Red Hat OpenStack Platform 11

手動インストール手順

Red Hat OpenStack Platform の手動インストールの手順

Last Updated: 2017-11-23

Red Hat OpenStack Platform 11 手動インストール手順

Red Hat OpenStack Platform の手動インストールの手順

OpenStack Documentation Team
Red Hat Customer Content Services
rhos-docs@redhat.com

法律上の通知

概要

本ガイドは、Red Hat OpenStack Platform のコンポーネントのインストールと設定方法についての参考情報を提供し、デプロイメントプロセスの実習方法を記載しています。 現在 Red Hat では、本リリースのガイドに記載の情報および手順のレビューを行なっています。 本書は、 で提供している Red Hat OpenStack Platform 10 のドキュメントをベースにしています。 Red Hat OpenStack Platform 11 でサポートが必要な場合には、Red Hat サポートまでお問い合わせください。

目次

第1章 はじめに **5**
1.1. 必要なチャンネルのサブスクライブ 5
1.2. インストールの前提条件のチェックリスト 7

第2章 前提条件 **13**
2.1. ファイアウォールの設定 13
2.2. データベースサーバーのインストール 15
2.3. メッセージブローカーのインストール 17
2.4. ネットワークタイムプロトコルサーバー (NTP) 21
2.5. OPENSTACK コマンドラインクライアントのインストール 22

第3章 IDENTITY サービスのインストール **23**
3.1. IDENTITY サービスのパッケージのインストール 23
3.2. アイデンティティーデータベースの作成 23
3.3. IDENTITY サービスの設定 24
3.4. IDENTITY サービスの起動 28
3.5. 管理者アカウントおよび IDENTITY サービスエンドポイントの作成 28
3.6. 一般ユーザーアカウントの作成 30
3.7. サービステナントの作成 31
3.8. IDENTITY サービスのインストールの検証 32

第4章 OBJECT サービスのインストール **35**
4.1. OBJECT STORAGE サービスの要件 35
4.2. RSYNCD の設定 35
4.3. OBJECT STORAGE サービスのパッケージのインストール 37
4.4. OBJECT STORAGE サービスの設定 38
4.5. OBJECT STORAGE サービスのインストールの検証 45

第5章 IMAGE サービスのインストール **47**
5.1. IMAGE サービスの要件 47
5.2. IMAGE サービスのパッケージのインストール 47
5.3. IMAGE サービスのデータベースの作成 47
5.4. IMAGE サービスの設定 48
5.5. イメージの API およびレジストリーサービスの起動 57
5.6. IMAGE サービスインストールの検証 57

第6章 BLOCK STORAGE サービスのインストール **60**
6.1. BLOCK STORAGE サービスのパッケージのインストール 60
6.2. BLOCK STORAGE サービスのデータベースの作成 60
6.3. BLOCK STORAGE サービスの設定 61
6.4. ボリュームサービスの設定 66
6.5. BLOCK STORAGE サービスの起動 68
6.6. BLOCK STORAGE サービスのインストールの検証 69

第7章 OPENSTACK NETWORKING のインストール **71**
7.1. OPENSTACK NETWORKING パッケージのインストール 71
7.2. OPENSTACK NETWORKING の設定 71
7.3. DHCP エージェントの設定 81
7.4. プラグインエージェントの設定 83
7.5. 外部ネットワークの作成 84
7.6. L3 エージェントの設定 86
7.7. OPENSTACK NETWORKING をインストールしたシステムの検証 89

第8章 COMPUTE サービスのインストール **91**
8.1. COMPUTE の VNC プロキシーのインストール 91
8.2. コンピュートノードのインストール 94

第9章 ORCHESTRATIONサービスのインストール **108**
9.1. ORCHESTRATION サービスのパッケージのインストール 108
9.2. ORCHESTRATION サービスの設定 108
9.3. ORCHESTRATION サービスの起動 115
9.4. ORCHESTRATION テンプレートを使用したスタックのデプロイ 116
9.5. TELEMETRY および ORCHESTRATION サービスの統合 117

第10章 DASHBOARD のインストール **118**
10.1. DASHBOARD サービスの要件 118
10.2. DASHBOARD パッケージのインストール 118
10.3. APACHE WEB サービスの起動 119
10.4. DASHBOARD の設定 119
10.5. DASHBOARD のインストールの検証 124

第11章 DATA PROCESSING サービスのインストール **126**
11.1. DATA PROCESSING サービスのパッケージのインストール 126
11.2. DATA PROCESSING サービスの設定 126
11.3. DATA PROCESSING サービスの設定と起動 130

第12章 TELEMETRY サービスのインストール **131**
12.1. TELEMETRY サービスのデプロイメントの概要 131
12.2. TELEMETRY サービスのパッケージのインストール 131
12.3. MONGODB バックエンドの設定および TELEMETRY データベースの作成 132
12.4. TELEMETRY サービスのデータベース接続の設定 133
12.5. TELEMETRY アイデンティティーレコードの作成 134
12.6. TELEMETRY サービスの認証の設定 136
12.7. TELEMETRY サービスのトラフィックを許可するためのファイアウォール設定 137
12.8. TELEMETRY サービスのための RABBITMQ メッセージブローカーの設定 137
12.9. コンピュートノードの設定 138
12.10. 監視対象サービスの設定 139
12.11. TELEMETRY の API およびエージェントの起動 141

第13章 TELEMETRY ALARMING サービスのインストール **142**
13.1. TELEMETRY ALARMING サービスのパッケージのインストール 142
13.2. TELEMETRY ALARMING サービスデータベースの作成 142
13.3. TELEMETRY ALARMING サービスの設定 143
13.4. TELEMETRY ALARMING サービスの起動 147

第14章 TIME-SERIES-DATABASE-AS-A-SERVICE のインストール **148**
14.1. TIME-SERIES-DATABASE-AS-A-SERVICE パッケージのインストール 148
14.2. TIME-SERIES-DATABASE-AS-A-SERVICE の初期化 149
14.3. TIME-SERIES-DATABASE-AS-A-SERVICE の設定 149
14.4. TIME-SERIES-DATABASE-AS-A-SERVICE データベースの作成 150
14.5. TELEMETRY サービスのバックエンドとしての TIME-SERIES-DATABASE-AS-A-SERVICE の設定 151

第15章 SHARED FILE SYSTEM サービスのインストール **152**
15.1. SHARED FILE SYSTEM サービスのバックエンドの要件 152
15.2. SHARED FILE SYSTEM サービスのパッケージのインストール 152
15.3. SHARED FILE SYSTEM サービス用のアイデンティティーレコードの作成 152
15.4. 基本的な SHARED FILE SYSTEM サービスの設定 153
15.5. SHARED FILE SYSTEM サービスのデータベースの作成 155

15.6. SHARED FILE SYSTEM サービスのバックエンドの定義 156
15.7. SHARED FILE SYSTEM サービスの起動 156
15.8. 定義済みのバックエンドに対する共有種別の作成 157

第16章 DATABASE-AS-A-SERVICE のインストール (テクノロジープレビュー) 158
16.1. DATABASE-AS-A-SERVICE の要件 158
16.2. DATABASE-AS-A-SERVICE パッケージのインストール 159
16.3. DATABASE-AS-A-SERVICE の設定 159

第17章 OPENSTACK INTEGRATION TEST SUITE のインストール 167

第18章 OPENSTACK BENCHMARKING サービスのインストール 168
18.1. OPENSTACK BENCHMARKING サービスパッケージのインストール 168
18.2. OPENSTACK BENCHMARKING サービスの設定 169
18.3. BENCHMARKING サービスのテナントネットワークの拡張 170
18.4. OPENSTACK BENCHMARKING サービスの検証 173

第1章 はじめに

本ガイドは、Red Hat OpenStack Platform 環境のコンポーネントのインストール/設定方法に関するリファレンスを提供します。インストールと設定の情報は、以下のコンポーネントごとに分類されています。

- MariaDB データベースサーバー
- RabbitMQ メッセージブローカー
- Identity サービス
- Object Storage サービス
- Image サービス
- Block Storage サービス
- OpenStack Networking
- Compute サービス
- Orchestration サービス
- Dashboard
- Data Processing サービス
- Telemetry サービス
- Time-Series-as-a-Service
- File Share サービス (テクノロジープレビュー)
- Database-as-a-Service (テクノロジープレビュー)

注記

OpenStack のコンポーネントとそのインターフェースに関する概要は、『アーキテクチャーガイド』 (https://access.redhat.com/documentation/ja/red-hat-openstack-platform/) を参照してください。

本ガイドには、すべてのコンポーネントに共通するデータベースの設定やファイアウォールの設定などのタスクや、各コンポーネントの設定に固有のタスクが含まれます。

1.1. 必要なチャンネルのサブスクライブ

Red Hat OpenStack Platform をインストールするには、OpenStack 環境にある全システムを Red Hat サブスクリプションマネージャーで登録して、必要なチャンネルをサブスクライブします。

手順1.1 必要なチャンネルのサブスクライブ

1. コンテンツ配信ネットワークにシステムを登録します。プロンプトが表示されたら、カスタマーポータルのユーザー名とパスワードを入力します。

```
# subscription-manager register
```

2. 自分が使用することのできる Red Hat OpenStack Platform のサブスクリプションについての詳しい情報を確認するには、以下のコマンドを実行します。

```
# subscription-manager list --available --all --
matches="*OpenStack*"
```

このコマンドでは、以下のような出力が表示されるはずです。

```
+-------------------------------------------+
    Available Subscriptions
+-------------------------------------------+
Subscription Name:   Red Hat Enterprise Linux OpenStack Platform,
Standard (2-sockets)
Provides:            Red Hat Beta
...
                     Red Hat OpenStack
...
SKU:                 ABC1234
Contract:            12345678
Pool ID:             0123456789abcdef0123456789abcdef
Provides Management: No
Available:           Unlimited
Suggested:           1
Service Level:       Standard
Service Type:        L1-L3
Subscription Type:   Stackable
Ends:                12/31/2099
System Type:         Virtual
```

3. 上記のコマンドで表示された **Pool ID** を使用して、Red Hat OpenStack Platform のエンタイトルメントをアタッチします。

```
# subscription-manager attach --pool=Pool ID
```

4. 関係のないチャンネルは無効にして、必要なチャンネルを有効にします。

```
# subscription-manager repos --disable=* \
--enable=rhel-7-server-rpms \
--enable=rhel-7-server-openstack-11-devtools-rpms \
--enable=rhel-7-server-openstack-11-rpms \
--enable=rhel-7-server-rh-common-rpms \
--enable=rhel-7-server-extras-rpms
```

5. **yum update** コマンドを実行してからリブートし、カーネルを含む最新のパッケージが確実にインストールされて実行されるようにします。

```
# yum update
# reboot
```

Red Hat OpenStack Platform パッケージを受信するための設定が正常に完了しました。リポジトリーの設定は、**yum repolist** コマンドを使用して随時確認することができます。

1.2. インストールの前提条件のチェックリスト

以下の表には、Red Hat OpenStack Platform 環境を正常にインストールするための前提条件をまとめています。チェックリストの項目は、インストールを開始する前に認識または確認しておく必要のある最小条件です。

「値/確認済み」の列は、適切な値を記入したり、確認済みの項目に「チェック」を付けたりするのに使用することができます。

注記

Red Hat OpenStack Platform の初期インストール後に単独のコンポーネントをインストールする場合には、次のパーミッションがあることを確認してください。

- ホストマシンへの `root` アクセス (コンポーネントをインストールしたり、ファイアウォールの更新などのその他の管理者タスクを実行したりするため)
- Identity サービスへの管理者アクセス
- データベースへの管理者アクセス (データベースおよびユーザーの両方を追加する機能)

表1.1 OpenStack インストール: 一般

項目	説明	値/確認済み
ハードウェア要件	ハードウェア要件を確認する必要があります。	はい | いいえ
オペレーティングシステム	Red Hat Enterprise Linux 7.3 Server	はい | いいえ
Red Hat サブスクリプション	お使いのシステムで以下の更新を取得する資格を提供するサブスクリプションが必要です。 • コンテンツ配信ネットワークまたは Red Hat Network Satellite サーバーなど同等のソースからのパッケージの更新 • Red Hat Enterprise Linux 7.3 Server と Red Hat OpenStack Platform の両方のソフトウェア更新	はい | いいえ
全インストール先マシンへの管理者アクセス	本ガイドに記載する手順はほぼすべて、root ユーザーとして実行しなければならないため、root アクセスが必要です。	はい | いいえ
Red Hat サブスクリプションのユーザー名とパスワード	Red Hat サブスクリプションのユーザー名とパスワードが必要です。	• 名前: • パスワード:

項目	説明	値/確認済み
マシンのアドレス	OpenStack のコンポーネントおよび補足のソフトウェアをインストールする先のサーバーの IP アドレスまたはホスト名を知っておく必要があります。	以下のサービスのホストアドレスを指定します。 • Identity サービス • OpenStack Networking • Block Storage サービス • Compute サービス • Image サービス • Object Storage サービス • Dashboard サービス • データベースサーバー

表1.2 OpenStack Identity サービス

項目	説明	値
ホストアクセス	Identity サービスをホストするシステムは以下のコンポーネントへのアクセス権が必要です。 • コンテンツ配信サービスまたは同等のサービス • OpenStack の全ホストがアドレス指定可能なネットワークインターフェース • データベースサーバーへのネットワークアクセス • LDAP を使用する場合には、そのディレクトリーサーバーへのネットワークアクセス	システムが以下のコンポーネントへアクセスできるかどうかを確認します。 • はい | いいえ • はい | いいえ • はい | いいえ • はい | いいえ
SSL 証明書	外部の SSL 証明書を使用する場合には、データベースと証明書の場所およびそれらへのアクセスが必要です。	はい | いいえ
LDAP 情報	LDAP を使用する場合には、新規ディレクトリーサーバーのスキーマを設定するために管理者アクセスが必要です。	はい | いいえ
接続	Identity サービスをホストするシステムは、他の全 OpenStack サービスに接続されている必要があります。	はい | いいえ

表1.3 OpenStack Object Storage サービス

項目	説明	値
ファイルシステム	Red Hat は現在、オブジェクトストレージ用に **XFS** および **`ext4`** のファイルシステムをサポートしています。これらのいずれかのファイルシステムが利用可能である必要があります。	• XFS • ext4
マウントポイント	**`/srv/node`** マウントポイントが使用可能である必要があります。	はい | いいえ
接続	Object Storage サービスをホストするシステムが Identity サービスに接続されている必要があります。	はい | いいえ

表1.4 OpenStack Image サービス

項目	説明	値
バックエンド	Image サービスは多数のストレージバックエンドをサポートします。次のオプションのいずれかを選択する必要があります。 • ファイル (ローカルディレクトリー) • Object Storage サービス	ストレージ種別:
接続	Image サービスをホストするサーバーは、Identity サービス、Dashboard サービス、および Compute サービスに接続されている必要があります。また、このサーバーは Object Storage をバックエンドとして使用する場合には、Object Storage サービスにアクセスできる必要があります。	はい | いいえ

表1.5 OpenStack Block Storage サービス

項目	説明	値

項目	説明	値
バックエンド	Block Storage サービスは、多数のストレージバックエンドをサポートします。以下のいずれかに決定する必要があります。 • Red Hat Ceph • LVM/iSCSI • ThinLVM • NFS • NetApp • Dell EqualLogic • Dell Storage Center	ストレージ種別:
接続	Block Storage サービスをホストするサーバーは Identity サービス、Dashboard サービス、および Compute サービスに接続されている必要があります。	はい | いいえ

表1.6 OpenStack Networking

項目	説明	値

項目	説明	値
プラグインエージェント	標準の OpenStack Networking コンポーネントに加えて、さまざまなネットワークメカニズムを実装するプラグインエージェントの幅広い選択肢が利用可能です。 以下の中から、ネットワークに適用する項目を決定してインストールする必要があります。	該当するプラグインに丸印を付けてください。 • Open vSwitch • Cisco UCS/Nexus • Linux Bridge • VMware NSX ネットワーク仮想化プラットフォーム • Ryu OpenFlow Controller • NEC OpenFlow • Big Switch Controller Plugin • Cloudbase Hyper-V • MidoNet • Brocade Neutron Plugin • PLUMgrid
接続	OpenStack Networking をホストするサーバーは Identity サービス、Dashboard サービス、および Compute サービスに接続されている必要があります。	はい | いいえ

表1.7 OpenStack Compute サービス

項目	説明	値
ハードウェア仮想化サポート	Compute サービスには、ハードウェア仮想化サポートが必要です。	はい | いいえ
VNC クライアント	Compute サービスは、Web ブラウザーを介したインスタンスへの Virtual Network Computing (VNC) コンソールアクセスをサポートしています。このサポートをユーザーに提供するかどうかを決める必要があります。	はい | いいえ

項目	説明	値
リソース: CPU とメモリー	OpenStack は、コンピュートノード上における CPU およびメモリーリソースのオーバーコミットをサポートしています。 • デフォルトの CPU オーバーコミット率 16 とは、物理コア 1 つにつき 最大 16 の仮想コアをノードに割り当てることができるという意味です。 • デフォルトのオーバーコミット率 1.5 とは、インスタンスのメモリー使用量合計が、物理メモリーの空き容量の 1.5 倍未満の場合には、インスタンスを物理ノードに割り当てることができるという意味です。	以下の値を決定します。 • CPU の設定: • メモリーの設定:
リソース: ホスト	リソースをホスト用に確保して、一定の容量のメモリーやディスクリソースがそのホスト上の別のリソースに自動的に割り当てられないようにすることができます。	以下の値を決定します。 • ホストのディスク (デフォルト 0 MB): • ホストのメモリー (デフォルト 512 MB):
libvirt のバージョン	仮想インターフェースの結線を設定するには、使用する libvirt のバージョンを知っておく必要があります。	バージョン:
接続	Compute サービスをホストするサーバーは、他の全 OpenStack サービスに接続されている必要があります。	はい | いいえ

表1.8 OpenStack Dashboard サービス

項目	説明	値
ホストのソフトウェア	Dashboard サービスをホストするシステムは、以下のパッケージがインストール済みである必要があります。 • httpd • mod_wsgi • mod_ssl	はい | いいえ
接続	Dashboard サービスをホストするシステムは、他の全 OpenStack サービスに接続されている必要があります。	はい | いいえ

第2章 前提条件

本章は、すべてのノードでファイアウォール機能を提供する **iptables** を使用するように設定する方法を説明します。また、Red Hat OpenStack Platform 環境の全コンポーネントで使用するデータベースサービスとメッセージブローカーのインストール方法も説明します。MariaDB データベースサービスは、各コンポーネントに必要なデータベースを作成してアクセスするためのツールを提供します。RabbitMQ メッセージブローカーにより、コンポーネント間の内部通信が可能になります。メッセージは、メッセージブローカーを使用するように設定されたコンポーネントであればどこからでもメッセージの送受信ができます。

注記

Red Hat OpenStack Platform をデプロイする前には、利用可能なデプロイメソッドの特性を考慮することが重要です。詳しくは、「Installing and Managing Red Hat OpenStack Platform」の記事を参照してください。

2.1. ファイアウォールの設定

各コンポーネントをホストするサーバーが **iptables** を使用するように設定します。この際、Network Manager サービスを無効にして、サーバーが **firewalld** で提供されるファイアウォール機能ではなく、**iptables** のファイアウォール機能を使用するように設定する必要があります。本書で記載するその他のファイル設定はすべて、**iptables** を使用します。

2.1.1. Network Manager の無効化

OpenStack Networking は、Networking Manager サービスが有効化されているシステムでは機能しません。以下の手順に記載するステップはすべて、ネットワークトラフィックを処理する環境の各サーバーに **root** ユーザーとしてログインして実行する必要があります。これには、OpenStack Networking、ネットワークノードすべて、コンピュートノードすべてをホストするサーバーが含まれます。

