ロードバランサーを使用するように OpenStack Platform 環境を設定する

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

本ガイドでは、Red Hat OpenStack Platform 環境をオーバークラウド用に外部ロードバランサーを使用するように設定する方法について説明します。これには、OpenStack Platform director を使用したオーバークラウドのロードバランサーおよび設定ガイドラインが含まれます。

目次

第1章 はじめに

Red Hat OpenStack Platform director は、**オーバークラウド** と呼ばれるクラウド環境を作成します。オーバークラウドには、特定のロールを実行するノード種別のセットが含まれます。これらのノード種別の 1 つが **コントローラー** ノードです。コントローラーはオーバークラウドの管理を行い、特定の OpenStack コンポーネントを使用します。オーバークラウドは、複数のコントローラーを合わせて、高可用性クラスターとして使用し、OpenStack サービスのオペレーションパフォーマンスを最大限に保つようにします。さらに、クラスターにより、OpenStack サービスへのアクセスの負荷分散が行われ、コントローラーノードに均等にトラフィックを分配して、各ノードのサーバーで過剰負荷を軽減します。

また、外部のロードバランサーを使用して、この分散を実行することも可能です。たとえば、組織で、コントローラーノードへのトラフィックの分散処理に、ハードウェアベースのロードバランサーを使用する場合などです。本ガイドでは、外部ロードバランサーとオーバークラウドの作成の両方の設定を定義するのに役立つ必要な情報を提供します。これには、以下のプロセスが含まれます。

1. **ロードバランサーのインストールと設定**: 本ガイドでは、負荷分散およびサービス用の HAProxy オプションをいくつか紹介します。設定を独自の外部ロードバランサーと同等のに変換します。

2. **オーバークラウドの設定およびデプロイメント**: 本ガイドには、オーバークラウドの外部ロードバランサーとの統合に役立つ Heat テンプレートのパラメーターが複数含まれています。これには主に、ロードバランサーの IP アドレスと潜在的なノードの IP アドレスが含まれます。本ガイドには、オーバークラウドのデプロイメントを起動するコマンドや、外部ロードバランサーを使用するための設定も含まれます。

1.1. オーバークラウドでの負荷分散の使用

オーバークラウドは、**HAProxy** と呼ばれるオープンソースツールを使用します。HAProxy は、OpenStack サービスを実行しているコントローラーノードへのトラフィックの負荷分散を行います。**haproxy** パッケージには、**haproxy** systemd サービスから起動される haproxy デーモンと、ロギング機能やサンプルの設定が含まれます。ただし、オーバークラウドは高可用性リソースマネージャー (Pacemaker)を使用して、高可用性サービス(haproxy-clone)として HAProxy 自体も制御します。これは、HAProxy が各コントローラーノードで実行され、各設定で定義される一連のルールに従ってトラフィックを分散することを意味します。

1.2. サンプルシナリオの定義

この記事では、例として以下のシナリオを使用しています。

- HAProxy を使用する外部読み込み用サーバー。これは、フェデレーションされた HAProxy サーバーを使用する方法を示しています。これをサポートされている別の外部ロードバランサーに置き換えることができます。
- OpenStack Platform director ノード 1 台
- 以下で構成されるオーバークラウド
- 高可用性クラスター内のコントローラーノード 3 台
- 1 コンピュートノード
- VLAN を使用したネットワーク分離

このシナリオでは、各ネットワークに以下の IP アドレスの割り当てを使用します。

- Internal API: 172.16.20.0/24
- Tenant: 172.16.22.0/24
- ストレージ： 172.16.21.0/24
- ストレージ管理： 172.16.19.0/24
- External: 172.16.23.0/24

これらの IP 範囲には、コントローラーノードおよびロードバランサーが OpenStack サービスにバインドする仮想 IP に対する IP 割り当てが含まれます。

第2章 デフォルト設定の定義

外部のロードバランサーを使用せずにオーバークラウドを作成して設定する場合には、director はトラフィックを複数の OpenStack サービスに分散するように HAProxy を設定します。director は、この設定を各コントローラーノードの **/etc/haproxy/haproxy.conf** ファイルに提供します。デフォルト設定には、global、default、および複数のサービス設定の 3 つの主要部分が含まれます。

次の数セクションでは、各設定セクションのデフォルトのパラメーターについて説明します。ここでは、外部ロードバランサーをインストールおよび設定するための設定例を説明します。これらのパラメーターは、合計の HAProxy パラメーターの一部のみであることに注意してください。これらのパラメーターおよびその他のパラメーターの詳細は、コントローラーノード（または **haproxy** パッケージがインストールされているシステム）の **/usr/share/doc/haproxy-*/configuration.txt** にある「HAProxy 設定マニュアル」を参照してください。

2.1. グローバル設定

```
global
  daemon
  group  haproxy
  log  /dev/log local0
  maxconn  10000
  pidfile  /var/run/haproxy.pid
  user  haproxy
```

このセクションでは、プロセス全体のパラメーターのセットを定義します。これには、以下のパラメーターが含まれます。

- **デーモン**: バックグラウンドプロセスとして実行します。
- **ユーザー haproxy、グループ haproxy**: プロセスを所有する Linux ユーザーおよびグループを定義します。
- **Log**: 使用する syslog サーバーを定義します。
- **maxconn**: プロセスへの同時接続の最大数を設定します。
- **pidfile**: プロセス ID に使用するファイルを設定します。

2.2. デフォルト設定

```
defaults
  log  global
  mode  tcp
  retries  3
  timeout  http-request 10s
  timeout  queue 1m
  timeout  connect 10s
  timeout  client 1m
  timeout  server 1m
  timeout  check 10s
```

