Red Hat OpenStack Platform 10

ベアメタルプロビジョニング

Bare Metal サービス (Ironic) のインストール、設定、および使用方法

Last Updated: 2020-11-05

Red Hat OpenStack Platform 10 ベアメタルプロビジョニング

Bare Metal サービス (Ironic) のインストール、設定、および使用方法

OpenStack Team
rhos-docs@redhat.com

法律上の通知

概要

本ガイドには、Red Hat OpenStack Platform 環境のオーバークラウドに Bare Metal サービスをインストール、設定、および使用するための手順を記載しています。

目次

前書き

本ガイドには、オーバークラウドに Bare Metal サービス (ironic) をインストールして設定し、そのサービスを使用してエンドユーザー向けの物理マシンのプロビジョニングと管理を行うための手順を記載しています。

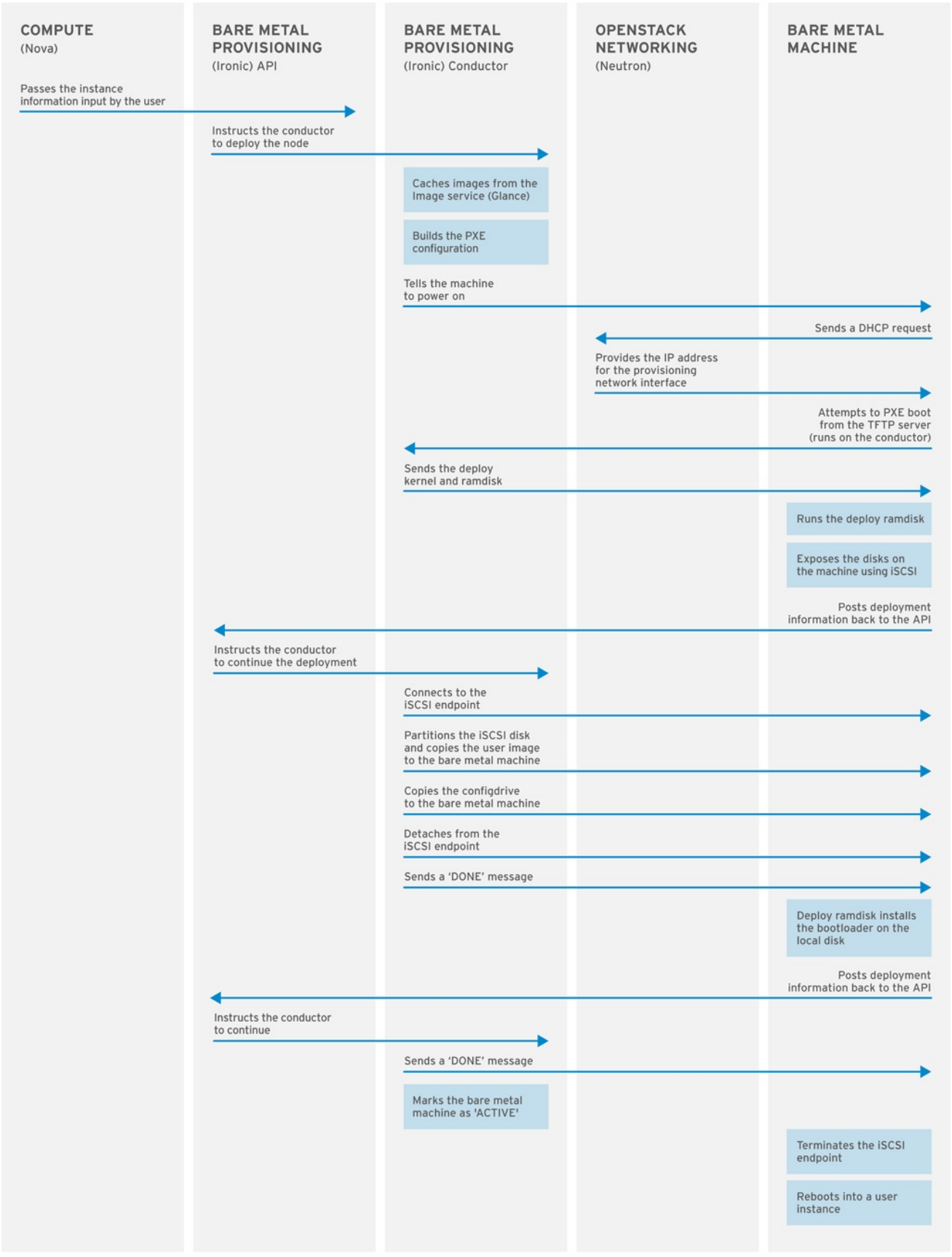
Bare Metal サービスのコンポーネントは、Red Hat OpenStack Platform director で OpenStack 環境 (オーバークラウド) を構成するベアメタルノードのプロビジョニングと管理を行うためにアンダークラウドの一部としても使用されます。director による Bare Metal サービスの使用方法については、『director のインストールと使用方法』を参照してください。

第1章 BARE METAL サービスについて

OpenStack Bare Metal サービス (ironic) は、エンドユーザー向けの物理マシンのプロビジョニングと管理に必要なコンポーネントを提供します。オーバークラウドの Bare Metal サービスは、以下の OpenStack サービスと対話します。

- OpenStack Compute (nova) は、スケジューリング、テナントレベルのクォータ設定、IP の割り当ての機能と、仮想マシンインスタンスを管理するためのユーザー向けの API を提供します。一方、Bare Metal サービスは、ハードウェア管理のための管理 API を提供します。
- OpenStack Identity (keystone) は、要求の認証機能を提供し、Bare Metal サービスが他の OpenStack サービスを特定するのを補助します。
- OpenStack Image サービス (glance) は、イメージとイメージのメタデータを管理します。
- OpenStack Networking (neutron) は、DHCP とネットワーク設定を提供します。
- OpenStack Object Storage (swift) は、特定のドライバーがイメージの一時的な URL を公開するために使用されます。

Bare Metal サービスは、iPXE を使用して物理マシンをプロビジョニングします。以下の図は、ユーザーがデフォルトのドライバーを使用して新規マシンを起動した場合、プロビジョニングプロセス中に OpenStack のサービスがどのように対話するかを概説しています。



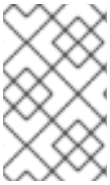
OPENSTACK_377593_1215

第2章 ベアメタルプロビジョニングのプランニング

本章では、インストールの前提条件、ハードウェア要件、ネットワーク要件など、Bare Metal サービスを設定するための要件について説明します。

2.1. インストールの前提条件

本ガイドでは、アンダークラウドに director をインストール済みで、Bare Metal サービスと残りのオーバークラウドをインストールする準備が整っていることを前提とします。director のインストールについての詳しい情報は、『Director Installation and Usage』の「アンダークラウドのインストール」を参照してください。



注記

ベアメタルノードは、OpenStack インストール環境のコントロールプレーンネットワークにアクセスできるため、オーバークラウドの Bare Metal サービスは、信頼済みのテナント環境向けに設計されています。

2.2. ハードウェア要件

オーバークラウドの要件

Bare Metal サービスを有効にしたオーバークラウドのハードウェア要件は、標準のオーバークラウドと同じです。詳しい情報は、**『director のインストールと使用方法』**の「オーバークラウドの要件」を参照してください。

ベアメタルマシンの要件

プロビジョニングするベアメタルマシンのハードウェア要件は、インストールするオペレーティングシステムによって異なります。Red Hat Enterprise Linux 7 の場合は、『Red Hat Enterprise Linux 7 インストールガイド』を参照してください。Red Hat Enterprise Linux 6 の場合は、『Red Hat Enterprise Linux 6 インストールガイド』を参照してください。

