Red Hat OpenStack Platform 17.0

インスタンスの作成と管理

CLI を使用したインスタンスの作成および管理

Last Updated: 2023-09-20

Red Hat OpenStack Platform 17.0 インスタンスの作成と管理

CLI を使用したインスタンスの作成および管理

OpenStack Team
rhos-docs@redhat.com

法律上の通知

概要

本ガイドでは、インスタンスの作成/管理手順について説明します。

目次

はじめに

注記

インスタンスの作成中に、ロールベースのアクセス制御 (RBAC) 共有セキュリティーグループをインスタンスに直接適用することはできません。RBAC 共有セキュリティーグループをインスタンスに適用するには、最初にポートを作成し、共有セキュリティーグループをそのポートに適用してから、そのポートをインスタンスに割り当てる必要があります。セキュリティーグループのポートへの追加 を参照してください。

多様性を受け入れるオープンソースの強化

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

RED HAT ドキュメントへのフィードバック (英語のみ)

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

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

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

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

第1章 インスタンスについて

インスタンスとは、クラウド内の物理コンピュートノードで実行される個別の仮想マシンです。インスタンスを起動するには、フレーバーとイメージまたはブート可能なボリュームのいずれかが必要です。イメージを使用してインスタンスを起動すると、指定されたイメージは、ブート可能なオペレーティングシステムと共にインストールされる仮想ディスクが含まれるベースイメージになります。各インスタンスには、インスタンスディスクとして参照するルートディスクが必要です。Compute サービス (nova) は、インスタンスディスクのサイズを変更し、インスタンスに指定したフレーバーの仕様と一致するようにします。

イメージは、Image サービス (glance) により管理されます。Image サービスのイメージストアには、多数の事前定義済みのイメージが含まれます。コンピュートノードは、インスタンス用に利用可能な仮想 CPU、メモリー、およびローカルディスクリソースを提供します。Block Storage サービス (cinder) は、事前定義されたボリュームを提供します。インスタンスディスクのデータは、インスタンスの削除時に削除される一時ストレージまたは Block Storage サービスによって提供される永続ボリュームのいずれかに保存されます。

Compute サービスは、インスタンスをオンデマンドで提供するための中核的なコンポーネントです。Compute サービスはインスタンスの作成、スケジュールおよび管理を行い、認証には Identity サービスと、インスタンスの起動に使用するイメージには Image サービスと、ユーザー/管理者用のインターフェイスには Dashboard サービス (horizon) と、それぞれ対話します。クラウドユーザーは、インスタンスの作成および管理時に Compute サービスと対話します。OpenStack CLI または Dashboard を使用してインスタンスを作成および管理することができます。

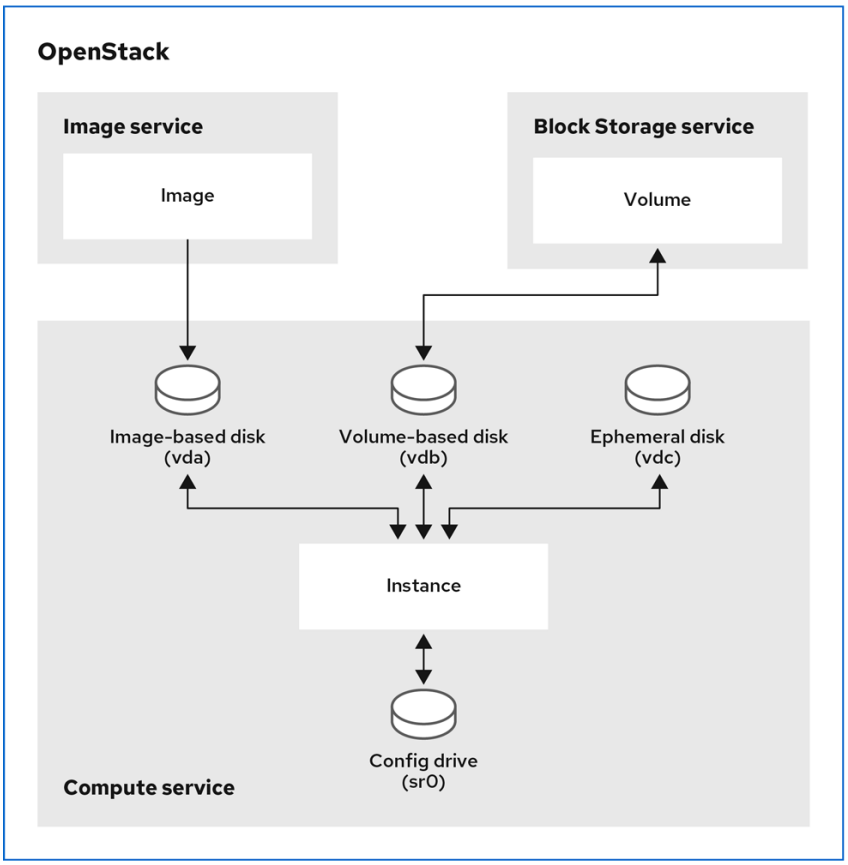
第2章 インスタンスのブートソース

インスタンスのブートソースは、イメージまたはブート可能なボリュームになります。イメージからブートするインスタンスのインスタンスディスクは、Compute サービスにより制御され、インスタンスが削除されると削除されます。ボリュームからブートするインスタンスのインスタンスディスクは、Block Storage サービスにより制御され、リモートに保存されます。

イメージには、ブート可能なオペレーティングシステムが含まれます。Image サービス (glance) は、イメージのストレージと管理をコントロールします。同じベースイメージから、任意の数のインスタンスを起動することができます。各インスタンスは、ベースイメージのコピーから実行されます。インスタンスに変更を加えても、ベースイメージには影響を及ぼしません。

ブート可能なボリュームは、ブート可能なオペレーティングシステムを含むイメージから作成されるブロックストレージボリュームです。インスタンスは、インスタンスの削除時に、ブート可能なボリュームを使用してインスタンスのデータを永続化できます。インスタンスの起動時に、既存の永続ルートボリュームを使用できます。イメージからインスタンスを起動する際に永続ストレージを作成することも可能です。これにより、インスタンスの削除時にインスタンスのデータを保存することができます。新規の永続ストレージボリュームは、ボリュームスナップショットからインスタンスを作成すると自動的に作成されます。

以下の図は、インスタンスの起動時に作成できるインスタンスディスクおよびストレージを示しています。作成される実際のインスタンスディスクおよびストレージは、使用するブートソースとフレーバーによって異なります。



145_OpenStack_0321

第3章 インスタンスのストレージの種別

インスタンスで利用可能な仮想ストレージは、インスタンスの起動に使用するフレーバーにより定義されます。以下の仮想ストレージリソースをインスタンスに関連付けることができます。

- インスタンスディスク
- 一時ストレージ
- スワップストレージ
- 永続ブロックストレージボリューム
- コンフィグドライブ

3.1. インスタンスディスク

インスタンスのデータを保存するために作成されるインスタンスディスクは、インスタンスの作成に使用するブートソースによって異なります。イメージからブートするインスタンスのインスタンスディスクは、Compute サービスにより制御され、インスタンスが削除されると削除されます。ボリュームからブートするインスタンスのインスタンスディスクは、Block Storage サービスにより提供される永続ボリュームです。

3.2. インスタンスの一時ストレージ

一時ディスクを設定するフレーバーを選択して、インスタンス用に一時ディスクが作成されるように指定できます。この一時ストレージは、インスタンスで利用可能な空の追加ディスクです。このストレージの値は、インスタンスのフレーバーにより定義されます。デフォルト値は 0 で、セカンダリー一時ストレージが作成されないことを意味します。

一時ディスクは、外付けのハードドライブや USB ドライブと同じ方法で表示されます。一時ディスクはブロックデバイスとして利用でき、lsblk コマンドを使用して確認することができます。ただし、通常ブロックデバイスを使用するように、それをマウントして使用することができます。接続先のインスタンス以外には、そのディスクを確保したり参照したりすることはできません。

注記

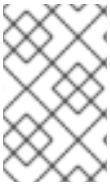
一時ストレージデータはインスタンスのスナップショットには含まれず、退避した後に復元したインスタンスでは利用できません。

3.3. インスタンスのスワップストレージ

スワップディスクを設定するフレーバーを選択して、インスタンス用にスワップディスクが作成されるように指定できます。このスワップストレージは、実行中のオペレーティングシステムのスワップ領域として使用するためにインスタンスで利用可能な追加のディスクです。

3.4. インスタンスのブロックストレージ

ブロックストレージボリュームは、実行中のインスタンスの状態に関係なく、インスタンスで利用可能な永続ストレージです。複数のブロックデバイスをインスタンスに接続することができます。1 つはブート可能なボリュームです。



注記

インスタンスディスクのデータにブロックストレージボリュームを使用する場合、ブロックストレージボリュームは、インスタンスが新規ボリュームの作成を要求する新規イメージで再ビルドされる場合でも、インスタンスの再ビルド後に維持されます。

3.5. コンフィグドライブ

インスタンスのブート時に、インスタンスにコンフィグドライブを接続することができます。コンフィグドライブは読み取り専用ドライブとしてインスタンスに提示されます。インスタンスはこのドライブをマウントして、そこからファイルを読み取ることができます。このコンフィグドライブを **cloud-init** の情報源として使用できます。コンフィグドライブは、**cloud-init** (サーバーのブートストラップ用) と組み合わせる場合や、インスタンスに大容量のファイルを渡す場合に有用です。たとえば、**cloud-init** を設定して、インスタンスの初回ブート中にコンフィグドライブを自動的にマウントして設定スクリプトを実行することができます。コンフィグドライブは **config-2** のボリュームラベルで作成され、インスタンスのブート時にインスタンスにアタッチされます。コンフィグドライブに渡される追加ファイルの内容は、コンフィグドライブの **openstack/{version}/** ディレクトリー内の **user_data** ファイルに追加されます。**cloud-init** はこのファイルからユーザーデータを取得します。