手順2.1 Network Manager サービスの無効化

1. Network Manager が現在有効化されているかどうかを確認します。

   ```
   # systemctl status NetworkManager.service | grep Active:
   ```

   - Network Manager サービスが現在インストールされていない場合には、エラーが表示されます。このエラーが表示された場合には、この先の操作を実行して Network Manager サービスを無効にする必要はありません。
   - Network Manager が稼働している場合には、システムは **Active: active (running)** と表示し、稼働していない場合は **Active: inactive (dead)** と表示します。Networking Manager がアクティブでない場合には、この先の操作は必要ありません。

2. Network Manager が稼働している場合には、Networking Manager を停止してから無効化する必要があります。

   ```
   # systemctl stop NetworkManager.service
   # systemctl disable NetworkManager.service
   ```

3. システムの各インターフェースの設定ファイルをテキストエディターで開きます。インターフェースの設定ファイルは、**/etc/sysconfig/network-scripts/** ディレクトリーにあり、ファイル名は **ifcfg-*X*** の形式です (*X* は、インターフェース名に置き換えます)。有効な

インターフェース名には、**eth0**、**p1p5**、**em1** などがあります。

標準のネットワークサービスがインターフェースを制御して、ブート時に自動的にアクティブ化されるように、以下のキーが各インターフェースの設定ファイルで設定されているか確認して、設定されていない場合には手動で以下を追加します。

```
NM_CONTROLLED=no
ONBOOT=yes
```

4. 標準のネットワークサービスを起動します。

```
# systemctl start network.service
```

5. ネットワークサービスがブート時に起動するように設定します。

```
# systemctl enable network.service
```

2.1.2. firewalld サービスの無効化

コンピュートノードおよび OpenStack Networking ノードの **firewalld** サービスを無効にして、**iptables**サービスを有効にします。

手順2.2 firewalld サービスの無効化

1. **iptables** サービスをインストールします。

```
# yum install iptables-services
```

2. **/etc/sysconfig/iptables** に定義されている iptables ルールを確認します。

注記

以下のコマンドで、現在の **firewalld** 設定を確認できます。

```
# firewall-cmd --list-all
```

3. **iptables** ルールに問題がなければ、**firewalld** を無効化します。

```
# systemctl disable firewalld.service
```

4. **firewalld** サービスを停止して、**iptables** サービスを起動します。

```
# systemctl stop firewalld.service; systemctl start
iptables.service; systemctl start ip6tables.service
```

5. **iptables** サービスがブート時に起動するようにを設定します。

```
# systemctl enable iptables.service
# systemctl enable ip6tables.service
```

2.2. データベースサーバーのインストール

各 OpenStack コンポーネントには、実行中の MariaDB データベースサービスが必要です。全 Red Hat OpenStack Platform サービスをデプロイしたり、単一の OpenStack コンポーネントをインストールしたりする前に、データベースサービスをデプロイしてください。

2.2.1. MariaDB データベースパッケージのインストール

MariaDB データベースサーバーには以下のパッケージが必要です。

mariadb-galera-server

MariaDB データベースサービスを提供します。

mariadb-galera-common

MariaDB サーバーの共有ファイルを提供します。このパッケージは、mariadb-galera-server パッケージの依存関係としてインストールされます。

galera

Galera wsrep (Write Set REPlication) provider をインストールします。このパッケージは、mariadb-galera-server パッケージの依存関係としてインストールされます。

パッケージをインストールします。

```
# yum install mariadb-galera-server
```

2.2.2. データベースのトラフィックを許可するためのファイアウォール設定

OpenStack 環境のすべてのコンポーネントは、データベースサーバーを使用し、そのデータベースにアクセスできる必要があります。データベースサービスをホストするサーバーのファイアウォールは、必要なポートでのネットワークトラフィックを許可するように設定する必要があります。以下の手順に記載するステップはすべて、データベースサービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順2.3 データベースのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

2. このファイルに、ポート **3306** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

   ```
   -A INPUT -p tcp -m multiport --dports 3306 -j ACCEPT
   ```

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

4. **iptables** サービスを再起動して、変更を有効にします。

   ```
   # systemctl restart iptables.service
   ```

2.2.3. データベースサービスの起動

以下の手順に記載するステップはすべて、データベースサービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順2.4 データベースサービスの起動

1. **mariadb** サービスを起動します。

```
# systemctl start mariadb.service
```

2. **mariadb** サービスがブート時に起動するように設定します。

```
# systemctl enable mariadb.service
```

2.2.4. データベース管理者アカウントの設定

デフォルトでは、MariaDB は **root** というデータベースユーザーを作成します。これにより、MariaDB サーバーがインストールされているマシンから MariaDB サーバーへのアクセスが提供されます。MariaDB サービスをホストするサーバーにセキュアにアクセスするには、このアカウントにパスワードを設定する必要があります。また、MariaDB サーバーがインストールされているマシン以外のマシンから MariaDB サーバーにアクセスする必要がある場合には、MariaDB サーバーへのアクセスを有効化する必要があります。インストール時に作成された匿名ユーザーおよびテストデータベースは削除することを推奨します。

手順2.5 データベース管理者アカウントの設定

1. MariaDB サービスがインストールされているマシンにログインします。

2. **mysql_secure_installation** を使用して **root** パスワードの設定、リモートの root ログインの許可、匿名ユーザーアカウントおよびテストデータベースの削除を行います。

```
# mysql_secure_installation
```

注記

必要に応じて、データベースユーザーのパスワードを変更します。以下の例で、*OLDPASS* はユーザーの既存のパスワードに、*NEWPASS* は新規パスワードに置き換えてください。ただし、**-p** と古いパスワードの間にはスペースをあけないようにしてください。

```
# mysqladmin -u root -pOLDPASS password NEWPASS
```

2.2.5. 接続性のテスト

データベースユーザーアカウントが正しく設定されていることを確認するには、MariaDB データベースがインストールされているマシン (ローカルの接続性)、および MariaDB サーバーがインストールされているのとは別のマシン (リモートの接続性) から、そのユーザーアカウントの MariaDB サーバーとの接続性をテストします。

2.2.5.1. ローカルの接続性のテスト

MariaDB サービスがインストールされているマシンからデータベースサービスをホストするサーバーに接続できるかどうかをテストします。

手順2.6 ローカルの接続性のテスト

1. データベースサービスに接続します。**USER** は接続先のユーザー名に置き換えます。

```
# mysql -u USER -p
```

2. プロンプトが表示されたら、データベースユーザー名を入力します。

```
Enter password:
```

データベースユーザーのパーミッションが正しく設定されている場合には、接続が成功して MariaDB の Welcome 画面とプロンプトが表示されます。データベースユーザーのパーミッションが正しく設定されていない場合には、そのデータベースユーザーにはデータベースサーバーへの接続が許可されていないことを示すエラーメッセージが表示されます。

2.2.5.2. リモートの接続性のテスト

データベースサーバーがインストールされているのとは別のマシンから MariaDB サーバーへの接続が可能かどうかをテストします。

手順2.7 リモートの接続性のテスト

1. MySQL クライアントツールをインストールします。

```
# yum install mysql
```

2. データベースサービスに接続します。*USER* はデータベースのユーザー名、*HOST* はデータベースサービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

```
# mysql -u USER -h HOST -p
```

3. プロンプトが表示されたら、データベースユーザー名を入力します。

```
Enter password:
```

データベースユーザーのパーミッションが正しく設定されている場合には、接続が成功して MariaDB の Welcome 画面とプロンプトが表示されます。データベースユーザーのパーミッションが正しく設定されていない場合には、そのデータベースユーザーにはデータベースサーバーへの接続が許可されていないことを示すエラーメッセージが表示されます。

2.3. メッセージブローカーのインストール

完全な Red Hat OpenStack Platform 環境をデプロイするには、稼働中のメッセージブローカーを以下の OpenStack コンポーネントに設定する必要があります。

- Block Storage サービス
- Compute サービス

- OpenStack Networking
- Orchestration サービス
- Image サービス
- Telemetry サービス

2.3.1. RabbitMQ メッセージブローカーパッケージのインストール

RabbitMQ はデフォルト (かつ推奨) のメッセージブローカーです。RabbitMQ メッセージングサービスは rabbitmq-server パッケージにより提供されます。

RabbitMQ をインストールします。

```
# yum install rabbitmq-server
```

2.3.2. メッセージブローカーのトラフィックを許可するためのファイアウォール設定

メッセージブローカーをインストールおよび設定する前には、使用するポートで受信接続を許可しておく必要があります。メッセージブローカー (AMQP) のトラフィック用のデフォルトポートは **5672** です。以下の手順で記載するステップはすべて、メッセージングサービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順2.8 メッセージブローカーのトラフィックのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。
2. このファイルに、ポート **5672** で 受信接続を許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

   ```
   -A INPUT -p tcp -m tcp --dport 5672  -j ACCEPT
   ```

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。
4. **iptables** サービスを再起動して、ファイアウォールの変更を有効にします。

   ```
   # systemctl restart iptables.service
   ```

2.3.3. RabbitMQ メッセージブローカーの起動と設定

手順2.9 RabbitMQ メッセージブローカーを OpenStack で使用するための起動および設定手順

1. **rabbitmq-server** サービスを立ち上げ、ブート時に起動するように設定します。

   ```
   # systemctl start rabbitmq-server.service
   # systemctl enable rabbitmq-server.service
   ```

2. rabbitmq-server パッケージのインストール時には、RabbitMQ サービスの **guest** ユーザーは、デフォルトの **guest** パスワードとともに自動的に作成されます。Red Hat では、特に IPv6 が利用可能な場合には、このデフォルトパスワードを変更することを強くお勧めします。IPv6 では、RabbitMQ はネットワーク外部からアクセスが可能となる場合があります。

```
# rabbitmqctl change_password guest NEW_RABBITMQ_PASS
```

NEW_RABBITMQ_PASS は、よりセキュアなパスワードに置き換えます。

3. Block Storage サービス、Compute サービス、OpenStack Networking、Orchestration サービス、Image サービス、Telemetry サービス用の RabbitMQ ユーザーアカウントを作成します。

```
# rabbitmqctl add_user cinder CINDER_PASS
# rabbitmqctl add_user nova NOVA_PASS
# rabbitmqctl add_user neutron NEUTRON_PASS
# rabbitmqctl add_user heat HEAT_PASS
# rabbitmqctl add_user glance GLANCE_PASS
# rabbitmqctl add_user ceilometer CEILOMETER_PASS
```

CINDER_PASS、*NOVA_PASS*、*NEUTRON_PASS*、*HEAT_PASS*、*GLANCE_PASS*、*CEILOM* は、各サービスのセキュアなパスワードに置き換えてください。

4. これらの RabbitMQ ユーザーに、全リソースに対する読み取り/書き込みのパーミッションを付与します。

```
# rabbitmqctl set_permissions cinder ".*" ".*" ".*"
# rabbitmqctl set_permissions nova ".*" ".*" ".*"
# rabbitmqctl set_permissions neutron ".*" ".*" ".*"
# rabbitmqctl set_permissions heat ".*" ".*" ".*"
# rabbitmqctl set_permissions glance ".*" ".*" ".*"
# rabbitmqctl set_permissions ceilometer ".*" ".*" ".*"
```

2.3.4. RabbitMQ メッセージブローカーでの SSL の有効化

RabbitMQ メッセージブローカーには SSL 機能が組み込まれており、トラフィックのセキュリティー保護に使用することができます。SSL 通信に必要な証明書を作成して、**/etc/rabbitmq/rabbitmq.config** 設定ファイルで RabbitMQ に SSL を設定します。

手順2.10 RabbitMQ メッセージブローカーでの SSL の有効化

1. 必要な証明書を保管するためのディレクトリーを作成します。

```
# mkdir /etc/pki/rabbitmq
```

2. 証明書用のセキュアなパスワードを選択して、**/etc/pki/rabbitmq** ディレクトリー内にファイル形式で保存します。

```
# echo SSL_RABBITMQ_PW > /etc/pki/rabbitmq/certpw
```

SSL_RABBITMQ_PW は、証明書のパスワードに置き換えます。このパスワードは、後で必要な証明書をさらにセキュリティー保護する際に使用します。

3. 証明書のディレクトリーとパスワードファイルのパーミッションを設定します。

```
# chmod 700 /etc/pki/rabbitmq
# chmod 600 /etc/pki/rabbitmq/certpw
```

4. **/etc/pki/rabbitmq/certpw** ファイル内のパスワードを使用して **/etc/pki/rabbitmq**

ディレクトリーに証明書データベースファイル (***.db**) を作成します。

```
# certutil -N -d /etc/pki/rabbitmq -f /etc/pki/rabbitmq/certpw
```

5. 実稼働環境では、信頼のできるサードパーティーの証明局 (CA) を使用して証明書に署名することを推奨します。サードパーティーの CA には、証明書署名要求 (CSR) が必要となります。

```
# certutil -R -d /etc/pki/rabbitmq -s "CN=RABBITMQ_HOST" \
 -a -f /etc/pki/rabbitmq/certpw > RABBITMQ_HOST.csr
```

RABBITMQ_HOST は、RabbitMQ メッセージブローカーをホストするサーバーの IP アドレスまたはホスト名に置き換えます。このコマンドにより、***RABBITMQ_HOST*.csr** という名前の CSR とキーファイル (*keyfile.key*) が生成されます。このキーファイルは、後で RabbitMQ メッセージブローカーが SSL を使用するように設定する際に使用します。

注記

一部の CA には、**"CN=*RABBITMQ_HOST*"** 以外の追加の値が必要な場合があります。

6. ***RABBITMQ_HOST*.csr** をサードパーティーの CA に提供して署名を受けます。CA は署名済みの証明書 (*server.crt*) と CA ファイル (*ca.crt*) を提供します。これらのファイルを証明書のデータベースに追加します。

```
# certutil -A -d /etc/pki/rabbitmq -n RABBITMQ_HOST -f
/etc/pki/rabbitmq/certpw \
 -t u,u,u -a -i /path/to/server.crt
# certutil -A -d /etc/pki/rabbitmq -n "Your CA certificate" \
 -f /etc/pki/rabbitmq/certpw -t CT,C,C -a -i /path/to/ca.crt
```

7. RabbitMQ メッセージブローカーがセキュアな通信に証明書ファイルを使用するように設定します。テキストエディターで **/etc/rabbitmq/rabbitmq.config** の設定ファイルを開き、以下のように **rabbit** のセクションを編集します。

   1. 以下の行を探します。

   ```
       %% {ssl_listeners, [5671]},
   ```

   パーセントの記号を削除して、設定をアンコメントします。

   ```
       {ssl_listeners, [5671]},
   ```

   2. 次に以下の行まで、下方向にスクロールします。

   ```
       %% {ssl_options, [{cacertfile,
   "/path/to/testca/cacert.pem"},
   ```

   この行と、その次の **ssl_options** で構成される数行を、以下の内容に置き換えます。

   ```
       {ssl_options, [{cacertfile,           "/path/to/ca.crt"},
                      {certfile,
   "/path/to/server.crt"},
   ```

```
                   {keyfile,
"/path/to/keyfile.key"},
                   {verify,               verify_peer},
                   {versions,
['tlsv1.2','tlsv1.1',tlsv1]},
                   {fail_if_no_peer_cert, false}]}
```

- */path/to/ca.crt* は、CA 証明書への絶対パスに置き換えます。
- */path/to/server.crt* は、署名済みの証明書への絶対パスに置き換えます。
- */path/to/keyfile.key* はキーファイルへの絶対パスに置き換えます。

8. 特定の TLS 暗号化バージョンのみのサポートを含めるように **rabbitmq.config** を編集して、SSLv3 を無効化します。

```
{rabbit, [
{ssl_options, [{versions, ['tlsv1.2','tlsv1.1',tlsv1]}]},
]}
```

9. RabbitMQ サービスを再起動し、変更を有効にします。

```
# systemctl restart rabbitmq-server.service
```

2.3.5. クライアント用 SSL 証明書のエクスポート

サーバーで SSL を有効にする場合には、セキュアな接続を確立するために、クライアントにその SSL 証明書のコピーが必要です。

以下のコマンド例は、メッセージブローカーの証明書データベースからのクライアント用証明書と秘密鍵をエクスポートするのに使用することができます。

```
# pk12util -o <p12exportfile> -n <certname> -d <certdir> -w
<p12filepwfile>
# openssl pkcs12 -in <p12exportfile> -out <clcertname> -nodes -clcerts -
passin pass:<p12pw>
```

SSL コマンドとオプションに関する詳細情報は、OpenSSL のマニュアル を参照してください。または、Red Hat Enterprise Linux では、**openssl** のマニュアルページを参照してください。

2.4. ネットワークタイムプロトコルサーバー (NTP)

OpenStack 環境の各システムでネットワークタイムプロトコル (NTP) を使用して全システムを同期します。まず、コントローラーノードで NTP を設定して、組織内で一般的に使用されている同じ外部の NTP サーバーが設定されているようにします。次に、OpenStack 環境の残りのシステムをコントローラーノードからの同期情報を取得するように設定します。

重要

さまざまなソースから同期して、異なるネットワークを介してルーティングする、外部の NTP サーバーを使用します。

OpenStack 環境に複数のコントローラーノードがある場合には、少しのずれでも他のシステムに問題が発生する可能性があるため、クロックの同期に細心の注意を払う必要があります。またこのような環境では、コントローラーノードの 1 つが利用できなくなった場合のために、システムが複数のコントローラーノードから同期情報を取得するように設定しておくと役立ちます。

NTP の設定方法については Red Hat Enterprise Linux 7 の『システム管理者のガイド』を参照してください。

2.5. OPENSTACK コマンドラインクライアントのインストール

openstack コマンドラインクライアントで、OpenStack サービスを設定してユーザーやプロジェクトを作成するには、python-openstackclient パッケージを必ずインストールするようにしてください。

```
# yum install python-openstackclient
```

第3章 IDENTITY サービスのインストール

本章では、OpenStack Identity サービスのインストール/設定方法およびこのサービスの使用に必要な基本的なユーザーアカウントとテナントの設定方法を説明します。

3.1. IDENTITY サービスのパッケージのインストール

Identity サービスには以下のパッケージが必要です。

openstack-keystone
: OpenStack Identity サービスを提供します。

openstack-utils
: 設定ファイルの編集をはじめとする数々のタスクに役立つサポートユーティリティーを提供します。

openstack-selinux
: OpenStack 固有の SELinux ポリシーモジュールを提供します。

httpd
: Apache Web サーバーを提供します。

mod_wsgi
: Apache 下で Python ベースの Web アプリケーションをホストするための WSGI 対応のインターフェースを提供します。

パッケージをインストールします。

```
# yum install -y openstack-keystone \
      openstack-utils \
      openstack-selinux
      httpd \
      mod_wsgi
```

3.2. アイデンティティーデータベースの作成

Identity サービスが使用するデータベースとデータベースユーザーを作成します。以下の手順の全ステップは、データベースサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順3.1 Identity サービスのデータベースの作成

1. データベースサービスに接続します。

   ```
   # mysql -u root -p
   ```

2. **keystone** データベースを作成します。

   ```
   mysql> CREATE DATABASE keystone;
   ```

3. **keystone** データベースユーザーを作成して、**keystone** データベースへのアクセスを許可します。

   ```
   mysql> GRANT ALL ON keystone.* TO 'keystone'@'%' IDENTIFIED BY
   'PASSWORD';
   mysql> GRANT ALL ON keystone.* TO 'keystone'@'localhost' IDENTIFIED
   BY 'PASSWORD';
   ```

   PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. データベースの特権をフラッシュして、設定が即時に反映されるようにします。

   ```
   mysql> FLUSH PRIVILEGES;
   ```

5. **mysql** クライアントを終了します。

   ```
   mysql> quit
   ```

3.3. IDENTITY サービスの設定

3.3.1. Identity サービスのデータベース接続の設定

Identity サービスによって使用されるデータベース接続文字列は、**/etc/keystone/keystone.conf** ファイルで定義されます。サービスを起動する前に、有効なデータベースサーバーをポイントするように更新しておく必要があります。

