Red Hat OpenStack Platform 15

オーバークラウドの IPv6 ネットワーク

IPv6 ネットワークを使用するようにオーバークラウドを設定する

Last Updated: 2022-09-15

Red Hat OpenStack Platform 15 オーバークラウドの IPv6 ネットワーク

IPv6 ネットワークを使用するようにオーバークラウドを設定する

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 director を使用して、IPv6 をエンドポイントに使用するオーバークラウドを作成する方法を説明します。また、director が IPv6 ベースのオーバークラウドをデプロイする方法や、これを実行する設定オプションについて説明します。

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。デフォルトでは、オーバークラウドは、インターネットプロトコルのバージョン 4 (IPv4) を使用してサービスのエンドポイントを設定します。ただし、オーバークラウドはインターネットプロトコルのバージョン 6 (IPv6) のエンドポイントもサポートします。これは、IPv6 のインフラストラクチャーをサポートする組織には便利です。本ガイドでは、オーバークラウドで IPv6 を使用するための情報と設定例を紹介します。

1.1. IPV6 ネットワークの定義

IPv6 は、最新バージョンのインターネットプロトコルです。Internet Engineering Task Force(IETF) は、現在の共通の IPv4 規格で IP アドレスが枯渇している問題に対応するため、IPv6 を開発しました。IPv6 には、以下のように IPv4 とさまざまな違いがあります。

大規模な IP アドレス範囲

IPv6 範囲は IPv4 範囲よりもはるかに大きくなります。

エンドツーエンドの接続性の改善

ネットワークアドレス変換に依存性が低くなるため、IP 範囲が大きいほど、エンドツーエンドの接続性が向上します。

ブロードキャストなし

IPv6 は従来の IP ブロードキャストに対応していません。代わりに、IPv6 はマルチキャストを使用して、階層的な方法で該当するホストにパケットを送信します。

ステートレスアドレス自動設定(SLAAC)

IPv6 には、IP アドレスを自動設定し、ネットワーク上の重複するアドレスを検出する機能があります。これにより、アドレスの割当に DHCP サーバーに依存する状況が抑えられます。

IPv6 は 128 ビット（16 ビットのグループを使用する 4 進数で表現）を使用してアドレスを定義します。一方、IPv4 は 32 ビット（グループの 8 ビットを使用した 10 進数の数字で表現）を使用します。たとえば、IPv4 アドレス(192.168.0.1)表現は以下のようになります。

bits	表現
11000000	192
10101000	168
00000000	0
00000001	1

IPv6 アドレス(2001:db8:88ec:9fb3::1)の場合、表現は以下のようになります。

bits	表現
0010 0000 0000 0001	2001
0000 1101 1011 1000	0db8

bits	表現
1000 1000 1110 1100	88ec
1001 1111 1011 0011	9fb3
0000 0000 0000 0000	0000
0000 0000 0000 0000	0000
0000 0000 0000 0000	0000
0000 0000 0000 0001	0001

各ビットグループの先頭の 0 なしで IPv6 アドレスを表現したり、各 IP アドレスの 0 のビットグループをすべて省略したりすることも可能である点に注意してください。この例では、0db8 ビットグループを db8 としてだけで表現し、0000 ビットグループ 3 個を省略でき、2001:0db8:88ec:9fb3:0000:0000:0000:0001 から 2001:db8:88ec:9fb3::1 に短縮できます。詳細は「RFC 5952: A Recommendation for IPv6 Address Text Representation」 を参照してください。

IPv6 でのサブネット化

IPv4 と同様に、IPv6 アドレスはビットマスクを使用してアドレスプレフィックスをネットワークとして定義します。たとえば、サンプル IP アドレス 2001:db8:88ec:9fb3::1/64 に /64 ビットマスクを追加すると、最初の 64 ビット(2001:db8:88ec:9fb3)をネットワークとして定義する接頭辞としてビットマスクが動作します。残りのビット (0000:0000:0000:0001) はホストを定義します。