第4章 インスタンスのフレーバー

インスタンスのフレーバーは、インスタンス用の仮想ハードウェアプロファイルを指定するリソースのテンプレートです。インスタンスの起動時にフレーバーを選択し、インスタンスに割り当てる仮想リソースを指定します。フレーバーは、仮想 CPU の数、RAM の容量、ルートディスクのサイズ、およびセカンダリー一時ストレージおよびスワップディスクを含む仮想ストレージのサイズを定義して、これらと共にインスタンスを作成します。クラウド内のプロジェクト用に定義した利用可能なフレーバーのセットからフレーバーを選択します。

第5章 インスタンスの作成

インスタンスを作成する前に、その他の Red Hat OpenStack Platform (RHOSP) のコンポーネント (フレーバー、ブートソース、ネットワーク、キーペア、セキュリティーグループなど) が利用できる状態でなければなりません。これらのコンポーネントは、インスタンスの作成に使用され、デフォルトでは提供されません。

インスタンスの作成時に、インスタンスに必要なブート可能なオペレーティングシステムが含まれるブートソース、インスタンスに必要なハードウェアプロファイルを持つフレーバー、インスタンスを接続するネットワーク、およびデータボリュームや一時ストレージなどの必要な追加のストレージを選択します。

5.1. 前提条件

- ブートソースとして必要なイメージまたはボリュームが利用できる。
  - イメージの作成方法の詳細は、**イメージの作成および管理** の イメージの作成 を参照してください。
  - ボリュームの作成方法の詳細は、**ストレージガイド** の Block Storage ボリュームの作成 を参照してください。
  - インスタンスのブートソースで利用可能なオプションの詳細は、インスタンスのブートソース を参照してください。
- 必要な CPU の数、メモリー、およびストレージ容量を指定するフレーバーが利用可能である。フレーバーの設定は、選択したイメージで指定されたディスクおよびメモリーサイズの最小要件を満たす必要があります。そうでないと、インスタンスの起動に失敗します。
- 必要なネットワークが利用可能である。ネットワークの作成方法の詳細は、**ネットワークガイド** の ネットワークの作成 を参照してください。

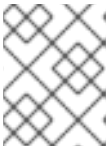
5.2. イメージからのインスタンスの作成

ブートソースとしてイメージを使用して、インスタンスを作成することができます。

手順

1. インスタンスが要求するハードウェアプロファイルを持つフレーバーの名前または ID を取得します。

```
$ openstack flavor list
```



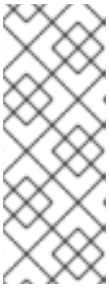
注記

イメージが正常に起動するのに十分なサイズを持つフレーバーを選択します。そうでないと、インスタンスの起動に失敗します。

2. インスタンスが要求するソフトウェアプロファイルを持つイメージの名前または ID を取得します。

```
$ openstack image list
```

必要なイメージが利用できない場合は、新しいイメージをダウンロードまたは作成できます。クラウドイメージの作成またはダウンロードの方法は、イメージの作成 を参照してください。



注記

26 を超えるボリュームをインスタンスにアタッチする必要がある場合は、インスタンスの作成に使用するイメージに以下のプロパティーが必要です。

- **hw_scsi_model=virtio-scsi**
- **hw_disk_bus=scsi**

3. インスタンスを接続するネットワークの名前または ID を取得します。

```
$ openstack network list
```

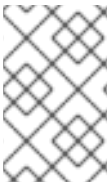
4. インスタンスを作成します。

```
$ openstack server create --flavor <flavor> \
  --image <image> --network <network> \
  --wait myInstanceFromImage
```

- **<flavor>** を、ステップ 1 で取得したフレーバーの名前または ID に置き換えてください。
- **<image>** を、ステップ 2 で取得したイメージの名前または ID に置き換えてください。
- **<network>** を、ステップ 3 で取得したネットワークの名前または ID に置き換えてください。必要に応じて、**--network** オプションを複数回使用して、インスタンスを複数のネットワークに接続することができます。

5.3. ブート可能なボリュームからのインスタンスの作成

ブートソースとしてブート可能なボリュームを使用して、インスタンスを作成することができます。障害発生時にインスタンスデータの可用性を改善する必要がある場合は、ボリュームからインスタンスをブートします。



注記

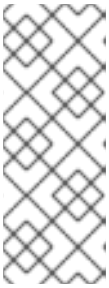
インスタンスディスクのデータにブロックストレージボリュームを使用する場合、ブロックストレージボリュームは、インスタンスが新規ボリュームの作成を要求する新規イメージで再ビルドされる場合でも、インスタンスの再ビルド後に維持されます。

手順

1. インスタンスが要求するソフトウェアプロファイルを持つイメージの名前または ID を取得します。

```
$ openstack image list
```

必要なイメージが利用できない場合は、新しいイメージをダウンロードまたは作成できます。クラウドイメージの作成またはダウンロードの方法は、イメージの作成 を参照してください。



注記

26 を超えるボリュームをインスタンスにアタッチする必要がある場合は、インスタンスの作成に使用するイメージに以下のプロパティーが必要です。

- **hw_scsi_model=virtio-scsi**
- **hw_disk_bus=scsi**

2. イメージからブート可能なボリュームを作成します。

```
$ openstack volume create --image <image> \
--size <size_gb> --bootable myBootableVolume
```

- **<image>** を、ステップ 1 で取得したボリュームに書き込むイメージの名前または ID に置き換えてください。
- **<size_gb>** をボリュームのサイズ (GB 単位) に置き換えます。

3. インスタンスが要求するハードウェアプロファイルを持つフレーバーの名前または ID を取得します。

```
$ openstack flavor list
```

4. インスタンスを接続するネットワークの名前または ID を取得します。

```
$ openstack network list
```

5. ブート可能なボリュームでインスタンスを作成します。

```
$ openstack server create --flavor <flavor> \
 --volume myBootableVolume --network <network> \
 --wait myInstanceFromVolume
```

- **<flavor>** を、ステップ 3 で取得したフレーバーの名前または ID に置き換えてください。
- **<network>** を、ステップ 4 で取得したネットワークの名前または ID に置き換えてください。必要に応じて、**--network** オプションを複数回使用して、インスタンスを複数のネットワークに接続することができます。

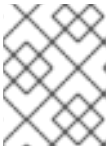
5.4. SR-IOV ネットワークインターフェイスを持つインスタンスの作成

Single Root I/O Virtualization (SR-IOV) ネットワークインターフェイスを持つインスタンスを作成するには、必要な SR-IOV ポートを作成する必要があります。

手順

1. インスタンスが要求するハードウェアプロファイルを持つフレーバーの名前または ID を取得します。

```
$ openstack flavor list
```



注記

イメージが正常に起動するのに十分なサイズを持つフレーバーを選択します。そうでないと、インスタンスの起動に失敗します。

ヒント

必要なポリシーを持つフレーバーを選択して、PCI パススルーデバイスおよび SR-IOV インターフェイス用にインスタンスに適用される NUMA アフィニティーポリシーを指定することができます。使用可能なポリシーの詳細は、**インスタンス作成のための Compute サービスの設定** ガイドの フレーバーメタデータ の **インスタンス PCI NUMA アフィニティーポリシー** を参照してください。NUMA アフィニティーポリシーが指定されたフレーバーを選択する場合には、使用するイメージには同じ NUMA アフィニティーポリシーが設定されているか、NUMA アフィニティーポリシーが設定されていない必要があります。

2. インスタンスが要求するソフトウェアプロファイルを持つイメージの名前または ID を取得します。

```
$ openstack image list
```

必要なイメージが利用できない場合は、新しいイメージをダウンロードまたは作成できます。クラウドイメージの作成またはダウンロードの方法は、イメージの作成 を参照してください。

ヒント

必要なポリシーを持つイメージを選択して、PCI パススルーデバイスおよび SR-IOV インターフェイス用にインスタンスに適用される NUMA アフィニティーポリシーを指定することができます。使用可能なポリシーの詳細は、**インスタンス作成のための Compute サービスの設定** ガイドの フレーバーメタデータ の **インスタンス PCI NUMA アフィニティーポリシー** を参照してください。NUMA アフィニティーポリシーが指定されたイメージを選択する場合には、使用するフレーバーには同じ NUMA アフィニティーポリシーが設定されているか、NUMA アフィニティーポリシーが設定されていない必要があります。

3. インスタンスを接続するネットワークの名前または ID を取得します。

```
$ openstack network list
```

4. SR-IOV インターフェイスに必要なポート種別を作成します。

```
$ openstack port create --network <network> \
 --vnic-type <vnic_type> mySriovPort
```

- **<network>** を、ステップ 3 で取得したネットワークの名前または ID に置き換えてください。
- **<vnic_type>** を以下の値のいずれかに置き換えます。
  - **direct**: ダイレクトモードの SR-IOV Virtual Function (VF) ポートを作成します。
  - **direct-physical**: ダイレクトモードの SR-IOV Physical Function (PF) ポートを作成します。
  - **macvtap**: MacVTap を使用して virtio インターフェイスをインスタンスに公開する間接モードの SR-IOV VF ポートを作成します。

5. インスタンスを作成します。

```
$ openstack server create --flavor <flavor> \
  --image <image> --port <port> \
  --wait mySriovInstance
```

- **<flavor>** を、ステップ 1 で取得したフレーバーの名前または ID に置き換えてください。
- **<image>** を、ステップ 2 で取得したイメージの名前または ID に置き換えてください。
- **<port>** を、ステップ 4 で作成したポートの名前または ID に置き換えてください。

5.5. ポート上での NUMA アフィニティーを持つインスタンスの作成

ポートで NUMA アフィニティーを使用してインスタンスを作成するには、必要な NUMA アフィニティーポリシーを使用してポートを作成し、インスタンスの作成時にポートを指定します。

注記

ポート NUMA アフィニティーポリシーは、フレーバー、イメージ、および PCI NUMA アフィニティーポリシーよりも優先されます。クラウド Operator は、PCI パススルーデバイスごとにデフォルトの NUMA アフィニティーポリシーを設定できます。インスタンスのフレーバー、イメージ、またはポートを使用して、インスタンスに適用されるデフォルトの NUMA アフィニティーポリシーをオーバーライドできます。

前提条件

- **port-numa-affinity-policy** 拡張機能はクラウドプラットフォームで有効にする必要があります。
- サービスプラグインは Networking サービス (neutron) で設定する必要があります。