このセクションでは、各サービスのデフォルトパラメーターセットを定義します。これには、以下のパラメーターが含まれます。

- **Log**: サービスのロギングを有効にします。**グローバル** 値は、ロギング関数が **global** セクションの **log** パラメーターを使用することを意味します。
- **Mode**: 使用するプロトコルを設定します。ここでは、デフォルトは TCP です。
- **retries**: 接続の障害を報告する前にサーバーで実行する再試行の数を設定します。
- **timeout**: 特定の関数について待機する最大時間を設定します。たとえば、**timeout http-request** は、完全な HTTP 要求を待つ最大の時間を設定します。

2.3. サービスの設定

```
listen ceilometer
  bind 172.16.20.250:8777
  bind 172.16.23.250:8777
  server overcloud-controller-0 172.16.20.150:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8777 check fall 5 inter 2000 rise 2
```

デフォルトのファイルには、複数のサービス設定セクションがあります。各サービス設定には以下が含まれます。

- **listen**: 要求をリッスンするサービスの名前
- **bind**: サービスがリッスンする IP アドレスおよび TCP ポート番号
- **server**: サービスを提供する各サーバー名、サーバーの IP アドレス、リッスンするポート、その他の情報

上記の例では、**ceilometer** サービスの HAProxy 設定を示しています。このサービスは、ceilometer サービスが提供する IP アドレスとポートを特定します（ポート 8777 は 172.16.20.2500 および 172.16.23.250）。HAProxy はこれらのアドレスに対する要求を **overcloud-controller-0** (172.16.20.150:8777)、**overcloud-controller-1** (172.16.20.151:8777)、または **overcloud-controller-2** (172.16.0.152:8777)に転送します。

さらに、**サーバー** パラメーターの例では、以下を有効にします。

- **チェック**: ヘルスチェックの有効化
- **fall 5**: ヘルスチェックに 5 回失敗すると、サービスは停止中とみなされます。
- **inter 2000**: 連続する 2 つのヘルスチェックの間隔は 2000 ミリ秒（または 2 秒）に設定されます。
- **増加 2**: ヘルスチェックが 2 回成功すると、サーバーは動作とみなされます。

各サービスは異なるネットワークトラフィックタイプを表す異なるアドレスにバインドします。サービスによっては、追加の設定オプションも含まれているものもあります。次章では、外部ロードバランサーでこれらの詳細を複製できるように、それぞれの特定のサービス設定について説明します。

第3章 サービス設定のリファレンス

本章では、負荷分散を使用するオーバークラウドの特定のサービスの設定について説明します。この設定は、独自の外部ロードバランサーを設定するためのガイドとして使用します。これらのパラメーターおよびその他のパラメーターの詳細は、コントローラーノード（または **haproxy** パッケージがインストールされているシステム）の **/usr/share/doc/haproxy-*/configuration.txt** にある「HAProxy 設定マニュアル」を参照してください。

注記

ほとんどのサービスは、デフォルトのヘルスチェック設定を使用します。

- 連続する 2 つのヘルスチェックの間隔は 2000 ミリ秒（2 秒）に設定されます。
- ヘルスチェックに2回成功すると、サービスは稼働状態とみなされます。
- ヘルスチェックに 5 回失敗すると、サービスは dead（停止）と見なされます。

各サービスは、各サービスの **Other information** セクションのデフォルトのヘルスチェックまたは追加のオプションを示します。

3.1. AODH

ポート番号： 8042

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen aodh
  bind 172.16.20.250:8042
  bind 172.16.23.250:8042
  server overcloud-controller-0 172.16.20.150:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8042 check fall 5 inter 2000 rise 2
```

3.2. CEILOMETER

ポート番号： 8777

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen ceilometer
  bind 172.16.20.250:8777
  bind 172.16.23.250:8777
  server overcloud-controller-0 172.16.20.150:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8777 check fall 5 inter 2000 rise 2
```

3.3. CINDER

ポート番号： 8776

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen cinder
  bind 172.16.20.250:8776
  bind 172.16.23.250:8776
  server overcloud-controller-0 172.16.20.150:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8776 check fall 5 inter 2000 rise 2
```

3.4. GLANCE_API

ポート番号： 9292

バインド先：storage、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 のストレージ

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen glance_api
  bind 172.16.23.250:9292
  bind 172.16.21.250:9292
  server overcloud-controller-0 172.16.21.150:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:9292 check fall 5 inter 2000 rise 2
```

3.5. GLANCE_REGISTRY

ポート番号： 9191

バインド先：internal_api

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen glance_registry
  bind 172.16.20.250:9191
  server overcloud-controller-0 172.16.20.150:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9191 check fall 5 inter 2000 rise 2
```

3.6. GNOCCHI

ポート番号： 8041

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen gnocchi
  bind 172.16.20.250:8041
  bind 172.16.23.250:8041
  server overcloud-controller-0 172.16.20.150:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8041 check fall 5 inter 2000 rise 2
```

3.7. HEAT_API

ポート番号： 8004

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

- このサービスは、デフォルトの TCP モードの代わりに HTTP モードを使用します。

HAProxy の例：

```
listen heat_api
  bind 172.16.20.250:8004
  bind 172.16.23.250:8004
  mode http
  server overcloud-controller-0 172.16.20.150:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8004 check fall 5 inter 2000 rise 2
```

3.8. HEAT_CFN

ポート番号： 8000

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen heat_cfn
  bind 172.16.20.250:8000
  bind 172.16.23.250:8000
  server overcloud-controller-0 172.16.20.150:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.152:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.151:8000 check fall 5 inter 2000 rise 2
```

3.9. HEAT_CLOUDWATCH

ポート番号： 8003

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen heat_cloudwatch
  bind 172.16.20.250:8003
  bind 172.16.23.250:8003
```

```
server overcloud-controller-0 172.16.20.150:8003 check fall 5 inter 2000 rise 2
server overcloud-controller-1 172.16.20.151:8003 check fall 5 inter 2000 rise 2
server overcloud-controller-2 172.16.20.152:8003 check fall 5 inter 2000 rise 2
```