以下の手順に記載するステップはすべて、Identity サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順3.2 Identity サービスの SQL データベース接続の設定

- **connection** 設定キーの値を設定します。

  ```
  # openstack-config --set /etc/keystone/keystone.conf \
     sql connection mysql://USER:PASS@IP/DB
  ```

  以下の値を置き換えてください。

  - *USER* は、Identity サービスのデータベースのユーザー名 (通常は **keystone**) に置き換えます。
  - *PASS* は選択したデータベースユーザーのパスワードに置き換えます。
  - *IP* は、Identity サーバーの IP アドレスまたはホスト名に置き換えます。
  - *DB* は、Identity サービスのデータベースの名前 (通常は **keystone**) に置き換えます。

重要

この接続設定キーに指定する IP アドレスまたはホスト名は、keystone データベースの作成時に keystone データベースユーザーがアクセスを許可された IP アドレスまたはホスト名と一致する必要があります。また、データベースがローカルでホストされ、keystone データベースの作成時に「localhost」へのアクセス権を付与した場合には、「localhost」と入力する必要があります。

3.3.2. 公開鍵インフラストラクチャーの設定

3.3.2.1. 公開鍵インフラストラクチャーの概要

Identity サービスは、ユーザーおよびその他のサービスが認証に使用する、暗号により署名されたドキュメントであるトークンを生成します。トークンは、秘密鍵で署名される一方、公開鍵は X509 証明書で提供されます。

証明書および関連する設定キーは、**keystone-manage pki_setup** コマンドで自動的に生成されますが、サードパーティーの証明機関を使用して必要な証明書の作成と署名を手動で行うことも可能です。サードパーティーの証明書を使用する場合には、Identity サービスの設定を手動で更新して、証明書と関連ファイルをポイントするようにする必要があります。

/etc/keystone/keystone.conf 設定ファイルの **[signing]** セクションには、PKI 設定に関連する以下のような設定キーが表示されます。

ca_certs

certfile 設定キーによって示された証明書を発行した認証局向けに証明書の場所を指定します。デフォルト値は **/etc/keystone/ssl/certs/ca.pem** です。

ca_key

certfile 設定キーによって示された証明書を発行した認証局のキーを指定します。デフォルト値は **/etc/keystone/ssl/certs/cakey.pem** です。

ca_password

認証局のファイルを開くために必要なパスワード (該当する場合) を指定します。値が指定されていない場合に、デフォルトのアクションではパスワードを使用しません。

certfile

トークンの検証に使用する必要のある証明書の場所を指定します。値が指定されていない場合には、デフォルト値の **/etc/keystone/ssl/certs/signing_cert.pem** が使用されます。

keyfile

トークンの署名時に使用する必要のある秘密鍵の場所を指定します。値が指定されていない場合には、デフォルト値の **/etc/keystone/ssl/private/signing_key.pem** が使用されます。

token_format

トークン生成時に使用するアルゴリズムを指定します。使用可能な値は **UUID** と **PKI** です。デフォルトは **PKI** です。

3.3.2.2. 公開鍵インフラストラクチャーファイルの作成

以下のセクションでは、Identity サービスが使用する PKI ファイルの作成と設定の方法を説明します。以下の手順で記載するステップはすべて、Identity サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順3.3 Identity サービスが使用する PKI ファイルの作成

1. **keystone-manage pki_setup** コマンドを実行します。

```
# keystone-manage pki_setup \
   --keystone-user keystone \
   --keystone-group keystone
```

2. **keystone** ユーザーが **/var/log/keystone/** および **/etc/keystone/ssl/** のディレクトリーを所有するように設定します。

```
# chown -R keystone:keystone /var/log/keystone \
   /etc/keystone/ssl/
```

3.3.2.3. 公開鍵インフラストラクチャーファイルを使用するための Identity サービスの設定

Identity サービスが使用するように PKI ファイルを生成した後には、Identity サービスがこのファイルを使用するように有効化する必要があります。

/etc/keystone/keystone.conf ファイルの属性値を設定します。

```
# openstack-config --set /etc/keystone/keystone.conf \
 signing token_format PKI
# openstack-config --set /etc/keystone/keystone.conf \
 signing certfile /etc/keystone/ssl/certs/signing_cert.pem
# openstack-config --set /etc/keystone/keystone.conf \
 signing keyfile /etc/keystone/ssl/private/signing_key.pem
# openstack-config --set /etc/keystone/keystone.conf \
 signing ca_certs /etc/keystone/ssl/certs/ca.pem
# openstack-config --set /etc/keystone/keystone.conf \
 signing key_size 1024
# openstack-config --set /etc/keystone/keystone.conf \
 signing valid_days 3650
# openstack-config --set /etc/keystone/keystone.conf \
 signing ca_password None
```

また、**/etc/keystone/keystone.conf** ファイルを直接編集して、これらの値を更新します。

3.3.3. Identity サービスのトラフィックを許可するためのファイアウォール設定

OpenStack 環境内の各コンポーネントは、認証に Idenity サービスを使用するため、このサービスへアクセスできる必要があります。

Block Storage サービスをホストするシステムのファイアウォール設定を変更して、これらのポートでのネットワークトラフィックを許可する必要があります。以下の手順に記載するステップはすべて、Identity サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順3.4 Identity サービスのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

2. このファイルに、ポート **5000** および **35357** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

```
-A INPUT -p tcp -m multiport --dports 5000,35357 -j ACCEPT
```

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

4. **iptables** サービスを再起動して、変更を有効にします。

```
# systemctl restart iptables.service
```

3.3.4. Identity サービスのデータベースへのデータ投入

Identity サービスのデータベース接続文字列を適切に設定した後には、Identiy サービスのデータベースにデータを投入します。

手順3.5 Identity サービスのデータベースへのデータ投入

1. Identity サービスをホストするシステムにログインします。

2. **keystone** ユーザーに切り替え、**/etc/keystone/keystone.conf** で特定されているデータベースを初期化してデータを投入します。

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

3.3.5. コレクション内のエンティティー数の制限

以下の手順を使用して、list コマンドが返す結果の数を制限します。結果数が利用可能なメモリーよりも大きい場合に発生する問題を回避したり、一覧の応答時間が長くならないようにしたりするために、上限を指定できます。

手順3.6 コレクション内のエンティティー数の制限

1. テキストエディターで **/etc/keystone/keystone.conf** を開きます。

2. **[DEFAULT]** セクションで **list_limit** を使用してグローバル値を設定します。

3. オプションで、個別のセクションで特定の制限を指定することで、このグローバル値を無効にすることができます。以下に例を示します。

```
[assignment]
list_limit = 100
```

list_{entity} の呼び出しへの応答が省略された場合には、応答の状態コードが 200 (OK) のままですが、コレクション内の **truncated** 属性が **true** に設定されます。

3.3.6. Apache HTTP サーバーの設定

Identity サービスが適切に機能するようにするには、Apache サーバーが **keystone** サービスおよび **wsgi** モジュール向けの適切な構成を使用するように設定する必要があります。

手順3.7 Apache HTTP サーバーの設定

1. テキストエディターで **/etc/httpd/conf/httpd.conf** ファイルを編集します。**ServerName** オプションがコントローラーノードを参照するように設定してください。

```
ServerName controller
```

2. **/etc/httpd/conf/httpd.conf** ファイルへのリンクを作成します。

```
# ln -s /usr/share/keystone/wsgi-keystone.conf /etc/httpd/conf.d
```

3.4. IDENTITY サービスの起動

以下の手順に記載するステップはすべて、Identity サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順3.8 Identity サービスの起動

1. **httpd** サービスを起動します。

```
# systemctl start httpd.service
```

2. **httpd** サービスがブート時に起動するように設定します。

```
# systemctl enable httpd.service
```

3.5. 管理者アカウントおよび IDENTITY サービスエンドポイントの作成

以下の手順では、管理ユーザーアカウントと関連付けられたテナントおよびロールを作成します。Identity サービスのエンドポイントは同時に作成されます。

以下の手順に記載するステップは、Identity サービスをホストするシステムで実行する必要があります。

手順3.9 管理者アカウントおよび Identity サービスエンドポイントの作成

1. **admin** ユーザー、ロール、テナントを作成します。

```
# keystone-manage bootstrap \
--bootstrap-password PASSWORD \
--bootstrap-username admin \
--bootstrap-project-name admin \
--bootstrap-role-name admin \
--bootstrap-service-name keystone \
--bootstrap-region-id RegionOne \
--bootstrap-admin-url http://IP:35357 \
--bootstrap-public-url http://IP:5000 \
--bootstrap-internal-url http://IP:5000
```

PASSWORD は **admin** ユーザーのパスワードに、*IP* は Identity サーバーの IP アドレスまたはホスト名に置き換えます。

2. 新規作成された **admin** アカウントは、Identity サービスの今後の管理に使用されます。認証を円滑化するためには、セキュリティー保護された場所 (**root** ユーザーのホームディレクトリーなど) に **keystonerc_admin** ファイルを作成します。

   ファイルに以下の行を追加して、認証に使用する環境変数を設定します。

   ```
   export OS_USERNAME=admin
   export OS_TENANT_NAME=admin
   export OS_PASSWORD=PASSWORD
   export OS_AUTH_URL=http://IP:35357/v2.0/
   export PS1='[\u@\h \W(keystone_admin)]\$ '
   ```

   今回も *PASSWORD* は **admin** ユーザーのパスワードに、*IP* は Identity サーバーの IP アドレスまたはホスト名に置き換えます。

3. 認証に使用する環境変数を読み込みます。

   ```
   # source ~/keystonerc_admin
   ```

注記

Red Hat は、以前使用されていた *管理トークン* に代わり、**keystone-manage bootstrap** コマンドを使用して管理者アカウントを作成することを推奨します。管理トークンを設定して使用する場合は、Red Hat OpenStack Platform 11 『手動インストール手順』の「管理者アカウントおよび Identity サービスエンドポイントの作成」の項を参照してください。

3.5.1. サービスのリージョン

Identity サービスにカタログ化されている各サービスは、そのリージョンによって特定されます。リージョンは通常、地理的な場所およびそのエンドポイントを示します。複数の Compute デプロイメントを使用する Red Hat OpenStack Platform 環境では、リージョンによりサービスを個別に分離することが可能となります。これは、Compute がインストールされた複数のシステム間でインフラストラクチャーを共有する強固な方法であるとともに、高度の耐障害性を実現します。

管理者は、複数のリージョン間で共有するサービスと、特定のリージョンのみで使用されるサービスを決定します。デフォルトでは、エンドポイントが定義されていてリージョンの指定がない場合には、**RegionOne** という名前のリージョンに作成されます。

個別のリージョンの使用を開始するには、サービスエンドポイントの追加時に ***--region*** の引数を指定します。

```
[(keystone_admin)]# openstack endpoint create --region REGION \
   --publicurl PUBLICURL \
   --adminurl ADMINURL \
   --internalurl INTERNALURL \
   SERVICENAME
```

REGION は、エンドポイントが属するリージョンの名前に置き換えます。リージョン間でエンドポイントを共有する場合には、該当する各リージョンに、同じ URL を含むエンドポイントエントリーを作成

します。各サービスの URL を設定する方法についての説明は、対象とするサービスの Identity サービスの設定情報を参照してください。

例3.1 個別のリージョン内のエンドポイント

以下に記載する例では、**APAC** および **EMEA** のリージョンが Identity サーバー (**identity.example.com**) エンドポイントを共有する一方、リージョン固有の Compute API エンドポイントを提供します。

```
$ openstack endpoint list --long
+--------+----------+-------------+-------------+---------------
-------------------------------------+...
| ID     | Region   | Service Name | Service Type | PublicURL
|...
+--------+----------+-------------+-------------+---------------
-------------------------------------+...
| b02... | APAC     | compute      | compute      | http://nova-
apac.example.com:8774/v2/%(tenant_id)s  |...
| c46... | APAC     | keystone     | identity     |
http://identity.example.com:5000/v3                  |...
| 31d... | EMEA     | compute      | compute      | http://nova-
emea.example.com:8774/v2/%(tenant_id)s  |...
| 727... | EMEA     | keystone     | identity     |
http://identity.example.com:5000/v3                  |...
+--------+----------+-------------+-------------+---------------
-------------------------------------+...
```

3.6. 一般ユーザーアカウントの作成

一般ユーザーおよびテナントを作成します。テナントは、サービスリソースの集約に使用されており、プロジェクトとしても知られています。

以下の手順に記載するステップは、Identity サービスをホストするシステムで実行する必要があります。

手順3.10 一般ユーザーアカウントの作成

1. Keystone 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 ~/keystonerc_admin
   ```

2. テナントを作成します。

   ```
   [(keystone_admin)]# openstack project create TENANT
   +-------------+----------------------------------+
   | Field       | Value                            |
   +-------------+----------------------------------+
   | description | None                             |
   | enabled     | True                             |
   | id          | 99c674e3fead4237ace2a0d86dab76e4 |
   | name        | TENANT                           |
   +-------------+----------------------------------+
   ```

TENANT は、テナント名に置き換えます。

3. 一般ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --project TENANT --
password PASSWORD USER
+------------+----------------------------------+
| Field      | Value                            |
+------------+----------------------------------+
| email      | None                             |
| enabled    | True                             |
| id         | 246b1342a8684bf39d7cc5165ef835d4 |
| name       | USER                             |
| project_id | 99c674e3fead4237ace2a0d86dab76e4 |
| username   | USER                             |
+------------+----------------------------------+
```

USER はこのアカウントのユーザー名に、*TENANT* は前のステップで使用したテナント名に、*PASSWORD* はこのアカウントのセキュアなパスワードに置き換えます。

注記

--project オプションが指定されているため、ユーザーはデフォルトで Identity の **_member_** ロールが自動的に関連付けられます。

4. 認証を円滑化するためには、セキュリティー保護された場所 (例: **root** ユーザーのホームディレクトリーなど) に **keystonerc_*user*** ファイルを作成します。

認証に使用する以下の環境変数を設定します。

```
export OS_USERNAME=USER
export OS_TENANT_NAME=TENANT
export OS_PASSWORD=PASSWORD
export OS_AUTH_URL=http://IP:5000/v2.0/
export PS1='[\u@\h \W(keystone_user)]\$ '
```

USER、*TENANT*、*PASSWORD* は、テナントおよびユーザーの作成時に指定した値に置き換えます。また *IP* は Identity Server の IP アドレスまたはホスト名に置き換えてください。

3.7. サービステナントの作成

テナントごとにクォータ制御を使用してリソース数を制限することができます。

注記

クォータに関する詳細は、Red Hat OpenStack Platform 『管理ガイド』の「プロジェクトの管理」セクションを参照してください。このガイドは以下のリンクから入手することができます。

https://access.redhat.com/site/documentation/ja/Red_Hat_Enterprise_Linux_OpenStack_Pla

ユーザーごとに、テナントが 1 つ割り当てられます。一般ユーザーの場合には、テナントは通常、そのユーザーのグループ、プロジェクト、または組織を示します。サービスユーザー (サービスの代わりに

Identity サービスにアクセスするエンティティー) の場合には、テナントはサービスの地理的地域を示します。環境内でサービスが分散されている場合は、通常サービスが実行中のエンドポイントごとに、サービステナントが 1 つ作成されます (Identity サービスおよび Dashboard サービスを除く)。環境内のサービスが単一ノードにデプロイされる場合には、管理の目的でサービステナントを複数作成することも可能ですが、必要とされるサービステナント数は 1 つだけです。

本ガイドに記載のサービス設定例は、全サービスが単一のノードにデプロイされていることを前提としているため、必要なテナントは 1 つのみです。このような例ではすべて **services** テナントを使用します。

注記

管理者、一般ユーザー、サービスユーザーはすべてテナントを必要とするため、通常は各グループ用に少なくとも 3 テナントを作成します。管理ユーザー、一般ユーザー、およびテナントの作成方法については、「管理者アカウントおよび Identity サービスエンドポイントの作成」および「一般ユーザーアカウントの作成」を参照してください。

手順3.11 サービステナントの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **services** テナントを作成します。

```
[(keystone_admin)]# openstack project create --description "Services
Tenant" services
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Services Tenant                  |
| enabled     | True                             |
| id          | 42e1efb4bd5e49a49cb2b346078d6325 |
| name        | services                         |
+-------------+----------------------------------+
```

注記

Identity サービスの全テナントとそれらの ID の一覧を取得するには、次のコマンドを実行します。

```
[(keystone_admin)]# openstack project list
+----------------------------------+----------+
| ID                               | Name     |
+----------------------------------+----------+
| 42e1efb4bd5e49a49cb2b346078d6325 | services |
| 65d8216d98c64399b8f44929b634bc3f | admin    |
| 99c674e3fead4237ace2a0d86dab76e4 | test     |
+----------------------------------+----------+
```

3.8. IDENTITY サービスのインストールの検証

Identity サービスのインストールが正しく機能していることを確認します。以下の手順で記載するすべ

てのステップは、Identity サーバーまたは環境内の他のサービスで実行する必要があります。ログインするユーザーは、管理ユーザーおよび一般ユーザーとして認証するために、それぞれの必要な環境変数が含まれている **keystonerc_admin** と **keystonerc_user** のファイルへのアクセス権が必要です。また、システムには、httpd、mod_wsgi、mod_ssl (セキュリティー目的) をインストールしておく必要があります。

手順3.12 Identity サービスのインストールの検証

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

```
# source ~/keystonerc_admin
```

2. システムで定義されているユーザーの一覧を表示します。

```
[(keystone_admin)]# openstack user list
+----------------------------------+-------+
| ID                               | Name  |
+----------------------------------+-------+
| 23c56d02d3bc4b88b034e0b3720fcd1b | admin |
| 246b1342a8684bf39d7cc5165ef835d4 | USER  |
+----------------------------------+-------+
```

システムで定義されているユーザーの一覧が表示されます。一覧が表示されない場合には、インストールに問題があります。

a. 返されたメッセージでパーミッションまたは認証に問題があることが示されている場合には、管理者ユーザーアカウント、テナント、ロールが正しく作成されていることを確認します。また、3 つのオブジェクトが正しくリンクされていることも確認します。

b. 返されたメッセージで接続に問題がある (**Connection refused**) ことが示されている場合には、**openstack-keystone** サービスが実行中であることと、ポート **5000** および **35357** での接続を許可するようにファイアウォールサービスが設定されていることを確認してください。

3. 一般の Identityサービスのユーザーとして、Keystone にアクセスするためのシェルを設定します。

```
# source ~/keystonerc_user
```

4. システムで定義されているユーザーの一覧を表示してみます。

```
[(keystone_user)]# openstack user list
You are not authorized to perform the requested action:
admin_required (HTTP 403) (Request-ID: req-1cfd3869-ac97-424d-bd00-
f835a6ab9be6)
```

このコマンドを実行する権限がないことを示すエラーメッセージが表示されます。このエラーメッセージが表示されず、代わりにユーザー一覧が表示された場合には、その一般ユーザーアカウントに誤って **admin** ロールが関連付けられていたことになります。

3.8.1. Identity クライアント (keystone) の接続性における問題のトラブルシューティング

Identity クライアント (**keystone**) が Identity サービスと通信できない場合には、次のようなエラーが返されます。

```
Unable to communicate with identity service: [Errno 113] No route to host.
(HTTP 400)
```

この問題をデバッグするには、以下にあげる一般的な原因を確認してください。

Identity サービスが稼働していない場合

Identity サービスをホストするシステムで、サービスのステータスを確認します。

```
# systemctl status openstack-keystone
● openstack-keystone.service - OpenStack Identity Service (code-named
Keystone)
   Loaded: loaded (/usr/lib/systemd/system/openstack-keystone.service;
disabled; vendor preset: disabled)
   Active: active (running) since Tue 2016-06-07 02:31:14 EDT; 5h 29min
ago
 Main PID: 23236 (keystone-all)
   CGroup: /system.slice/openstack-keystone.service
           ├─23236 /usr/bin/python2 /usr/bin/keystone-all
           ├─23247 /usr/bin/python2 /usr/bin/keystone-all
           ├─23248 /usr/bin/python2 /usr/bin/keystone-all
           ├─23249 /usr/bin/python2 /usr/bin/keystone-all
           └─23250 /usr/bin/python2 /usr/bin/keystone-all

Jun 07 02:31:13 mitaka.localdomain systemd[1]: Starting OpenStack
Identity Service (code-named Keystone)...
Jun 07 02:31:14 mitaka.localdomain systemd[1]: Started OpenStack
Identity Service (code-named Keystone).
```