手順

1. 必要な NUMA アフィニティーポリシーでポートを作成します。

```
$ openstack port create --network <network> \
  [--numa-policy-required | --numa-policy-preferred | --numa-policy-legacy] \
  myNUMAAffinityPort
```

- **<network>** は、インスタンスを接続するテナントネットワークの名前または ID に置き換えます。
- 次のいずれかのオプションを使用して、ポートに適用する NUMA アフィニティーポリシーを指定します。
  - **--numa-policy-required** - このポートのスケジューリングに必要な NUMA アフィニティーポリシー。
  - **--numa-policy-preferred** - このポートのスケジューリング用に優先される NUMA アフィニティーポリシー。
  - **--numa-policy-legacy** - レガシーモードを使用してこのポートをスケジュールする NUMA アフィニティーポリシー。

2. インスタンスを作成します。

```
$ openstack server create --flavor <flavor> \
  --image <image> --port <port> \
  --wait myNUMAAffinityInstance
```

- **<flavor>** は、インスタンスに必要なハードウェアプロファイルを持つフレーバーの名前または ID に置き換えます。
- **<image>** は、インスタンスに必要なソフトウェアプロファイルを含むイメージの名前または ID に置き換えます。
- **<port>** は、ステップ 1 で作成したポートの名前または ID に置き換えてください。

5.6. 関連情報

- カスタムインスタンスの作成

第6章 最小帯域幅 QOS を確保したインスタンスの作成

Quality of Service (QoS) ポリシーを使用して、最小帯域幅の確保を要求するインスタンスを作成することができます。

QoS ポリシーと最小帯域幅を確保するルールの組み合わせは、特定の物理ネットワーク上のポートに割り当てられます。設定したポートを使用するインスタンスを作成する場合、Compute のスケジューリングサービスは、この要求を満たすインスタンス用ホストを選択します。Compute のスケジューリングサービスは、インスタンスをデプロイするホストを選択する前に、他のインスタンスが各物理インターフェイス上に確保する帯域幅を Placement サービスに問い合わせます。

制限/制約

- 新規インスタンスの作成時にのみ、最小帯域幅を確保する QoS ポリシーを割り当てることができます。Compute サービスがインスタンス用リソースの使用状況を更新するのは作成または移動操作時だけなので、すでに動作中のインスタンスに最小帯域幅を確保する QoS ポリシーを割り当てることはできません。つまり、インスタンスで利用可能な最小帯域幅を保証することはできません。
- 最小帯域幅を確保する QoS ポリシーなど、リソース要求が設定されたポートを使用するインスタンスのライブマイグレーションを行うことはできません。ポートにリソース要求が設定されているかどうかを確認するには、以下のコマンドを実行します。

```
$ openstack port show <port_name/port_id>
```

前提条件

- 最小帯域幅ルールの QoS ポリシーを利用することができる。詳細は、**ネットワーキングガイド** の Quality of Service (QoS) ポリシーの設定 を参照してください。

手順

1. 利用可能な QoS ポリシーをリスト表示します。

```
(overcloud)$ openstack network qos policy list
```

```
+--------------------------------------+---------+---------+---------+--------------------------------------+
| ID                                   | Name    | Shared | Default | Project                              |
+--------------------------------------+---------+---------+---------+--------------------------------------+
| 6d771447-3cf4-4ef1-b613-945e990fa59f | policy2 | True   | False   |
ba4de51bf7694228a350dd22b7a3dc24 |
| 78a24462-e3c1-4e66-a042-71131a7daed5 | policy1 | True   | False   |
ba4de51bf7694228a350dd22b7a3dc24 |
| b80acc64-4fc2-41f2-a346-520d7cfe0e2b | policy0 | True   | False   |
ba4de51bf7694228a350dd22b7a3dc24 |
+--------------------------------------+---------+---------+---------+--------------------------------------+
```

2. 利用可能な各ポリシーのルールを確認して、必要な最小帯域幅に関するものを特定します。

```
(overcloud)$ openstack network qos policy show policy0
```

```
+-------------------+--------------------------------------------------------------------------------------------+
| Field             | Value                                                                                      |
```

```
+-------------------------------------------------------------------------------------------------+
| description |
|
| id          | b80acc64-4fc2-41f2-a346-520d7cfe0e2b
|
| is_default  | False
|
| location    | cloud='', project.domain_id=, project.domain_name='Default',
project.id=ba4de51bf7694228a350dd22b7a3dc24, project.name=admin,
region_name=regionOne, zone=
|
| name        | policy0
|
| project_id  | ba4de51bf7694228a350dd22b7a3dc24
|
| rules       | [{min_kbps: 100000, direction: egress, id: d46218fe-9218-4e96-952b-
9f45a5cb3b3c, qos_policy_id: b80acc64-4fc2-41f2-a346-520d7cfe0e2b, type:
minimum_bandwidth}, {min_kbps: 100000, direction: ingress, id: 1202c4e3-a03a-464c-
80d5-0bf90bb74c9d, qos_policy_id: b80acc64-4fc2-41f2-a346-520d7cfe0e2b, type:
minimum_bandwidth}] |
| shared      | True
|
| tags        | []
|
+-------------------------------------------------------------------------------------------------+
```

3. 適切なポリシーからポートを作成します。

```
(overcloud)$ openstack port create port-normal-qos --network net0 --qos-policy policy0
```

4. 使用する NIC ポートを指定して、インスタンスを作成します。

```
$ openstack server create --flavor cirros256 --image cirros-0.3.5-x86_64-disk --nic port-
id=port-normal-qos --wait qos_instance
```

出力の ACTIVE ステータスは、要求された最小帯域幅を保証できるホストにインスタンスが正常に作成されていることを示しています。

6.1. インスタンスからの最小帯域幅を確保する QOS の削除

インスタンスから最小帯域幅を確保する QoS ポリシーの制約を外す場合、インターフェイスの割り当てを解除することができます。

手順

- インターフェイスの割り当てを解除するには、以下のコマンドを入力します。

```
$ openstack server remove port <vm_name|vm_id> <port_name|port_id>
```

第7章 VDPA インターフェイスを使用したインスタンスの作成

重要

この機能は、本リリースでは **テクノロジープレビュー** として提供しているため、Red Hat では全面的にはサポートしていません。これは、テスト用途にのみご利用いただく機能です。実稼働環境にはデプロイしないでください。テクノロジープレビュー機能についての詳細は、対象範囲の詳細 を参照してください。

VDPA の vNIC タイプを持つインスタンスのポートをリクエストすることで、VDPA インターフェイスを持つインスタンスを作成できます。

制限

- VDPA インターフェイスを持つインスタンスを一時停止、ライブマイグレーション、延期、または退避することはできません。
- VDPA インターフェイスをインスタンスから切り離してからインスタンスに再接続することはできません。

手順

1. 物理ネットワークにマップされるネットワークを作成します。

```
$ openstack network create vdpa_network \
 --provider-physical-network tenant \
 --provider-network-type vlan \
 --provider-segment 1337
```

2. ネットワークのサブネットを作成します。

```
$ openstack subnet create vdpa_subnet \
 --network vdpa_net1 \
 --subnet-range 192.0.2.0/24 \
 --dhcp
```

3. VDPA 対応の NIC からポートを作成します。

```
$ openstack port create vdpa_direct_port \
 --network vdpa_network \
 --vnic-type vdpa \
```

4. 使用する NIC ポートを指定して、インスタンスを作成します。

```
$ openstack server create vdpa_instance \
 --flavor cirros256 --image cirros-0.3.5-x86_64-disk \
 --nic port-id=vdpa_direct_port --wait
```

出力の "ACTIVE" ステータスは、要求された VDPA インターフェイスを提供できるホスト上にインスタンスが正常に作成されたことを示します。

第8章 インスタンスの更新

永続ボリュームストレージ、ネットワークインターフェイス、パブリック IP アドレスなど、追加のリソースを実行中のインスタンスに追加および削除できます。インスタンスのメタデータとインスタンスが属するセキュリティーグループを更新することもできます。

8.1. インスタンスへのネットワークの接続

稼働中のインスタンスにネットワークを接続することができます。インスタンスにネットワークを接続すると、Compute サービスはインスタンス用にネットワーク上にポートを作成します。デフォルトのセキュリティーグループを使用し、ネットワーク上にサブネットが 1 つだけある場合には、ネットワークを使用してネットワークインターフェイスをインスタンスに接続します。

手順

1. 利用可能なネットワークを特定し、インスタンスに接続するネットワークの名前または ID をメモします。

```
(overcloud)$ openstack network list
```

   必要なネットワークが利用できない場合は、新規ネットワークを作成します。

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

2. インスタンスにネットワークを接続します。

```
$ openstack server add network <instance> <network>
```

   - **<instance>** を、ネットワークを接続するインスタンスの名前または ID に置き換えてください。
   - **<network>** をインスタンスに接続するネットワークの名前または ID に置き換えてください。

関連情報

- Command Line Interface Reference の openstack network create コマンド。
- **ネットワークガイド** の ネットワークの作成。

8.2. インスタンスからのネットワークの切断

インスタンスからネットワークを切断することができます。

注記

ネットワークを切断すると、すべてのネットワークポートが切断されます。インスタンスにネットワーク上の複数ポートがあり、それらのポートの 1 つだけを接続解除する必要がある場合は、インスタンスからのポートの切断 の手順に従い、ポートの接続を解除します。

手順

1. インスタンスに接続されているネットワークを特定します。

```
(overcloud)$ openstack server show <instance>
```

2. インスタンスからネットワークを切断します。

```
$ openstack server remove network <instance> <network>
```

- **<instance>** を、ネットワークを削除するインスタンスの名前または ID に置き換えてください。
- **<network>** をインスタンスから削除するネットワークの名前または ID に置き換えてください。

8.3. インスタンスへのポートの接続

ポートを使用して、実行中のインスタンスにネットワークインターフェイスを接続することができます。ポートは、一度に 1 つのインスタンスにしか接続できません。カスタムセキュリティーグループを使用する場合や、ネットワークに複数のサブネットがある場合には、ポートを使用してインスタンスにネットワークインターフェイスを接続します。