3.10. HORIZON

ポート番号： 80

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する
- このサービスは、デフォルトの TCP モードの代わりに HTTP モードを使用します。
- このサービスは、UI との対話に cookie ベースの永続性を使用します。

HAProxy の例：

```
listen horizon
  bind 172.16.20.250:80
  bind 172.16.23.250:80
  mode http
  cookie SERVERID insert indirect nocache
  server overcloud-controller-0 172.16.20.150:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:80 check fall 5 inter 2000 rise 2
```

3.11. KEYSTONE_ADMIN

ポート番号： 35357

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen keystone_admin
  bind 172.16.23.250:35357
  bind 172.16.20.250:35357
  server overcloud-controller-0 172.16.20.150:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:35357 check fall 5 inter 2000 rise 2
```

3.12. KEYSTONE_ADMIN_SSH

ポート番号： 22

バインド先：internal_api

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen keystone_admin_ssh
  bind 172.16.20.250:22
  server overcloud-controller-0 172.16.20.150:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:22 check fall 5 inter 2000 rise 2
```

3.13. KEYSTONE_PUBLIC

ポート番号： 5000

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen keystone_public
  bind 172.16.20.250:5000
  bind 172.16.23.250:5000
  server overcloud-controller-0 172.16.20.150:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:5000 check fall 5 inter 2000 rise 2
```

3.14. MYSQL

Port Number: 3306

バインド先：internal_api

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用します。ただし、ヘルスチェックにはポート 9200 が使用されます。
- このサービスは、1度に1つのサーバーにのみ負荷分散されます。
- 各サーバーは、他のすべての非バックアップサーバーが利用できない場合にのみ負荷分散に使用されます。
- サーバーがダウンしていると、すべての接続が即座に終了します。
- 両サイドで TCP keepalive パケットの送信を有効にします。
- HTTP プロトコルを有効にしてサーバーの正常性でチェックします。
- スティッキーテーブルを設定して IP アドレスを保存します。これは永続性を維持するのに役立ちます。

重要

mysql サービスは、Galera を使用して高可用性のデータベースクラスターを提供します。Galera は **アクティブ/アクティブ** 設定をサポートしていますが、ロックの競合を避けるためにロードバランサーにより強制された **アクティブ/パッシブ** を使用することを推奨します。

HAProxy の例：

```
listen mysql
  bind 172.16.20.250:3306
  option tcpka
  option httpchk
  stick on dst
  stick-table type ip size 1000
  timeout client 0
  timeout server 0
  server overcloud-controller-0 172.16.20.150:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-1 172.16.20.151:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-2 172.16.20.152:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
```

3.15. NEUTRON

ポート番号： 9696

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen neutron
  bind 172.16.20.250:9696
  bind 172.16.23.250:9696
  server overcloud-controller-0 172.16.20.150:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9696 check fall 5 inter 2000 rise 2
```

3.16. NOVA_EC2

ポート番号： 8773

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen nova_ec2
  bind 172.16.20.250:8773
  bind 172.16.23.250:8773
  server overcloud-controller-0 172.16.20.150:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8773 check fall 5 inter 2000 rise 2
```

3.17. NOVA_METADATA

ポート番号： 8775

バインド先：internal_api

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen nova_metadata
  bind 172.16.20.250:8775
  server overcloud-controller-0 172.16.20.150:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8775 check fall 5 inter 2000 rise 2
```

3.18. NOVA_NOVNCPROXY

ポート番号： 6080

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する
- デフォルトの分散方法はラウンドロビンです。ただし、このサービスでは **source** メソッドが使用されます。このメソッドは、ソース IP アドレスをハッシュし、実行中のサーバーの重みの合計で除算します。これにより、要求を受信するサーバーが指定されます。これにより、サーバーが終了/起動しない限り、同じクライアント IP アドレスは常に同じサーバーに到達します。実行中のサーバー数の変更によりハッシュの結果が変更されると、バランサーは多くのクライアントを別のサーバーにリダイレクトします。

HAProxy の例：

```
listen nova_novncproxy
  bind 172.16.20.250:6080
  bind 172.16.23.250:6080
  balance source
  server overcloud-controller-0 172.16.20.150:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:6080 check fall 5 inter 2000 rise 2
```

3.19. NOVA_OSAPI

ポート番号： 8774

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen nova_osapi
  bind 172.16.20.250:8774
  bind 172.16.23.250:8774
  server overcloud-controller-0 172.16.20.150:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8774 check fall 5 inter 2000 rise 2
```

3.20. NOVA_PLACEMENT

ポート番号： 8778

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen nova_placement
  bind 172.16.20.250:8778
  bind 172.16.23.250:8778
  server overcloud-controller-0 172.16.20.150:8778 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8778 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8778 check fall 5 inter 2000 rise 2
```

3.21. PANKO

ポート番号： 8779

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen panko
  bind 172.16.20.250:8779
  bind 172.16.23.250:8779
  server overcloud-controller-0 172.16.20.150:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8779 check fall 5 inter 2000 rise 2
```

3.22. REDIS

ポート番号： 6379

バインド先：internal_api（redis サービス IP）

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用します。
- **tcp-check** send/expect シーケンスを使用してヘルスチェックを実行します。送信する文字列は "info\ replication\r\n" で、応答は "role:master" です。
- Redis サービスは認証にパスワードを使用します。たとえば、HAProxy 設定は、**tcp-check** と

AUTH メソッドおよび Redis 管理パスワードを使用します。通常、director は無作為にパスワードを生成しますが、カスタムの Redis パスワードを定義することができます。詳細は、「ロードバランシングオプションの設定」 を参照してください。

- デフォルトの分散方法はラウンドロビンです。ただし、このサービスでは、**最初** のメソッドを使用します。これにより、利用可能な接続スロットを持つ最初のサーバーが接続を受け取るようになります。

HAProxy の例：