サービスが実行されていない場合には (**Active: inactive (dead)** という出力)、**root** ユーザーとしてログインして起動します。

```
# systemctl start openstack-keystone
```

ファイアウォールが適切に設定されていない場合

ファイアウォールがポート **5000** と **35357** で TCP トラフィックを許可するように設定されていない可能性があります。「Identity サービスのトラフィックを許可するためのファイアウォール設定」で設定を正しく修正する方法を参照してください。

第4章 OBJECT サービスのインストール

4.1. OBJECT STORAGE サービスの要件

以下のアイテムは、Object Storage サービスのインストール要件です。

サポート対象のファイルシステム

Object Storage サービスは、ファイルシステムにオブジェクトを保管します。現在は **XFS** および **ext4** がサポートされています。ファイルシステムは、**拡張属性** (**xattr**) を有効にした状態でマウントする必要があります。

XFS を使用するように推奨します。これは、**/etc/fstab** で設定します。

例4.1 1 つの XFS ストレージディスクの/etc/fstab のエントリー例

```
/dev/sdb1 /srv/node/d1 xfs inode64,noatime,nodiratime 0 0
```

注記

デフォルトでは、拡張属性はすでに **XFS** で有効になっています。そのため、**/etc/fstab** エントリーで、**user_xattr** を指定する必要はありません。

許容されるマウントポイント

Object Storage サービスは、デバイスが **/srv/node/** にマウントされることを想定します。

4.2. RSYNCD の設定

複製が必ず行われるように、まずお使いのファイルシステムの **rsyncd** を設定してから、Object Storage サービスをインストールして設定します。以下の手順では、各ストレージノードに **root** ユーザーでログインして実行する必要があります。本手順では、XFS ストレージディスクが少なくとも 2 つ、各ストレージノードにマウントされていることが前提です。

例4.2 2 つの XFS ストレージディスクの/etc/fstab のエントリー例

```
/dev/sdb1 /srv/node/d1 xfs inode64,noatime,nodiratime 0 0
/dev/sdb2 /srv/node/d2 xfs inode64,noatime,nodiratime 0 0
```

手順4.1 rsyncd の設定

1. コントローラーの **/etc/hosts** ファイルからのアドレスをコピーして、ストレージノードの IP アドレスを追加します。また、すべてのノードに **/etc/hosts** ファイルの全アドレスが指定されているようにします。

2. rsync および xinetd パッケージをインストールします。

```
# yum install rsync xinetd
```

3. テキストエディターで **/etc/rsyncd.conf** ファイルを開いて以下の行を追加します。

```
##assumes 'swift' has been used as the Object Storage user/group
uid = swift
gid = swift
log file = /var/log/rsyncd.log
pid file = /var/run/rsyncd.pid
##address on which the rsync daemon listens
address = LOCAL_MGT_NETWORK_IP

[account]
max connections = 2
path = /srv/node/
read only = false
write only      = no
list            = yes
incoming chmod  = 0644
outgoing chmod  = 0644
lock file = /var/lock/account.lock

[container]
max connections = 2
path = /srv/node/
read only = false
write only      = no
list            = yes
incoming chmod  = 0644
outgoing chmod  = 0644
lock file = /var/lock/container.lock

[object]
max connections = 2
path = /srv/node/
read only = false
write only      = no
list            = yes
incoming chmod  = 0644
outgoing chmod  = 0644
lock file = /var/lock/object.lock
```

注記

複数のアカウント、コンテナー、オブジェクトのセクションを使用することができます。

4. **/etc/xinetd.d/rsync** ファイルを開いて、以下の情報を追加します。

```
service rsync
 {
     port            = 873
     disable         = no
     socket_type     = stream
     protocol        = tcp
     wait            = no
```

```
        user            = root
        group           = root
        groups          = yes
        server          = /usr/bin/rsync
        bind            = LOCAL_MGT_NETWORK_IP
        server_args = --daemon --config /etc/rsync.conf
    }
```

5. **xinetd** サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start xinetd.service
# systemctl enable xinetd.service
```

4.3. OBJECT STORAGE サービスのパッケージのインストール

以下のパッケージにより、Object Storage サービスのコンポーネントが提供されます。

OpenStack Object Storage の主要パッケージ

openstack-swift-proxy

オブジェクトに対してプロキシーを要求します。

openstack-swift-object

最大 5 GB のデータオブジェクトを格納します。

openstack-swift-container

各コンテナー内のオブジェクトをすべてトラッキングするデータベースを維持管理します。

openstack-swift-account

各アカウント内の全コンテナーをトラッキングするデータベースを維持管理します。

OpenStack Object Storage の依存関係

openstack-swift

特定のサービスに共通したコードが含まれます。

openstack-swift-plugin-swift3

OpenStack Object Storage 用の swift3 プラグインです。

memcached

プロキシーサーバーのソフト依存関係で、対話時に毎回再認証せず、認証済みのクライアントをキャッシュします。

openstack-utils

OpenStack の設定用ユーティリティーを提供します。

python-swiftclient

swift コマンドラインツールを提供します。

手順4.2 Object Storage サービスのパッケージのインストール

- 必要なパッケージをインストールします。

```
# yum install -y openstack-swift-proxy \
   openstack-swift-object \
   openstack-swift-container \
   openstack-swift-account \
   openstack-utils \
   memcached \
   python-swiftclient
```

4.4. OBJECT STORAGE サービスの設定

4.4.1. Object Storage サービス用のアイデンティティーレコードの作成

Object Storage サービスで必要な Identity サービスのレコードを作成して設定します。これらのエントリーは、Object Storage サービスに対する認証を提供し、Object Storage サービスによって提供される機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identityサービスサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順4.3 Object Storage サービス用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **swift** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password PASSWORD swift
+----------+----------------------------------+
| Field    | Value                            |
+----------+----------------------------------+
| email    | None                             |
| enabled  | True                             |
| id       | 00916f794cec438ea7f14ee0769e6964 |
| name     | swift                            |
| username | swift                            |
+----------+----------------------------------+
```

PASSWORD は、Object Storageサービスが Identityサービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **services** テナントのコンテキスト内で、**swift** ユーザーと **admin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
swift admin
```

4. **swift** Object Storage サービスのエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name swift \
    --description "Swift Storage Service" \
    object-store
```

5. **swift** エンドポイントエントリーを作成します。

```
[(keystone_admin)]# openstack endpoint create \
    --publicurl 'http://IP:8080/v1/AUTH_%(tenant_id)s' \
    --adminurl 'http://IP:8080/v1' \
    --internalurl 'http://IP:8080/v1/AUTH_%(tenant_id)s' \
    --region RegionOne \
    swift
```

IP は Object Storage のプロキシサービスをホストするサーバーの IP アドレスまたは完全修飾ドメイン名に置き換えます。

4.4.2. Object Storage サービスのストレージノードの設定

Object Storage サービスは、ファイルシステムにオブジェクトを保管します。これは通常、接続されている複数の物理ストレージデバイス上のファイルシステムです。オブジェクトの保管に使用するデバイスはすべて **ext4** または **XFS** の形式でフォーマットし、**/srv/node/** ディレクトリーの下にマウントする必要があります。また、指定されたノードで実行されるサービスはすべて有効化して、それらに使用するポートを開く必要があります。

プロキシサービスは、他のサービスとともに実行することが可能ですが、以下の手順ではプロキシサービスは対象外となっています。

手順4.4 Object Storage サービスのストレージノードの設定

1. **ext4** または **XFS** のファイルシステムでデバイスをフォーマットします。**xattr** を必ず有効化してください。

2. **/etc/fstab** ファイルにデバイスを追加して、ブート時には **/srv/node/** の下にマウントされるようにします。**blkid** コマンドを使用して、デバイスの一意 ID を検索して、この一意の ID を使用してデバイスをマウントします。

   注記

   ext4 を使用する場合には、**user_xattr** オプションを指定してファイルシステムをマウントすることにより、拡張属性を有効化するようにしてください (**XFS** の場合は、拡張属性はデフォルトで有効化されます)。

3. 各ノードで実行中の各サービスが使用する TCP ポートを開くようにファイアウォールを設定します。サービスデフォルトでは、アカウントサービスはポート 6202、コンテナーサービスはポート 6201、オブジェクトサービスはポート 6200 を使用します。

   a. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

b. アカウント、コンテナー、オブジェクトのサービスが使用するポートで TCP トラフィックを許可する **INPUT** ルールを追加します。この新規ルールは、**reject-with icmp-host-prohibited** よりも前に記載する必要があります。

```
-A INPUT -p tcp -m multiport --dports 6200,6201,6202,873 -j
ACCEPT
```

c. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

d. **iptables** サービスを再起動して、ファイアウォールの変更を有効にします。

```
# systemctl restart iptables.service
```

4. **/srv/node/** のコンテンツの所有者を **swift:swift** に変更します。

```
# chown -R swift:swift /srv/node/
```

5. **/srv/node/** 配下の全ディレクトリーの **SELinux** コンテキストを正しく設定します。

```
# restorecon -R /srv
```

6. **/etc/swift/swift.conf** ファイルにハッシュプレフィックスを追加します。

```
# openstack-config --set /etc/swift/swift.conf swift-hash
swift_hash_path_prefix \
   $(openssl rand -hex 10)
```

7. **/etc/swift/swift.conf** ファイルにハッシュサフィックスを追加します。

```
# openstack-config --set /etc/swift/swift.conf swift-hash
swift_hash_path_suffix \
   $(openssl rand -hex 10)
```

8. ストレージサービスがリッスンする IP アドレスを設定します。Object Storage クラスター内の全ノードにある全サービスに対して以下のコマンドを実行します。

```
# openstack-config --set /etc/swift/object-server.conf \
   DEFAULT bind_ip NODE_IP_ADDRESS
# openstack-config --set /etc/swift/account-server.conf \
   DEFAULT bind_ip NODE_IP_ADDRESS
# openstack-config --set /etc/swift/container-server.conf \
   DEFAULT bind_ip NODE_IP_ADDRESS
```

NODE_IP_ADDRESS は、設定するノードの IP アドレスに置き換えます。

9. 現在設定中のノードから全 Object Storage サービスノードに **/etc/swift/swift.conf** をコピーします。

重要

`/etc/swift/swift.conf` ファイルは、すべての Object Storage サービスノードで全く同じである必要があります。

10. ノードで実行するサービスを起動します。

```
# systemctl start openstack-swift-account.service
# systemctl start openstack-swift-container.service
# systemctl start openstack-swift-object.service
```

11. サービスがブート時に起動するように設定します。

```
# systemctl enable openstack-swift-account.service
# systemctl enable openstack-swift-container.service
# systemctl enable openstack-swift-object.service
```

4.4.3. Object Storage サービスのプロキシーサービスの設定

Object Storage のプロキシーサービスは、**`gets`** および **`puts`** の転送先のノードを決定します。

アカウント、コンテナー、オブジェクトのサービスは、プロキシーサービスと並行して実行することが可能ですが、以下の手順ではプロキシーサービスのみについて説明します。

注記

Object Storage サービスに組み込まれている SSL 機能は、主にテストを目的としており、実稼働環境での使用はお勧めできません。Red Hat は 実稼働環境のクラスターには SSL 接続の終了にロードバランサーを使用することを推奨します。

手順4.5 Object Storage サービスのプロキシサービスの設定

1. 適切なサービスユーザーの正しい認証情報でプロキシーサーバーの設定ファイルを更新します。

```
# openstack-config --set /etc/swift/proxy-server.conf \
     filter:authtoken auth_host IP
# openstack-config --set /etc/swift/proxy-server.conf \
     filter:authtoken admin_tenant_name services
# openstack-config --set /etc/swift/proxy-server.conf \
     filter:authtoken admin_user swift
# openstack-config --set /etc/swift/proxy-server.conf \
     filter:authtoken admin_password PASSWORD
```

以下の値を置き換えてください。

- *IP* は、Identity サーバーの IP アドレスまたはホスト名に置き換えます。
- *services* は、Object Storage サービス用に作成されたテナントの名前に置き換えます (上記の例では、この値を **`services`** に設定)。
- *swift* は、Object Storage サービス用に作成されたサービスユーザーの名前に置き換えます (上記の例では、この値を **`swift`** に設定)。

- *PASSWORD* は、サービスユーザーに関連付けられたパスワードに置き換えます。

2. **memcached** および **openstack-swift-proxy** サービスを起動します。

```
# systemctl start memcached.service
# systemctl start openstack-swift-proxy.service
```

3. **memcached** および **openstack-swift-proxy** サービスがブート時に起動するように設定します。

```
# systemctl enable memcached.service
# systemctl enable openstack-swift-proxy.service
```

4. Object Storage プロキシーサービスをホストするサーバーへの受信接続を許可します。テキストエディターで **/etc/sysconfig/iptables** ファイルを開き、ポート 8080 の TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載するようにしてください。

```
-A INPUT -p tcp -m multiport --dports 8080 -j ACCEPT
```

重要

上記のルールにより、全リモートホストから Swift プロキシーを実行するシステムへの通信がポート **8080** で許可されます。より制限の厳しいファイアウォールルールの作成についての説明は、『Red Hat Enterprise Linux セキュリティーガイド』を参照してください。

https://access.redhat.com/site/documentation/ja-JP/Red_Hat_Enterprise_Linux/

5. **iptables** サービスを再起動して、変更を有効にします。

```
# systemctl restart iptables.service
```

4.4.4. Object Storage サービスのリング

リングは、ストレージノードのクラスター内でデータが格納される場所を決定します。リングファイルは、**swift-ring-builder** ツールを使用して生成されます。必要なリングファイルは 3 つで、それぞれ **オブジェクト、コンテナー、アカウント** のサービスが対象です。

クラスター内の各ストレージデバイスは、パーティション分割されます。推奨されるパーティション数は 1 デバイスあたり 100 です。**partition power** (パーティションのべき乗) として知られる、パーティションディレクトリーへのファイルシステムパスの MD5 ハッシュから設定可能なビット数は、そのデバイスのパーティション指数として使用されます。1000 のデバイスがあるクラスターで、各デバイスが 100 パーティションに分かれている場合に、**partition count** (パーティション数) は 100 000 です。

partition count は partition power の計算に使用され、partition power を 2 で累乗した値が partition count となります。partition power が小数の場合には切り上げられます。partition count が 100 000 の場合には、その partition power は 17 となります (16.610 から切り上げした値)。数学的には、$2^{\text{partition power}}$ と表記します。

4.4.5. Object Storage サービスのリングファイルの構築

リングファイルは、Object Storage サービスに保管されているオブジェクトのトラッキング用に 1 つ、オブジェクトが配置されているコンテナーのトラッキング用に 1 つ、どのコンテナーにどのアカウントがアクセスできるかをトラッキングするのに 1 つ、合計で 3 つ作成する必要があります。リングファイルは、特定のデータが保管されている場所を推定するのに使用されます。

リングファイルは、パーティションのべき乗、レプリカ数、ゾーン、パーティションの再割り当て間隔の 4 つのパラメーターを使用することで生成されます。

表4.1 リングファイルの構築に使用されるパラメーター

リングファイルのパラメーター	説明
part_power	$2^{\text{partition power}}$ = partition count. パーティション数は、計算後に切り上げ
replica_count	クラスター内でデータが複製される回数
min_part_hours	パーティションが移動できるまでの最小時間。このパラメーターは、min_part_hours で指定された時間内に 1 つのデータ項目のコピーを複数移動しないようにすることで、データの可用性を向上させます。
zone	デバイスをリングに追加する際に使用されます (任意)。ゾーンは、柔軟な抽象化です。特定のデプロイメント内では、各ゾーンを他のゾーンから可能な限り分離する必要があります。ゾーンを使用してサイト、キャビネット、ノードに加えて、デバイスまでも示すことができます。

手順4.6 Object Storage サービスのリングファイルの構築

1. サービスごとに 1 リングを構築します。ビルダーファイル、*partition power*、*レプリカ数*、および *パーティション再割り当ての最小間隔* を指定します。

```
# swift-ring-builder /etc/swift/object.builder create part_power
replica_count min_part_hours
# swift-ring-builder /etc/swift/container.builder create part_power
replica_count min_part_hours
# swift-ring-builder /etc/swift/account.builder create part_power
replica_count min_part_hours
```

2. リングが作成されたら、account リングにデバイスを追加します。

```
# swift-ring-builder /etc/swift/account.builder add
zX-SERVICE_IP:6202/dev_mountpt part_count
```

以下の値を置き換えてください。

- *X* は、指定したゾーンに対応する整数に置き換えます (例: **z1** はゾーン 1 に対応)。
- *SERVICE_IP* は、アカウント、コンテナー、オブジェクトのサービスがリッスンする必要のある IP アドレスに置き換えます。IP は、Object Storage サービスのストレージノードの設定中に指定した **bind_ip** の値と一致する必要があります。

- *dev_mountpt* は、デバイスがマウントされる **/srv/node** のサブディレクトリーに置き換えます。

- *part_count* は、partition power (パーティションのべき乗) の計算に使用した partition count (パーティション数) に置き換えます。

注記

上記の手順は、リングに追加する (クラスター内の各ノード上の) デバイスごとに繰り返してください。

3. container と object のリングの両方にデバイスを追加します。

```
# swift-ring-builder /etc/swift/container.builder add
zX-SERVICE_IP:6201/dev_mountpt part_count
# swift-ring-builder /etc/swift/object.builder add
zX-SERVICE_IP:6200/dev_mountpt part_count
```

変数は前のステップで使用したのと同じ値に置き換えます。

注記

上記のコマンドは、リングに追加する (クラスター内の各ノード上の) デバイスごとに繰り返してください。

4. リング内の複数のデバイスにパーティションを分散します。

```
# swift-ring-builder /etc/swift/account.builder rebalance
# swift-ring-builder /etc/swift/container.builder rebalance
# swift-ring-builder /etc/swift/object.builder rebalance
```

5. **/etc/swift** ディレクトリーにリングファイルが 3 つあるかどうかを確認します。次のコマンドを実行してください。

```
# ls /etc/swift/*gz
```

これらのファイルは以下のように表示されるはずです。

```
/etc/swift/account.ring.gz  /etc/swift/container.ring.gz
/etc/swift/object.ring.gz
```

6. **openstack-swift-proxy** サービスを再起動します。

```
# systemctl restart openstack-swift-proxy.service
```

7. 前の手順で作成したばかりのファイルを含む、**/etc/swift/** ディレクトリー内の全ファイルの所有権を **root** ユーザーと **swift** グループに設定します。

重要

マウントポイントはすべて **root** が所有し、マウント済みファイルシステムの全 root は **swift** が所有する必要があります。以下のコマンドを実行する前に、すべてのデバイスがすでにマウント済みで、それらを **root** が所有していることを確認してください。

```
# chown -R root:swift /etc/swift
```

8. クラスター内の各ノードに各リングビルダーファイルをコピーして、 **/etc/swift/** 配下に保管します。

4.5. OBJECT STORAGE サービスのインストールの検証

Object Storage サービスのインストールおよび設定後には、そのインストールや設定を検証する必要があります。以下の手順は、プロキシーサービスをホストするサーバーまたは、**keystonerc_admin** ファイルをコピーして python-swiftclient パッケージをインストールするマシン上で実行する必要があります。

手順4.7 Object Storage サービスのインストールの検証

1. プロキシーサーバーノード上でデバッグレベルのロギングを有効化します。

```
# openstack-config --set /etc/swift/proxy-server.conf DEFAULT
log_level debug
```