ヒント

ネットワークを使用してネットワークインターフェイスを接続すると、ポートは自動的に作成されます。詳しくは、インスタンスへのネットワークの接続 を参照してください。

注記

Red Hat OpenStack Platform (RHOSP) は、インスタンスごとに最大 24 のインターフェイスを提供します。デフォルトでは、最大 16 個の PCIe デバイスをインスタンスに追加した後に、インスタンスを再起動しなければ、さらにデバイスを追加できません。RHOSP 管理者は **NovaLibvirtNumPciePorts** パラメーターを使用して、インスタンスに追加できる PCIe デバイスの数を設定できます。これにより、デバイスを追加するためにインスタンスの再起動が必要になります。

前提条件

- SR-IOV vNIC を使用してポートをインスタンスに接続する場合、適切な物理ネットワーク上のホストに空き SR-IOV デバイスが必要であり、インスタンスには空き PCIe スロットが必要です。

手順

1. インスタンスにアタッチするポートを作成します。

```
$ openstack port create --network <network> [--vnic-type <vnic-type>] <port>
```

- **<network>** を、ポートを作成するネットワークの名前または ID に置き換えてください。
- オプション: SR-IOV ポートを作成するには、**<vnic-type>** を次のいずれかの値に置き換えます。
  - **direct**: ダイレクトモードの SR-IOV Virtual Function (VF) ポートを作成します。

- **direct-physical**: ダイレクトモードの SR-IOV Physical Function (PF) ポートを作成します。
- **macvtap** : MacVTap デバイスを介してインスタンスに接続される SR-IOV ポートを作成します。

- **<port>** を、インスタンスに接続するポートの名前または ID に置き換えてください。

2. インスタンスにポートを接続します。

```
$ openstack server add port <instance> <port>
```

- **<instance>** を、ポートを接続するインスタンスの名前または ID に置き換えてください。
- **<port>** を、インスタンスに接続するポートの名前または ID に置き換えてください。

3. ポートがインスタンスにアタッチされていることを確認します。

```
$ openstack port list --device-id <instance_UUID>
```

<instance_UUID> は、ポートをアタッチしたインスタンスの UUID に置き換えます。

関連情報

- **コマンドラインインターフェイスリファレンス** の openstack port create コマンド

8.4. インスタンスからのポートの切断

インスタンスからポートを切断することができます。

手順

1. インスタンスに接続されているポートを特定します。

```
(overcloud)$ openstack server show <instance>
```

2. インスタンスからポートを切断します。

```
$ openstack server remove port <instance> <port>
```

- **<instance>** を、ポートを削除するインスタンスの名前または ID に置き換えてください。
- **<port>** をインスタンスから削除するポートの名前または ID に置き換えてください。

8.5. インスタンスへのボリュームの接続

永続ストレージ用にインスタンスにボリュームを接続することができます。ボリュームがマルチアタッチボリュームとして設定されていない限り、ボリュームは一度に 1 つのインスタンスにしか接続することができません。マルチ接続ボリュームの作成の詳細は、複数のインスタンスに接続できるボリュームを参照してください。

前提条件

- マルチアタッチボリュームをアタッチするために、環境変数 **OS_COMPUTE_API_VERSION** を 2.60 以降に設定する。
- インスタンスは完全に稼働しているか、完全に停止しています。インスタンスが起動中またはシャットダウン中の場合、インスタンスにボリュームをアタッチできません。
- 26 を超えるボリュームをインスタンスにアタッチするには、インスタンスの作成に使用するイメージに以下のプロパティーが必要です。
  - **hw_scsi_model=virtio-scsi**
  - **hw_disk_bus=scsi**

手順

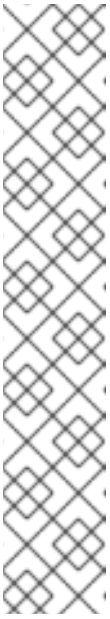
1. 利用可能なボリュームを特定し、インスタンスに接続するボリュームの名前または ID をメモします。

```
(overcloud)$ openstack volume list
```

2. インスタンスにボリュームを接続します。

```
$ openstack server add volume <instance> <volume>
```

- **<instance>** を、ボリュームを接続するインスタンスの名前または ID に置き換えてください。
- **<volume>** を、インスタンスに接続するボリュームの名前または ID に置き換えてください。



注記

コマンドが次のエラーを返した場合、インスタンスにアタッチするために選択したボリュームはマルチアタッチであるため、Compute API バージョン 2.60 以降を使用する必要があります。

```
Multiattach volumes are only supported starting with compute API version 2.60. (HTTP 400) (Request-ID: req-3a969c31-e360-4c79-a403-75cc6053c9e5)
```

インスタンスにボリュームを追加するときに、環境変数 **OS_COMPUTE_API_VERSION=2.72** を設定するか、**--os-compute-api-version** 引数を含めることができます。

```
$ openstack --os-compute-api-version 2.72 server add volume <instance> <volume>
```

ヒント

--os-compute-api-version 2.20 以降を指定して、ステータスが **SHELVED** または **SHELVED_OFFLOADED** のインスタンスにボリュームを追加します。

3. ボリュームがインスタンスにアタッチされていることを確認します。

```
$ openstack volume show <volume>
```

<volume> は、表示するボリュームの名前または ID に置き換えます。

出力例:

```
+-----------------------------------------------------+---------------------+---------+-----+---------------
-------------------------------------------------------------------------+
| ID                                                  | Name                | Status | Size| Attached to
+-----------------------------------------------------+---------------------+---------+-----+---------------
-------------------------------------------------------------------------+
| f3fb92f6-c77b-429f-871d-65b1e3afa750 | volMultiattach | in-use |   50 | Attached to
instance1 on /dev/vdb Attached to instance2 on /dev/vdb  |
+-----------------------------------------------------+---------------------+---------+-----+---------------
-------------------------------------------------------------------------+
```

8.6. インスタンスに接続されているボリュームの表示

特定のインスタンスに接続されているボリュームを表示できます。

前提条件

- **python-openstackclient 5.5.0** を使用しています。

手順

- インスタンスに接続されているボリュームをリスト表示します。

```
$ openstack server volume list <instance>
+--------------------+----------+--------------------+----------------------+
| ID                 | Device   | Server ID          | Volume ID            |
+--------------------+----------+--------------------+----------------------+
| 1f9dcb02-9a20-4a4b- | /dev/vda | ab96b635-1e63-4487- | 1f9dcb02-9a20-4a4b-9f |
| 9f25-c7846a1ce9e8  |          | a85c-854197cd537b  | 25-c7846a1ce9e8      |
+--------------------+----------+--------------------+----------------------+
```

8.7. インスタンスからのボリュームの切断

インスタンスからボリュームを切断することができます。

注記

ネットワークを切断すると、すべてのネットワークポートが切断されます。インスタンスにネットワーク上の複数ポートがあり、それらのポートの1つだけを接続解除する必要がある場合は、インスタンスからのポートの切断 の手順に従い、ポートの接続を解除します。

前提条件

- インスタンスは完全に稼働しているか、完全に停止しています。インスタンスが起動中またはシャットダウン中の場合、インスタンスからボリュームを切り離すことはできません。

手順

1. インスタンスに接続されているボリュームを特定します。

```
(overcloud)$ openstack server show <instance>
```

2. インスタンスからボリュームを切断します。

```
$ openstack server remove volume <instance> <volume>
```

- **<instance>** を、ボリュームを削除するインスタンスの名前または ID に置き換えてください。
- **<volume>** をインスタンスから削除するボリュームの名前または ID に置き換えてください。

注記

--os-compute-api-version 2.20 以降を指定して、ステータスが **SHELVED** または **SHELVED_OFFLOADED** のインスタンスからボリュームを削除します。

第9章 インスタンスへのパブリックアクセスの提供

新規インスタンスには、インスタンスが割り当てられるネットワークの固定 IP アドレスを持つポートが自動的に割り当てられます。この IP アドレスはプライベートであり、インスタンスが削除されるまでインスタンスに永続的に関連付けられます。インスタンス間の通信には、固定 IP アドレスが使用されます。

パブリックインスタンスを共有外部ネットワークに直接接続することができます。この場合、パブリック IP アドレスがインスタンスに直接割り当てられます。これは、プライベートクラウドで操作している場合に便利です。

また、外部プロバイダーネットワークへのルーティング接続が設定されているプロジェクトネットワークを介して、インスタンスへのパブリックアクセスを提供することもできます。パブリッククラウドで作業する場合や、パブリック IP アドレスが制限されている場合、これが望まれる方法です。プロジェクトネットワークを通じてパブリックアクセスを提供するには、プロジェクトネットワークを外部ネットワークに設定されたゲートウェイを持つルーターに接続する必要があります。外部トラフィックがインスタンスに到達するためには、クラウドユーザーは Floating IP アドレスをそのインスタンスに関連付ける必要があります。

インスタンスへのアクセスおよびインスタンスからのアクセスを提供するには、共有外部ネットワークまたはルーティング対応プロバイダーネットワークに接続されているかどうかに関わらず、SSH、ICMP、HTTP などの必要なプロトコルにセキュリティーグループルールを設定する必要があります。また、インスタンスにリモートでアクセスできるように、作成時にキーペアをインスタンスに渡す必要もあります。

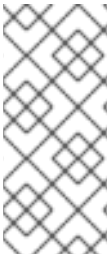
9.1. 前提条件

- 外部ネットワークには、Floating IP アドレスを提供するサブネットが必要です。
- プロジェクトネットワークが、ゲートウェイとして外部ネットワークが設定されたルーターに接続されている必要があります。

9.2. セキュリティーグループとキーペアを使用したインスタンスアクセスのセキュリティー保護

セキュリティーグループとは、インスタンスへの/からのネットワークおよびプロトコルアクセスを制御する IP フィルタールールのセットで、たとえば、ICMP によりインスタンスへの ping 送信が可能で、SSH によりインスタンスへの接続が可能となります。セキュリティーグループルールは、プロジェクト内のすべてのインスタンスに適用されます。

すべてのプロジェクトには、**default** という名前のデフォルトセキュリティーグループがあり、これは、インスタンスにセキュリティーグループを指定しない場合に使用されます。デフォルトでは、デフォルトのセキュリティーグループは、すべての送信トラフィックを許可し、同じセキュリティーグループのインスタンス以外のソースからの着信トラフィックをすべて拒否します。デフォルトのセキュリティーグループにルールを追加するか、プロジェクト用に新規セキュリティーグループを作成できます。インスタンスの作成時に 1 つ以上のセキュリティーグループをインスタンスに適用できます。セキュリティーグループを実行中のインスタンスに適用するには、セキュリティーグループをインスタンスに接続されているポートに適用します。