```
listen redis
  bind 172.16.20.249:6379 transparent
  balance first
  option tcp-check
  tcp-check send AUTH\ p@55w0rd!\r\n
  tcp-check send PING\r\n
  tcp-check expect string +PONG
  tcp-check send info\ replication\r\n
  tcp-check expect string role:master
  tcp-check send QUIT\r\n
  tcp-check expect string +OK
  server overcloud-controller-0 172.16.20.150:6379 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:6379 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:6379 check fall 5 inter 2000 rise 2
```

3.23. SAHARA

ポート番号： 8386

バインド先：internal_api、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 の internal_api

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する
- このサービスはオプションのオーバークラウドサービスです。オーバークラウドのデプロイメントに **environments/services/sahara.yaml** 環境ファイルを追加してインストールする場合。

HAProxy の例：

```
listen sahara
  bind 172.16.20.250:8386
  bind 172.16.23.250:8386
  server overcloud-controller-0 172.16.20.150:8386 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8386 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8386 check fall 5 inter 2000 rise 2
```

3.24. SWIFT_PROXY_SERVER

ポート番号： 8080

バインド先：storage、external

ターゲットネットワーク/サーバー： overcloud-controller-0、overcloud-controller-1、および overcloud-controller-2 のストレージ

その他の情報：

- 各ターゲットサーバーはデフォルトのヘルスチェックを使用する

HAProxy の例：

```
listen swift_proxy_server
  bind 172.16.23.250:8080
  bind 172.16.21.250:8080
  server overcloud-controller-0 172.16.21.150:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:8080 check fall 5 inter 2000 rise 2
```

第4章 オーバークラウドの設定

本項では、外部ロードバランサーを使用するオーバークラウドを作成するプロセスを実行します。これには、ノードの登録、ネットワーク設定、オーバークラウドの作成コマンドに必要な設定オプションの設定が含まれます。

4.1. 環境の設定

このセクションでは、『director のインストールと使用方法』 の「プロセスのカットダウンバージョン」を使用します。

以下のワークフローを使用して環境を設定します。

- ノード定義のテンプレートを作成して director で空のノードを登録します。
- 全ノードのハードウェアを検査します。
- 手動でノードをロールにタグ付けします。
- フレーバーを作成してロールにタグ付けします。

4.1.1. stack ユーザーの初期化

stack ユーザーとして director ホストにログインし、以下のコマンドを実行して director の設定を初期化します。

```
$ source ~/stackrc
```

このコマンドでは、director の CLI ツールにアクセスする認証情報が含まれる環境変数を設定します。

4.1.2. ノードの登録

ノード定義のテンプレート(**instackenv.json**)は JSON 形式のファイルで、ノード登録用のハードウェアおよび電源管理の情報が含まれています。以下に例を示します。

```
{
    "nodes":[
        {
            "mac":[
                "bb:bb:bb:bb:bb:bb"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"pxe_ipmitool",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.205"
        },
        {
            "mac":[
                "cc:cc:cc:cc:cc:cc"
            ],
```

```
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"pxe_ipmitool",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.206"
        },
        {
            "mac":[
                "dd:dd:dd:dd:dd:dd"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"pxe_ipmitool",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.207"
        },
        {
            "mac":[
                "ee:ee:ee:ee:ee:ee"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"pxe_ipmitool",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.208"
        }
    ]
}
```

テンプレートを作成したら、stack ユーザーのホームディレクトリーにファイルを保存し (**/home/stack/instackenv.json**)、それを director にインポートします。これを実行するには、以下のコマンドを使用します。

```
$ openstack overcloud node import ~/instackenv.json
```

このコマンドでテンプレートをインポートして、テンプレートから director に各ノードを登録します。

カーネルと ramdisk イメージを全ノードに割り当てます。

```
$ openstack overcloud node configure
```

director でのノードの登録、設定が完了しました。

4.1.3. ノードのハードウェアの検査

ノードの登録後に、各ノードのハードウェア属性を確認します。以下のコマンドを実行して、各ノードのハードウェア属性を検証します。

```
$ openstack overcloud node introspect --all-manageable
```

重要

ノードは **manageable** の状態である必要があります。このプロセスが最後まで実行されて正常に終了したことを確認してください。ベアメタルノードの場合には、通常 15 分ほどかかります。

4.1.4. ノードの手動でのタグ付け

各ノードのハードウェアを登録、検査した後には、特定のプロファイルにノードをタグ付けします。これらのプロファイルタグにより、ノードがフレーバーに照合され、そのフレーバーはデプロイメントロールに割り当てられます。

ノード一覧を取得して UUID を把握します。

```
$ ironic node-list
```

特定のプロファイルにノードを手動でタグ付けするには、各ノードの properties/capabilities パラメーターに profile オプションを追加します。たとえば、3 つのノードがControllerプロファイルを使用し、1 つのノードがComputeプロファイルを使用するようにタグ付けするには、以下のコマンドを使用します。