2. **rsyslog** サービスおよび **openstack-swift-proxy** サービスを再起動します。

```
# systemctl restart rsyslog.service
# systemctl restart openstack-swift-proxy.service
```

3. Keystone 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 ~/keystonerc_admin
```

4. プロキシーサーバーに接続できることを確認します。

```
[(keystone_admin)]# swift list
 Message from syslogd@example-swift-01 at Jun 14 02:46:00 ...
 135 proxy-server Server reports support for api versions: v3.0,
v2.0
```

5. Object Storage サービスノードへファイルをアップロードします。

```
[(keystone_admin)]# head -c 1024 /dev/urandom > data1.file ; swift
upload c1 data1.file
[(keystone_admin)]# head -c 1024 /dev/urandom > data2.file ; swift
upload c1 data2.file
[(keystone_admin)]# head -c 1024 /dev/urandom > data3.file ; swift
upload c1 data3.file
```

6. Object Storage サービスのクラスターに格納されているオブジェクトを一覧表示します。

```
[(keystone_admin)]# swift list
[(keystone_admin)]# swift list c1
data1.file
data2.file
data3.file
```

第5章 IMAGE サービスのインストール

5.1. IMAGE サービスの要件

Image サービスをインストールするには、以下の認証情報と情報にアクセスできる必要があります。

- MariaDB データベースサービスをホストするサーバーの IP アドレスと root の認証情報
- Identity サービスの管理者権限の認証情報およびエンドポイントの URL

OpenStack Object Storage サービスをストレージバックエンドとして使用する場合には、サービスのエンドポイントの公開 URL が必要となります。このエンドポイントは、「Object Storage サービス用のアイデンティティーレコードの作成」の手順の一環として設定されます。

5.2. IMAGE サービスのパッケージのインストール

OpenStack Image サービスには、以下のパッケージが必要です。

openstack-glance

OpenStack Image サービスを提供します。

openstack-utils

設定ファイルの編集をはじめとする数々のタスクに役立つサポートユーティリティーを提供します。

openstack-selinux

OpenStack 固有の SELinux ポリシーモジュールを提供します。

パッケージをインストールします。

```
# yum install -y openstack-glance openstack-utils openstack-selinux
```

5.3. IMAGE サービスのデータベースの作成

Image サービスが使用するデータベースとデータベースユーザーを作成します。以下の手順はすべて、データベースサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順5.1 Image サービスのデータベースの作成

1. データベースサービスに接続します。

   ```
   # mysql -u root -p
   ```

2. **glance** データベースを作成します。

   ```
   mysql> CREATE DATABASE glance;
   ```

3. **glance** データベースユーザーを作成し、**glance** データベースへのアクセスを許可します。

   ```
   mysql> GRANT ALL ON glance.* TO 'glance'@'%' IDENTIFIED BY
   ```

```
'PASSWORD';
mysql> GRANT ALL ON glance.* TO 'glance'@'localhost' IDENTIFIED BY
'PASSWORD';
```

PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. データベースの特権をフラッシュして、設定が即時に反映されるようにします。

```
mysql> FLUSH PRIVILEGES;
```

5. **mysql** クライアントを終了します。

```
mysql> quit
```

5.4. IMAGE サービスの設定

Image サービスを設定するには、以下のタスクを完了しておく必要があります。

- Image サービスの認証のための Identity サービスの設定 (データベースエントリーの作成、接続文字列の設定、設定ファイルの更新)
- ディスクイメージストレージバックエンドの設定 (本ガイドでは Object Storage サービスを使用)
- Image サービスへのアクセスのためのファイアウォール設定
- TLS/SSL の設定
- Image サービスデータベースへのデータ投入

5.4.1. Image サービスのデータベース接続の設定

Image サービスによって使用されるデータベース接続文字列は、**/etc/glance/glance-api.conf** および **/etc/glance/glance-registry.conf** のファイルで定義されます。サービスを起動する前に、有効なデータベースサーバーをポイントするように更新しておく必要があります。

以下の手順に記載するステップはすべて、Image サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順5.2 Image サービスの SQL データベース接続の設定

1. **glance-api.conf** ファイルで **sql_connection** の設定キーの値を設定します。

```
# openstack-config --set /etc/glance/glance-api.conf \
   DEFAULT sql_connection mysql://USER:PASS@IP/DB
```

以下の値を置き換えてください。

   - *USER* は、Image サービスのデータベースのユーザー名 (通常は **glance**) に置き換えます。
   - *PASS* は選択したデータベースユーザーのパスワードに置き換えます。

- *IP* は、データベースサービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。
- *DB* は、Image サービスのデータベースの名前 (通常は **glance**) に置き換えます。

2. **glance-registry.conf** ファイルで **sql_connection** の設定キーの値を設定します。

```
# openstack-config --set /etc/glance/glance-registry.conf \
   DEFAULT sql_connection mysql://USER:PASS@IP/DB
```

USER、*PASS*、*IP*、*DB* は、上記のステップで使用した値と同じ値に置き換えます。

重要

この接続設定キーに指定する IP アドレスまたはホスト名は、Image サービスのデータベースの作成時に Image サービスのデータベースユーザーがアクセスを許可された IP アドレスまたはホスト名と一致する必要があります。また、データベースがローカルでホストされ、Image サービスのデータベースの作成時に「localhost」へのアクセス権を付与した場合には、「localhost」と入力する必要があります。

5.4.2. Image サービス用のアイデンティティーレコードの作成

Image サービスで必要な Identity サービスのレコードを作成して設定します。これらのエントリーは、Image サービスによって提供されるボリューム機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identity サービスのサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順5.3 Image サービス用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **glance** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password PASSWORD glance
+----------+----------------------------------+
| Field    | Value                            |
+----------+----------------------------------+
| email    | None                             |
| enabled  | True                             |
| id       | b1f665b15a7943ccb4668c9e78e98a7c |
| name     | glance                           |
| username | glance                           |
+----------+----------------------------------+
```

PASSWORD は、Image サービスが Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **services** テナントのコンテキスト内で、**glance** ユーザーと **admin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
glance admin
```

4. **glance** の Image サービスのエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name glance \
        --description "Glance Image Service" \
        image
```

5. **glance** エンドポイントエントリーを作成します。

```
[(keystone_admin)]#openstack endpoint create \
        --publicurl 'http://IP:9292' \
        --adminurl 'http://IP:9292' \
        --internalurl 'http://IP:9292' \
        --region RegionOne \
        glance
```

IP は、Image サービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

5.4.3. Image サービスの認証の設定

Image サービスが認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、Image サービスをホストする各システムに **root** ユーザーとしてログインして実行する必要があります。

手順5.4 Image サービスが Identity サービスを使用して認証を行うための設定

1. **glance-api** サービスを設定します。

```
# openstack-config --set /etc/glance/glance-api.conf \
   paste_deploy flavor keystone
# openstack-config --set /etc/glance/glance-api.conf \
   keystone_authtoken auth_host IP
# openstack-config --set /etc/glance/glance-api.conf \
   keystone_authtoken auth_port 35357
# openstack-config --set /etc/glance/glance-api.conf \
   keystone_authtoken auth_protocol http
# openstack-config --set /etc/glance/glance-api.conf \
   keystone_authtoken admin_tenant_name services
# openstack-config --set /etc/glance/glance-api.conf \
   keystone_authtoken admin_user glance
# openstack-config --set /etc/glance/glance-api.conf \
   keystone_authtoken admin_password PASSWORD
```

2. **glance-registry** サービスを設定します。

```
# openstack-config --set /etc/glance/glance-registry.conf \
```

```
   paste_deploy flavor keystone
# openstack-config --set /etc/glance/glance-registry.conf \
   keystone_authtoken auth_host IP
# openstack-config --set /etc/glance/glance-registry.conf \
   keystone_authtoken auth_port 35357
# openstack-config --set /etc/glance/glance-registry.conf \
   keystone_authtoken auth_protocol http
# openstack-config --set /etc/glance/glance-registry.conf \
   keystone_authtoken admin_tenant_name services
# openstack-config --set /etc/glance/glance-registry.conf \
   keystone_authtoken admin_user glance
# openstack-config --set /etc/glance/glance-registry.conf \
   keystone_authtoken admin_password PASSWORD
```

以下の値を置き換えてください。

- *IP* は、Identity サーバーの IP アドレスまたはホスト名に置き換えます。
- *services* は、Image サービス用に作成されたテナントの名前に置き換えます (上記の例では、この値を **services** に設定)。
- *glance* は、Image サービス用に作成されたサービスユーザーの名前に置き換えます (上記の例では、この値を **glance** に設定)。
- *PASSWORD* は、サービスユーザーに関連付けられたパスワードに置き換えます。

5.4.4. Object Storage サービスをイメージの保管に使用する方法

デフォルトでは、Image サービスはストレージバックエンドにローカルファイルシステム (**file**) を使用しますが、アップロードしたディスクイメージを保管するには、次にあげるいずれかのストレージバックエンドを使用することができます。

- **file**: イメージサーバーのローカルファイルシステム (**/var/lib/glance/images/** ディレクトリー)
- **swift**: OpenStack Object Storage サービス

注記

以下の設定手順では、**openstack-config** コマンドを使用します。ただし、**/etc/glance/glance-api.conf** ファイルを手動で更新することもできます。このファイルを手動で更新する場合は、**default_store** パラメーターが正しいバックエンドに設定されていることを確認し (例: '**default_store=rbd**')、バックエンドのセクションのパラメーターを更新します ('**RBD Store Options**' のセクション)。

手順5.5 Image サービスが Object Storage サービスを使用するように設定する手順

1. **default_store** 設定キーを **swift** に設定します。

```
# openstack-config --set /etc/glance/glance-api.conf \
   DEFAULT default_store swift
```

2. **swift_store_auth_address** 設定キーを Identity サービスのパブリックエンドポイントに設定します。

```
# openstack-config --set /etc/glance/glance-api.conf \
   DEFAULT swift_store_auth_address http://IP:5000/v2.0/
```

3. Object Storage サービスでイメージを保管するコンテナーを追加します。

```
# openstack-config --set /etc/glance/glance-api.conf \
   DEFAULT swift_store_create_container_on_put True
```

4. **swift_store_user** 設定キーは、認証に使用するテナントとユーザーが含まれるように、*TENANT*:*USER*: の形式で設定します。

```
# openstack-config --set /etc/glance/glance-api.conf \
   DEFAULT swift_store_user services:swift
```

   - 本ガイドの手順に従って Object Storage をデプロイする場合には (上記のコマンド例に記載されているように)、上記の値をそれぞれ **services** テナントと **swift** ユーザーに置き換えてください。
   - 本ガイドの手順には従わずに Object Storage をデプロイする場合には、上記の値はその環境の適切な Object Storage テナントとユーザーに置き換える必要があります。

5. **swift_store_key** 設定キーを Object Storage サービスのデプロイ時に **swift** ユーザー用に指定したパスワードに設定します。

```
# openstack-config --set /etc/glance/glance-api.conf \
   DEFAULT swift_store_key PASSWORD
```

5.4.5. Image サービスのトラフィックを許可するためのファイアウォール設定

Image サービスは、ポート **9292** 経由でネットワークにアクセスできるようにします。以下の手順に記載するステップはすべて、Image サービスをホストするサーバーに **root** ユーザーとしてログインして実行してください。

手順5.6 Image サービスのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/glance/glance-api.conf** ファイルを開き、以下のパラメーターの前に付いているコメント文字を削除します。

```
bind_host = 0.0.0.0
bind_port = 9292
```

2. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

3. ポート **9292** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

```
-A INPUT -p tcp -m multiport --dports 9292 -j ACCEPT
```

4. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

5. **iptables** サービスを再起動して、変更を有効にします。

```
# systemctl restart iptables.service
```

5.4.6. Image サービスのための RabbitMQ メッセージブローカーの設定

RabbitMQ はデフォルト (かつ推奨の) メッセージブローカーです。RabbitMQ メッセージングサービスは、rabbitmq-server パッケージにより提供されます。以下の手順で記載する全ステップは、Image サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順5.7 Image サービス (glance) が RabbitMQ メッセージブローカーを使用するための設定

1. RabbitMQ を通知機能として設定します。

   ```
   # openstack-config --set /etc/glance/glance-api.conf \
      DEFAULT notification_driver messaging
   ```

2. RabbitMQ のホスト名を設定します。

   ```
   # openstack-config --set /etc/glance/glance-api.conf \
      DEFAULT rabbit_host RABBITMQ_HOST
   ```

   RABBITMQ_HOST は、メッセージブローカーの IP アドレスまたはホスト名に置き換えます。

3. メッセージブローカーのポートを **5672** に設定します。

   ```
   # openstack-config --set /etc/glance/glance-api.conf \
      DEFAULT rabbit_port 5672
   ```

4. RabbitMQ の設定時に Image サービス用に作成した RabbitMQ ユーザー名とパスワードを設定します。

   ```
   # openstack-config --set /etc/glance/glance-api.conf \
      DEFAULT rabbit_userid glance
   # openstack-config --set /etc/glance/glance-api.conf \
      DEFAULT rabbit_password GLANCE_PASS
   ```

   glance および *GLANCE_PASS* は、Image サービス用に作成した RabbitMQ ユーザー名とパスワードに置き換えます。

5. RabbitMQ の起動時に、**glance** ユーザーに全リソースに対するパーミッションが付与されます。このアクセスは、特別に仮想ホスト **/** を介して行われます。Image サービスがこの仮想ホストに接続されるように設定します。

   ```
   # openstack-config --set /etc/glance/glance-api.conf \
      DEFAULT rabbit_virtual_host /
   ```

5.4.7. Image サービスが SSL を使用するための設定

glance-api.conf ファイルで、以下のオプションを使用して SSL を設定します。

表5.1 Image サービスの SSL オプション

設定オプション	説明
cert_file	API サーバーをセキュアに起動する際に使用する証明書ファイルへのパス
key_file	API サーバーをセキュアに起動する際に使用する秘密鍵ファイルへのパス
ca_file	クライアントの接続を検証するのに使用する CA 証明書ファイルへのパス

5.4.8. Image サービスのデータベースへのデータ投入

Image サービスのデータベース接続文字列を適切に設定した後には、Identiy サービスのデータベースにデータを投入します。

手順5.8 Image サービスのデータベースへのデータ投入

1. Image サービスをホストするシステムにログインします。

2. **glance** ユーザーに切り替えます。

```
# su glance -s /bin/sh
```

3. **/etc/glance/glance-api.conf** および **/etc/glance/glance-registry.conf** で特定されているデータベースを初期化し、データを投入します。

```
$ glance-manage db_sync
```

5.4.9. ローカルファイルシステムを介したイメージのロードの有効化

デフォルトでは、Image サービスは HTTP プロトコルを使用してイメージをインスタンスに提供します。具体的には、イメージデータは、イメージストアからコンピュートノードのローカルディスクに HTTP を介して伝送されます。このプロセスは、Image サービスと Compute サービスが別々のホストにインストールされたデプロイメントの大半が対象であるため一般的なプロセスです。

注記

Image サービスと Compute サービスが同じホストにはインストールされていなくても、それらのサービスが共有ファイルシステムを共有している場合は、直接イメージへアクセスすることができます。この場合は、そのファイルシステムを同じ場所にマウントする必要があります。

両サービスが同じホストにインストールされた (その結果、同じファイルシステムを共有する) デプロイの場合には、HTTP のステップを完全にスキップした方がより効率的です。その代わりに、ローカルファイルシステムを介してイメージの送受信をするように Image サービスと Compute サービスの両方を設定する必要があります。

この手順で生成される Image のファイルシステムメタデータは、新規イメージにのみ適用されます。既存のイメージは、このメタデータを使用しません。

手順5.9 Image サービスと Compute サービスがローカルファイルシステムでイメージを送受信するための設定

1. **openstack-nova-compute** に要求される Image のファイルシステムメタデータを公開するための JSON ドキュメントを作成します。

2. Image サービスが JSON ドキュメントを使用するように設定します。

3. **openstack-nova-compute** が Image サービスによって提供されるファイルシステムメタデータを使用するように設定します。

5.4.9.1. Image サービスがローカルファイルシステムを介してイメージを提供するための設定

HTTP ではなく、ローカルファイルシステムを使用したイメージのロードを有効にするには、最初に Image サービスがローカルファイルシステムメタデータを **openstack-nova-compute** サービスに公開する必要があります。この操作は以下の手順に従って実行します。

手順5.10 Compute サービスが Compute サービスに対してローカルファイルシステムメタデータを公開するための設定

1. Image サービスが使用するファイルシステムのマウントポイントを特定します。

   ```
   # df
   Filesystem     1K-blocks     Used Available Use% Mounted on
   /dev/sda3       51475068 10905752  37947876  23% /
   devtmpfs         2005504        0   2005504   0% /dev
   tmpfs            2013248      668   2012580   1% /dev/shm
   ```

   たとえば、Image サービスが **/dev/sda3** ファイルシステムを使用する場合には、対応するマウントポイントは **/** です。

2. 以下のコマンドで、マウントポイントの一意識別子を作成します。

   ```
   # uuidgen
   ad5517ae-533b-409f-b472-d82f91f41773
   ```

   uuidgen の出力の内容をメモしておきます。この情報は、次のステップで使用します。

3. **.json** の拡張子でファイルを作成します。

4. テキストエディターでファイルを開き、以下の情報を追加します。

   ```
   {
   "id": "UID",
   "mountpoint": "MOUNTPT"
   }
   ```

   以下の値を置き換えてください。

   - *UID* は、前のステップで作成した一意識別子に置き換えます。
   - *MOUNTPT* は、Image サービスのファイルシステムのマウントポイント (最初のステップで特定したマウントポイント) に置き換えます。

5. Image サービスが JSON ファイルを使用するように設定します。

   ```
   # openstack-config --set /etc/glance/glance-api.conf \
   ```

```
 DEFAULT show_multiple_locations True
# openstack-config --set /etc/glance/glance-api.conf \
 DEFAULT filesystem_store_metadata_file JSON_PATH
```

JSON_PATH は、JSON ファイルへの完全なパスに置き換えます。

> **重要**
>
> 適切なポリシー設定を行わずに設定した場合には、Image サービスの管理者以外のユーザーでもアクティブなイメージデータを置き換えることができてしまいます (他のユーザーへの通知なしに現在のイメージが切り替えられる可能性があります)。設定情報に関する OSSN 通知 (推奨のアクション) は https://wiki.openstack.org/wiki/OSSN/OSSN-0065 を参照してください。

6. Image サービスを再起動します (すでに実行されていない場合)。

```
# systemctl restart openstack-glance-registry.service
# systemctl restart openstack-glance-api.service
```

この手順で生成される Image のファイルシステムメタデータは、新規イメージにのみ適用されます。既存のイメージは、このメタデータを使用しません。

5.4.9.2. Compute サービスがローカルファイルシステムメタデータを使用するための設定

Image サービスがファイルシステムメタデータを公開するように設定した後には、そのメタデータを Compute サービスが使用するように設定することができます。この操作により **openstack-nova-compute** がローカルファイルシステムからイメージをロードできるようになります。

手順5.11 Compute サービスが Image サービスによって提供されるローカルファイルシステムメタデータを使用するための設定

1. **file://** スキームを使ったダイレクト URL の使用を有効にするように **openstack-nova-compute** を設定します。

```
# openstack-config --set /etc/nova/nova.conf \
 DEFAULT allowed_direct_url_schemes file
```

2. Image サービスのファイルシステム用のエントリーを作成します。

```
# openstack-config --set /etc/nova/nova.conf \
 image_file_url filesystems FSENTRY
```