プロビジョニングするベアメタルマシンはすべて、以下の要件を満たす必要があります。

- ベアメタルネットワークに接続するための NIC 1 つ。
- ironic-conductor サービスから到達可能なネットワークに接続された電源管理インターフェース (例: IPMI)。テスト目的で SSH ドライバーを使用している場合は、これは必要ありません。デフォルトでは、ironic-conductor は全コントローラーノードで実行されます。
- ベアメタルネットワーク上での PXE ブート。デプロイメント内のその他すべての NIC については PXE ブートを無効にしてください。

2.3. ネットワーク要件

ベアメタルネットワーク:

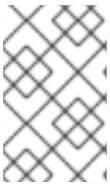
これは、Bare Metal サービスが以下の用途で使用するプライベートネットワークです。

- オーバークラウド上のベアメタルマシンのプロビジョニングと管理
- 再デプロイ前のベアメタルノードのクリーニング
- ベアメタルノードへのテナントアクセス

ベアメタルネットワークは、ベアメタルシステムを検出するための DHCP および PXE ブートの機能を提供します。このネットワークは、Bare Metal サービスが PXE ブートと DHCP 要求に対応できるように、トランキングされたインターフェースでネイティブの VLAN を使用する必要があります。

ベアメタルネットワークがコントロールプレーンネットワークに到達していること:

ベアメタルネットワークは、コントロールプレーンネットワークにルーティングする必要があります。分離したベアメタルネットワークを定義すると、ベアメタルノードは PXE ブートできなくなります。



注記

ベアメタルノードは、OpenStack インストール環境のコントロールプレーンネットワークに直接アクセスできるため、オーバークラウドの Bare Metal サービスは、信頼済みのテナント環境向けに設計されています。

ネットワークのタグ付け:

- コントロールプレーンネットワーク (director のプロビジョニングネットワーク) は常にタグなしです。
- ベアメタルネットワークは、プロビジョニングのためにタグなしである必要があり、また Ironic API にアクセスできなければなりません。
- その他のネットワークはタグ付けすることができます。

オーバークラウドコントローラー:

Bare Metal サービスを有効にしたコントローラーノードは、ベアメタルネットワークにアクセス可能である必要があります。

ベアメタルノード:

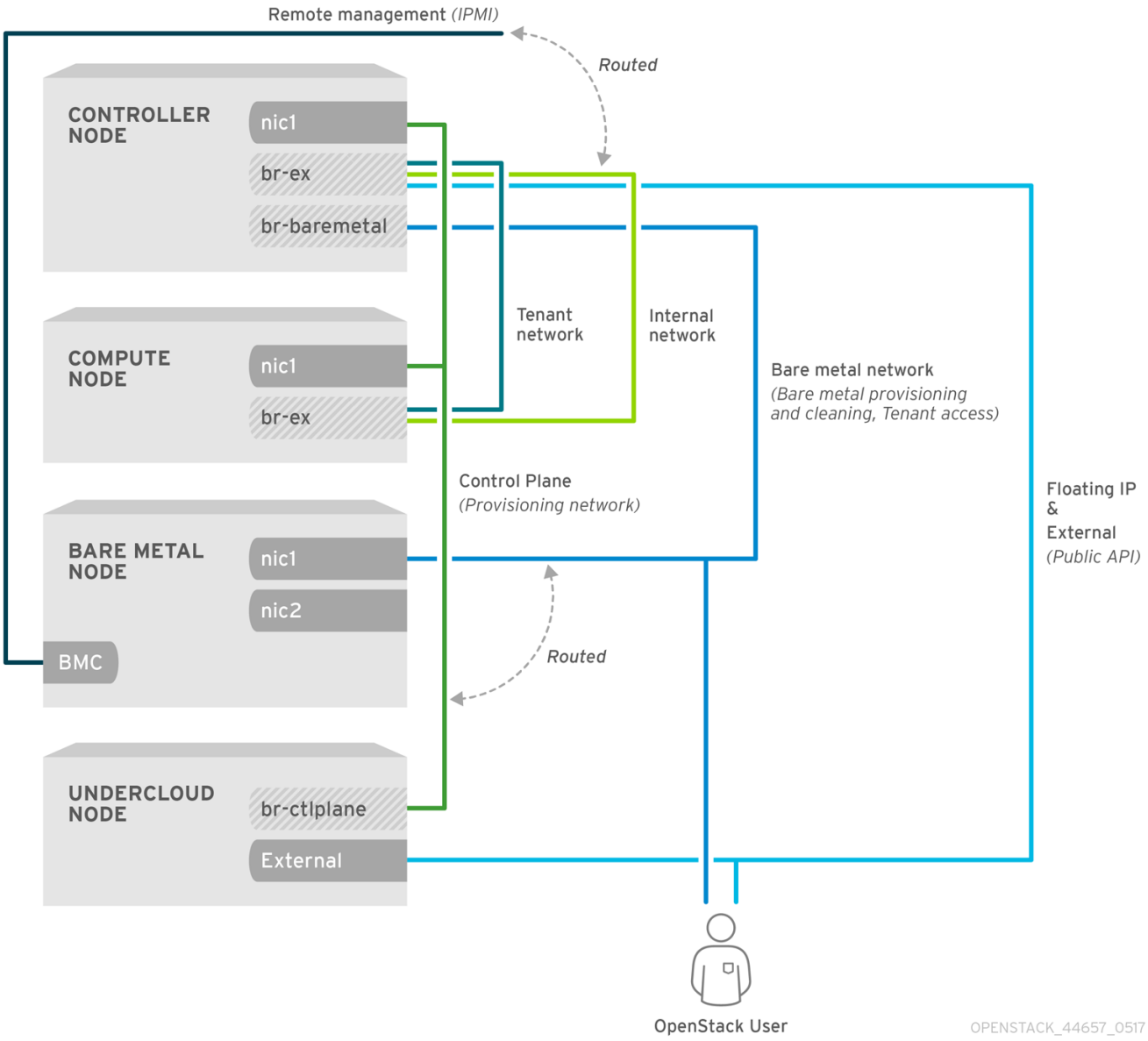
ベアメタルノードの PXE ブートに使用するように設定されている NIC は、ベアメタルネットワークにアクセス可能である必要があります。

2.3.1. デフォルトのベアメタルネットワーク

このアーキテクチャーでは、ベアメタルネットワークはコントロールプレーンネットワークとは分離されています。ベアメタルネットワークは、テナントネットワークとしても機能します。

- ベアメタルネットワークは、OpenStack のオペレーターが作成します。このネットワークには、director のプロビジョニングネットワークへのルートが必要です。
- Ironic ユーザーは、パブリックの OpenStack API とベアメタルネットワークにアクセスすることができます。ベアメタルネットワークは、director のプロビジョニングネットワークにルーティングされるので、ユーザーはコントロールプレーンにも間接的にアクセスできます。
- Ironic は、ノードのクリーニングにベアメタルネットワークを使用します。

デフォルトのベアメタルネットワークアーキテクチャー図


Remote management (IPMI)
Routed
CONTROLLER NODE
nic1
br-ex
br-baremetal
COMPUTE NODE
nic1
br-ex
BARE METAL NODE
nic1
nic2
BMC
UNDERCLOUD NODE
br-ctlplane
External
Tenant network
Internal network
Bare metal network
(Bare metal provisioning and cleaning, Tenant access)
Control Plane
(Provisioning network)
Routed
Floating IP & External
(Public API)
OpenStack User
OPENSTACK_44657_0517

第3章 BARE METAL サービスを有効にしたオーバークラウドのデプロイ

director を使用したオーバークラウドのデプロイメントについての詳しい情報は、『director のインストールと使用方法』を参照してください。本章では、ironic 固有のデプロイメント手順のみを説明します。

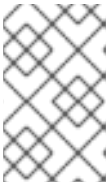
3.1. IRONIC のテンプレートの作成

環境ファイルを使用して、Bare Metal サービスを有効にしたオーバークラウドをデプロイします。テンプレートは、director ノードの **/usr/share/openstack-tripleo-heat-templates/environments/services/ironic.yaml** にあります。