注記

インスタンスの作成中に、ロールベースのアクセス制御 (RBAC) 共有セキュリティーグループをインスタンスに直接適用することはできません。RBAC 共有セキュリティーグループをインスタンスに適用するには、最初にポートを作成し、共有セキュリティーグループをそのポートに適用してから、そのポートをインスタンスに割り当てる必要があります。セキュリティーグループのポートへの追加 を参照してください。

キーペアは、インスタンスへのリモートアクセスを有効にするために起動時にインスタンスに挿入される SSH または x509 認証情報です。RHOSP で新規キーペアを作成するか、既存のキーペアをインポートできます。各ユーザーには、少なくとも 1 つのキーペアが必要です。キーペアは複数のインスタンスに使用できます。

注記

プロジェクト内のユーザーとキーペアを共有することはできません。各キーペアはプロジェクトではなくキーペアを作成またはインポートした個々のユーザーに属するためです。

9.2.1. セキュリティーグループの作成

新規セキュリティーグループを作成して、プロジェクト内のインスタンスおよびポートに適用できます。

手順

1. (オプション) 必要なセキュリティーグループが存在しないことを確認するには、利用可能なセキュリティーグループとそのルールを確認します。

```
$ openstack security group list
$ openstack security group rule list <sec_group>
```

   - **<sec_group>** を、利用可能なセキュリティーグループのリストから取得したセキュリティーグループの名前または ID に置き換えてください。

2. セキュリティーグループを作成します。

```
$ openstack security group create mySecGroup
```

3. セキュリティーグループにルールを追加します。

```
$ openstack security group rule create --protocol <protocol> \
[--dst-port <port-range>] \
[--remote-ip <ip-address> | --remote-group <group>] \
[--ingress | --egress] mySecGroup
```

   - **<protocol>** を、インスタンスとの通信に許可するプロトコルの名前に置き換えます。
   - (オプション) **<port-range>** を、プロトコル用に開く送信先ポートまたはポート範囲に置き換えます。IP プロトコル (TCP、UDP、および SCTP) には必須です。指定されたプロトコルに対するすべてのポートを許可するには、**-1** に設定します。
   - (オプション) 指定した IP アドレスからのアクセスのみを許可するには、**--remote-ip** を使用してリモート IP アドレスブロックを指定するか、**--remote-group** を使用して、ルール

がリモートグループのメンバーであるインターフェイスからのパケットにのみ適用されることを指定します。**--remote-ip** を使用する場合は、**<ip-address>** をリモート IP アドレスブロックに置き換えます。CIDR 表記を使用できます。**--remote-group** を使用する場合は、**<group>** を既存のセキュリティーグループの名前または ID に置き換えてください。いずれのオプションも指定されない場合は、リモート IP アクセス範囲のデフォルトは IPv4 が **0.0.0.0/0**、IPv6 が **::/0** なので、すべてのアドレスにアクセスが許可されます。

- プロトコルルールが適用されるネットワークトラフィックの方向、つまり受信 (**ingress**) または送信 (**egress**) のいずれかを指定します。指定されない場合、デフォルトは **ingress** に設定されます。

4. インスタンスへのアクセスを許可するすべてのプロトコルに対してルールが作成されるまで、ステップ 3 を繰り返します。以下の例では、セキュリティーグループ **mySecGroup** のインスタンスへの SSH 接続を許可するルールを作成します。

```
$ openstack security group rule create --protocol tcp \
 --dst-port 22 mySecGroup
```

9.2.2. セキュリティーグループルールの更新

アクセス可能なセキュリティーグループのルールを更新できます。

手順

1. ルールを更新するセキュリティーグループの名前または ID を取得します。

```
$ openstack security group list
```

2. セキュリティーグループに適用する必要があるルールを決定します。

3. セキュリティーグループにルールを追加します。

```
$ openstack security group rule create --protocol <protocol> \
[--dst-port <port-range>] \
[--remote-ip <ip-address> | --remote-group <group>] \
[--ingress | --egress] <group_name>
```

- **<protocol>** を、インスタンスとの通信に許可するプロトコルの名前に置き換えます。

- (オプション) **<port-range>** を、プロトコル用に開く送信先ポートまたはポート範囲に置き換えます。IP プロトコル (TCP、UDP、および SCTP) には必須です。指定されたプロトコルに対するすべてのポートを許可するには、**-1** に設定します。

- (オプション) 指定した IP アドレスからのアクセスのみを許可するには、**--remote-ip** を使用してリモート IP アドレスブロックを指定するか、**--remote-group** を使用して、ルールがリモートグループのメンバーであるインターフェイスからのパケットにのみ適用されることを指定します。**--remote-ip** を使用する場合は、**<ip-address>** をリモート IP アドレスブロックに置き換えます。CIDR 表記を使用できます。**--remote-group** を使用する場合は、**<group>** を既存のセキュリティーグループの名前または ID に置き換えてください。いずれのオプションも指定されない場合は、リモート IP アクセス範囲のデフォルトは IPv4 が **0.0.0.0/0**、IPv6 が **::/0** なので、すべてのアドレスにアクセスが許可されます。

- プロトコルルールが適用されるネットワークトラフィックの方向、つまり受信 (**ingress**) または送信 (**egress**) のいずれかを指定します。指定されない場合、デフォルトは **ingress** に設定されます。

- **<group_name>** を、ルールを適用するセキュリティーグループの名前または ID に置き換えてください。

4. インスタンスへのアクセスを許可するすべてのプロトコルに対してルールが作成されるまで、ステップ 3 を繰り返します。以下の例では、セキュリティーグループ **mySecGroup** のインスタンスへの SSH 接続を許可するルールを作成します。

```
$ openstack security group rule create --protocol tcp \
 --dst-port 22 mySecGroup
```

9.2.3. セキュリティーグループルールの削除

セキュリティーグループからルールを削除できます。

手順

1. ルールが適用されるセキュリティーグループを特定します。

```
$ openstack security group list
```

2. セキュリティーグループに関連付けられたルールの ID を取得します。

```
$ openstack security group show <sec-group>
```

3. ルールを削除します。

```
$ openstack security group rule delete <rule> [<rule> ...]
```

<rule> を、削除するルールの ID に置き換えます。削除するルールの ID のスペース区切りのリストを指定して、一度に複数のルールを削除できます。

9.2.4. セキュリティーグループのポートへの追加

default セキュリティーグループは、代替のセキュリティーグループを指定しないインスタンスに適用されます。別のセキュリティーグループを実行中のインスタンスのポートに適用することができます。

手順

1. セキュリティーグループを適用するインスタンスのポートを決定します。

```
$ openstack port list --server myInstancewithSSH
```

2. セキュリティーグループをポートに適用します。

```
$ openstack port set --security-group <sec_group> <port>
```

<sec_group> を、実行中のインスタンスのポートに適用するセキュリティーグループの名前または ID に置き換えてください。必要に応じて、**--security-group** オプションを複数回使用して、複数のセキュリティーグループを適用できます。

9.2.5. ポートからのセキュリティーグループの削除

ポートからセキュリティーグループを削除するには、まずすべてのセキュリティーグループを削除してから、ポートに割り当てられたままにするセキュリティーグループを再度追加します。

手順

1. ポートに関連付けられたセキュリティーグループのリストを表示し、ポートに関連付けたままにするセキュリティーグループの ID を記録します。

```
$ openstack port show <port>
```

2. ポートに関連付けられたセキュリティーグループをすべて削除します。

```
$ openstack port set --no-security-group <port>
```

3. セキュリティーグループをポートに再適用します。

```
$ openstack port set --security-group <sec_group> <port>
```

<sec_group> を、実行中のインスタンスのポートに再度適用するセキュリティーグループの ID に置き換えてください。必要に応じて、**--security-group** オプションを複数回使用して、複数のセキュリティーグループを適用できます。

9.2.6. セキュリティーグループの削除

ポートに関連付けられていないセキュリティーグループを削除できます。

手順

1. 削除するセキュリティーグループの名前または ID を取得します。

```
$ openstack security group list
```

2. 利用可能なポートのリストを取得します。

```
$ openstack port list
```

3. 各ポートで関連付けられたセキュリティーグループを確認します。

```
$ openstack port show <port-uuid> -c security_group_ids
```

削除するセキュリティーグループがポートのいずれかに関連付けられている場合は、まずポートからそのセキュリティーグループを削除する必要があります。詳細は、ポートからのセキュリティーグループの削除 を参照してください。

4. セキュリティーグループを削除します。

```
$ openstack security group delete <group> [<group> ...]
```

<group> を削除するグループの ID に置き換えます。削除するグループの ID のスペース区切りのリストを指定して、一度に複数のグループを削除できます。

9.2.7. 新しい SSH キーペアの生成

プロジェクト内で使用する新しい SSH キーペアを作成できます。

注記

x509 証明書を使用して、Windows インスタンス用のキーペアを作成します。

手順

1. キーペアを作成し、秘密鍵をローカルの **.ssh** ディレクトリーに保存します。

```
$ openstack keypair create <keypair> > ~/.ssh/<keypair>.pem
```

<keypair> を新しいキーペアの名前に置き換えます。

2. 秘密鍵を保護します。

```
$ chmod 600 ~/.ssh/<keypair>.pem
```

9.2.8. 既存の SSH キーペアのインポート

新しいキーペアの作成時に公開鍵ファイルを指定して、Red Hat OpenStack Platform (RHOSP) の外部で作成したプロジェクトに SSH 鍵をインポートできます。

手順

1. 既存のキーファイルからキーペアを作成し、秘密鍵をローカルの **.ssh** ディレクトリーに保存します。
   - 既存の公開鍵ファイルからキーペアをインポートするには、以下のコマンドを入力します。

```
$ openstack keypair create --public-key ~/.ssh/<public_key>.pub \
 <keypair> > ~/.ssh/<keypair>.pem
```