IPv6 は、いくつかの特別なアドレスタイプを使用します。

ループバック

ループバックデバイスは、ホスト内の内部通信に IPv6 を使用します。このデバイスは常に ::1/128 になります。

リンクローカル

リンクローカルアドレスは、特定のネットワークセグメント内で有効な IP アドレスです。IPv6 では、各ネットワークデバイスにリンクローカルアドレスがあり、プレフィックス fe80::/10 を使用する必要があります。ただし、ほとんどの場合、これらのアドレスには fe80::/64 のプレフィックスが付けられます。

一意のローカル

一意のローカルアドレスは、ローカル通信用です。これらのアドレスは、fc00::/7 接頭辞を使用します。

マルチキャスト

ホストはマルチキャストアドレスを使用してマルチキャストグループに参加します。これらのアドレスは、ff00::/8 接頭辞を使用します。たとえば、FF02::1 はネットワーク上の全ノードのマルチキャストグループで、FF02::2 はすべてのルーターのマルチキャストグループです。

グローバルユニキャスト

これらのアドレスは、通常パブリック IP アドレス用に予約されます。これらのアドレスは 2000::/3 接頭辞を使用します。

1.2. RED HAT OPENSTACK PLATFORM での IPV6 の使用

Red Hat OpenStack Platform director は、OpenStack サービスを分離されたネットワークにマッピングするための手段を提供します。これらのネットワークには、以下が含まれます。

- 内部 API
- ストレージ
- ストレージ管理
- テナントネットワーク（Neutron VLAN モード）
- 外部

これらのネットワークトラフィック種別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。

Red Hat OpenStack Platform director には、これらのネットワークに IPv6 通信を使用する方法もあります。つまり、必要な OpenStack サービス、データベース、その他の関連サービスが IPv6 アドレスを使用して通信できます。これは、複数のコントローラーノードが関係する高可用性ソリューションを使用する環境にも適用されます。これにより、Red Hat OpenStack Platform を IPv6 インフラストラクチャーと統合するのに役立ちます。

以下の表を、Red Hat OpenStack Platform で IPv6 をサポートするネットワークに関するガイドとして使用します。

ネットワーク種別	デュアルスタック (IPv4/v6)	シングルスタック (IPv6)	シングルスタック (IPv4)	注記
内部 API		あり	○	
ストレージ		あり	あり	
ストレージ管理		あり	あり	
テナントネットワーク	あり	はい	あり	
テナントネットワークエンドポイント	あり	はい	あり	これはテナントネットワーク自体ではなく、テナントネットワークトンネルをホストするネットワークの IP アドレスを指します。 ネットワークエンドポイントの IPv6 がサポートするのは VXLAN および Geneve だけです。GRE (Generic Routing Encapsulation) はまだサポートされていません。
External: パブリック API (および Horizon)		あり	あり	

ネットワーク種別	デュアルスタック (IPv4/v6)	シングルスタック (IPv6)	シングルスタック (IPv4)	注記
External: Floating IP	あり	はい	あり	**デュアルスタックと単一スタック(IPv6)のみ**: グローバルユニキャストアドレス(GUA)接頭辞が割り当てられ、アドレスには外部ゲートウェイポート上の NAT は必要ありません。
プロバイダーネットワーク	あり	はい	あり	IPv6 サポートはテナントのオペレーティングシステムによって異なります。
プロビジョニング (PXE/DHCP)			あり	このネットワークのインターフェースは IPv4 のみになります。
IPMI またはその他の BMC			あり	RHOSP は、プロビジョニングネットワーク (IPv4) を通じてベースボード管理コントローラー(BMC)インターフェースと通信します。 BMC インターフェースがデュアルスタック IPv4 または IPv6 に対応している場合には、RHOSP に含まれていないツールは IPv6 を使用して BMC と通信できます。
オーバークラウドプロビジョニングネットワーク				オーバークラウドの ironic に使用するプロビジョニングネットワーク。
オーバークラウドクリーニングネットワーク				再利用の準備が整うまでマシンをクリーンアップするために使用される分離ネットワーク。

1.3. 設定要件

本ガイドは 、『director のインストールと使用方法』 の補足情報を提供します。これは、「 director のインストールと使用方法」で指定した要件と 同じ要件が本ガイドにも適用されることを意味します。必要に応じて、この要件を実装してください。