テンプレートへの記入

提供されているテンプレートまたは追加の yaml ファイル (例: **~/templates/ironic.yaml**) で、追加の設定を指定することができます。

- ベアメタルと仮想インスタンスの両方を備えたハイブリッドのデプロイメントでは、**NovaSchedulerDefaultFilters** の一覧に **AggregateInstanceExtraSpecsFilter** を追加する必要があります。**NovaSchedulerDefaultFilters** をどこにも設定していない場合には、ironic.yaml に設定することができます。サンプルは、「テンプレートの例」を参照してください。



注記

SR-IOV を使用している場合には、NovaSchedulerDefaultFilters はすでに **tripleo-heat-templates/environments/neutron-sriov.yaml** で設定されています。このリストに **AggregateInstanceExtraSpecsFilter** を追記してください。

- 初回のデプロイメントおよび再デプロイメントの前に実行されるクリーニングの種別は、**IronicCleaningDiskErase** で設定されます。デフォルトでは、これは **puppet/services/ironic-conductor.yaml** によって「full」に設定されます。この設定を「metadata」にすると、パーティションテーブルのみがクリーニングされるので処理速度を大幅に向上させることができますが、複数のテナントがある環境ではデプロイメントのセキュリティーレベルが低くなるため、信頼済みのテナント環境でのみ適用すべきです。

- **IronicEnabledDrivers** パラメーターを使用してドライバーを追加することができます。デフォルトでは、**pxe_ipmitool**、**pxe_drac**、および **pxe_ilo** が有効です。

設定パラメーターの全一覧は、『オーバークラウドのパラメーター』の「Bare Metal (ironic) パラメーター」セクションを参照してください。

3.2. ネットワーク設定

ironic が使用する **br-baremetal** という名前のブリッジを作成します。これは、追加のテンプレートで指定することができます。

~/templates/network-environment.yaml

```
parameter_defaults:
  NeutronBridgeMappings: datacentre:br-ex,baremetal:br-baremetal
  NeutronFlatNetworks: datacentre,baremetal
```

このブリッジをコントローラーのプロビジョニングネットワーク (コントローラープレーン) 内に設定して、このネットワークをベアメタルネットワークとして再利用できるようにするか、専用のネットワークを追加することができます。設定の要件は同じですが、ベアメタルネットワークはプロビジョニングに使用するので VLAN タグ付けはできません。

~/templates/nic-configs/controller.yaml

```
network_config:
    -
      type: ovs_bridge
        name: br-baremetal
        use_dhcp: false
        members:
          -
            type: interface
            name: eth1
```

3.3. テンプレートの例

テンプレートファイルの例を以下に示します。このファイルは、お使いの環境の要件を満たさない可能性があります。このサンプルを使用する前には、お使いの環境内の既存の設定を干渉しないことを確認してください。

~/templates/ironic.yaml

```
parameter_defaults:

    NovaSchedulerDefaultFilters:
        - RetryFilter
        - AggregateInstanceExtraSpecsFilter
        - AvailabilityZoneFilter
        - RamFilter
        - DiskFilter
        - ComputeFilter
        - ComputeCapabilitiesFilter
        - ImagePropertiesFilter

    IronicCleaningDiskErase: metadata
```

この例では、

- **AggregateInstanceExtraSpecsFilter** は、ハイブリッドデプロイメント向けに、仮想インスタンスとベアメタルインスタンスの両方を許可します。
- 初回のデプロイメントまたは再デプロイメントの前に実行されるディスククリーニングでは、パーティションテーブル (metadata) のみが消去されます。

3.4. オーバークラウドのデプロイ

Bare Metal サービスを有効にするには、オーバークラウドの初回のデプロイメントまたは再デプロイメントの時に **-e** を使用して ironic の環境ファイルをオーバークラウドの残りの設定と共に追加します。

以下に例を示します。

```
$ openstack overcloud deploy \
  --templates \
  -e ~/templates/node-info.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml \
  -e ~/templates/network-environment.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/ironic.yaml \
  -e ~/templates/ironic.yaml \
```

オーバークラウドのデプロイについての詳しい情報は、『director のインストールと使用方法』の「CLI ツールを使用したオーバークラウドの作成」および「オーバークラウド作成時の環境ファイルの追加」を参照してください。

3.5. BARE METAL サービスのテスト

OpenStack Integration Test Suite を使用して、Red Hat OpenStack デプロイメントを検証することができます。詳しい情報は、『OpenStack Integration Test Suite Guide』 を参照してください。

Bare Metal サービスを検証するその他の方法

1. Identity に管理ユーザーとしてアクセスするためのシェルを設定します。

   ```
   $ source ~/overcloudrc
   ```

2. **nova-compute** サービスがコントローラーノードで実行中であることを確認します。

   ```
   $ openstack compute service list -c Binary -c Host -c Status
   ```

3. デフォルトの ironic ドライバーを変更した場合には、必要なドライバーを必ず有効にしてください。

   ```
   $ openstack baremetal driver list
   ```

4. ironic のエンドポイントがリストされていることを確認します。

   ```
   $ openstack catalog list
   ```

第4章 デプロイ後の BARE METAL サービスの設定

本項では、デプロイ後のオーバークラウドの設定に必要な手順について説明します。

4.1. OPENSTACK NETWORKING の設定

DHCP、PXE ブート、およびその他の必要な場合に OpenStack Networking が Bare Metal サービスと通信するように設定します。以下の手順では、ベアメタルのプロビジョニングに使用する単一のフラットなネットワークのユースケース向けに OpenStack Networking を設定します。この設定では、ML2 プラグインと Open vSwitch エージェントを使用します。**flat** ネットワークのみがサポートされます。

この手順では、ベアメタルネットワークインターフェースを使用してブリッジを作成し、リモート接続をすべて破棄します。

以下の手順はすべて、**root** ユーザーとしてログインしている間、OpenStack Networking をホストするサーバーで実行する必要があります。

OpenStack Networking が Bare Metal サービスと通信するための設定

1. Identity に管理ユーザーとしてアクセスするためのシェルを設定します。

```
$ source ~/overcloudrc
```

2. ベアメタルインスタンスをプロビジョニングするためのフラットなネットワークを作成します。

```
$ openstack network create \
  --provider-network-type flat \
  --provider-physical-network baremetal \
  --share NETWORK_NAME
```

NETWORK_NAME は、このネットワークの名前に置き換えます。仮想ネットワークの実装先となる物理ネットワークの名前 (この場合は **baremetal**) は、以前の手順で **~/templates/network-environment.yaml** に **NeutronBridgeMappings** パラメーターで設定されています。

3. フラットネットワーク上にサブネットを作成します。

```
$ openstack subnet create \
  --network NETWORK_NAME \
  --subnet-range NETWORK_CIDR \
  --ip-version 4 \
  --gateway GATEWAY_IP \
  --allocation-pool start=START_IP,end=END_IP \
  --dhcp SUBNET_NAME
```

以下の値を置き換えてください。

- SUBNET_NAME は、サブネットの名前に置き換えます。
- NETWORK_NAME は、以前のステップで作成済みのプロビジョニングネットワークの名前に置き換えます。
- NETWORK_CIDR は、サブネットが示す IP アドレスブロックの Classless Inter-Domain

Routing (CIDR) 表記に置き換えます。**START_IP** で始まり **END_IP** で終る範囲で指定する IP アドレスブロックは、**NETWORK_CIDR** で指定されている IP アドレスブロックの範囲内に入る必要があります。

- **GATEWAY_IP** は、新しいサブネットのゲートウェイとして機能するルーターインターフェースの IP アドレスまたはホスト名に置き換えます。このアドレスは、**NETWORK_CIDR** で指定されている IP アドレスブロック内で、かつ **START_IP** で始まり **END_IP** で終わる範囲で指定されている IP アドレスブロック外である必要があります。