     - **<public_key>** を、キーペアの作成に使用する秘密鍵ファイルの名前に置き換えます。
     - **<keypair>** を新しいキーペアの名前に置き換えます。
   - 既存の秘密鍵ファイルからキーペアをインポートするには、以下のコマンドを入力します。

```
$ openstack keypair create --private-key ~/.ssh/<private_key> \
 <keypair> > ~/.ssh/<keypair>.pem
```

     - **<private_key>** を、キーペアの作成に使用する秘密鍵ファイルの名前に置き換えます。
     - **<keypair>** を新しいキーペアの名前に置き換えます。
2. 秘密鍵を保護します。

```
$ chmod 600 ~/.ssh/<keypair>.pem
```

9.2.9. 関連情報

- **ネットワークガイド** の セキュリティーグループ
- **ユーザーおよびアイデンティティー管理ガイド** の プロジェクトのセキュリティー管理。

9.3. インスタンスへの FLOATING IP アドレスの割り当て

パブリックの Floating IP アドレスをインスタンスに割り当て、インターネット等のクラウド外にあるネットワークとの通信を有効にすることができます。クラウド管理者は、外部ネットワーク用の Floating IP アドレスの利用可能なプールを設定します。このプールからプロジェクトに Floating IP アドレスを確保してから、その Floating IP アドレスをインスタンスに割り当てることができます。

プロジェクトでは、プロジェクト内のインスタンスで使用できる Floating IP アドレスのクォータに制限があります (デフォルトでは 50)。そのため、必要がなくなった場合、再利用するために IP アドレスを解放します。

前提条件

- インスタンスは、外部ネットワーク、またはゲートウェイとして外部ネットワークが設定されたルーターに接続されているプロジェクトネットワーク上になければなりません。
- インスタンスが接続する外部ネットワークには、Floating IP アドレスを提供するサブネットが必要です。

手順

1. 現在のプロジェクトに確保されている Floating IP アドレスを確認します。

```
$ openstack floating ip list
```

使用できる Floating IP アドレスがない場合は、外部ネットワークの割り当てプールから Floating IP アドレスを現在のプロジェクトに確保します。

```
$ openstack floating ip create <provider-network>
```

<provider-network> を、外部アクセスを提供するのに使用する外部ネットワークの名前または ID に置き換えてください。

ヒント

デフォルトでは、Floating IP アドレスは外部ネットワークのプールから無作為に確保されます。クラウド管理者は、--floating-ip-address オプションを使用して、外部ネットワークから特定の Floating IP アドレスを確保することができます。

2. インスタンスに Floating IP アドレスを割り当てます。

```
$ openstack server add floating ip [--fixed-ip-address <ip_address>] \
 <instance> <floating_ip>
```

- **<instance>** をパブリックアクセスを提供するインスタンスの名前または ID に置き換えてください。
- **<floating_ip>** を、インスタンスに割り当てる Floating IP アドレスに置き換えてください。

- (オプション) **<ip_address>** を、Floating IP を割り当てるインターフェイスの IP アドレスに置き換えます。デフォルトでは、Floating IP アドレスは最初のポートに割り当てられます。

3. Floating IP アドレスがインスタンスに割り当てられていることを確認します。

```
$ openstack server show <instance>
```

関連情報

- **ネットワークガイド** の Floating IP プールの作成

9.4. インスタンスからの FLOATING IP アドレスの割り当て解除

インスタンスがパブリックアクセスを要求しなくなった場合には、インスタンスへの割り当てを解除し、割り当てプールに戻します。

手順

1. インスタンスへの Floating IP アドレスの割り当てを解除します。

```
$ openstack server remove floating ip <instance> <ip_address>
```

- **<instance>** を、パブリックアクセスを削除するインスタンスの名前または ID に置き換えてください。
- **<floating_ip>** をインスタンスに割り当てられた Floating IP アドレスに置き換えてください。

2. 割り当てプールに Floating IP アドレスを解放します。

```
$ openstack floating ip delete <ip_address>
```

3. Floating IP アドレスが削除され、割り当てに利用できないことを確認します。

```
$ openstack floating ip list
```

9.5. SSH アクセスが設定されたインスタンスの作成

インスタンスの作成時にキーペアを指定して、インスタンスへの SSH アクセスを提供することができます。キーペアは、起動時にインスタンスに挿入される SSH または x509 認証情報です。各プロジェクトには、少なくとも 1 つのキーペアが必要です。キーペアは、プロジェクトではなく、個々のユーザーに属します。

注記

インスタンスの作成後にキーペアをインスタンスに関連付けることはできません。

インスタンスの作成時にセキュリティーグループをインスタンスに直接適用することや、実行中のインスタンスのポートに適用することができます。

注記

インスタンスの作成中に、ロールベースのアクセス制御 (RBAC) 共有セキュリティーグループを直接インスタンスに適用することはできません。RBAC 共有セキュリティーグループをインスタンスに適用するには、最初にポートを作成し、共有セキュリティーグループをそのポートに適用してから、そのポートをインスタンスに割り当てる必要があります。セキュリティーグループのポートへの追加 を参照してください。

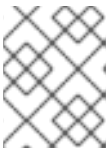
前提条件

- インスタンスに SSH 接続するのに使用できるキーペアが利用可能である。詳細は、新しい SSH キーペアの生成 を参照してください。
- インスタンスを作成するネットワークは、外部ネットワーク、またはゲートウェイとして外部ネットワークが設定されたルーターに接続されているプロジェクトネットワークでなければなりません。詳細は、**ネットワークガイド** の ルーターの追加 を参照してください。
- インスタンスが接続する外部ネットワークには、Floating IP アドレスを提供するサブネットが必要です。
- セキュリティーグループが、インスタンスへの SSH アクセスを許可する。詳細は、セキュリティーグループとキーペアを使用したインスタンスアクセスのセキュリティー保護 を参照してください。
- インスタンスのベースとなっているイメージに、インスタンスに SSH 公開鍵を挿入する **cloud-init** パッケージが含まれている。
- インスタンスに割り当てるフローティング IP アドレスが利用可能である。詳細は、インスタンスへの Floating IP アドレスの割り当て を参照してください。

手順

1. インスタンスが要求するハードウェアプロファイルを持つフレーバーの名前または ID を取得します。

```
$ openstack flavor list
```



注記

イメージが正常に起動するのに十分なサイズを持つフレーバーを選択します。そうでないと、インスタンスの起動に失敗します。

2. インスタンスが要求するソフトウェアプロファイルを持つイメージの名前または ID を取得します。

```
$ openstack image list
```

必要なイメージが利用できない場合は、新しいイメージをダウンロードまたは作成できます。クラウドイメージの作成またはダウンロードの情報は、イメージの作成 を参照してください。

3. インスタンスを接続するネットワークの名前または ID を取得します。

```
$ openstack network list
```

4. インスタンスにリモートでアクセスするために使用するキーペアの名前を取得します。

```
$ openstack keypair list
```

5. SSH アクセスを設定してインスタンスを作成します。

```
$ openstack server create --flavor <flavor> \
  --image <image> --network <network> \
  [--security-group <secgroup>] \
  --key-name <keypair> --wait myInstancewithSSH
```

- **<flavor>** を、ステップ 1 で取得したフレーバーの名前または ID に置き換えてください。
- **<image>** を、ステップ 2 で取得したイメージの名前または ID に置き換えてください。
- **<network>** を、ステップ 3 で取得したネットワークの名前または ID に置き換えてください。必要に応じて、**--network** オプションを複数回使用して、インスタンスを複数のネットワークに接続することができます。
- (オプション) **default** セキュリティーグループは、代替のセキュリティーグループを指定しないインスタンスに適用されます。インスタンスの作成時に別のセキュリティーグループをインスタンスに直接適用することや、実行中のインスタンスのポートに適用することができます。インスタンスの作成時に別のセキュリティーグループを指定するには、**--security-group** オプションを使用します。実行中のインスタンスのポートにセキュリティーグループを追加する方法は、セキュリティーグループのポートへの追加 を参照してください。
- **<keypair>** を、ステップ 4 で取得したキーペアの名前または ID に置き換えてください。

6. インスタンスに Floating IP アドレスを割り当てます。

```
$ openstack server add floating ip myInstancewithSSH <floating_ip>
```

<floating_ip> を、インスタンスに割り当てる Floating IP アドレスに置き換えてください。

7. 自動的に作成された **cloud-user** アカウントを使用して、SSH を使用してインスタンスにログインできることを確認します。

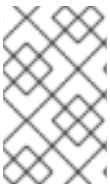
```
$ ssh -i ~/.ssh/<keypair>.pem cloud-user@<floatingIP>
[cloud-user@demo-server1 ~]$
```

9.6. 関連情報

- **ネットワークガイド** の ネットワークの作成。
- **ネットワークガイド** での ルーターの追加。

第10章 インスタンスへの接続

インスタンスのセキュリティーグループルールでプロトコルを許可している場合、SSH や WinRM などのリモートシェルを使用して、クラウド外部の場所からインスタンスにアクセスできます。また、ネットワーク接続に失敗してもデバッグを行うことができるように、インスタンスのコンソールに直接接続することもできます。



注記

インスタンスにキーペアを指定しなかった場合や、インスタンスにセキュリティーグループを割り当てなかった場合には、VNC を使用してクラウド内からのみインスタンスにアクセスできます。インスタンスに ping 送信を行うことはできません。

10.1. インスタンスのコンソールへのアクセス

ブラウザーに VNC コンソールの URL を入力して、インスタンスの VNC コンソールに直接接続することができます。

手順

1. インスタンスの VNC コンソールの URL を表示するには、以下のコマンドを入力します。

```
$ openstack console url show <vm_name>
+-------+-----------------------------------------------------+
| Field | Value                                               |
+-------+-----------------------------------------------------+
| type  | novnc                                               |
| url | http://172.25.250.50:6080/vnc_auto.html?token=       |
| | 962dfd71-f047-43d3-89a5-13cb88261eb9                      |
+-------+-----------------------------------------------------+
```