```
$ ironic node-update 1a4e30da-b6dc-499d-ba87-0bd8a3819bc0 add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 6faba1a9-e2d8-4b7c-95a2-c7fbdc12129a add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 5e3b2f50-fcd9-4404-b0a2-59d79924b38e add
properties/capabilities='profile:control,boot_option:local'
$ ironic node-update 58c3d07e-24f2-48a7-bbb6-6843f0e8ee13 add
properties/capabilities='profile:compute,boot_option:local'
```

profile:compute および **profile:control** オプションを追加することで、適切なプロファイルにノードをタグ付けします。

4.2. ネットワークの設定

本項では、オーバークラウドのネットワーク設定を検証します。これには、特定のネットワークトラフィックを使用し、負荷分散オプションでオーバークラウドを設定できるようにサービスを分離することが含まれます。

4.2.1. ネットワークの分離

director は、分離オーバークラウドネットワークを構成する手段を提供します。つまり、オーバークラウドの環境はネットワークトラフィックの種別を異なるネットワークに分割し、ネットワークトラフィックを特定のネットワークインターフェースまたはボンディングに割り当てます。分離ネットワークの設定後、director は OpenStack サービスが分離ネットワークを使用するように設定します。分離ネットワークが設定されていない場合には、すべてのサービスがプロビジョニングネットワーク上で実行されます。

まず、オーバークラウドには、ネットワークインターフェースのテンプレートセットが必要です。これらのテンプレートをカスタマイズして、ロールごとにノードインターフェースを設定します。これらのテンプレートは YAML 形式の標準の Heat テンプレートです。director には、使用を開始するためのテンプレート例が含まれています。

- **/usr/share/openstack-tripleo-heat-templates/network/config/single-nic-vlans**: ロールごとに VLAN 設定を持つ単一 NIC 用のテンプレートが含まれるディレクトリー。
- **/usr/share/openstack-tripleo-heat-templates/network/config/bond-with-vlans**: ロールごとにボンディングされた NIC 設定用のテンプレートが含まれるディレクトリー。

ネットワークインターフェースの設定に関する詳しい情報は、『director のインストールと使用方法』を参照してください。

次に、ネットワーク環境ファイルを作成します。このファイルは、オーバークラウドのネットワーク環境を記述し、ネットワークインターフェース設定テンプレートをポイントする Heat 環境ファイルです。このファイルは、IP アドレス範囲と共にネットワークのサブネットおよび VLAN も定義します。これらの値は、ローカル環境用にカスタマイズすることができます。

このシナリオでは、**/home/stack/network-environment.yaml** として保存された以下のネットワーク環境ファイルを使用します。

```
resource_registry:
  OS::TripleO::BlockStorage::Net::SoftwareConfig: /home/stack/templates/my-
overcloud/network/config/bond-with-vlans/cinder-storage.yaml
  OS::TripleO::Compute::Net::SoftwareConfig: /home/stack/templates/my-
overcloud/network/config/bond-with-vlans/compute.yaml
  OS::TripleO::Controller::Net::SoftwareConfig: /home/stack/templates/my-
overcloud/network/config/bond-with-vlans/controller.yaml
  OS::TripleO::ObjectStorage::Net::SoftwareConfig: /home/stack/templates/my-
overcloud/network/config/bond-with-vlans/swift-storage.yaml
  OS::TripleO::CephStorage::Net::SoftwareConfig: /home/stack/templates/my-
overcloud/network/config/bond-with-vlans/ceph-storage.yaml

parameter_defaults:
  InternalApiNetCidr: 172.16.20.0/24
  TenantNetCidr: - 172.16.22.0/24
  StorageNetCidr: 172.16.21.0/24
  StorageMgmtNetCidr: 172.16.19.0/24
  ExternalNetCidr: 172.16.23.0/24
  InternalApiAllocationPools: [{'start': '172.16.20.10', 'end': '172.16.20.200'}]
  TenantAllocationPools: [{'start': '172.16.22.10', 'end': '172.16.22.200'}]
  StorageAllocationPools: [{'start': '172.16.21.10', 'end': '172.16.21.200'}]
  StorageMgmtAllocationPools: [{'start': '172.16.19.10', 'end': '172.16.19.200'}]
  # Leave room for floating IPs in the External allocation pool
  ExternalAllocationPools: [{'start': '172.16.23.10', 'end': '172.16.23.60'}]
  # Set to the router gateway on the external network
  ExternalInterfaceDefaultRoute: 172.16.23.1
  # Gateway router for the provisioning network (or Undercloud IP)
  ControlPlaneDefaultRoute: 192.0.2.254
  # The IP address of the EC2 metadata server. Generally the IP of the Undercloud
  EC2MetadataIp: 192.0.2.1
  # Define the DNS servers (maximum 2) for the overcloud nodes
  DnsServers: ["8.8.8.8","8.8.4.4"]
  InternalApiNetworkVlanID: 201
  StorageNetworkVlanID: 202
```