- **START_IP** は、Floating IP アドレスを確保する新規サブネット内の IP アドレス範囲の開始アドレスを示す IP アドレスに置き換えます。

- **END_IP** は、Floating IP アドレスを確保する新規サブネット内の IP アドレス範囲の終了アドレスを示す IP アドレスに置き換えます。

4. ネットワークとサブネットをルーターに接続して、メタデータ要求が OpenStack Networking サービスによって処理されるようにします。

```
$ openstack router create ROUTER_NAME
```

ROUTER_NAME は、ルーターの名前に置き換えます。

5. このルーターに Bare Metal サブネットを追加します。

```
$ openstack router add subnet ROUTER_NAME BAREMETAL_SUBNET
```

ROUTER_NAME をルーターの名前に、**BAREMETAL_SUBNET** を以前のステップで作成したサブネットの ID または名前に、それぞれ置き換えます。これにより、**cloud-init** からのメタデータ要求に対応すると共に、ノードを設定することができます。

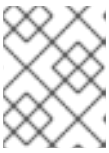
6. Bare Metal サービスを実行しているコントローラー上のプロバイダーネットワークの UUID を指定して、クリーニングを設定します。
~/templates/ironic.yaml

```
parameter_defaults:
  ControllerExtraConfig:
    ironic::conductor::cleaning_network_uuid: UUID
```

UUID は、以前のステップで作成されたベアメタルネットワークの UUID に置き換えます。

UUID は、**openstack network show** で確認することができます。

```
openstack network show NETWORK_NAME -f value -c id
```



注記

ネットワークの UUID は、オーバークラウドの初回のデプロイメントが完了するまで利用できないので、この設定はデプロイ後に実行する必要があります。

7. 「オーバークラウドのデプロイ」の説明に従って **openstack overcloud deploy** コマンドを実行し、オーバークラウドを再デプロイして変更を適用します。

4.2. ベアメタルフレーバーの作成

デプロイメントの一部として使用するフレーバーを作成する必要があります。このフレーバーの仕様 (メモリー、CPU、ディスク) はベアメタルノードが提供する仕様以下である必要があります。

1. Identity に管理ユーザーとしてアクセスするためのシェルを設定します。

```
$ source ~/overcloudrc
```

2. 既存のフレーバーを一覧表示します。

```
$ openstack flavor list
```

3. Bare Metal サービス向けに新規フレーバーを作成します。

```
$ openstack flavor create \
  --id auto --ram RAM \
  --vcpus VCPU --disk DISK \
  --property baremetal=true \
  --public baremetal
```

RAM はメモリー量、**VCPU** は仮想 CPU 数、 **DISK** はディスクストレージの値に置き換えます。**baremetal** プロパティーは、ベアメタルを仮想インスタンスと区別するために使用されます。

4. 指定したそれぞれの値を使用して新規フレーバーが作成されたことを確認します。

```
$ openstack flavor list
```

4.3. ベアメタルイメージの作成

デプロイメントには 2 セットのイメージが必要です。

- **デプロイイメージ** は、Bare Metal サービスがベアメタルノードをブートしてユーザーイメージをベアメタルノードにコピーするのに使用されます。デプロイイメージは、**カーネル** イメージと **ramdisk** イメージで構成されます。
- **ユーザーイメージ** は、ベアメタルノードにデプロイされるイメージです。ユーザーイメージにも **カーネル** イメージと **ramdisk** イメージが含まれますが、追加で **メイン** イメージも含まれます。メインイメージは、ルートパーティションイメージまたは完全なディスクイメージのいずれかです。
  - **完全なディスクイメージ** は、パーティションテーブルとブートローダーを含むイメージです。完全なディスクイメージを使用してデプロイされたノードはローカルブートをサポートするので、Bare Metal サービスはデプロイ後のノードのリブートは制御しません。
  - **ルートパーティションイメージ** には、オペレーティングシステムのルートパーティションのみが含まれています。ルートパーティションを使用する場合には、デプロイイメージが Image サービスに読み込まれた後に、ノードのプロパティーにデプロイイメージをノードのブートイメージとして設定することができます。デプロイ後のノードのリブートでは、netboot を使用してユーザーイメージがプルダウンされます。

本項に記載する例では、ルートパーティションイメージを使用してベアメタルノードをプロビジョニングします。

4.3.1. デプロイイメージの準備

デプロイイメージを作成する必要はありません。アンダークラウドによるオーバークラウドのデプロイ時に、すでにデプロイイメージが使用されているためです。デプロイイメージは、以下に示したように、kernel イメージと ramdisk イメージの 2 つのイメージで構成されます。

```
ironic-python-agent.kernel
ironic-python-agent.initramfs
```

これらのイメージは、削除したり他の場所でアンパックしたりしていない限りは、多くの場合、ホームディレクトリーにあります。ホームディレクトリーにない場合でも、**rhosp-director-images-ipa** パッケージがインストールされているので、これらのイメージは **/usr/share/rhosp-director-images/ironic-python-agent*.tar** ファイル内にあります。

イメージを抽出して Image サービスにアップロードします。

```
$ openstack image create \
  --container-format aki \
  --disk-format aki \
  --public \
  --file ./ironic-python-agent.kernel bm-deploy-kernel
$ openstack image create \
  --container-format ari \
  --disk-format ari \
  --public \
  --file ./ironic-python-agent.initramfs bm-deploy-ramdisk
```

4.3.2. ユーザーイメージの準備

最後に必要となるイメージは、ベアメタルノードにデプロイされるユーザーイメージです。ユーザーイメージには、カーネルイメージと ramdisk イメージに加えて、メインイメージが含まれます。

1. カスタマーポータル (ログインが必要) から Red Hat Enterprise Linux KVM ゲストイメージをダウンロードします。
2. DIB_LOCAL_IMAGE をダウンロードしたイメージとして定義します。

   ```
   $ export DIB_LOCAL_IMAGE=rhel-server-7.4-beta-1-x86_64-kvm.qcow2
   ```

3. **diskimage-builder** ツールを使用してユーザーイメージを作成します。

   ```
   $ disk-image-create rhel7 baremetal -o rhel-image
   ```

   これで kernel は **rhel-image.vmlinuz** として、初期 ramdisk は **rhel-image.initrd** として抽出されます。

4. イメージを Image サービスにアップロードします。

   ```
   $ KERNEL_ID=$(openstack image create \
     --file rhel-image.vmlinuz --public \
     --container-format aki --disk-format aki \
     -f value -c id rhel-image.vmlinuz)
   $ RAMDISK_ID=$(openstack image create \
     --file rhel-image.initrd --public \
   ```

```
  --container-format ari --disk-format ari \
  -f value -c id rhel-image.initrd)
$ openstack image create \
  --file rhel-image.qcow2   --public \
  --container-format bare \
  --disk-format qcow2 \
  --property kernel_id=$KERNEL_ID \
  --property ramdisk_id=$RAMDISK_ID \
  rhel-image
```

4.4. ベアメタルノードとしての物理マシンの追加

ベアメタルノードの登録には 2 つの方法があります。

1. ノードの詳細情報を記載したインベントリーファイルを作成し、そのファイルを Bare Metal サービスにインポートしてからノードを利用できるようにします。

2. 物理マシンをベアメタルノードとして登録してから、手動でハードウェア情報を追加し、各イーサネットの MAC アドレス用にポートを作成します。これらの手順は、overcloudrc ファイルが配置されている任意のノードで実行できます。

本項では、両方の方法について詳しく説明します。

物理マシンの登録後、新規リソースは Compute に直ぐには通知されません。これは、Compute のリソーストラッカーが定期的にしか同期していないためです。次の定期タスクが実行されると、変更が認識されるようになります。この値 **scheduler_driver_task_period** は、**/etc/nova/nova.conf** で更新することができます。デフォルトの間隔は 60 秒です。