2. VNC コンソールに直接接続するには、ブラウザーに表示される URL を入力します。

10.2. インスタンスへのログイン

パブリックインスタンスにリモートでログインできます。

前提条件

- インスタンスのキーペアの証明書がある。証明書は、キーペアの作成時にダウンロードされます。キーペアを自分で作成しなかった場合には、管理者に問い合わせてください。
- インスタンスがパブリックインスタンスとして設定されている。パブリックインスタンスの要件についての詳細は、インスタンスへのパブリックアクセスの提供 を参照してください。
- クラウドユーザーアカウントがある。

手順

1. ログインするインスタンスの Floating IP アドレスを取得します。

```
$ openstack server show <instance>
```

<instance> を接続するインスタンスの名前または ID に置き換えてください。

2. 自動的に作成された **cloud-user** アカウントを使用して、インスタンスにログインします。

```
$ ssh -i ~/.ssh/<keypair>.pem cloud-user@<floatingIP>
[cloud-user@demo-server1 ~]$
```

- **<keypair>** をキーペアの名前に置き換えます。
- **<floating_ip>** をインスタンスの Floating IP アドレスに置き換えます。

ヒント

次のコマンドを使用して、Floating IP アドレスなしでインスタンスにログインすることができます。

```
$ openstack server ssh --login cloud-user \
 --identity ~/.ssh/<keypair>.pem --private <instance>
```

  - **<keypair>** をキーペアの名前に置き換えます。
  - **<instance>** を接続するインスタンスの名前または ID に置き換えてください。

第11章 インスタンスの管理

インスタンスのサイズ変更やインスタンスの退避など、インスタンスの管理操作を実行することができます。管理操作の全一覧については、インスタンス管理の操作 を参照してください。

11.1. インスタンスのリサイズ

インスタンスのメモリーまたは CPU 数の増減が必要な場合、インスタンスのサイズを変更できます。インスタンスのサイズを変更するには、必要な容量を持つインスタンス用の新規フレーバーを選択します。インスタンスのサイズを変更すると、インスタンスは再ビルドされ再起動します。

手順

1. サイズ変更するインスタンスの名前または ID を取得します。

```
$ openstack server list
```

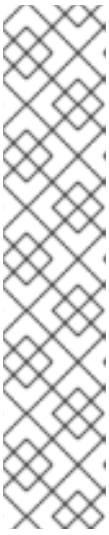
2. インスタンスのサイズ変更に使用するフレーバーの名前または ID を取得します。

```
$ openstack flavor list
```

3. インスタンスのサイズを変更します。

```
$ openstack server resize --flavor <flavor> \
  --wait <instance>
```

- **<flavor>** を、ステップ 2 で取得したフレーバーの名前または ID に置き換えてください。
- **<instance>** を、サイズを変更するインスタンスの名前または ID に置き換えてください。



注記

サイズ変更には時間がかかる場合があります。インスタンスの電源がオフになり、インスタンスのサイズが変更される前に、インスタンスのオペレーティングシステムは制御されたシャットダウンを実行します。この間、インスタンスのステータスは **RESIZE** になります。

```
$ openstack server list
+--------------------+---------------+--------+--------------------------+
| ID                 | Name          | Status | Networks                 |
+--------------------+---------------+--------+--------------------------+
| 67bc9a9a-5928-47c... | myCirrosServer | RESIZE |
admin_internal_net=192.168.111.139    |
+--------------------+---------------+--------+--------------------------+
```

4. サイズ変更が完了すると、インスタンスのステータスが **VERIFY_RESIZE** に変わります。ここで、サイズ変更を確認するか、元に戻す必要があります。

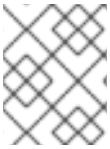
- サイズ変更を確認するには、以下のコマンドを実行します。

```
$ openstack server resize confirm <instance>
```

- サイズ変更を元に戻すには、以下のコマンドを入力します。

```
$ openstack server resize revert <instance>
```

インスタンスは元のフレーバーに戻され、ステータスは **ACTIVE** に変わります。



注記

設定した時間枠内で確認しないと、自動的にインスタンスのサイズ変更を確認するようにクラウドが設定されている場合があります。

11.2. インスタンスのスナップショットの作成

スナップショットは、インスタンスの実行中のディスクの状態をキャプチャーするイメージです。インスタンスのスナップショットを作成して、新規インスタンスを作成するためのテンプレートとして使用できるイメージを作成することができます。スナップショットでは、別のインスタンスから新規インスタンスを作成し、インスタンスの状態を復元することができます。スナップショットのベースとなっているインスタンスを削除する場合、スナップショットイメージを使用して、スナップショットと同じ状態に新規インスタンスを作成できます。

手順

1. スナップショットを作成するインスタンスの名前または ID を取得します。

```
$ openstack server list
```

2. スナップショットを作成します。

```
$ openstack server image create --name <image_name> <instance>
```

- **<image_name>** を新規スナップショットイメージの名前に置き換えます。
- **<instance>** を、スナップショットを作成するインスタンスの名前または ID に置き換えてください。

3. (オプション) インスタンスのスナップショットをテンプレートとして使用し、新規インスタンスを作成する際、ディスクの状態が維持されるようにするには、QEMU ゲストエージェントを有効にし、スナップショットイメージに以下のメタデータを追加して、スナップショット処理中にファイルシステムが休止状態になるようにします。

```
$ openstack image set --property hw_qemu_guest_agent=yes \
 --property os_require_quiesce=yes <image_name>
```

QEMU ゲストエージェントは、管理アプリケーションがインスタンスの OS レベルのコマンドを実行するのに役立つバックグラウンドプロセスです。このエージェントを有効にすると、別のデバイスがインスタンスに追加され、PCI スロットが使用され、インスタンスに割り当てることができるその他のデバイスの数が制限されます。また、これにより、Windows インスタンスでは、未知のハードウェアデバイスについての警告のメッセージが表示されます。

11.3. インスタンスのレスキュー

システムの障害やアクセスの失敗などの緊急事態では、インスタンスをレスキューモードに配置できます。これにより、インスタンスをシャットダウンして、新しいインスタンスディスクを使用してリブー

トし、元のインスタンスディスクおよびコンフィグドライブをボリュームとしてリブートしたインスタンスにマウントします。リブートしたインスタンスに接続し、元のインスタンスディスクを表示してシステムを修復し、データを復元できます。

手順

1. インスタンスのレスキューを実行します。

```
$ openstack server rescue [--image <image>] <instance>
```

   - (オプション) デフォルトでは、インスタンスはクラウド管理者が提供するレスキューイメージまたは元のインスタンスイメージの新規コピーから起動します。レスキューモードでインスタンスをリブートするときに使用する代替イメージを指定するには、**--image** オプションを使用します。
   - **<instance>** を、レスキューするインスタンスの名前または ID に置き換えてください。

2. レスキューされたインスタンスに接続して、問題を修正します。
3. 通常のブートディスクからインスタンスを再起動します。

```
$ openstack server unrescue <instance>
```

11.4. インスタンスの退避

退避は、使用していないが削除したくないインスタンスがある場合に便利です。インスタンスを退避すると、インスタンスのデータとリソースの割り当てが保持されますが、インスタンスのメモリーはクリアされます。クラウドの設定によっては、退避されたインスタンスは、即座に、または一定時間の後に、**SHELVED_OFFLOADED** 状態に移行します。**SHELVED_OFFLOADED** の場合、インスタンスのデータおよびリソースの割り当てが削除されます。

インスタンスを退避すると、Compute サービスはインスタンスの状態をキャプチャーするスナップショットイメージを生成し、**<instance>-shelved** の形式でイメージに名前を割り当てます。このスナップショットイメージは、インスタンスが復元または削除される際に削除されます。

退避されたインスタンスが必要なくなったら、これを削除できます。一度に複数のインスタンスを退避できます。

手順

1. 復元するインスタンスの名前または ID を取得します。

```
$ openstack server list
```

2. インスタンスを退避させます。

```
$ openstack server shelve <instance> [<instance> ...]
```

   <instance> を、退避するインスタンスの名前または ID に置き換えてください。必要に応じて、複数のインスタンスを指定して退避できます。

3. インスタンスが退避されていることを確認します。

```
$ openstack server list
```

退避したインスタンスのステータスは **SHELVED_OFFLOADED** になります。

11.5. インスタンス管理の操作

インスタンスを作成したら、以下の管理操作を実行できます。

表11.1 管理操作

操作	説明	コマンド
インスタンスの停止	インスタンスを停止します。	**openstack server stop**
インスタンスの起動	停止しているインスタンスを起動します。	**openstack server start**
実行中のインスタンスの一時停止	即座に実行中のインスタンスを一時停止します。インスタンスの状態がメモリー (RAM) に保存されます。一時停止されたインスタンスは、引き続きフリーズの状態で実行されます。一時停止アクションの確認は求められません。	**openstack server pause**
一時停止されたインスタンスの実行再開	一時停止されたインスタンスをすぐに再開します。再開アクションの確認は求められません。	**openstack server unpause**
実行中のインスタンスの中断	即座に実行中のインスタンスを中断します。インスタンスの状態がインスタンスのディスクに保存されます。中断アクションの確認は求められません。	**openstack server suspend**
中断されたインスタンスの実行再開	中断されたインスタンスを直ちに再開します。インスタンスの状態がインスタンスのディスクに保存されます。再開アクションの確認は求められません。	**openstack server resume**

操作	説明	コマンド
インスタンスの削除	インスタンスを完全に破棄します。破棄アクションの確認は求められません。ソフト削除を有効にするようにクラウドが設定されていない限り、削除されたインスタンスは復元できません。  **注記** インスタンスを削除しても、接続されていたボリュームは削除されません。接続されていたボリュームは別途削除する必要があります。詳細は、**ストレージガイド** の Block Storage サービス ボリュームの削除 を参照してください。	**openstack server delete**
インスタンスメタデータの編集	インスタンスのメタデータを使用して、インスタンスの属性を指定することができます。詳細は、カスタムインスタンスの作成 を参照してください。	**openstack server set --property <key=value> [--property <key=value>] <instance>**
セキュリティーグループの追加	指定したセキュリティーグループをインスタンスに追加します。	**openstack server add security group**
セキュリティーグループの削除	指定したセキュリティーグループをインスタンスから削除します。	**openstack remove security group**