FSENTRY は Image サービスのファイルシステムに割り当てる名前に置き換えます。

3. Image サービスがローカルファイルシステムメタデータを公開するために使用する **.json** ファイルを開きます。次のステップでは、このファイルに記載されている情報を使用します。

4. Image サービスによって公開されるファイルシステムメタデータにエントリーを関連付けます。

```
# openstack-config --set /etc/nova/nova.conf \
 image_file_url:FSENTRY id UID
# openstack-config --set /etc/nova/nova.conf \
```

```
image_file_url:FSENTRY mountpoint MOUNTPT
```

以下の値を置き換えてください。

- *UID* は、Image サービスが使用する一意識別子に置き換えます。Image サービスによって使用される **.json** ファイルでは、*UID* は **"id":** の値です。
- *MOUNTPT* は Image サービスのファイルシステムが使用するマウントポイントに置き換えます。Image サービスが使用する **.json** ファイルでは、*MOUNTPT* は **"mountpoint":** の値です。

5.5. イメージの API およびレジストリーサービスの起動

Glance の設定後には、**glance-api** および **glance-registry** サービスを起動して、各サービスがブート時に起動するように設定します。

```
# systemctl start openstack-glance-registry.service
# systemctl start openstack-glance-api.service
# systemctl enable openstack-glance-registry.service
# systemctl enable openstack-glance-api.service
```

5.6. IMAGE サービスインストールの検証

本項では、ディスクイメージを Image サービスにアップロードする際に必要な手順を説明します。このイメージは、OpenStack の環境で仮想マシンを起動するためのベースとして使用することができます。

5.6.1. テストディスクイメージの取得

Image サービスへのイメージのインポートをテストする際に使用可能なディスクイメージを Red Hat からダウンロードします。新規イメージは、Red Hat Enterprise Linux 7 の各マイナーリリースで提供され、Red Hat Enterprise Linux の **製品のダウンロード** から入手することができます。

手順5.12 テストディスクイメージのダウンロード

1. https://access.redhat.com に進み、自分のアカウント情報を使用して Red Hat カスタマーポータルにログインします。
2. メニューバーで **ダウンロード** をクリックします。
3. **A-Z** をクリックして、製品のダウンロードをアルファベット順に並べ替えます。
4. **Red Hat Enterprise Linux** をクリックして **製品のダウンロード** ページにアクセスします。
5. **KVM Guest Image** のダウンロードリンクをクリックします。

5.6.2. ディスクイメージのアップロード

Image サービスに保管されているイメージをベースにインスタンスを起動するには、Image サービスに 1 つまたは複数のイメージをあらかじめアップロードしておく必要があります。OpenStack 環境での使用に適したイメージにアクセスできる必要があります。

重要

Linux ベースの仮想マシンイメージはすべて、Image サービスにアップロードする前に **virt-sysprep** コマンドを実行しておくことを推奨します。**virt-sysprep** コマンドは、仮想環境で使用するための準備として、仮想イメージを再初期化します。デフォルトの操作には、SSH キーの削除、永続的な MAC アドレスの削除、ユーザーアカウントの削除などが含まれます。

virt-sysprep コマンドは、Red Hat Enterprise Linux libguestfs-tools パッケージによって提供されます。パッケージをインストールしてディスクイメージを再度初期化します。

```
# yum install -y libguestfs-tools
# virt-sysprep --add FILE
```

特定の操作の有効化/無効化についての説明は、**virt-sysprep** の man ページを参照してください。

手順5.13 Image サービスへのディスクイメージのアップロード

1. 設定済みユーザー (管理者アカウントは必要なし) として keystone にアクセスするシェルを設定します。

```
# source ~/keystonerc_userName
```

2. ディスクイメージをインポートします。

```
[(keystone_userName)]# glance image-create --name "NAME" \
        --is-public IS_PUBLIC \
        --disk-format DISK_FORMAT \
        --container-format CONTAINER_FORMAT \
        --file IMAGE
```

以下の値を置き換えてください。

- *NAME* は、ディスクイメージにより参照されるユーザー名に置き換えます。
- *IS_PUBLIC* は **true** または **false** に置き換えます。
  - **true**: 全ユーザーがイメージを表示/使用することができます。
  - **false**: 管理者のみがイメージを表示/使用することができます。
- *DISK_FORMAT* には、ディスクイメージの形式を指定します。有効な値には、**aki**、**ami**、**ari**、**iso**、**qcow2**、**raw**、**vdi**、**vhd**、**vmdk** が含まれます。仮想マシンディスクイメージの形式が不明な場合には、**qemu-img info** コマンドを使用してその形式の特定を試みます。
- *CONTAINER_FORMAT* には、イメージのコンテナーの形式を指定します。イメージに関連した追加のメタデータが含まれる **ovf** や **ami** などのファイル形式でイメージがパッケージされていない限りは、コンテナーの形式は **bare** となります。
- *IMAGE* は (アップロード用の) イメージファイルへのローカルパスに置き換えます。アップロードするイメージがローカルではアクセスできないが、リモートの URL でアクセスでき

る場合には、**--file** パラメーターの代わりに **--location** パラメーターで URL を指定します。イメージをオブジェクトストアにコピーするには **--copy-from** 引数も指定する必要があります。この引数を指定しない場合は、毎回必要に応じてリモートからイメージへアクセスされます。

glance image-create の構文についての詳しい情報は、help ページを参照してください。

```
[(keystone_userName)]# glance help image-create
```

上記のコマンドの出力からイメージの一意名をメモしてください。

3. イメージが正常にアップロードされていることを確認します。

```
 [(keystone_userName)]# glance image-show IMAGE_ID
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | 2f81976cae15c16ef0010c51e3a6c163     |
| container_format | bare                                 |
| created_at       | 2013-01-25T14:45:48                  |
| deleted          | False                                |
| disk_format      | qcow2                                |
| id               | 0ce782c6-0d3e-41df-8fd5-39cd80b31cd9 |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | RHEL 6.6                             |
| owner            | b1414433c021436f97e9e1e4c214a710     |
| protected        | False                                |
| size             | 25165824                             |
| status           | active                               |
| updated_at       | 2013-01-25T14:45:50                  |
+------------------+--------------------------------------+
```

IMAGE_ID は、イメージの一意名に置き換えます。

ディスクイメージは、OpenStack 環境で仮想マシンインスタンスを起動するためのベースとして使用することができるようになりました。

第6章 BLOCK STORAGE サービスのインストール

6.1. BLOCK STORAGE サービスのパッケージのインストール

OpenStack Block Storage サービスには以下のパッケージが必要です。

openstack-cinder
: Block Storage サービスおよび関連する設定ファイルを提供します。

openstack-utils
: 設定ファイルの編集をはじめとする数々のタスクに役立つサポートユーティリティーを提供します。

openstack-selinux
: OpenStack 固有の SELinux ポリシーモジュールを提供します。

device-mapper-multipath
: デバイスマッパーでマルチパスデバイスを管理するツールを提供します。Block Storage の操作を正しく行うには、これらのツールが必要です。

パッケージをインストールします。

```
# yum install -y openstack-cinder openstack-utils openstack-selinux
device-mapper-multipath
```

6.2. BLOCK STORAGE サービスのデータベースの作成

Block Storage サービスで使用するデータベースおよびデータベースユーザーを作成します。以下の手順はすべて、データベースサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順6.1 Block Storage サービスのデータベースの作成

1. データベースサービスに接続します。

   ```
   # mysql -u root -p
   ```

2. **cinder** データベースを作成します。

   ```
   mysql> CREATE DATABASE cinder;
   ```

3. **cinder** データベースユーザーを作成し、**cinder** データベースへのユーザーアクセスを許可します。

   ```
   mysql> GRANT ALL ON cinder.* TO 'cinder'@'%' IDENTIFIED BY
   'PASSWORD';
   mysql> GRANT ALL ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY
   'PASSWORD';
   ```

PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. データベースの特権をフラッシュして、設定が即時に反映されるようにします。

```
mysql> FLUSH PRIVILEGES;
```

5. **mysql** クライアントを終了します。

```
mysql> quit
```

6.3. BLOCK STORAGE サービスの設定

6.3.1. Block Storage サービスのデータベース接続の設定

Block Storage サービスによって使用されるデータベース接続文字列は、**/etc/neutron/plugin.ini** ファイルで定義されます。サービスを起動する前に、有効なデータベースサーバーをポイントするように更新しておく必要があります。

Block Storage サービスをホストする各システムの **sql_connection** 設定キーの値を設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT sql_connection mysql://USER:PASS@IP/DB
```

以下の値を置き換えてください。

- *USER* は、Block Storage サービスのデータベースのユーザー名 (通常は **cinder**) に置き換えます。
- *PASS* は選択したデータベースユーザーのパスワードに置き換えます。
- *IP* は、データベースサービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。
- *DB* は、Block Storage サービスのデータベースの名前 (通常は **cinder**) に置き換えます。

重要

この接続設定キーに指定する IP アドレスまたはホスト名は、Block Storage サービスのデータベースの作成時に Block Storage サービスのデータベースユーザーがアクセスを許可された IP アドレスまたはホスト名と一致する必要があります。また、データベースがローカルでホストされ、Block Storage サービスのデータベースの作成時に「localhost」へのアクセス権を付与した場合には、「localhost」と入力する必要があります。

6.3.2. Block Storage サービス用のアイデンティティーレコードの作成

Block Storage サービスで必要な Identity サービスのレコードを作成して設定します。これらのエントリーは、Block Storage サービスに対する認証を提供し、Block Storage が提供するボリューム機能を探してアクセスを試みる他の OpenStack サービスを誘導します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identity サービスのサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順6.2 Block Storage サービス用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **cinder** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password PASSWORD cinder
+----------+----------------------------------+
| Field    | Value                            |
+----------+----------------------------------+
| email    | None                             |
| enabled  | True                             |
| id       | cb4dafc849df4ea180e9e29346a29038 |
| name     | cinder                           |
| username | cinder                           |
+----------+----------------------------------+
```

PASSWORD は、Block Storage サービスが Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **services** テナントのコンテキスト内で、**cinder** ユーザーと **admin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
cinder admin
```

4. **cinder** および **cinderv2** Block Storage のサービスエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name cinder \
        --description "Cinder Volume Service" \
        volume
```

5. **cinder** エンドポイントエントリーを作成します。

```
[(keystone_admin)]# openstack endpoint create \
   --publicurl 'http://IP:8776/v1/%(tenant_id)s' \
   --adminurl 'http://IP:8776/v1/%(tenant_id)s' \
   --internalurl 'http://IP:8776/v1/%(tenant_id)s'\
   --region RegionOne \
   cinder
```

IP は、Block Storage API サービス (**openstack-cinder-api**) をホストするシステムの IP アドレスまたはホスト名に置き換えます。複数の API サービスインスタンスをインストールして実行するには、各インスタンスの IP アドレスまたはホスト名を使用してこのステップを繰り返します。

6.3.3. Block Storage サービスの認証設定

Block Storage サービスが認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、Block Storage サービスをホストする各サーバーに **root** ユーザーとしてログインして実行する必要があります。

手順6.3 Block Storage サービスが Identity サービスを使用して認証を行うための設定

1. 認証ストラテジーを **keystone** に設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT auth_strategy keystone
```

2. Block Storage サービスが使用する必要のある Identity サービスのホストを設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   keystone_authtoken auth_host IP
```

IP は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

3. Block Storage サービスが正しいテナントとして認証を行うように設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   keystone_authtoken admin_tenant_name services
```

services は、OpenStack Networking を使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

4. Block Storage サービスが **cinder** の管理ユーザーアカウントを使用して認証を行うように設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   keystone_authtoken admin_user cinder
```

5. Block Storage サービスが正しい **cinder** の管理ユーザーアカウントを使用するように設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   keystone_authtoken admin_password PASSWORD
```

PASSWORD は、**cinder** ユーザーの作成時に設定したパスワードに置き換えます。

6.3.4. Block Storage サービスのトラフィックを許可するためのファイアウォール設定

OpenStack 環境内の各コンポーネントは、認証に Idenity サービスを使用するため、このサービスへアクセスできる必要があります。Block Storage サービスをホストするシステムのファイアウォール設定を変更して、これらのポートでのネットワークトラフィックを許可する必要があります。以下の手順に記載するステップはすべて、Block Storage サービスをホストする各システムに **root** ユーザーとしてログインして実行する必要があります。

手順6.4 Block Storage サービスのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

2. このファイルに、ポート **3260** および **8776** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

   ```
   -A INPUT -p tcp -m multiport --dports 3260,8776 -j ACCEPT
   ```

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

4. **iptables** サービスを再起動して、変更を有効にします。

   ```
   # systemctl restart iptables.service
   ```

6.3.5. Block Storage サービスが SSL を使用するための設定

以下に記載する **cinder.conf** ファイル内のオプションを使用して、SSL を設定します。

表6.1 Block Storage の SSL オプション

設定オプション	説明
backlog	ソケットの設定を行うバックログ要求の数
tcp_keepidle	サーバーソケットごとに設定する TCP_KEEPIDLE の値 (秒単位)
ssl_ca_file	クライアントの接続を検証するのに使用する CA 証明書ファイル
ssl_cert_file	サーバーをセキュアに起動する際に使用する証明書ファイル
ssl_key_file	サーバーをセキュアに起動する際に使用する秘密鍵ファイル

6.3.6. Block Storage サービスのための RabbitMQ メッセージブローカーの設定

RabbitMQ はデフォルト (かつ推奨の) メッセージブローカーです。RabbitMQ メッセージングサービスは、rabbitmq-server パッケージにより提供されます。以下の手順で記載する全ステップは、Block Storage サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順6.5 Block Storage サービスが RabbitMQ メッセージブローカーを使用するための設定

1. RPC バックエンドとして RabbitMQ を設定します。

   ```
   # openstack-config --set /etc/cinder/cinder.conf \
      DEFAULT rpc_backend cinder.openstack.common.rpc.impl_kombu
   ```

2. RabbitMQ のホスト名を設定します。

   ```
   # openstack-config --set /etc/cinder/cinder.conf \
      DEFAULT rabbit_host RABBITMQ_HOST
   ```

   RABBITMQ_HOST は、メッセージブローカーの IP アドレスまたはホスト名に置き換えます。

3. メッセージブローカーのポートを **5672** に設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT rabbit_port 5672
```

4. RabbitMQ の設定時に Block Storage サービス用に作成した RabbitMQ ユーザー名とパスワードを設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT rabbit_userid cinder
```

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT rabbit_password CINDER_PASS
```

cinder および *CINDER_PASS* は、Block Storage サービス用に作成された RabbitMQ ユーザー名とパスワードに置き換えます。

5. RabbitMQ の起動時に、**cinder** ユーザーに全リソースに対するパーミッションが付与されます。このアクセスは、特別に仮想ホスト **/** を介して行われます。Block Storage サービスがこの仮想ホストに接続されるように設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT rabbit_virtual_host /
```

6.3.7. Block Storage サービスとメッセージブローカーとの間の SSL 通信の有効化

メッセージブローカーで SSL を有効化した場合は、Block Storage サービスも相応に設定する必要があります。以下の手順では、エクスポートしたクライアントの証明書とキーファイルが必要です。これらのファイルのエクスポートの方法に関する説明は、「クライアント用 SSL 証明書のエクスポート」を参照してください。

1. メッセージブローカーとの SSL 通信を有効化します。

```
# openstack-config --set /etc/cinder/cinder.conf \
 DEFAULT rabbit_use_ssl True
# openstack-config --set /etc/cinder/cinder.conf \
 DEFAULT kombu_ssl_certfile /path/to/client.crt
# openstack-config --set /etc/cinder/cinder.conf \
 DEFAULT kombu_ssl_keyfile /path/to/clientkeyfile.key
```

以下の値を置き換えてください。

- */path/to/client.crt* はエクスポートされたクライアント証明書の絶対パスに置き換えます。
- */path/to/clientkeyfile.key* はエクスポートされたキーファイルの絶対パスに置き換えます。

2. 証明書がサードパーティーの認証局 (CA) によって署名されている場合には、次のコマンドを実行する必要もあります。

```
# openstack-config --set /etc/cinder/cinder.conf \
 DEFAULT kombu_ssl_ca_certs /path/to/ca.crt
```

/path/to/ca.crt は、サードパーティー CA によって提供された CA ファイルの絶対パスに置き換えます (詳細は「RabbitMQ メッセージブローカーでの SSL の有効化」を参照)。

6.3.8. Block Storage データベースへのデータ投入

Block Storage データベース接続文字列を適切に設定した後には、Block Storage データベースにデータを投入します。

重要

この手順は、データベースの初期化とデータ投入のために 1 回のみ実行する必要があります。Block Storage サービスをホストするシステムの追加時には繰り返す必要はありません。

手順6.6 Block Storage サービスのデータベースへのデータ投入

1. Block Storage サービスをホストするシステムにログインします。
2. **`cinder`** ユーザーに切り替えます。

   ```
   # su cinder -s /bin/sh
   ```

3. **`/etc/cinder/cinder.conf`** で特定されているデータベースを初期化し、データを投入します。

   ```
   $ cinder-manage db sync
   ```

6.3.9. Block Storage API サービスのスループットの増加

デフォルトでは、Block Storage API サービス (**`openstack-cinder-api`**) は 1 プロセスで実行されます。これにより、Block Storage サービスが常時処理可能な要求件数が制限されます。実稼働環境では、マシンのキャパシティーが許す限り多くのプロセスで **`openstack-cinder-api`** の実行を許可することによって、Block Storage API のスループットを増加させることをお勧めします。

Block Storage API サービスオプション ***`osapi_volume_workers`*** により、**`openstack-cinder-api`** 向けに起動する API サービスワーカーの数 (または OS プロセス数) を指定することができます。

このオプションを設定するには、**`openstack-cinder-api`** ホストで以下のコマンドを実行します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT osapi_volume_workers CORES
```

CORES はマシン上の CPU コア/スレッド数に置き換えてください。

6.4. ボリュームサービスの設定

6.4.1. Block Storageドライバーのサポート

ボリュームサービス (**`openstack-cinder-volume`**) には、適切なブロックストレージへのアクセスが必要です。Red Hat OpenStack Platform は以下のサポートされているブロックストレージタイプに対してボリュームドライバーを提供しています。

- Red Hat Ceph
- LVM/iSCSI
- ThinLVM
- NFS
- NetApp
- Dell EqualLogic
- Dell Storage Center

https://access.redhat.com/site/documentation/ja/Red_Hat_Enterprise_Linux_OpenStack_Platform

LVM バックエンドの設定手順については、「OpenStack Block Storage で LVM ストレージバックエンドを使用するための設定」を参照してください。

6.4.2. OpenStack Block Storage で LVM ストレージバックエンドを使用するための設定

openstack-cinder-volume は、そのサービスが実行されるサーバーに直接接続されたボリュームグループを利用することができます。このボリュームグループは、Block Storage サービスが専用で使用するように作成し、そのボリュームグループをポイントするように設定を更新する必要があります。

以下の手順に記載するステップはすべて、**openstack-cinder-volume** サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順6.7 openstack-cinder-volume で LVM ストレージをバックエンドとして使用するための設定

1. 物理ボリュームを作成します。

   ```
   # pvcreate DEVICE
     Physical volume "DEVICE" successfully created
   ```

   DEVICE は、有効かつ未使用のデバイスへのパスに置き換えます。以下はその例です。

   ```
   # pvcreate /dev/sdX
   ```

2. ボリュームグループを作成します。

   ```
   # vgcreate cinder-volumes DEVICE
     Volume group "cinder-volumes" successfully created
   ```

   DEVICE は、物理ボリュームの作成時に使用したデバイスへのパスに置き換えます。また、オプションとして、*cinder-volumes* を別の新規ボリュームグループ名に置き換えます。

3. **volume_group** 設定キーを、前のステップで作成したボリュームグループ名に設定します。

   ```
   # openstack-config --set /etc/cinder/cinder.conf \
     DEFAULT volume_group cinder-volumes
   ```