4.4.1. インベントリーファイルを使用したベアメタルノードの登録

1. ノードの詳細情報を記載したファイル **overcloud-nodes.yaml** を作成します。1 つのファイルで複数のノードを登録することが可能です。

```
nodes:
    - name: node0
      driver: pxe_ipmitool
      driver_info:
        ipmi_address: <IPMI_IP>
        ipmi_username: <USER>
        ipmi_password: <PASSWORD>
      properties:
        cpus: <CPU_COUNT>
        cpu_arch: <CPU_ARCHITECTURE>
        memory_mb: <MEMORY>
        local_gb: <ROOT_DISK>
        root_device:
            serial: <SERIAL>
      ports:
        - address: <PXE_NIC_MAC>
```

以下の値を置き換えてください。

- **<IPMI_IP>** は、Bare Metal コントローラーの IP アドレスに置き換えます。

- **<USER>** は、ユーザー名に置き換えます。

- **<PASSWORD>** は、パスワードに置き換えます。
- **<CPU_COUNT>** は、CPU の数に置き換えます。
- **<CPU_ARCHITECTURE>** は、CPU のアーキテクチャー種別に置き換えます。
- **<MEMORY>** は、メモリー容量 (MiB 単位) に置き換えます。
- **<ROOT_DISK>** は、ルートディスクの容量 (GiB 単位) に置き換えます。
- **<MAC_ADDRESS>** は、PXE ブートで使用する NIC の MAC アドレスに置き換えます。マシンに複数のディスクがある場合に、含める必要があるのは **root_device** のみです。**<SERIAL>** は、デプロイメントに使用するディスクのシリアル番号に置き換えます。

2. Identity を管理ユーザーとして使用するためのシェルを設定します。

```
$ source ~/overcloudrc
```

3. インベントリーファイルを ironic にインポートします。

```
$ openstack baremetal create overcloud-nodes.yaml
```

4. これで、ノードは **enroll** の状態となります。各ノードでデプロイカーネルとデプロイ ramdisk を指定して、利用できるようにします。

```
$ openstack baremetal node set NODE_UUID \
  --driver-info deploy_kernel=KERNEL_UUID \
  --driver-info deploy_ramdisk=INITRAMFS_UUID
```

以下の値を置き換えてください。

- NODE_UUID は、ノードの一意識別子に置き換えます。もしくは、ノードの論理名を使用します。
- KERNEL_UUID は、Image サービスにアップロードした **カーネル** デプロイイメージの一意識別子に置き換えます。この値は以下のコマンドで確認します。

```
$ openstack image show bm-deploy-kernel -f value -c id
```

- INITRAMFS_UUID は、Image サービスにアップロードした **ramdisk** イメージの一意識別子に置き換えます。この値は以下のコマンドで確認します。

```
$ openstack image show bm-deploy-ramdisk -f value -c id
```

5. ノードが正常に登録されたことを確認します。

```
$ openstack baremetal node list
```

ノードを登録した後にその状態が表示されるまで時間がかかる場合があります。

4.4.2. ベアメタルノードの手動登録

1. Identity を管理ユーザーとして使用するためのシェルを設定します。

```
$ source ~/overcloudrc
```

2. 新しいノードを追加します。

```
$ openstack baremetal node create --driver pxe_impitool --name NAME
```

ノードを作成するには、ドライバー名を指定する必要があります。この例では **pxe_impitool** を使用しています。異なるドライバーを使用するには、**IronicEnabledDrivers** パラメーターを設定してそのドライバーを有効化する必要があります。サポートされているドライバーについての詳しい情報は、「付録A *Bare Metal のドライバー*」を参照してください。

重要

ノードの一意識別子を書き留めておきます。

3. ノードのドライバーの情報を更新して、Bare Metal サービスがノードを管理できるようにします。

```
$ openstack baremetal node set NODE_UUID \
  --driver_info PROPERTY=VALUE \
  --driver_info PROPERTY=VALUE
```

以下の値を置き換えてください。

- **NODE_UUID** は、ノードの一意識別子に置き換えます。もしくは、ノードの論理名を使用します。
- **PROPERTY** は、**ironic driver-properties** コマンドで返された必要なプロパティーに置き換えます。
- **VALUE** は、プロパティーの有効な値に置き換えます。

4. ノードドライバーのデプロイカーネルとデプロイ ramdisk を指定します。

```
$ openstack baremetal node set NODE_UUID \
  --driver-info deploy_kernel=KERNEL_UUID \
  --driver-info deploy_ramdisk=INITRAMFS_UUID
```

以下の値を置き換えてください。

- **NODE_UUID** は、ノードの一意識別子に置き換えます。もしくは、ノードの論理名を使用します。
- **KERNEL_UUID** は、Image サービスにアップロードされた **.kernel** イメージの一意識別子に置き換えます。
- **INITRAMFS_UUID** は、Image サービスにアップロードされた **.initramfs** イメージの一意識別子に置き換えます。

5. ノードのプロパティーを更新して、ノード上のハードウェアの仕様と一致するようにします。

```
$ openstack baremetal node set NODE_UUID \
  --property cpus=CPU \
  --property memory_mb=RAM_MB \
```

```
--property local_gb=DISK_GB \
--property cpu_arch=ARCH
```

以下の値を置き換えてください。

- **NODE_UUID** は、ノードの一意識別子に置き換えます。もしくは、ノードの論理名を使用します。
- **CPU** は、CPU の数に置き換えます。
- **RAM_MB** は、メモリー (MB 単位) に置き換えます。
- **DISK_GB** は、ディスク容量 (GB 単位) に置き換えます。
- **ARCH** は、アーキテクチャー種別に置き換えます。

6. オプション: 初回のデプロイの後には、**ironic-conductor** から PXE を使用する代わりに、ノードのディスクにインストールされたローカルのブートローダーからリブートするようにノードを設定します。ノードのプロビジョニングに使用するフレーバーでも、ローカルブートの機能を設定する必要があります。ローカルブートを有効にするには、ノードのデプロイに使用するイメージに **grub2** が含まれている必要があります。ローカルブートを以下のように設定します。

```
$ openstack baremetal node set NODE_UUID \
  --property capabilities="boot_option:local"
```

NODE_UUID は、ノードの一意識別子に置き換えます。もしくは、ノードの論理名を使用します。

7. プロビジョニングネットワーク上の NIC の MAC アドレスを使用してポートを作成することにより、Bare Metal サービスにノードのネットワークカードを通知します。

```
$ openstack baremetal port create --node NODE_UUID MAC_ADDRESS
```

NODE_UUID は、ノードの一意識別子に置き換えます。**MAC_ADDRESS** は、PXE ブートに使用する NIC の MAC アドレスに置き換えます。

8. 複数のディスクがある場合には、ルートデバイスのヒントを設定してください。これにより、デプロイメントに使用すべきディスクがデプロイ ramdisk に通知されます。

```
$ openstack baremetal node set NODE_UUID \
  --property root_device={"PROPERTY": "VALUE"}
```

以下の値に置き換えてください。

- **NODE_UUID** *は、ノードの一意識別子に置き換えます。もしくは、ノードの論理名を使用します。*
- *PROPERTY と VALUE は、デプロイメントに使用するディスクの情報に置き換えます (例:* ***root_device='{"size": 128}'****)。*
  以下のプロパティーがサポートされています。
- ***model*** *(文字列): デバイスの ID*
- ***vendor*** *(文字列): デバイスのベンダー*

- ***serial*** *(文字列): ディスクのシリアル番号*
- ***hctl*** *(文字列): SCSI のホスト、チャンネル、ターゲット、Lun*
- ***size*** *(整数): デバイスのサイズ (GB 単位)*
- ***wwn*** *(文字列): 一意のストレージ ID*
- ***wwn_with_extension*** *(文字列): ベンダー拡張子を追加した一意のストレージ ID*
- ***wwn_vendor_extension*** *(文字列): 一意のベンダーストレージ ID*
- ***rotational*** *(ブール値): ディスクを用いるデバイス (HDD) の場合は true、それ以外 (SSD) の場合は false。*
- ***name****: デバイス名 (例: /dev/sdb1)。これは、永続デバイス名が付いたデバイスのみに使用してください。*