```
  StorageMgmtNetworkVlanID: 203
  TenantNetworkVlanID: 204
  ExternalNetworkVlanID: 100
  # Set to "br-ex" if using floating IPs on native VLAN on bridge br-ex
  NeutronExternalNetworkBridge: """"
  # Customize bonding options if required
  BondInterfaceOvsOptions:
    "bond_mode=balance-tcp lacp=active other-config:lacp-fallback-ab=true"
```

ネットワーク環境設定に関する詳しい情報は、『 director のインストールと使用方法』 を参照してください。

keystone_admin_ssh の仮想 IP に接続できるように、director ホストが Internal API ネットワークにアクセスできるようにします。

4.2.2. ロードバランシングオプションの設定

director は、外部ロードバランサーが内部で管理する HAProxy ではなく、仮想 IP をホストするオーバークラウドを作成する方法を提供します。この設定では、オーバークラウドのデプロイメントが開始される前に、多数の仮想 IP が外部ロードバランサー（分離ネットワークごとに 1 つ、ならびに Redis サービス用に 1 つ）で設定されていることを前提としています。オーバークラウドノードの NIC 設定がそれを許可すると、仮想 IP の一部が同じであることがあります。

前章の設定を使用して、外部ロードバランサーを設定していました。これらの設定には、director がオーバークラウドノードに割り当ててサービス設定に使用する IP が含まれます。

外部のロードバランサーを使用するためのオーバークラウド設定が含まれる Heat 環境ファイル (**external-lb.yaml**)の例を以下に示します。

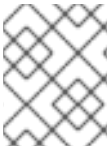
```
parameter_defaults:
  ControlFixedIPs: [{'ip_address':'192.0.2.250'}]
  PublicVirtualFixedIPs: [{'ip_address':'172.16.23.250'}]
  InternalApiVirtualFixedIPs: [{'ip_address':'172.16.20.250'}]
  StorageVirtualFixedIPs: [{'ip_address':'172.16.21.250'}]
  StorageMgmtVirtualFixedIPs: [{'ip_address':'172.16.19.250'}]
  RedisVirtualFixedIPs: [{'ip_address':'172.16.20.249'}]
  # IPs assignments for the Overcloud Controller nodes. Ensure these IPs are from each respective
allocation pools defined in the network environment file.
  ControllerIPs:
    external:
    - 172.16.23.150
    - 172.16.23.151
    - 172.16.23.152
    internal_api:
    - 172.16.20.150
    - 172.16.20.151
    - 172.16.20.152
    storage:
    - 172.16.21.150
    - 172.16.21.151
    - 172.16.21.152
    storage_mgmt:
    - 172.16.19.150
    - 172.16.19.151
    - 172.16.19.152
    tenant:
```

```
    - 172.16.22.150
    - 172.16.22.151
    - 172.16.22.152
    # CIDRs
    external_cidr: "24"
    internal_api_cidr: "24"
    storage_cidr: "24"
    storage_mgmt_cidr: "24"
    tenant_cidr: "24"
  RedisPassword: p@55w0rd!
  ServiceNetMap:
    NeutronTenantNetwork: tenant
    CeilometerApiNetwork: internal_api
    AodhApiNetwork: internal_api
    GnocchiApiNetwork: internal_api
    MongoDbNetwork: internal_api
    CinderApiNetwork: internal_api
    CinderIscsiNetwork: storage
    GlanceApiNetwork: storage
    GlanceRegistryNetwork: internal_api
    KeystoneAdminApiNetwork: internal_api
    KeystonePublicApiNetwork: internal_api
    NeutronApiNetwork: internal_api
    HeatApiNetwork: internal_api
    NovaApiNetwork: internal_api
    NovaMetadataNetwork: internal_api
    NovaVncProxyNetwork: internal_api
    SwiftMgmtNetwork: storage_mgmt
    SwiftProxyNetwork: storage
    HorizonNetwork: internal_api
    MemcachedNetwork: internal_api
    RabbitMqNetwork: internal_api
    RedisNetwork: internal_api
    MysqlNetwork: internal_api
    CephClusterNetwork: storage_mgmt
    CephPublicNetwork: storage
    ControllerHostnameResolveNetwork: internal_api
    ComputeHostnameResolveNetwork: internal_api
    BlockStorageHostnameResolveNetwork: internal_api
    ObjectStorageHostnameResolveNetwork: internal_api
    CephStorageHostnameResolveNetwork: storage
```

parameter_defaults セクションには、OpenStack 上の各ネットワークの仮想 IP および IP 割り当てが含まれます。これらの設定は、ロードバランサーのサービスごとに同じ IP 設定と一致する必要があります。本セクションでは、Redis サービス(**RedisPassword**)の管理パスワードも定義します。このセクションには、各 OpenStack サービスを特定のネットワークにマッピングする **ServiceNetMap** パラメーターも含まれています。負荷分散設定には、このサービスの再マッピングが必要です。

4.3. ロードバランシング用の SSL の設定

デフォルトでは、オーバークラウドはサービスに暗号化されていないエンドポイントを使用します。これは、オーバークラウドの設定に、エンドポイントに対して SSL/TLS を有効化するための追加の環境ファイルが必要であることを意味します。



注記

外部ロードバランサーに SSL 証明書とキーのコピーがインストールされていることを確認します。

内部ロードバランサーを持つオーバークラウドは、Heat テンプレートコレクションからの **enable-tls.yaml** 環境ファイルを使用して、鍵と証明書のペアをインストールします。外部ロードバランサーにはこのファイルは必要ありません。ただし、オーバークラウドには引き続き SSL/TLS エンドポイントの一覧が必要です。IP アドレスまたはドメイン名を使用してパブリックエンドポイントにアクセスするかどうかに応じて、以下の環境ファイルを使用します。

- パブリックエンドポイントへのアクセスに DNS 名を使用する場合には、**/usr/share/openstack-tripleo-heat-templates/environments/tls-endpoints-public-dns.yaml**を使用します。
- パブリックエンドポイントへのアクセスに 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**を使用します。

自己署名証明書を使用するか、または証明書の署名者がオーバークラウドイメージのデフォルトのトラストストアにない場合は、その証明書をオーバークラウドイメージに挿入します。Heat テンプレートコレクションから **inject-trust-anchor.yaml** 環境ファイルをコピーします。

```
$ cp -r /usr/share/openstack-tripleo-heat-templates/environments/inject-trust-anchor.yaml
~/templates/.
```

このファイルを編集して、下記のパラメーターに以下の変更を加えます。

SSLRootCertificate

SSLRootCertificate パラメーターにルート認証局ファイルの内容をコピーします。以下に例を示します。