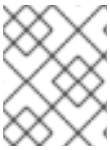
本ガイドでは、以下の要件も満たす必要があります。

- Red Hat OpenStack Platform director をインストールしたアンダークラウドホスト。『director のインストールと使用方法』を 参照してください。
- お使いのネットワークが IPv6 ネイティブ VLAN と IPv4 ネイティブ VLAN をサポートしている。いずれもデプロイメントで使用されます。

1.4. シナリオの定義

本ガイドのシナリオでは、IPv6 を使用する分離ネットワークでオーバークラウドを作成します。本書で

は、Heat テンプレートおよび環境ファイルを使用して設定したネットワーク分離で、オーバークラウドを作成することが目的です。このシナリオでは、設定の違いを示すために、これらの Heat テンプレートと環境ファイルに特定のバリアントも提供されます。



注記

このシナリオでは、アンダークラウドは引き続き PXE ブート、イントロスペクション、デプロイメント、およびその他のサービスに IPv4 の接続を使用します。

本ガイドは、『 director のインストールと使用方法』 の「オーバークラウドの管理」のシナリオに類似したシナリオを使用します。主な違いは、Ceph Storage ノードが省略される点です。

このシナリオ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。

重要

本ガイドでは、RFC 3849 で定義されているように文書用に 2001:DB8::/32 IPv6 接頭辞を使用します。これらのサンプルアドレスは、独自のネットワークの IPv6 アドレスに置き換えるようにしてください。

第2章 作成前のオーバークラウドの設定

本章では、**openstack overcloud deploy** コマンドを実行する前に必要となる設定について説明します。これには、プロビジョニング用のノードの準備、アンダークラウドでの IPv6 アドレスの設定、およびオーバークラウドの IPv6 パラメーターを定義するためのネットワーク環境ファイルの作成が含まれます。

2.1. STACK ユーザーの初期化

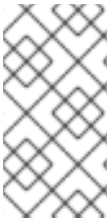
stack ユーザーとして director ホストにログインし、以下のコマンドを実行して director の設定を初期化します。

```
$ 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。

2.2. アンダークラウドの IPV6 アドレスの設定

アンダークラウドには、オーバークラウドのパブリック API（外部ネットワーク）へのアクセスが必要です。そのためには、アンダークラウドのホストに、外部ネットワークにアクセスするインターフェースで IPv6 アドレスが必要です。



注記

プロビジョニングネットワークには、引き続き全ノードで IPv4 接続が必要です。アンダークラウドおよびオーバークラウドノードは、PXE ブート、イントロスペクション、およびデプロイメントにこのネットワークを使用します。また、ノードはこのネットワークを使用して IPv4 経由で DNS および NTP サービスにアクセスします。

ネイティブ VLAN または専用インターフェース

アンダークラウドがネイティブ VLAN または外部ネットワークに接続されている専用インターフェースを使用する場合は、**ip** コマンドを使用して、インターフェースに IPv6 アドレスを追加します。以下の例では、専用のインターフェースは **eth0** です。

```
$ sudo ip link set dev eth0 up; sudo ip addr add 2001:db8::1/64 dev eth0
```

省略した VLAN インターフェース

外部ネットワークにアクセスするためのコントロールプレーンブリッジ (**br-ctlplane**) と同じインターフェースで、トランキングされた VLAN がアンダークラウドにより使用される場合には、新しい VLAN インターフェースを作成して、コントロールプレーンにアタッチし、IPv6 アドレスを VLAN に追加します。たとえば、このシナリオでは、外部ネットワークの VLAN ID に 100 を使用しています。

```
$ sudo ovs-vsctl add-port br-ctlplane vlan100 tag=100 -- set interface vlan100 type=internal
$ sudo ip l set dev vlan100 up; sudo ip addr add 2001:db8::1/64 dev vlan100
```

IPv6 アドレスの確認

ip コマンドを使用して、IPv6 アドレスが追加されたことを確認します。

```
$ ip addr
```

IPv6 アドレスは、選択したインターフェースに表示されます。

永続的な IPv6 アドレスの設定

上記に加えて、IPv6 アドレスを永続化する場合があります。この場合は、**/etc/sysconfig/network-scripts/** で適切なインターフェースファイルを変更または作成します（この例では、**ifcfg-eth0** または **ifcfg-vlan100**）。以下の行を追加します。