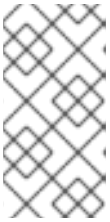
注記

複数のプロパティーを指定する場合には、デバイスはそれらの全プロパティーと一致する必要があります。

9. *ノードの設定を検証します。*

```
$ openstack baremetal node validate NODE_UUID
+------------+--------+--------------------------------------------+
| Interface  | Result | Reason                                     |
+------------+--------+--------------------------------------------+
| boot       | False  | Cannot validate image information for node  |
|            |        | a02178db-1550-4244-a2b7-d7035c743a9b       |
|            |        | because one or more parameters are missing  |
|            |        | from its instance_info. Missing are:       |
|            |        | ['ramdisk', 'kernel', 'image_source']      |
| console    | None   | not supported                              |
| deploy     | False  | Cannot validate image information for node  |
|            |        | a02178db-1550-4244-a2b7-d7035c743a9b       |
|            |        | because one or more parameters are missing  |
|            |        | from its instance_info. Missing are:       |
|            |        | ['ramdisk', 'kernel', 'image_source']      |
| inspect    | None   | not supported                              |
| management | True   |                                            |
| network    | True   |                                            |
| power      | True   |                                            |
| raid       | True   |                                            |
| storage    | True   |                                            |
+------------+--------+--------------------------------------------+
```

NODE_UUID *は、ノードの一意識別子に置き換えます。もしくは、ノードの論理名を使用します。上記のコマンドの出力には、各インターフェースが* ***True*** *または* ***None*** *のいずれかと報告されるはずです。* ***None*** *とマークされたインターフェースは、設定していないか、ドライバーがサポートしていないインターフェースです。*



注記

「ramdisk」、「kernel」、および「image_source」のパラメーターが指定されていないと、インターフェースの検証に失敗する場合があります。Compute サービスは、デプロイメントプロセスの最初に未指定のパラメーターを設定するので、この結果は問題ありません。

4.5. ホストアグリゲートを使用した物理マシンと仮想マシンのプロビジョニングの分離

OpenStack Compute は、ホストアグリゲートを使用してアベイラビリティーゾーンをパーティション分割し、特定の共有プロパティーが指定されたノードをグループ化します。インスタンスがプロビジョニングされると、Compute のスケジューラーがフレーバーのプロパティーをホストアグリゲートに割り当てられたプロパティーと比較して、インスタンスが正しいアグリゲート内の正しいホストに (物理マシン上または仮想マシンとして) プロビジョニングされたことを確認します。

以下の手順は、次の作業の方法を説明します。

- ***baremetal** プロパティーをフレーバーに追加して、**true** または **false** に設定する。*
- *一致する **baremetal** プロパティーを設定して、ベアメタルホスト用とコンピュートノード用のホストアグリゲートを別々に作成する。1 つのアグリゲートでグループ化されたノードは、このプロパティーを継承します。*

ホストアグリゲートの作成

1. *ベアメタル用のフレーバーで **baremetal** プロパティーを **true** に設定します。*

```
$ openstack flavor set baremetal --property baremetal=true
```

2. *仮想インスタンスに使用するフレーバーで **baremetal** プロパティーを **false** に設定します。*

```
$ openstack flavor set FLAVOR_NAME --property baremetal=false
```

3. ***baremetal-hosts** という名前のホストアグリゲートを作成します。*

```
$ openstack aggregate create --property baremetal=true baremetal-hosts
```

4. *各コントローラーノードを **baremetal-hosts** アグリゲートに追加します。*

```
$ openstack aggregate add host baremetal-hosts HOSTNAME
```

注記

***NovaIronic** サービスでコンポーザブルロールを作成していた場合には、このサービスがあるノードをすべて **baremetal-hosts** アグリゲートに追加します。デフォルトでは、**NovaIronic** サービスがあるのはコントローラーノードのみです。*

5. ***virtual-hosts** という名前のホストアグリゲートを作成します。*

```
$ openstack aggregate create --property baremetal=false virtual-hosts
```

6. *各コンピュートノードを **virtual-hosts** アグリゲートに追加します。*

```
$ openstack aggregate add host virtual-hosts HOSTNAME
```

7. *オーバークラウドのデプロイ時に以下の Compute フィルタースケジューラーを追加していなかった場合には、この時点で /etc/nova/nova.conf の **scheduler_default_filters** セクションの既存リストに追加します。*

```
AggregateInstanceExtraSpecsFilter
```

第5章 ベアメタルノードの管理

本章では、登録済みのベアメタルノードで物理マシンをプロビジョニングする方法について説明します。インスタンスは、コマンドラインまたは OpenStack Dashboard で起動することができます。

5.1. コマンドラインインターフェースを使用したインスタンスの起動

openstack コマンドラインインターフェースを使用してベアメタルインスタンスをデプロイします。

コマンドライン上でのインスタンスのデプロイ

1. Identity に管理ユーザーとしてアクセスするためのシェルを設定します。

```
$ source ~/overcloudrc
```

2. インスタンスをデプロイします。

```
$ openstack server create \
  --nic net-id=NETWORK_UUID \
  --flavor baremetal \
  --image IMAGE_UUID \
  INSTANCE_NAME
```

以下の値を置き換えてください。

- **NETWORK_UUID** は、Bare Metal サービスで使用するために作成したネットワークの一意識別子に置き換えます。
- **IMAGE_UUID** は、Image サービスにアップロードされたディスクイメージの一意識別子に置き換えます。
- **INSTANCE_NAME** は、ベアメタルインスタンスの名前に置き換えます。

3. インスタンスのステータスを確認します。

```
$ openstack server list --name INSTANCE_NAME
```

5.2. DASHBOARD を使用したインスタンスの起動

Dashboard のグラフィカルユーザーインターフェースを使用してベアメタルインスタンスをデプロイします。

Dashboard でのインスタンスのデプロイ

1. **http[s]://DASHBOARD_IP/dashboard** で Dashboard にログインします。
2. **プロジェクト > コンピュート > インスタンス** の順にクリックします。
3. **インスタンスの起動** をクリックします。
   - **詳細** タブで **インスタンス名** を指定して、**インスタンス数** に **1** を選択します。

- ***ソース** タブで **ブートソースを選択してください** のドロップダウンメニューから **イメージ** を選択し、続いて↑ (上向き矢印) の記号をクリックしてオペレーティングシステムのディスクイメージを選択します。選択したイメージは **割り当て済み** に移動します。*

- ***フレーバー** タブで **baremetal** を選択します。*

- ***ネットワーク** タブで、↑ (上向き矢印) および↓ (下向き矢印) ボタンを使用して必要なネットワークを **利用可能** から **割り当て済み** に移動します。ここでは、必ずBare Metal サービス用に作成した共有ネットワークを選択してください。*