```
parameter_defaults:
  SSLRootCertificate: |
  -----BEGIN CERTIFICATE-----
  MIIDgzCCAmugAwIBAgIJAKk46qw6ncJaMA0GCSqGSIb3DQEBCwUAMFgxCzAJBgNV
  ...
  sFW3S2roS4X0Af/kSSD8mlBBTFTCMBAj6rtLBKLaQbIxEpIzrgvp
  -----END CERTIFICATE-----
```

重要

この認証局の内容に新しく追加する行は、すべて同じレベルにインデントする必要があります。

OS::TripleO::NodeTLSCAData

OS::TripleO::NodeTLSCAData:' のリソース URL を絶対 URL に変更します。

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

DNS ホスト名を使用して SSL/TLS でオーバークラウドにアクセスする場合には、新しい環境ファイル (**~/templates/cloudname.yaml**)を作成して、オーバークラウドのエンドポイントのホスト名を定義します。以下のパラメーターを使用してください。

CloudName

オーバークラウドエンドポイントの DNS ホスト名

DnsServers

使用する DNS サーバー一覧。設定済みの DNS サーバーには、パブリック API の IP と一致する設定済みの CloudName のエントリーが含まれている必要があります。

以下は、このファイルの内容の例です。

```
parameter_defaults:
 CloudName: overcloud.example.com
 DnsServers: 10.0.0.1
```

「オーバークラウドの作成」 のデプロイメントコマンド(**openstack overcloud deploy**)は、**-e** オプションを使用して環境ファイルを追加します。本項の環境ファイルは、以下の順序で追加します。

- SSL/TLS エンドポイントが含まれる環境ファイル（**tls-endpoints-public-dns.yaml** または **tls-endpoints-public-ip.yaml**）
- DNS ホスト名を設定する環境ファイル (**cloudname.yaml**)
- ルート認証局を注入する環境ファイル (**inject-trust-anchor.yaml**)

以下に例を示します。

```
$ openstack overcloud deploy --templates [...] -e /usr/share/openstack-tripleo-heat-templates/environments/tls-endpoints-public-dns.yaml -e ~/templates/cloudname.yaml -e ~/templates/inject-trust-anchor.yaml
```

4.4. オーバークラウドの作成

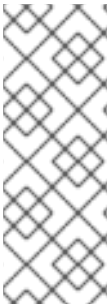
外部のロードバランサーを使用するオーバークラウドを作成する場合には、**openstack overcloud deploy** コマンドに追加の引数を指定する必要があります。以下に例を示します。

```
$ openstack overcloud deploy --templates -e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml -e ~/network-environment.yaml  -e /usr/share/openstack-tripleo-heat-templates/environments/external-loadbalancer-vip.yaml -e ~/external-lb.yaml --control-scale 3 --compute-scale 1 --control-flavor control --compute-flavor compute [ADDITIONAL OPTIONS]
```

上記のコマンドは、以下のオプションを使用します。

- **--templates**: デフォルトの Heat テンプレートコレクションからオーバークラウドを作成します。
- **-e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml**: 追加の環境ファイルをオーバークラウドデプロイメントに追加します。ここでは、リソースネットワーク分離の設定を初期化する環境ファイルを追加します。
- **-e ~/network-environment.yaml**: 追加の環境ファイルをオーバークラウドデプロイメントに追加します。ここでは、前のステップで作成したネットワーク環境ファイルです。

- **-e /usr/share/openstack-tripleo-heat-templates/environments/external-loadbalancer-vip.yaml**: 追加の環境ファイルをオーバークラウドデプロイメントに追加します。ここでは、外部負荷分散設定を初期化する環境ファイルを追加します。ネットワーク設定ファイルの後に、この環境ファイルを追加する必要があります。
- **-e ~/external-lb.yaml**: 追加の環境ファイルをオーバークラウドデプロイメントに追加します。ここでは、外部ロードバランサー設定が含まれる環境ファイルです。ネットワーク設定ファイルの後に、この環境ファイルを追加する必要があります。
- **--control-scale 3** - コントローラーノードを 3 つにスケーリングします。
- **--compute-scale 3** - コンピュートノードを 3 つにスケーリングします。
- **--control-flavor コントロール**: コントローラーノードに特定のフレーバーを使用します。
- **--compute-flavor compute**: コンピュートノードに特定のフレーバーを使用します。



注記

オプションの完全な一覧を表示するには、以下を実行します。

```
$ openstack help overcloud deploy
```

パラメーターの例については、『director のインストールと使用方法』 も併せて参照してください。

オーバークラウドの作成プロセスが開始され、director によりノードがプロビジョニングされます。このプロセスは完了するまで多少時間がかかります。オーバークラウドの作成のステータスを確認するには、stack ユーザーとして別のターミナルを開き、以下のコマンドを実行します。

```
$ source ~/stackrc
$ heat stack-list --show-nested
```

4.5. オーバークラウドへのアクセス

director は、director ホストからオーバークラウドに対話するための設定を行い、認証をサポートするスクリプトを作成します。director は、このファイル overcloudrc を stack ユーザーのホームディレクトリーに保存します。このファイルを使用するには、以下のコマンドを実行します。

```
$ source ~/overcloudrc
```

これにより、director ホストの CLI からオーバークラウドと対話するために必要な環境変数が読み込まれます。director のホストとの対話に戻るには、以下のコマンドを実行します。

```
$ source ~/stackrc
```

4.6. オーバークラウド設定の完了

これでオーバークラウドの作成が完了しました。

高可用性クラスターのフェンシングについては、『 director のインストールと使用方法』 を参照してください。

作成後の機能については、『 director のインストールと使用方法』 を参照してください。

付録A HAPROXY のデフォルト設定例

以下は、オーバークラウドのコントローラーノード上の HAProxy のデフォルト設定ファイルの例です。このファイルは、各コントローラーノードの **/etc/haproxy/haproxy.conf** にあります。