```
IPV6INIT=yes
IPV6ADDR=2001:db8::1/64
```

詳細は、Red Hat カスタマーポータルの 「How do I configure a network interface for IPv6?」 を参照してください。

2.3. 環境の設定

本項では、『 **director のインストールと** 使用方法 』の「CLI ツールを使用した基本的なオーバークラウド要件の設定 」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。

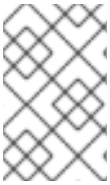
2.3.1. ノードの登録

ノード定義のテンプレート (**instackenv.json**) は JSON ファイル形式で、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"
  }
  {
    "mac":[
      "ff:ff:ff:ff:ff:ff"
    ],
    "cpu":"4",
    "memory":"6144",
    "disk":"40",
    "arch":"x86_64",
    "pm_type":"pxe_ipmitool",
    "pm_user":"admin",
    "pm_password":"p@55w0rd!",
    "pm_addr":"192.0.2.209"
  }
  {
    "mac":[
      "gg:gg:gg:gg:gg:gg"
    ],
    "cpu":"4",
    "memory":"6144",
    "disk":"40",
    "arch":"x86_64",
    "pm_type":"pxe_ipmitool",
    "pm_user":"admin",
```

```
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.210"
        }
    ]
}
```



注記

プロビジョニングネットワークは IPv4アドレスを使用します。IPMI アドレスも IPv4 アドレスでなければなりません。また、プロビジョニングネットワークを介してルーティングに直接接続または到達可能である必要があります。

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**)、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。

2.3.2. 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。

2.3.3. 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、次にそのフレーバーが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
```

各ノードの **properties/capabilities** パラメーターに profile オプションを追加して、ノードを特定のプロファイルに手動でタグ付けします。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 484587b2-b3b3-40d5-925b-a26a2fa3036f add
properties/capabilities='profile:compute,boot_option:local'
$ ironic node-update d010460b-38f2-4800-9cc4-d69f0d067efe add
properties/capabilities='profile:compute,boot_option:local'
$ ironic node-update d930e613-3e14-44b9-8240-4f3559801ea6 add
properties/capabilities='profile:compute,boot_option:local'
```

profile:compute と **profile:control** オプションを追加して、このノードがそれぞれのプロファイルにタグ付けされます。

注記

手動でのタグを付ける代わりに、自動プロファイルタグ付け機能を使用し、ベンチマークデータに基づいて、多数のノードに自動でタグ付けします。

2.4. 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、サービスを分離して、特定のネットワークトラフィックを使用し、IPv6 オプションを使用したオーバークラウドの設定が含まれます。

2.4.1. コンポーザブルネットワークの詳細設定

1. デフォルトの **network_data** ファイルのコピーします。

```
$ cp /usr/share/openstack-tripleo-heat-templates/network_data.yaml /home/stack/.
```

2. **network_data.yaml** ファイルのローカルコピーを編集し、IPv6 ネットワーク要件に応じてパラメーターを変更します。たとえば、外部ネットワークには以下のデフォルトネットワーク情報が含まれます。

```
- name: External
  vip: true
  name_lower: external
  vlan: 10
  ipv6: true
  ipv6_subnet: '2001:db8:fd00:1000::/64'
  ipv6_allocation_pools: [{'start': '2001:db8:fd00:1000::10', 'end':
'2001:db8:fd00:1000:ffff:ffff:ffff:fffe'}]
  gateway_ipv6: '2001:db8:fd00:1000::1'
```

- **name** は、唯一の必須の値です。ただし、**name_lower** を使用して名前を正規化し、読みやすくすることができます。たとえば、**InternalApi** を **internal_api** に変更します。
- **vip: true** は、新規ネットワークのデフォルトを設定する残りのパラメーターを使用して、新規ネットワーク上に仮想 IP アドレス(VIP)を作成します。
- **ipv6** は、IPv6 を有効にするかどうかを定義します。

- **ipv6_subnet** および **ipv6_allocation_pools** と**gateway_ip6** は、ネットワークのデフォルトIPv6 サブネットおよび IP 範囲を設定します。