4. ***インスタンスの起動** をクリックします。*

第6章 BARE METAL サービスのトラブルシューティング

以下の項には、Bare Metal サービスを有効にした環境における問題を診断するのに役立つ可能性のある情報と手順を記載します。

6.1. PXE ブートエラー

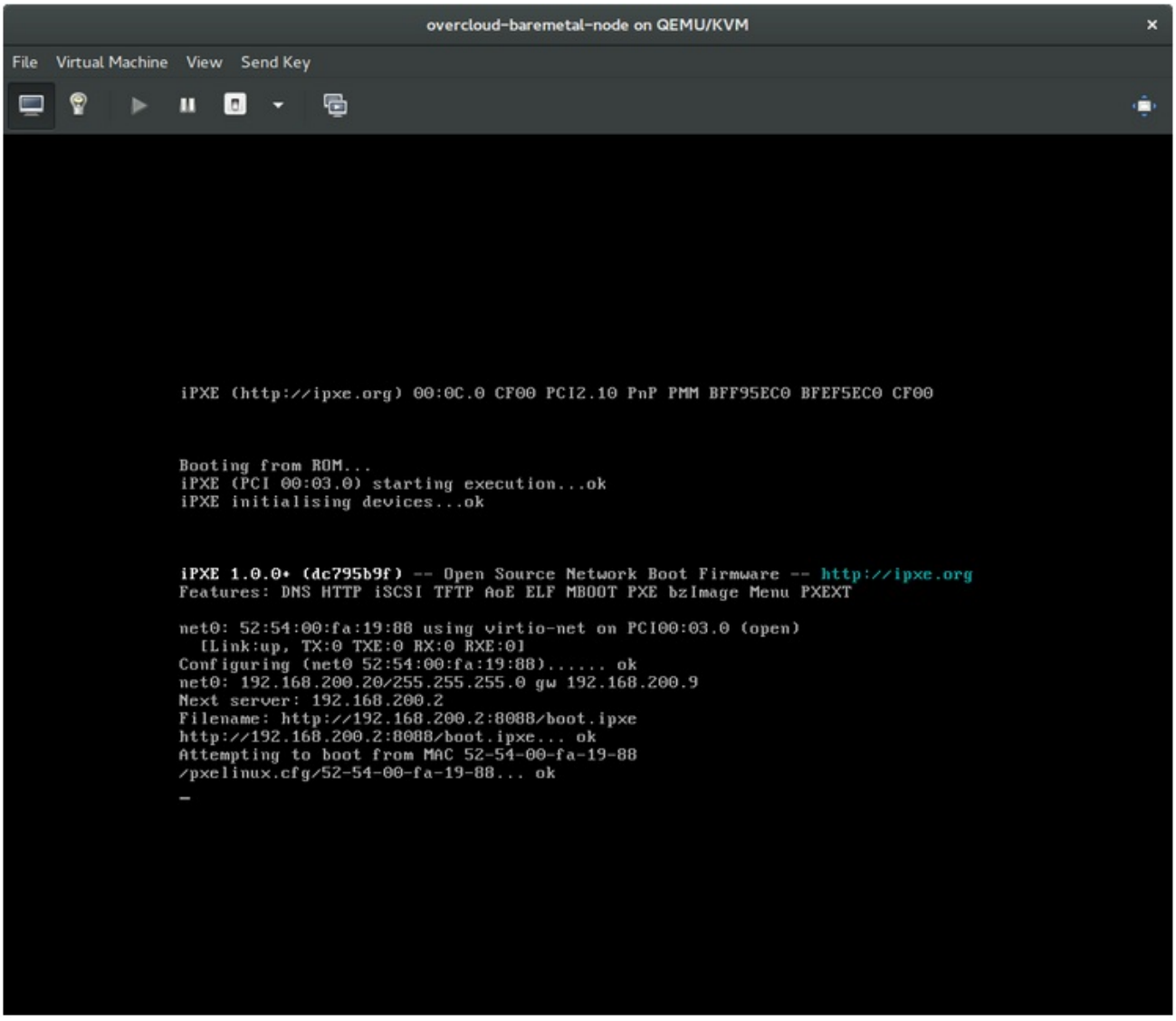
Permission Denied エラー

Bare Metal サービスノードのコンソールで「Permission Denied」エラーが表示された場合には、以下に示したように、適切な SELinux コンテキストを **/httpboot** と **/tftpboot** のディレクトリーに必ず適用してください。

```
# semanage fcontext -a -t httpd_sys_content_t "/httpboot(/.*)?"
# restorecon -r -v /httpboot
# semanage fcontext -a -t tftpdir_t "/tftpboot(/.*)?"
# restorecon -r -v /tftpboot
```

/pxelinux.cfg/XX-XX-XX-XX-XX-XX でのブートプロセスのフリーズ

以下の図に示したように、ノードのコンソールで、IP アドレスは取得されているがプロセスが停止しているように表示されている場合:



*これは、**ironic.conf** ファイルで誤った PXE ブートテンプレートを使用している可能性があることを示しています。*

```
$ grep ^pxe_config_template ironic.conf
pxe_config_template=$pybasedir/drivers/modules/ipxe_config.template
```

*デフォルトのテンプレートは **ipxe_config.template** です。*

6.2. ベアメタルノードの起動後のログインエラー

*ノードのコンソールのログインプロンプトで、設定手順中に設定した **root** パスワードを使用してログインを試みてもログインできない場合には、デプロイしたイメージでブートしていないことを意味します。**deploy-kernel**/**deploy-ramdisk** イメージにスタックしてしまって、システムが正しいイメージをまだ取得していない可能性があります。*

*この問題を修正するには、Compute または Bare Metal サービスノードの **/httpboot/pxelinux.cfg/MAC_ADDRESS** にある PXE ブートの設定ファイルをチェックして、このファイルにリストされている全 IP アドレスがベアメタルネットワークの IP アドレスに対応していることを確認してください。*

注記

Bare Metal サービスノードが認識している唯一のネットワークはベアメタルネットワークです。エンドポイントの 1 つがこのネットワーク上にない場合には、そのエンドポイントはブートプロセスの一環として Bare Metal サービスノードに到達することはできません。

たとえば、ファイルの kernel の行は以下のようになります。

```
kernel http://192.168.200.2:8088/5a6cdbe3-2c90-4a90-b3c6-85b449b30512/deploy_kernel selinux=0
disk=cciss/c0d0,sda,hda,vda iscsi_target_iqn=iqn.2008-10.org.openstack:5a6cdbe3-2c90-4a90-b3c6-
85b449b30512 deployment_id=5a6cdbe3-2c90-4a90-b3c6-85b449b30512
deployment_key=VWDYDVVEFCQJNOSTO9R67HKUXUGP77CK
ironic_api_url=http://192.168.200.2:6385 troubleshoot=0 text nofb nomodeset vga=normal
boot_option=netboot ip=${ip}:${next-server}:${gateway}:${netmask} BOOTIF=${mac}  ipa-api-
url=http://192.168.200.2:6385 ipa-driver-name=pxe_ssh boot_mode=bios initrd=deploy_ramdisk
coreos.configdrive=0 || goto deploy
```

上記の例の kernel 行の値	対応する情報
http://192.168.200.2:8088	**/etc/ironic/ironic.conf** ファイルのパラメーター **http_url**。この IP アドレスはベアメタルネットワーク上にある必要があります。
5a6cdbe3-2c90-4a90-b3c6-85b449b30512	**ironic node-list** 内のベアメタルノードの UUID
deploy_kernel	これは、**/httpboot/<NODE_UUID>/deploy_kernel** としてコピーされた Image サービス内のデプロイカーネルイメージです。

上記の例の kernel 行の値	対応する情報
http://192.168.200.2:6385	**/etc/ironic/ironic.conf** ファイルのパラメーター **api_url**。この IP アドレスはベアメタルネットワーク上にある必要があります。
pxe_ssh	このノードの Bare Metal サービスが使用している IPMI ドライバー
deploy_ramdisk	これは、**/httpboot/<NODE_UUID>/deploy_ramdisk** としてコピーされた Image サービス内のデプロイ ramdisk イメージです。

/httpboot/pxelinux.cfg/MAC_ADDRESS *と* ***ironic.conf*** *ファイルの間で値が一致していない場合:*

1. ***ironic.conf*** *ファイル内の値を更新します。*
2. *Bare Metal サービスを再起動します。*
3. *ベアメタルインスタンスを再デプロイします。*

6.3. BARE METAL サービスが正しいホスト名を取得しない

*Bare Metal サービスが正しいホスト名を取得しない場合は、****cloud-init*** *でエラーが発生していることを意味します。この問題を修正するには、ベアメタルのサブネットを OpenStack Networking サービス内のルーターに接続します。meta-data エージェントへの要求はこれで正しくルーティングされるようになるはずです。*