操作	説明	コマンド
インスタンスのレスキュー	システムの障害やアクセスの失敗などの緊急事態では、インスタンスをレスキューモードに配置できます。これにより、インスタンスがシャットダウンされ、ルートディスクが一時サーバーにマウントされます。一時サーバーに接続して、システムを修復し、データを復元できます。 実行中のインスタンスをレスキューモードにリブートすることもできます。たとえば、インスタンスのファイルシステムが破損している場合は、この操作が必要になる場合があります。	**openstack server rescue**
レスキューされたインスタンスの復元	レスキューされたインスタンスをリブートします。	**openstack server unrescue**
インスタンスログの表示	インスタンスのコンソールログの最新のセクションを表示します。	**openstack console log show**
インスタンスの退避	インスタンスを退避すると、インスタンスのデータとリソースの割り当てが保持されますが、インスタンスのメモリーはクリアされます。クラウドの設定によっては、退避されたインスタンスは、即座に、または一定時間の後に、**SHELVED_OFFLOADED** 状態に移行します。インスタンスの状態が **SHELVED_OFFLOADED** の場合、インスタンスのデータおよびリソースの割り当てが削除されます。インスタンスの状態がインスタンスのディスクに保存されます。インスタンスがボリュームからブートすると、すぐに **SHELVED_OFFLOADED** の状態に移行します。退避アクションの確認は求められません。	**openstack server shelve**
インスタンスの復元	退避したインスタンスのディスクイメージを使用して、インスタンスを復元します。	**openstack server unshelve**

操作	説明	コマンド
インスタンスのロック	管理者以外のユーザーがインスタンスでアクションを実行しないようにインスタンスをロックします。	**openstack server lock** **openstack server unlock**
インスタンスのソフトリブート	インスタンスを正常に停止して再起動します。ソフトリブートは、全プロセスを正常にシャットダウンしてから、インスタンスを再起動するように試みます。デフォルトでは、インスタンスをリブートするとソフトリブートになります。	**openstack server reboot --soft <server>**
インスタンスのハードリブート	インスタンスを停止して再起動します。ハードリブートにより、インスタンスの電源をシャットダウンしてから再びオンにします。	**openstack server reboot --hard <server>**
インスタンスの再ビルド	新しいイメージおよびディスクパーティションのオプションを使用して、インスタンスを再ビルドします。これには、インスタンスのシャットダウン、イメージの再作成、およびリブートが伴います。オペレーティングシステムの問題が発生する場合は、インスタンスを終了して起動する代わりに、このオプションを使用します。	**openstack server rebuild**

第12章 カスタムインスタンスの作成

クラウドユーザーは、インスタンスがブート時に実行するシェルスクリプトなど、インスタンスを起動する際に使用する追加データを指定することができます。クラウドユーザーは、以下の手段を使用してデータをインスタンスに渡すことができます。

ユーザーデータ

cloud-init が実行するインスタンスの起動コマンドに指示を追加するのに使用します。

インスタンスメタデータ

インスタンスの作成または更新時に指定することができるキー/値のペアのリスト

コンフィグドライブまたはメタデータサービスを使用して、インスタンスに渡される追加データにアクセスすることができます。

コンフィグドライブ

インスタンスのブート時に、インスタンスにコンフィグドライブを接続することができます。コンフィグドライブは読み取り専用ドライブとしてインスタンスに提示されます。インスタンスはこのドライブをマウントして、そこからファイルを読み取ることができます。このコンフィグドライブを **cloud-init** の情報源として使用できます。コンフィグドライブは、**cloud-init** (サーバーのブートストラップ用) と組み合わせる場合や、インスタンスに大容量のファイルを渡す場合に有用です。たとえば、**cloud-init** を設定して、インスタンスの初回ブート中にコンフィグドライブを自動的にマウントして設定スクリプトを実行することができます。コンフィグドライブは **config-2** のボリュームラベルで作成され、インスタンスのブート時にインスタンスにアタッチされます。コンフィグドライブに渡される追加ファイルの内容は、コンフィグドライブの **openstack/{version}/** ディレクトリー内の **user_data** ファイルに追加されます。**cloud-init** はこのファイルからユーザーデータを取得します。

メタデータサービス

インスタンス固有のデータを取得するための REST API を提供します。インスタンスは、**169.254.169.254** または **fe80::a9fe:a9fe** からこのサービスにアクセスします。

cloud-init は、コンフィグドライブとメタデータサービスの両方を使用して、インスタンスのカスタマイズ用の追加データを利用することができます。**cloud-init** パッケージは、さまざまなデータ入力形式をサポートします。シェルスクリプトおよび **cloud-config** 形式が、最も一般的な入力形式です。

- シェルスクリプト: データ宣言は **#!** または **Content-Type: text/x-shellscript** で始まります。シェルスクリプトは、ブートプロセスの最後に呼び出されます。
- **cloud-config** format: データ宣言は **#cloud-config** または **Content-Type: text/cloud-config** で始まります。**cloud-config** ファイルが **cloud-init** により解析および実行されるためには、有効な YAML でなければなりません。

注記

cloud-init では、インスタンスに渡されるユーザーデータの最大サイズは 16384 バイトです。サイズの制限を変更することはできないため、サイズの制限を超えるデータが必要な場合は、gzip 圧縮を使用します。

ベンダー固有のデータ

RHOSP 管理者は、インスタンスの作成時にデータを渡すこともできます。クラウドユーザーは、このデータ (例: インスタンスを Active Directory に登録するための暗号化トークン) にアクセスすることができない場合があります。

RHOSP 管理者は、ベンダーデータ機能を使用してデータをインスタンスに渡します。ベンダーデータの設定は読み取り専用で、以下のファイルのいずれかにあります。

- **/openstack/{version}/vendor_data.json**
- **/openstack/{version}/vendor_data2.json**

メタデータサービスを使用して、またはインスタンス上のコンフィグドライブから、これらのファイルを確認することができます。メタデータサービスを使用してファイルにアクセスするには、**http://169.254.169.254/openstack/{version}/vendor_data.json** または **http://169.254.169.254/openstack/{version}/vendor_data2.json** のいずれかに対して GET リクエストを行います。

12.1. ユーザーデータを使用したインスタンスのカスタマイズ

ユーザーデータを使用して、インスタンスの起動コマンドに指示を追加することができます。**cloud-init** はこれらのコマンドを実行して、ブートプロセスの最後のステップとしてインスタンスをカスタマイズします。

手順

1. **cloud-init** への指示が含まれるファイルを作成します。たとえば、インスタンス上に Web サーバーをインストールして有効にする bash スクリプトを作成します。

```
$ vim /home/scripts/install_httpd
#!/bin/bash

yum -y install httpd python-psycopg2
systemctl enable httpd --now
```

2. **--user-data** オプションを使用してインスタンスを起動し、bash スクリプトを渡します。

```
$ openstack server create \
--image rhel8 \
--flavor default \
--nic net-id=web-server-network \
--security-group default \
--key-name web-server-keypair \
--user-data /home/scripts/install_httpd \
--wait web-server-instance
```

3. インスタンスの状態がアクティブになったら、Floating IP アドレスを割り当てます。

```
$ openstack floating ip create web-server-network
$ openstack server add floating ip web-server-instance 172.25.250.123
```

4. SSH でインスタンスにログインします。

```
$ ssh -i ~/.ssh/web-server-keypair cloud-user@172.25.250.123
```

5. カスタマイズが正常に実行されたことを確認します。たとえば、Web サーバーがインストールされ有効になっていることを確認するには、以下のコマンドを入力します。

```
$ curl http://localhost | grep Test
<title>Test Page for the Apache HTTP Server on Red Hat Enterprise Linux</title>
<h1>Red Hat Enterprise Linux <strong>Test Page</strong></h1>
```

6. **/var/log/cloud-init.log** が実行したかどうかなど、関連するメッセージの有無を **cloud-init** ファイルで確認します。

```
$ sudo less /var/log/cloud-init.log
...output omitted...
...util.py[DEBUG]: Cloud-init v. 0.7.9 finished at Sat, 23 Jun 2018 02:26:02 +0000.
Datasource DataSourceOpenStack [net,ver=2].  Up 21.25 seconds
```

12.2. メタデータを使用したインスタンスのカスタマイズ

インスタンスのメタデータを使用して、インスタンスの起動コマンドにインスタンスの属性を指定することができます。

手順

1. **--property <key=value>** オプションでインスタンスを起動します。たとえば、インスタンスを Web サーバーとして識別するには、以下の属性を設定します。

```
$ openstack server create \
--image rhel8 \
--flavor default \
--property role=webservers \
--wait web-server-instance
```

2. (オプション) インスタンスの作成後に、さらにインスタンスに属性を追加します。以下に例を示します。

```
$ openstack server set \
--property region=emea \
--wait web-server-instance
```

12.3. コンフィグドライブを使用したインスタンスのカスタマイズ

インスタンスのブートプロセス中にアタッチされる、インスタンス用のコンフィグドライブを作成することができます。コンフィグドライブに渡したコンテンツは、コンフィグドライブによりインスタンスに提供されます。

手順

1. コンフィグドライブを有効にし、コンフィグドライブで利用可能にするコンテンツが含まれるファイルを指定します。たとえば、以下のコマンドは **config-drive-instance** という名前の新規インスタンスを作成し、ファイル **my-user-data.txt** のコンテンツが含まれるコンフィグドライブをアタッチします。

```
(overcloud)$ openstack server create --flavor m1.tiny \
  --config-drive true \
  --user-data ./my-user-data.txt \
  --image cirros config-drive-instance
```

このコマンドにより、ボリュームラベルが **config-2** のコンフィグドライブが作成され、インスタンスのブート時にアタッチされます。また、**my-user-data.txt** のコンテンツが、コンフィグドライブの **openstack/{version}/** ディレクトリーにある **user_data** ファイルに追加されます。

2. インスタンスにログインします。

3. コンフィグドライブをマウントします。

   - インスタンスの OS が **udev** を使用する場合:

     ```
     # mkdir -p /mnt/config
     # mount /dev/disk/by-label/config-2 /mnt/config
     ```

   - インスタンスの OS が **udev** を使用しない場合は、まずコンフィグドライブに対応するブロックデバイスを特定する必要があります。

     ```
     # blkid -t LABEL="config-2" -odevice
     /dev/vdb
     # mkdir -p /mnt/config
     # mount /dev/vdb /mnt/config
     ```