-n オプションを使用して、カスタム **network_data** ファイルをデプロイメントに含めます。**-n** オプションを設定しないと、デプロイメントコマンドはデフォルトのネットワーク情報を使用します。

2.4.2. ネットワーク分離

デフォルトでは、オーバークラウドはサービスをプロビジョニングネットワークに割り当てます。ただし、Red Hat OpenStack Platform director は、オーバークラウドのネットワークトラフィックを分離したネットワークに分割することができます。これらのネットワークは、デフォルトで **network_data.yaml** という名前のデプロイメントコマンドラインに追加するファイルで定義されます。

IPv6 アドレスを使用してサービスがネットワークをリッスンしている場合は、パラメーターのデフォルト値で、そのサービスが IPv6 ネットワーク上で実行されていることを指定する必要があります。各サービスが実行されるネットワークは、**network/service_net_map.yaml** のファイルで定義され、**ServiceNetMap** エントリーのパラメーターのデフォルトを個別に宣言することで上書きできます。以下のサービスでは、パラメーターのデフォルトを環境ファイルで設定する必要があります。

```
parameter_defaults:
  # Enable IPv6 for Ceph.
  CephIPv6: True
  # Enable IPv6 for Corosync. This is required when Corosync is using an IPv6 IP in the cluster.
  CorosyncIPv6: True
  # Enable IPv6 for MongoDB. This is required when MongoDB is using an IPv6 IP.
  MongoDbIPv6: True
  # Enable various IPv6 features in Nova.
  NovaIPv6: True
  # Enable IPv6 environment for RabbitMQ.
  RabbitIPv6: True
  # Enable IPv6 environment for Memcached.
  MemcachedIPv6: True
  # Enable IPv6 environment for MySQL.
  MysqlIPv6: True
  # Enable IPv6 environment for Manila
  ManilaIPv6: True
  # Enable IPv6 environment for Redis.
  RedisIPv6: True
```

director のコア Heat テンプレートの **environments/network-isolation.j2.yaml** ファイルは Jinja2 形式のファイルで、コンポーザブルネットワークファイル内の各 IPv6 ネットワークのポートおよび仮想 IP をすべて定義します。レンダリングすると、すべてのリソースレジストリーと共に **network-isolation.yaml** ファイルが同じ場所に生成されます。

2.4.3. インターフェースの設定

オーバークラウドには、ネットワークインターフェーステンプレートのセットが必要です。director には、**network_data** ファイルをベースにレンダリングされる Jinja2 ベースの Heat テンプレートセットが含まれています。

NIC ディレクトリー	説明	環境ファイル

NIC ディレクトリー	説明	環境ファイル
single-nic-vlans	単一の NIC (**nic1**) がコントロールプレーンネットワークにアタッチされ、VLAN 経由でデフォルトの Open vSwitch ブリッジにアタッチされる。	**environments/net-single-nic-with-vlans-v6.j2.yaml**
single-nic-linux-bridge-vlans	単一の NIC (**nic1**) がコントロールプレーンネットワークにアタッチされ、VLAN 経由でデフォルトの Linux ブリッジにアタッチされる。	**environments/net-single-nic-linux-bridge-with-vlans-v6.yaml**
bond-with-vlans	コントロールプレーンネットワークが **nic1** にアタッチされる。ボンディング構成の NIC (**nic2** および **nic3**) が VLAN 経由でデフォルトの Open vSwitch ブリッジにアタッチされる。	**environments/net-bond-with-vlans-v6.yaml**
multiple-nics	コントロールプレーンネットワークが **nic1** にアタッチされる。それ以降の NIC は **network_data** ファイルで定義されるネットワークに割り当てられる。デフォルトでは、Storage が **nic2** に、Storage Management が **nic3** に、Internal API が **nic4** に、Tenant が **br-tenant** ブリッジ上の **nic5** に、External がデフォルトの Open vSwitch ブリッジ上の **nic6** に割り当てられる。	**environments/net-multiple-nics-v6.yaml**