6.4. BARE METAL サービスのコマンド実行時に OPENSTACK IDENTITY サービスの認証情報が無効

*Identity サービスへの認証で問題がある場合には、****ironic.conf*** *ファイルの* ***identity_uri*** *パラメーターをチェックして、****keystone*** *AdminURL から* ***/v2.0*** *が削除されていることを確認してください。たとえば、****identity_uri*** *は* ***http://IP:PORT*** *に設定する必要があります。*

6.5. ハードウェアの登録

ハードウェア登録での問題は、ノードの登録情報が誤っていることが原因となっている可能性があります。プロパティー名と値が正しく入力されていることを確認してください。プロパティー名に誤りやタイプミスがあってもノードの情報には正常に追加されますが、そのプロパティー名は無視されます。

ノードの情報を更新します。以下の例では、登録するノードのメモリー使用量を 2 GB に更新します。

```
$ openstack baremetal node set --property memory_mb=2048 NODE_UUID
```

6.6. NO VALID HOST エラー

*Compute スケジューラーがインスタンスを起動するのに適切なベアメタルノードを見つけられない場合、****NoValidHost*** *エラーが /var/log/nova/nova-conductor.log に表示されるか、起動に失敗した直後に Dashboard に表示されます。通常これは、Compute が想定するリソースとベアメタルノードが提供するリソースが一致しないことが原因です。*

1. *利用可能なハイパーバイザーのリソースを確認します。*

```
$ openstack hypervisor stats show
```

このコマンドで返されるリソースは、Bare Metal が提供するリソースと一致する必要があります。

2. *Compute がベアメタルノードをハイパーバイザーとして認識していることを確認します。*

```
$ openstack hypervisor list
```

ノードは UUID で識別され、一覧に表示されるはずです。

3. *ベアメタルノードの詳細を確認します。*

```
$ openstack baremetal node list
$ openstack baremetal node show NODE_UUID
```

ノードの詳細が、Compute によって返された情報と一致することを確認します。

4. *選択したフレーバーがベアメタルノードで利用可能なリソースを超えていないことを確認します。*

```
$ openstack flavor show FLAVOR_NAME
```

5. ***openstack baremetal node list** の出力をチェックして、ベアメタルノードがメンテナンスモードに入っていないことを確認します。必要な場合には、メンテナンスモードを解除してください。*

```
$ openstack baremetal node maintenance unset NODE_UUID
```

6. ***openstack baremetal node list** の出力をチェックして、ベアメタルノードが **available** の状態であることを確認します。必要な場合には、ノードを **available** に切り替えます。*

```
$ openstack baremetal node provide NODE_UUID
```

付録A BARE METAL のドライバー

ベアメタルノードは、Bare Metal サービスで有効にしたドライバーの1 つを使用するように設定することができます。各ドライバーは、プロビジョニングメソッドと電源管理のタイプで構成されます。ドライバーによっては追加の設定が必要な場合があります。このセクションに記述された各ドライバーはプロビジョニングに PXE を使用します。ドライバーは電源管理タイプ別にリストされます。

ironic.yaml ファイルの **IronicEnabledDrivers** パラメーターを使用して、ドライバーを追加することができます。デフォルトでは、**pxe_ipmitool**、**pxe_drac**、および **pxe_ilo** が有効です。

サポートされているプラグインとドライバーの全一覧は、「Component, Plug-In, and Driver Support in Red Hat OpenStack Platform」のアーティクルを参照してください。

A.1. INTELLIGENT PLATFORM MANAGEMENT INTERFACE (IPMI)

IPMI は、電源管理やサーバー監視などの帯域外 (OOB) リモート管理機能を提供するインターフェースです。この電源管理タイプを使用するには、全 Bare Metal サービスノードで IPMI が共有ベアメタルネットワークに接続されている必要があります。**pxe_ipmitool** ドライバーを有効にし、ノードの **driver_info** に以下の情報を設定します。

- **ipmi_address**: IPMI NIC の IP アドレス
- **ipmi_username**: IPMI のユーザー名
- **ipmi_password**: IPMI のパスワード

A.2. DELL REMOTE ACCESS CONTROLLER (DRAC)

DRAC は、電源管理やサーバー監視などの帯域外 (OOB) リモート管理機能を提供するインターフェースです。この電源管理タイプを使用するには、全 Bare Metal サービスノードで DRAC が共有ベアメタルネットワークに接続されている必要があります。**pxe_drac** ドライバーを有効にし、ノードの **driver_info** に以下の情報を設定します。

- **drac_address**: DRAC NIC の IP アドレス
- **drac_username**: DRAC のユーザー名
- **drac_password**: DRAC のパスワード

A.3. INTEGRATED REMOTE MANAGEMENT CONTROLLER (IRMC)

富士通の iRMC は、電源管理やサーバー監視などの帯域外 (OOB) リモート管理機能を提供するインターフェースです。Bare Metal サービスノードでこの電源管理タイプを使用するには、このノードで iRMC インターフェースが共有ベアメタルネットワークに接続されている必要があります。**pxe_irmc** ドライバーを有効にし、ノードの **driver_info** に以下の情報を設定します。

- **irmc_address**: iRMC インターフェースの NIC の IP アドレス
- **irmc_username**: iRMC のユーザー名
- **irmc_password**: iRMC のパスワード

IPMI を使用してブートモードを設定する場合、または SCCI を使用してセンサーデータを取得する場合には、追加で以下のステップを完了する必要があります。

1. ironic.conf でセンサーメソッドを有効にします。

```
$ openstack-config --set /etc/ironic/ironic.conf \
  irmc sensor_method METHOD
```

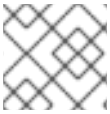
METHOD は ***scci*** または ***ipmitool*** に置き換えます。

2. SCCI を有効にした場合は、***python-scciclient*** パッケージをインストールします。

```
# yum install python-scciclient
```

3. Bare Metal Conductor サービスを再起動します。

```
# systemctl restart openstack-ironic-conductor.service
```



注記

iRMC ドライバーを使用するには、iRMC S4 以降が必要です。

A.4. INTEGRATED LIGHTS-OUT (ILO)

*Hewlett-Packard の iLO は、電源管理やサーバー監視などの帯域外 (OOB) リモート管理機能を提供するインターフェースです。この電源管理タイプを使用するには、全ベアメタルノードで iLO インターフェースが共有ベアメタルネットワークに接続されている必要があります。**pxe_ilo** ドライバーを有効にし、ノードの **driver_info** に以下の情報を設定します。*

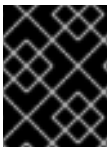
- ***ilo_address***: *iLO インターフェースの NIC の IP アドレス*
- ***ilo_username***: *iLO のユーザー名*
- ***ilo_password***: *iLO のパスワード*

python-proliantutils *パッケージもインストールして、Bare Metal Conductor サービスを再起動する必要があります。*

```
# yum install python-proliantutils
# systemctl restart openstack-ironic-conductor.service
```

A.5. SSH と VIRSH

Bare Metal サービスは、libvirt を実行するホストにアクセスして、仮想マシンをノードとして使用することができます。virsh はノードの電源管理機能を制御します。



重要

SSH ドライバーは、テストおよび評価の目的でのみ利用いただけます。Red Hat OpenStack Platform のエンタープライズ環境には推奨していません。

*この電源管理タイプを使用するには、Bare Metal サービスは仮想ノードを設定するホスト上の libvirt 環境に完全にアクセス可能なアカウントに SSH でアクセスできる必要があります。**pxe_ssh** ドライバーを有効にし、ノードの **driver_info** に以下の情報を設定します。*

- ***ssh_virt_type***: *このオプションは* ***virsh*** *に設定します。*
- ***ssh_address***: *virsh ホストの IP アドレス*
- ***ssh_username***: *SSH ユーザー名*
- ***ssh_key_contents***: *Bare Metal コンダクターノード上の SSH 秘密鍵の内容。対応する公開鍵が virsh ホストにコピーされている必要があります。*