4. **volume_driver** 設定キーを **cinder.volume.drivers.lvm.LVMVolumeDriver** に設定することにより、LVM ストレージへのアクセスに正しいボリュームドライバーが使用されるようにします。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT volume_driver cinder.volume.drivers.lvm.LVMVolumeDriver
```

6.4.3. SCSI ターゲットデーモンの設定

openstack-cinder-volume サービスは、ストレージのマウントに SCSI ターゲットデーモンを使用します。**root** ユーザーとしてログインして、**openstack-cinder-volume** サービスをホストする各サーバーに SCSI ターゲットデーモンをインストールする必要があります。

手順6.8 SCSI ターゲットデーモンの設定

1. targetcli パッケージをインストールします。

```
# yum install targetcli
```

2. **target** デーモンを起動し、ブート時に起動するように設定します。

```
# systemctl start target.service
# systemctl enable target.service
```

3. **lioadm** iSCSI ターゲットのユーザーランドツールを使用するようにボリュームサービスを設定します。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT iscsi_helper lioadm
```

4. iSCSI デーモンがリッスンする必要のある正しい IP アドレスを設定します (*ISCSIIP*)。

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT iscsi_ip_address ISCSIIP
```

   ISCSI_IP は、**openstack-cinder-volume** サービスをホストするサーバーのホスト名または IP アドレスに置き換えます。

6.5. BLOCK STORAGE サービスの起動

Block Storage の機能を有効にするには、3 つのサービスそれぞれのインスタンスを少なくとも 1 つ起動する必要があります。

- API サービス (**openstack-cinder-api**)
- スケジューラーサービス (**openstack-cinder-scheduler**)
- ボリュームサービス (**openstack-cinder-volume**)

これらのサービスは、同じシステムに配置する必要はありませんが、同じメッセージブローカーとデータベースを使用して通信するように設定する必要があります。サービスが稼働すると、API がボリュームの受信要求を受け入れ、スケジューラーがそれらの要求を必要に応じて割り当て、ボリュームサービ

スが処理します。

手順6.9 Block Storage サービスの起動

1. API を実行する予定の各サーバーに **root** ユーザーとしてログインして、API サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-cinder-api.service
# systemctl enable openstack-cinder-api.service
```

2. スケジューラーを実行する予定の各サーバーに **root** ユーザーとしてログインして、スケジューラーサービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-cinder-scheduler.service
# systemctl enable openstack-cinder-scheduler.service
```

3. Block Storage のアタッチ先のサーバーに **root** ユーザとしてログインして、Volume サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-cinder-volume.service
# systemctl enable openstack-cinder-volume.service
```

6.6. BLOCK STORAGE サービスのインストールの検証

6.6.1. Block Storage サービスのインストールのローカルでの検証

ローカルでボリュームを作成/削除して、ブロックストレージのインストールが完了し、使用の準備ができていることを確認します。**root** ユーザーまたは **keystonerc_admin** ファイルにアクセス権があるユーザーとしてログインして、Block Storage API サービスをホストするサーバーに対して、このテストを実行します。続行する前に、**keystonerc_admin** ファイルをこのシステムにコピーします。

手順6.10 Block Storage サービスのインストールのローカルでの検証

1. 管理者ユーザーの識別と認証に使用する環境変数を読み込みます。

```
# source ~/keystonerc_admin
```

2. このコマンドの出力としてエラーが何も返されないことを確認してください。

```
# cinder list
```

3. ボリュームを作成します。

```
# cinder create SIZE
```

SIZE は、作成するボリュームのサイズを指定するギガバイト (GB) 単位の値に置き換えます。

4. ボリュームを削除します。

```
# cinder delete ID
```

ID は、ボリュームの作成時に返された識別子に置き換えます。

6.6.2. Block Storage サービスのインストールのリモートでの検証

リモートのマシンを使用してボリュームを作成/削除して、ブロックストレージのインストールが完了し、使用の準備ができていることを確認します。**root** ユーザーまたは **keystonerc_admin** ファイルにアクセス権があるユーザーとしてログインして、Block Storage API サービスをホストするサーバー以外のサーバーに対して、このテストを実行します。続行する前に、**keystonerc_admin** ファイルをこのシステムにコピーします。

手順6.11 Block Storage サービスのインストールのリモートでの検証

1. python-cinderclient パッケージをインストールします。

   ```
   # yum install -y python-cinderclient
   ```

2. 管理者ユーザーの識別と認証に使用する環境変数を読み込みます。

   ```
   # source ~/keystonerc_admin
   ```

3. このコマンドの出力としてエラーが何も返されないことを確認してください。

   ```
   # cinder list
   ```

4. ボリュームを作成します。

   ```
   # cinder create SIZE
   ```

   SIZE は、作成するボリュームのサイズを指定するギガバイト (GB) 単位の値に置き換えます。

5. ボリュームを削除します。

   ```
   # cinder delete ID
   ```

   ID は、ボリュームの作成時に返された識別子に置き換えます。

第7章 OPENSTACK NETWORKING のインストール

7.1. OPENSTACK NETWORKING パッケージのインストール

OpenStack Networking には、次のパッケージが必要です。

openstack-neutron
: OpenStack Networking および関連する設定ファイルを提供します。

openstack-neutron-ml2
: OpenStack Networking プラグインを提供します。

openstack-utils
: 設定ファイルの編集をはじめとする数々のタスクに役立つサポートユーティリティーを提供します。

openstack-selinux
: OpenStack 固有の SELinux ポリシーモジュールを提供します。

このパッケージは、ネットワークトラフィックを処理する全システムにインストールする必要があります。これには、OpenStack Networking ノード、全ネットワークノード、全コンピュートノードが含まれます。

パッケージをインストールします。

```
# yum install -y openstack-neutron \
   openstack-neutron-ml2 \
   openstack-utils \
   openstack-selinux
```

PLUGIN は **ml2**、**openvswitch** または **linuxbridge** に置き換えます (どちらのプラグインをインストールするかを決定します)。

7.2. OPENSTACK NETWORKING の設定

重要

Red Hat Openstack Platform のカスタマイズなしの状態では、**/etc/nova/nova.conf** の **[DEFAULT]** セクションの **vif_plugging_is_fatal** オプションはデフォルトで **True** となっています。このオプションは、VIF プラグに失敗した場合にインスタンスが機能しないようにするかどうかを制御します。同様に、**/etc/neutron/neutron.conf** ファイルの **[DEFAULT]** セクションの **notify_nova_on_port_status_changes** および **notify_nova_on_port_data_changes** オプションはデフォルトで **False** に指定されています。これらのオプションは、ポートの状態またはデータの変更に関する通知を nova に送信するかどうかを制御します。ただし、これらの値の組み合わせが原因でインスタンスが起動しなくなる可能性があります。インスタンスを正しく起動できるようにするには、これらすべてのオプションを **True** または **False** に設定してください。**True** に設定するには、以下のコマンドを実行してください。

```
# openstack-config --set /etc/nova/nova.conf \
DEFAULT vif_plugging_is_fatal True
# openstack-config --set /etc/neutron/neutron.conf \
DEFAULT notify_nova_on_port_status_changes True
# openstack-config --set /etc/neutron/neutron.conf \
DEFAULT notify_nova_on_port_data_changes True
```

False に設定するには、代わりに以下のコマンドを実行します。

```
# openstack-config --set /etc/nova/nova.conf \
DEFAULT vif_plugging_is_fatal False
# openstack-config --set /etc/neutron/neutron.conf \
DEFAULT notify_nova_on_port_status_changes False
# openstack-config --set /etc/neutron/neutron.conf \
DEFAULT notify_nova_on_port_data_changes False
```

7.2.1. OpenStack Networking プラグインの設定

OpenStack Networking プラグインは、以下に示す例のように **neutron.conf** で長いクラス名ではなく、指定した短い名前で参照することができます。

```
core_plugin = neutron.plugins.ml2.plugin:Ml2Plugin
```

上記の代わりに以下のように参照することができます。

```
core_plugin = ml2
```

誤って空白文字を入れないように気をつけてください。空白文字によって解析エラーが発生する可能性があります。

service_plugins オプションには、複数のサービスプラグインをコンマ区切りリストで指定することができます。

表7.1 service_plugins

ショートネーム	クラス名
dummy	neutron.tests.unit.dummy_plugin:DummyServicePlugin
router	neutron.services.l3_router.l3_router_plugin:L3RouterPlugin
firewall	neutron.services.firewall.fwaas_plugin:FirewallPlugin
lbaas	neutron.services.loadbalancer.plugin:LoadBalancerPlugin
metering	neutron.services.metering.metering_plugin:MeteringPlugin

7.2.1.1. ML2 プラグインの有効化

`neutron-server` サービスを実行しているノードで、ML2 プラグインを有効化します。

手順7.1 ML2 プラグインの有効化

1. OpenStack Networking を **`ml2_conf.ini`** ファイルにダイレクトするためのシンボリックリンクを作成します。

   ```
   # ln -s /etc/neutron/plugins/ml2/ml2_conf.ini
   /etc/neutron/plugin.ini
   ```

2. テナントのネットワーク種別を設定します。サポートされる値は、**`gre`**、**`local`**、**`vlan`**、**`vxlan`** です。デフォルト値は **`local`** ですが、エンタープライズのデプロイメントには推奨していません。

   ```
   # openstack-config --set /etc/neutron/plugin.ini \
      ml2 tenant_network_types TYPE
   ```

   TYPE は、テナントのネットワーク種別に置き換えます。

3. **`flat`** または **`vlan`** ネットワークが選択されている場合は、物理ネットワークを VLAN 範囲にマッピングする必要もあります。

   ```
   # openstack-config --set /etc/neutron/plugin.ini \
      ml2 network_vlan_ranges NAME:START:END
   ```

   以下の値を置き換えてください。

   - *NAME* は、物理ネットワーク名に置き換えます。
   - *START* は、VLAN 範囲の開始を示す識別子に置き換えます。
   - *END* は、VLAN 範囲の終了を示す識別子に置き換えます。

   以下の例のようにコンマ区切りリストを使用して、複数の範囲を指定することができます。

   ```
   physnet1:1000:2999,physnet2:3000:3999
   ```

4. ドライバー種別を設定します。サポートされる値は **local**、**flat**、**vlan**、**gre**、**vxlan** です。

```
# openstack-config --set /etc/neutron/plugin.ini \
   ml2 type_drivers TYPE
```

TYPE は、ドライバーの種別に置き換えます。複数のドライバーを指定する場合は、コンマ区切りリストを使用します。

5. メカニズムドライバーを設定します。利用可能な値は、**openvswitch**、**linuxbridge**、**l2population** です。

```
# openstack-config --set /etc/neutron/plugin.ini \
   ml2 mechanism_drivers TYPE
```

TYPE は、メカニズムドライバーの種別に置き換えます。複数のメカニズムドライバーを指定する場合は、コンマ区切りリストを使用します。

重要

Red Hat OpenStack Platform 11 では、Neutron の Linux Bridge ML2 ドライバーおよびエージェントは非推奨となり、Red Hat OpenStack Platform 12 では削除される予定です。OpenStack Platform director によってデフォルトでデプロイされるのは、Open vSwitch のプラグインです。一般的な用途の場合には、Red Hat はこのプラグインを推奨しています。

6. L2 Population を有効化します。

```
# openstack-config --set /etc/neutron/plugin.ini \
   agent l2_population True
```

7. 使用しているプラグインエージェントに合わせて、**/etc/neutron/plugins/ml2/openvswitch_agent.ini** ファイルまたは **/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルでファイアウォールドライバーを設定します。

   a. **Open vSwitch ファイアウォールドライバー**

   ```
   # openstack-config --set
   /etc/neutron/plugins/ml2/openvswitch_agent.ini
      securitygroup firewall_driver
   neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDr
   iver
   ```

   b. **Linux Bridge ファイアウォールドライバー**

   ```
   # openstack-config --set
   /etc/neutron/plugins/ml2/linuxbridge_agent.ini
      securitygroup firewall_driver
   neutron.agent.linux.iptables_firewall.IptablesFirewallDriver
   ```

8. ML2 プラグインおよび L3 ルーターを有効化します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   DEFAULT core_plugin ml2
# openstack-config --set /etc/neutron/neutron.conf \
   DEFAULT service_plugins router
```

7.2.2. OpenStack Networking データベースの作成

OpenStack Networking を使用するデータベースおよびデータベースユーザーを作成します。この手順に記載するステップはすべて、**neutron-server** サービスを起動する前に、データベースサーバーに**root** ユーザーとしてログインして実行する必要があります。

手順7.2 OpenStack Networking データベースの作成

1. データベースサービスに接続します。

   ```
   # mysql -u root -p
   ```

2. 以下の名前のいずれかを指定してデータベースを作成します。

   - ML2 プラグインを使用する場合に、推奨されるデータベース名は **neutron_ml2** です。
   - Open vSwitch プラグインを使用する場合に、推奨されるデータベース名は **ovs_neutron** です。
   - Linux Bridge プラグインを使用する場合に、推奨されるデータベース名は **neutron_linux_bridge** です。

   以下の例では ML2 用の **neutron_ml2** データベースを作成します。

   ```
   mysql> CREATE DATABASE neutron_ml2 character set utf8;
   ```

3. **neutron** データベースユーザーを作成し、**neutron_ml2** データベースへのアクセスを許可します。

   ```
   mysql> GRANT ALL ON neutron_ml2.* TO 'neutron'@'%' IDENTIFIED BY
   'PASSWORD';
   mysql> GRANT ALL ON neutron_ml2.* TO 'neutron'@'localhost'
   IDENTIFIED BY 'PASSWORD';
   ```

   PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. データベースの特権をフラッシュして、設定が即時に反映されるようにします。

   ```
   mysql> FLUSH PRIVILEGES;
   ```

5. **mysql** クライアントを終了します。

   ```
   mysql> quit
   ```

7.2.3. OpenStack Networking データベース接続の設定

OpenStack Networking で使用するデータベース接続は **/etc/neutron/plugin.ini** ファイルで定義します。有効なデータベースサーバーを参照するように更新してから、サービスを起動する必要があります。以下の手順で記載のステップはすべて、OpenStack Networking をホストするサーバーに **root** ユーザーでログインして実行する必要があります。

手順7.3 OpenStack Networking SQL データベース接続の設定

1. **connection** 設定キーの値を設定します。

```
# openstack-config --set /etc/neutron/plugin.ini \
   DATABASE sql_connection mysql://USER:PASS@IP/DB
```

   以下の値を置き換えてください。

   - *USER* は、OpenStack Networking データベースのユーザー名 (通常は **neutron**) に置き換えます。
   - *PASS* は選択したデータベースユーザーのパスワードに置き換えます。
   - *IP* は、データベースサーバーの IP アドレスまたはホスト名に置き換えます。
   - *DB* は、OpenStack Networking データベースの名前に置き換えます。

   重要

   この接続設定キーに指定する IP アドレスまたはホスト名は、OpenStack Networking データベースの作成時に OpenStack Networking データベースユーザーがアクセスを許可された IP アドレスまたはホスト名と一致する必要があります。また、データベースがローカルでホストされ、OpenStack Networking データベースの作成時に「localhost」へのアクセス権を付与した場合には、「localhost」と入力する必要があります。

2. OpenStack Networking のデータベーススキーマをアップグレードします。

```
# neutron-db-manage --config-file /usr/share/neutron/neutron-
dist.conf \
   --config-file /etc/neutron/neutron.conf --config-file
/etc/neutron/plugin.ini upgrade head
```

7.2.4. OpenStack Networking 用のアイデンティティーレコードの作成

OpenStack Networking で必要な Identity サービスを作成して設定します。これらのエントリーは、OpenStack Networking によって提供されるボリューム機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identity サービスのサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順7.4 OpenStack Networking 用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
   ```

2. **neutron** ユーザーを作成します。

   ```
   [(keystone_admin)]# openstack user create --password PASSWORD
   neutron
   +----------+----------------------------------+
   | Field    | Value                            |
   +----------+----------------------------------+
   | email    | None                             |
   | enabled  | True                             |
   | id       | 8f0d819a4ae54bf9b12d01d0fb095805 |
   | name     | neutron                          |
   | username | neutron                          |
   +----------+----------------------------------+
   ```

   PASSWORD は、OpenStack Networking が Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **services** テナントのコンテキスト内で、**neutron** ユーザーと **admin** ロールを関連付けます。

   ```
   [(keystone_admin)]# openstack role add --project services --user
   neutron admin
   ```

4. **neutron** の OpenStack Networking サービスのエントリーを作成します。

   ```
   [(keystone_admin)]# openstack service create --name neutron \
      --description "OpenStack Networking" \
      network
   ```

5. **neutron** のエンドポイントエントリーを作成します。

   ```
   [(keystone_admin)]# openstack endpoint create
      --publicurl 'http://IP:9696' \
      --adminurl 'http://IP:9696' \
      --internalurl 'http://IP:9696' \
      --region RegionOne \
      neutron
   ```

   IP は、OpenStack Networking ノードとして機能するサーバーの IP アドレスまたはホスト名に置き換えます。

7.2.5. OpenStack Networking の認証の設定

OpenStack Networking が認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、OpenStack Networking をホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順7.5 OpenStack Networking が Identity サービスを使用して認証を行うための設定

手順7.5 OpenStack Networking サービスが Identity サービスを使用して認証を行うための設定

1. 認証ストラテジーを **keystone** に設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   DEFAULT auth_strategy keystone
```

2. OpenStack Networking が使用する必要のある Identity サービスのホストを設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   keystone_authtoken auth_host IP
```

   IP は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

3. OpenStack Networking が正しいテナントとして認証を行うように設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   keystone_authtoken admin_tenant_name services
```

   services は、OpenStack Networking を使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

4. OpenStack Networking が **neutron** の管理ユーザーアカウントを使用して認証を行うように設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   keystone_authtoken admin_user neutron
```

5. OpenStack Networking が **neutron** の管理ユーザーアカウントのパスワードを使用して認証を行うように設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   keystone_authtoken admin_password PASSWORD
```

   PASSWORD は、**neutron** ユーザーの作成時に設定したパスワードに置き換えます。

7.2.6. OpenStack Networking のトラフィックを許可するためのファイアウォール設定

OpenStack Networking は、TCP ポート **9696** で接続を受信します。OpenStack Networking のファイアウォールは、このポートのネットワークトラフィックを許可するように設定する必要があります。以下の手順に記載するステップはすべて、OpenStack Networking をホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順7.6 OpenStack Networking のトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

2. ポート **9696** で TCP トラフィックを許可する INPUT ルールを追加します。

```
-A INPUT -p tcp -m multiport --dports 9696 -j ACCEPT
```

3. ファイアウォールがポート **4789** で VXLAN 接続を受け入れるように INPUT ルールを追加します。新しいルールは、トラフィックを REJECT する INPUT ルールの前に記載する必要があります。

   ```
   -A INPUT -p udp -m udp --dport 4789 -j ACCEPT
   ```

4. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

5. **iptables** サービスを再起動して、変更を有効にします。

   ```
   # systemctl restart iptables.service
   ```

7.2.7. OpenStack Networking のための RabbitMQ メッセージブローカーの設定

RabbitMQ はデフォルト (かつ推奨の) メッセージブローカーです。RabbitMQ メッセージングサービスは、rabbitmq-server パッケージにより提供されます。以下の手順で記載する全ステップは、OpenStack Networking をホストするシステムに **root** ユーザーとしてログインして実行する必要があります。

手順7.7 OpenStack Networking サービスが RabbitMQ メッセージブローカーを使用するための設定

1. RPC バックエンドとして RabbitMQ を設定します。

   ```
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT rpc_backend neutron.openstack.common.rpc.impl_kombu
   ```

2. OpenStack Networking が RabbitMQ ホストに接続するように設定します。

   ```
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT rabbit_host RABBITMQ_HOST
   ```

   RABBITMQ_HOST は、メッセージブローカーの IP アドレスまたはホスト名に置き換えます。

3. メッセージブローカーのポートを **5672** に設定します。

   ```
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT rabbit_port 5672
   ```

4. RabbitMQ の設定時に OpenStack Networking 用に作成した RabbitMQ ユーザー名とパスワードを設定します。

   ```
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT rabbit_userid neutron
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT rabbit_password NEUTRON_PASS
   ```

   neutron および *NEUTRON_PASS* は、OpenStack Networking 用に作成した RabbitMQ ユーザー名とパスワードに置き換えます。

5. RabbitMQ の起動時に、**neutron** ユーザーに全リソースに対するパーミッションが付与されます。このアクセスは、特別に仮想ホスト **/** を介して行われます。Networking サービスがこの仮想ホストに接続されるように設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   DEFAULT rabbit_virtual_host /
```