この例では、**sinle-nic-vlans** テンプレートコレクションを使用します。

2.4.4. IPv6 分離ネットワークの設定

デフォルトの heat テンプレートコレクションには、デフォルトのネットワーク設定用に Jinja2 ベースの環境ファイルが含まれます。このファイルは **environments/network-environment.j2.yaml** です。**network_data** ファイルでレンダリングすると、**network-environment.yaml** という標準の YAML ファイルが作成されます。このファイルの一部にはオーバーライドが必要になる場合があるので、独自のカスタム **network-environment.yaml** ファイルを作成する必要があります。このシナリオでは、カスタム環境ファイル (**/home/stack/network-environment.yaml**) を以下の内容で作成します。

```
parameter_defaults:
  DnsServers: ["8.8.8.8","8.8.4.4"]
  ControlPlaneDefaultRoute: 192.0.2.1
  ControlPlaneSubnetCidr: "24"
  EC2MetadataIp: 192.0.2.1
```

parameter_defaults セクションには、IPv4 の特定のサービスのカスタマイズが含まれます。

2.5. オーバークラウドの全設定

これにより、IPv6 ベースのオーバークラウドの設定に必要なステップが完了します。次の章では、**openstack overcloud deploy** コマンドを使用して、本章の設定を使用してオーバークラウドを作成します。

第3章 オーバークラウドの作成

IPv6 ネットワークを使用するオーバークラウドを作成するには、**openstack overcloud deploy** コマンドに追加の引数が必要です。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 /usr/share/openstack-tripleo-heat-templates/environments/network-environment.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/net-single-nic-with-vlans.yaml \
  -e /home/stack/templates/network-environment.yaml \
  --ntp-server pool.ntp.org \
  [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** - オーバークラウドデプロイメントに別の環境ファイルを追加します。ここでは、IPv6 向けの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 /usr/share/openstack-tripleo-heat-templates/environments/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、IPv6 向けの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 /usr/share/openstack-tripleo-heat-templates/environments/net-single-nic-with-vlans.yaml**: オーバークラウドのデプロイメントに別の環境ファイルを追加します。ここでは、IPv6 向けの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 /home/stack/templates/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。
- **--ntp-server pool.ntp.org**: NTP サーバーを設定します。

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、以下を実行します。

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

3.1. 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
```

第4章 作成後のオーバークラウドの設定

作成プロセスでは、IPv6 ネットワークを使用するオーバークラウドが完全に機能します。ただし、オーバークラウドには、作成後の設定が必要です。

4.1. オーバークラウドのテナントネットワークの作成

オーバークラウドには、インスタンスに IPv6 ベースのテナントネットワークが必要です。source コマンドで **overcloudrc** ファイルを読み込み、**neutron** に初期テナントネットワークを作成します。以下に例を示します。

```
$ source ~/overcloudrc
$ neutron net-create default --provider:physical_network datacentre --provider:network_type vlan --provider:segmentation_id 101
$ neutron subnet-create default 2001:db8:fd00:6000::/64 --ipv6-ra-mode slaac --ipv6-address-mode slaac --ip-version 6 --name default
```

上記のステップにより、**default** という名前の基本的な **neutron** ネットワークが作成されます。neutron net-list により、作成したネットワークを確認します。

```
$ neutron net-list
```

4.2. オーバークラウドのパブリックネットワークの作成

このシナリオでは、外部ネットワークを使用するようにノードインターフェースを設定しています。ただし、ネットワークアクセスができるように、オーバークラウド上にこのネットワークを作成する必要があります。

```
$ neutron net-create public --router:external --provider:physical_network datacentre --provider:network_type vlan --provider:segmentation_id 100
$ neutron subnet-create public 2001:db8:0:2::/64 --ip-version 6 --gateway 2001:db8::1 --allocation-pool start=2001:db8:0:2::2,end=2001:db8:0:2::ffff --ip-version 6 --ipv6_address_mode=slaac --ipv6_ra_mode=slaac
```

このコマンドは、**public** と呼ばれるネットワークを作成します。このネットワークは、インスタンスに 65000 個以上 IPv6 アドレスが含まれる割当プールを提供します。

インスタンストラフィックを外部ネットワークにルーティングするためのルーターを作成します。

```
neutron router-create public-router
neutron router-gateway-set public-router public
```

第5章 まとめ

これで IPv6 ベースのオーバークラウドの作成と設定が完了しました。オーバークラウドの作成後の全般的な機能については、『 director のインストールと使用方法』 を参照してください。