```
global
  daemon
  group  haproxy
  log  /dev/log local0
  maxconn  10000
  pidfile  /var/run/haproxy.pid
  user  haproxy

defaults
  log  global
  mode  tcp
  retries  3
  timeout  http-request 10s
  timeout  queue 1m
  timeout  connect 10s
  timeout  client 1m
  timeout  server 1m
  timeout  check 10s

listen aodh
  bind 172.16.20.250:8042
  bind 172.16.20.250:8042
  mode http
  server overcloud-controller-0 172.16.20.150:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.252:8042 check fall 5 inter 2000 rise 2

listen ceilometer
  bind 172.16.20.250:8777
  bind 172.16.23.250:8777
  server overcloud-controller-0 172.16.20.150:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8777 check fall 5 inter 2000 rise 2

listen cinder
  bind 172.16.20.250:8776
  bind 172.16.23.250:8776
  server overcloud-controller-0 172.16.20.150:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8776 check fall 5 inter 2000 rise 2

listen glance_api
  bind 172.16.23.250:9292
  bind 172.16.21.250:9292
  server overcloud-controller-0 172.16.21.150:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:9292 check fall 5 inter 2000 rise 2

listen glance_registry
  bind 172.16.20.250:9191
```

```
  server overcloud-controller-0 172.16.20.150:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9191 check fall 5 inter 2000 rise 2

listen gnocchi
  bind 172.16.23.250:8041
  bind 172.16.21.250:8041
  mode http
  server overcloud-controller-0 172.16.20.150:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8041 check fall 5 inter 2000 rise 2

listen heat_api
  bind 172.16.20.250:8004
  bind 172.16.23.250:8004
  mode http
  server overcloud-controller-0 172.16.20.150:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8004 check fall 5 inter 2000 rise 2

listen heat_cfn
  bind 172.16.20.250:8000
  bind 172.16.23.250:8000
  server overcloud-controller-0 172.16.20.150:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.152:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.151:8000 check fall 5 inter 2000 rise 2

listen heat_cloudwatch
  bind 172.16.20.250:8003
  bind 172.16.23.250:8003
  server overcloud-controller-0 172.16.20.150:8003 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8003 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8003 check fall 5 inter 2000 rise 2

listen horizon
  bind 172.16.20.250:80
  bind 172.16.23.250:80
  mode http
  cookie SERVERID insert indirect nocache
  server overcloud-controller-0 172.16.20.150:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:80 check fall 5 inter 2000 rise 2

listen keystone_admin
  bind 172.16.23.250:35357
  bind 172.16.20.250:35357
  server overcloud-controller-0 172.16.20.150:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:35357 check fall 5 inter 2000 rise 2

listen keystone_admin_ssh
  bind 172.16.20.250:22
  server overcloud-controller-0 172.16.20.150:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:22 check fall 5 inter 2000 rise 2
```

```
listen keystone_public
  bind 172.16.20.250:5000
  bind 172.16.23.250:5000
  server overcloud-controller-0 172.16.20.150:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:5000 check fall 5 inter 2000 rise 2

listen mysql
  bind 172.16.20.250:3306
  option tcpka
  option httpchk
  stick on dst
  stick-table type ip size 1000
  timeout client 0
  timeout server 0
  server overcloud-controller-0 172.16.20.150:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-1 172.16.20.151:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-2 172.16.20.152:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2

listen neutron
  bind 172.16.20.250:9696
  bind 172.16.23.250:9696
  server overcloud-controller-0 172.16.20.150:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9696 check fall 5 inter 2000 rise 2

listen nova_ec2
  bind 172.16.20.250:8773
  bind 172.16.23.250:8773
  server overcloud-controller-0 172.16.20.150:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8773 check fall 5 inter 2000 rise 2

listen nova_metadata
  bind 172.16.20.250:8775
  server overcloud-controller-0 172.16.20.150:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8775 check fall 5 inter 2000 rise 2

listen nova_novncproxy
  bind 172.16.20.250:6080
  bind 172.16.23.250:6080
  balance source
  server overcloud-controller-0 172.16.20.150:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:6080 check fall 5 inter 2000 rise 2

listen nova_osapi
  bind 172.16.20.250:8774
  bind 172.16.23.250:8774
  server overcloud-controller-0 172.16.20.150:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8774 check fall 5 inter 2000 rise 2
```

```
listen nova_placement
  bind 172.16.20.250:8778
  bind 172.16.23.250:8778
  mode http
  server overcloud-controller-0 172.16.20.150:8778 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8778 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8778 check fall 5 inter 2000 rise 2

listen panko
  bind 172.16.20.250:8779 transparent
  bind 172.16.23.250:8779 transparent
  server overcloud-controller-0 172.16.20.150:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8779 check fall 5 inter 2000 rise 2

listen redis
  bind 172.16.20.249:6379
  balance first
  option tcp-check
  tcp-check send AUTH\ p@55w0rd!\r\n
  tcp-check send PING\r\n
  tcp-check expect string +PONG
  tcp-check send info\ replication\r\n
  tcp-check expect string role:master
  tcp-check send QUIT\r\n
  tcp-check expect string +OK
  server overcloud-controller-0 172.16.20.150:6379 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:6379 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:6379 check fall 5 inter 2000 rise 2

listen swift_proxy_server
  bind 172.16.23.250:8080
  bind 172.16.21.250:8080
  server overcloud-controller-0 172.16.21.150:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:8080 check fall 5 inter 2000 rise 2
```