7.2.8. OpenStack Networking とメッセージブローカーとの間の SSL 通信の有効化

メッセージブローカーで SSL を有効化した場合は、OpenStack Networking も相応に設定する必要があります。以下の手順では、エクスポートしたクライアントの証明書とキーファイルが必要です。これらのファイルのエクスポートの方法に関する説明は、「クライアント用 SSL 証明書のエクスポート」を参照してください。

手順7.8 OpenStack Networking と RabbitMQ メッセージブローカーとの間の SSL 通信の有効化

1. メッセージブローカーとの SSL 通信を有効化します。

   ```
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT rabbit_use_ssl True
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT kombu_ssl_certfile /path/to/client.crt
   # openstack-config --set /etc/neutron/neutron.conf \
      DEFAULT kombu_ssl_keyfile /path/to/clientkeyfile.key
   ```

   以下の値を置き換えてください。

   - */path/to/client.crt* はエクスポートされたクライアント証明書の絶対パスに置き換えます。
   - */path/to/clientkeyfile.key* はエクスポートされたキーファイルの絶対パスに置き換えます。

2. 証明書がサードパーティーの認証局 (CA) によって署名されている場合には、次のコマンドを実行する必要もあります。

   ```
   # openstack-config --set /etc/neutron/neutron.conf \
    DEFAULT kombu_ssl_ca_certs /path/to/ca.crt
   ```

   /path/to/ca.crt は、サードパーティー CA によって提供された CA ファイルの絶対パスに置き換えます (詳細は 「RabbitMQ メッセージブローカーでの SSL の有効化」 を参照)。

7.2.9. OpenStack Networking が Compute サービスとネットワークトポロジーの変更について通信するように設定する手順

OpenStack Networking が Compute サービスとネットワークトポロジーの変更について通信するように設定します。

手順7.9 OpenStack Networking が Compute サービスとネットワークトポロジーの変更について通信するように設定する手順

1. OpenStack Networking がコンピュートコントローラーノードに接続されるように設定します。

   ```
   # openstack-config --set /etc/neutron/neutron.conf \
      nova_url http://CONTROLLER_IP:9696
   ```

   CONTROLLER_IP は、コンピュートコントローラーノードの IP アドレスまたはホスト名に置き換えます。

2. **nova** ユーザーのユーザー名、パスワード、テナントを設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
    nova username nova
# openstack-config --set /etc/neutron/neutron.conf \
   nova auth_type password
# openstack-config --set /etc/neutron/neutron.conf \
   nova password = PASSWORD
 # openstack-config --set /etc/neutron/neutron.conf \
   nova project_name SERVICES
```

   SERVICES は、nova プロジェクトの現在の名前に、*PASSWORD* は、**nova** ユーザーの作成時に設定したパスワードに置き換えます。

3. 管理者権限で、OpenStack Networking がコンピュートコントローラーノードに接続されるように設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   DEFAULT nova_admin_auth_url http://CONTROLLER_IP:35357
```

   CONTROLLER_IP は、コンピュートコントローラーノードの IP アドレスまたはホスト名に置き換えます。

4. OpenStack Networking がコンピュートコントローラーノードに対応する正しいリージョンを使用するように設定します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   nova region_name RegionOne
```

7.2.10. OpenStack Networking の起動

`neutron-server` サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start neutron-server.service
# systemctl enable neutron-server.service
```

重要

デフォルトでは、前のリリースとの後方互換性を維持するため、OpenStack Networking による IP アドレスの Classless Inter-Domain Routing (CIDR) チェックは有効化されていません。このようなチェックが必要な場合には、**`/etc/neutron/neutron.conf`** ファイルで **`force_gateway_on_subnet`** 設定キーの値を **`True`** に設定します。

7.3. DHCP エージェントの設定

DHCP エージェントを設定します。以下の手順に記載するステップはすべて、OpenStack Networking をホストするサーバーに **`root`** ユーザーとしてログインして実行する必要があります。

手順7.10 DHCP エージェントの設定

1. DHCP エージェントが認証に Identity サービスを使用するように設定します。

a. 認証ストラテジーを **keystone** に設定します。

```
# openstack-config --set /etc/neutron/dhcp_agent.ini \
   DEFAULT auth_strategy keystone
```

b. DHCP エージェントが使用する必要のある Identity サービスのホストを設定します。

```
# openstack-config --set /etc/neutron/dhcp_agent.ini \
   keystone_authtoken auth_host IP
```

IP は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

c. DHCP エージェントが正しいテナントとして認証を行うように設定します。

```
# openstack-config --set /etc/neutron/dhcp_agent.ini \
   keystone_authtoken admin_tenant_name services
```

services は、OpenStack Networking を使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

d. DHCP エージェントが **neutron** の管理ユーザーアカウントを使用して認証を行うように設定します。

```
# openstack-config --set /etc/neutron/dhcp_agent.ini \
   keystone_authtoken admin_user neutron
```

e. DHCP エージェントが **neutron** の管理ユーザーアカウントのパスワードを使用して認証を行うように設定します。

```
# openstack-config --set /etc/neutron/dhcp_agent.ini \
   keystone_authtoken admin_password PASSWORD
```

PASSWORD は、**neutron** ユーザーの作成時に設定したパスワードに置き換えます。

2. 使用する OpenStack Networking プラグインに応じて、**/etc/neutron/dhcp_agent.ini** ファイルでインターフェースドライバーを設定します。ML2 を使用する場合は、いずれかのドライバーを選択します。環境で使用するプラグインに適したコマンドを使用してください。

   - **Open vSwitch インターフェースドライバー**

```
# openstack-config --set /etc/neutron/dhcp_agent.ini \
   DEFAULT interface_driver
neutron.agent.linux.interface.OVSInterfaceDriver
```

   - **Linux Bridge インターフェースドライバー**

```
# openstack-config --set /etc/neutron/dhcp_agent.ini \
   DEFAULT interface_driver \
   neutron.agent.linux.interface.BridgeInterfaceDriver
```

3. **neutron-dhcp-agent** サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start neutron-dhcp-agent.service
# systemctl enable neutron-dhcp-agent.service
```

7.4. プラグインエージェントの設定

お使いの環境で使用するプラグインに関連付けるエージェントを設定します。ML2 プラグインを使用している場合には、Open vSwitch エージェントを設定します。

7.4.1. Open vSwitch プラグインエージェントの設定

設定の前に ML2 プラグインをインストールして有効化する必要があります。「ML2 プラグインの有効化」を参照してください。

Open vSwitch プラグインには、対応するエージェントがあります。Open vSwitch プラグインを使用している場合には、パケットを処理する環境内の全ノードにエージェントをインストールして設定する必要があります。これには、専用の DHCP および L3 エージェントをホストするシステムおよび全コンピュートノードが含まれます。

注記

VXLAN および GRE に対する Open vSwitch の TCP segmentation offload (TSO) および Generic Segmentation Offload (GSO) サポートは、デフォルトで有効化されています。

手順7.11 Open vSwitch プラグインエージェントの設定

1. **openvswitch** サービスを起動します。

   ```
   # systemctl start openvswitch.service
   ```

2. **openvswitch** サービスがブート時に起動するように設定します。

   ```
   # systemctl enable openvswitch.service
   ```

3. Open vSwitch エージェントを実行する各ホストには、プライベートのネットワークトラフィックに使用される **br-int** という名前の Open vSwitch ブリッジも必要です。このブリッジは自動的に作成されます。

   警告

   br-int ブリッジは、このエージェントが正しく機能するために必要です。**br-int** ブリッジは、作成後に、削除したり変更したりしないでください。

4. **bridge_mappings** 設定キーの値を、物理ネットワークとそれらに関連付けられたネットワークブリッジ (コンマ区切りリスト) に設定します。

```
# openstack-config --set
/etc/neutron/plugins/ml2/openvswitch_agent.ini \
   ovs bridge_mappings PHYSNET:BRIDGE
```

PHYSNET は物理ネットワークの名前に、*BRIDGE* はネットワークブリッジの名前に置き換えます。

5. **neutron-openvswitch-agent** サービスを起動します。

```
# systemctl start neutron-openvswitch-agent.service
```

6. **neutron-openvswitch-agent** サービスがブート時に起動するように設定します。

```
# systemctl enable neutron-openvswitch-agent.service
```

7. **neutron-ovs-cleanup** サービスがブート時に起動するように設定します。このサービスを使用することで、OpenStack Networking エージェントが tap デバイスの作成と管理を完全に制御できるようにします。

```
# systemctl enable neutron-ovs-cleanup.service
```

7.5. 外部ネットワークの作成

OpenStack Networking は、レイヤー 3 (L3) エージェントを外部ネットワークに接続する 2 つのメカニズムを提供します。第 1 の方法は、外部ブリッジ (**br-ex**) への直接接続で、Open vSwitch プラグインが使用されている場合 (または機能的には ML2 で実装されている場合) にのみサポートされます。ML2 プラグインがサポートする第 2 の方法は、外部プロバイダーネットワークを使用した接続で、Open vSwitch プラグインと Linux Bridge プラグインの両方でサポートされています。

以下の手順に記載するステップはすべて、OpenStack Networking コマンドラインインターフェース (python-neutronclient パッケージにより提供) がインストールされたサーバーで実行する必要があります。Identity サービスの管理者ユーザーの認証情報が格納されている **keystonerc_admin** ファイルへのアクセスも必要です。

以下の手順に記載するステップで生成される一意識別子をメモしておきます。これらの識別子は、L3 エージェントの設定時に必要となります。

手順7.12 外部ネットワークの作成と設定

1. Keystone 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 ~/keystonerc_admin
```

2. 新規プロバイダーネットワークを作成します。

```
[(keystone_admin)]# neutron net-create EXTERNAL_NAME \
   --router:external \
   --provider:network_type TYPE \
   --provider:physical_network PHYSNET \
   --provider:segmentation_id VLAN_TAG
```

以下の値を置き換えてください。

- *EXTERNAL_NAME* は、新規の外部ネットワークプロバイダーの名前に置き換えます。
- *TYPE* は、使用するプロバイダーネットワークの種別に置き換えます。サポートされている値は **flat** (フラットネットワーク用)、**vlan** (VLAN ネットワーク用)、**local** (ローカルネットワーク用) です。
- *PHYSNET* は、物理ネットワークの名前に置き換えます。これは、ローカルネットワークを使用する場合には適用されません。*PHYSNET* は、**/etc/neutron/plugin.ini** ファイルの **bridge_mappings** の下で定義されている値と一致する必要があります。
- *VLAN_TAG* は、ネットワークトラフィックを識別するために使用する VLAN タグを指定します。指定の VLAN タグは、ネットワーク管理者により定義する必要があります。**network_type** が **vlan** 以外の値に設定されている場合は、パラメーターは必要ありません。

次のステップで必要となるため、返された外部ネットワークの一意識別子をメモしておきます。

3. 外部プロバイダーネットワークの新しいサブネットを作成します。

```
[(keystone_admin)]# neutron subnet-create --gateway GATEWAY \
   --allocation-pool start=IP_RANGE_START,end=IP_RANGE_END \
   --disable-dhcp EXTERNAL_NAME EXTERNAL_CIDR
```

以下の値を置き換えてください。

- *GATEWAY* は、新しいサブネットのゲートウェイとして機能するシステムの 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、*EXTERNAL_CIDR* が指定する IP アドレスのブロック内にあり、開始値 *IP_RANGE_START*、終了値 *IP_RANGE_END* で指定した IP アドレスブロックの範囲外で**なければなりません**。
- *IP_RANGE_START* は、Floating IP アドレスが確保される新規サブネット内の IP アドレス範囲の開始値を指定します。
- *IP_RANGE_END* は、Floating IP アドレスが確保される新しいサブネット内の IP アドレス範囲の終了値に置き換えます。
- *EXTERNAL_NAME* は、サブネットが関連付ける外部ネットワークの名前に置き換えます。これは、上記の手順の **net-create** アクションで指定した名前と一致する必要があります。
- *EXTERNAL_CIDR* は、**192.168.100.0/24** などの、サブネットが表現する IP アドレスブロックの CIDR (Classless Inter-Domain Routing) 表記に置き換えます。開始値 *IP_RANGE_START* と終了値 *IP_RANGE_END* の範囲で指定された IP アドレスのブロックは、*EXTERNAL_CIDR* で指定した IP アドレスのブロック内にも収まる **必要があります**。

次のステップで必要となるため、返されたサブネットの一意識別子をメモしておきます。

4. 新しいルーターを作成します。

```
[(keystone_admin)]# neutron router-create NAME
```

NAME は、新規ルーターの名前に置き換えます。次のステップや、L3 エージェントの設定時に必要となるため、返されたルーターの一意識別子をメモしておきます。

5. ルーターと外部プロバイダーネットワークを関連付けます。

```
[(keystone_admin)]# neutron router-gateway-set ROUTER NETWORK
```

ROUTER はルーターの一意識別子に、*NETWORK* は外部プロバイダーネットワークの一意識別子に置き換えます。

6. 各プライベートネットワークのサブネットにルーターを関連付けます。

```
[(keystone_admin)]# neutron router-interface-add ROUTER SUBNET
```

ROUTER はルーターの一意識別子に、*SUBNET* はプライベートネットワークサブネットの一意識別子に置き換えます。ルーターのリンク先となる、既存のプライベートネットワークサブネットごとにこのステップを実行してください。

7.6. L3 エージェントの設定

レイヤー 3 エージェントを設定します。以下の手順に記載するステップはすべて、OpenStack Networking をホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順7.13 L3 エージェントの設定

1. L3 エージェントが認証に Identity サービスを使用するように設定します。

   a. 認証ストラテジーを **keystone** に設定します。

   ```
   # openstack-config --set /etc/neutron/metadata_agent.ini \
      DEFAULT auth_strategy keystone
   ```

   b. L3 エージェントが使用する必要のある Identity サービスのホストを設定します。

   ```
   # openstack-config --set /etc/neutron/metadata_agent.ini \
      keystone_authtoken auth_host IP
   ```

   IP は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

   c. L3 エージェントが正しいテナントとして認証を行うように設定します。

   ```
   # openstack-config --set /etc/neutron/metadata_agent.ini \
      keystone_authtoken admin_tenant_name services
   ```

   services は、OpenStack Networking を使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

   d. L3 エージェントが **neutron** の管理ユーザーアカウントを使用して認証を行うように設定します。

   ```
   # openstack-config --set /etc/neutron/metadata_agent.ini \
      keystone_authtoken admin_user neutron
   ```

e. L3 エージェントが **neutron** の管理ユーザーアカウントのパスワードを使用して認証を行うように設定します。

```
# openstack-config --set /etc/neutron/metadata_agent.ini \
   keystone_authtoken admin_password PASSWORD
```

PASSWORD は、**neutron** ユーザーの作成時に設定したパスワードに置き換えます。

f. **neutron-metadata-agent** サービスと **nova-metadata-api** サービスが同じサーバー上にインストールされていない場合は、**nova-metadata-api** サービスのアドレスを設定します。

```
# openstack-config --set /etc/neutron/metadata_agent.ini \
   DEFAULT nova_metadata_ip IP
```

IP は、**nova-metadata-api** サービスをホストするサーバーの IP アドレスに置き換えます。

2. 使用する OpenStack Networking プラグインに応じて、**/etc/neutron/l3_agent.ini** ファイルでインターフェースドライバーを設定します。環境で使用するプラグインに適したコマンドを使用してください。

   - **Open vSwitch インターフェースドライバー**

```
# openstack-config --set /etc/neutron/l3_agent.ini \
   DEFAULT interface_driver
neutron.agent.linux.interface.OVSInterfaceDriver
```

   - **Linux Bridge インターフェースドライバー**

```
# openstack-config --set /etc/neutron/l3_agent.ini \
   DEFAULT interface_driver
neutron.agent.linux.interface.BridgeInterfaceDriver
```

3. L3 エージェントは、外部ブリッジまたは外部プロバイダーネットワークを使用して外部ネットワークに接続します。Open vSwitch プラグインを使用する場合には、これらのいずれの方法もサポートされます。Linux Bridge プラグインを使用する場合は、外部プロバイダーネットワークの使用のみがサポートされます。環境に最も適したオプションを設定してください。

   - **外部ブリッジの使用**
     外部ブリッジを作成/設定して、OpenStack Networking がこのブリッジを使用するように設定します。L3 エージェントのインスタンスをホストする各システムで以下の手順を実行してください。

     a. 外部ブリッジ **br-ex** を作成します。

```
# ovs-vsctl add-br br-ex
```

     b. **/etc/sysconfig/network-scripts/ifcfg-br-ex** ファイルを作成して以下の行を追加し、**br-ex** デバイスが再起動後も永続的に存在するようにします。

```
DEVICE=br-ex
DEVICETYPE=ovs
```

```
TYPE=OVSBridge
ONBOOT=yes
BOOTPROTO=none
```

c. L3 エージェントが確実に外部ブリッジを使用するようにします。

```
# openstack-config --set /etc/neutron/l3_agent.ini \
   DEFAULT external_network_bridge br-ex
```

- **プロバイダーネットワークの使用**
  プロバイダーネットワークを使用して L3 エージェントを外部ネットワークに接続するには、最初にプロバイダーネットワークを作成する必要があります。また、そのネットワークに関連付けるサブネットとルーターも作成する必要があります。以下のステップを完了するには、ルーターの一意識別子が必要となります。

  external_network_bridge 設定キーの値は空欄にしてください。これの設定により、L3 エージェントはこの外部ブリッジの使用を試みません。

```
# openstack-config --set /etc/neutron/l3_agent.ini \
   DEFAULT external_network_bridge ""
```

4. **neutron-l3-agent** サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start neutron-l3-agent.service
# systemctl enable neutron-l3-agent.service
```

5. OpenStack Networking のメタデータエージェントにより、仮想マシンインスタンスはコンピュートメタデータサービスと通信することができます。このエージェントは、L3 エージェントと同じホストで実行されます。**neutron-metadata-agent** サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start neutron-metadata-agent.service
# systemctl enable neutron-metadata-agent.service
```

6. **leastrouter** スケジューラーは、L3 エージェントのルーター割り当てを列挙し、その結果として、最もルーター数が少ない L3 エージェントにルーターをスケジュールします。これは、L3 エージェントの候補プールから無作為に選択する ChanceScheduler の動作とは異なります。

a. **leastrouter** スケジューラーを有効化します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   DEFAULT router_scheduler_driver
neutron.scheduler.l3_agent_scheduler.LeastRoutersScheduler
```

b. Keystone 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 ~/keystonerc_admin
```

c. ネットワークに接続されると、ルーターがスケジュールされます。ルーターのスケジュールを解除するには、以下のコマンドを使用します。

```
[(keystone_admin)]# neutron l3-agent-router-remove L3_NODE_ID
ROUTER_ID
```

L3_NODE_ID はルーターが現在ホストされるエージェントの一意識別子に、*ROUTER_ID* はルーターの一意識別子に置き換えます。

d. ルーターを割り当てます。

```
[(keystone_admin)]# neutron l3-agent-router-add L3_NODE_ID
ROUTER_ID
```

L3_NODE_ID はルーターを割り当てるエージェントの一意識別子に、*ROUTER_ID* はルーターの一意識別子に置き換えます。

7.7. OPENSTACK NETWORKING をインストールしたシステムの検証

OpenStack Networking の使用を開始するには、コンピュートノードにネットワークコンポーネントをデプロイし、初期ネットワークおよびルーターの定義も行う必要がありますが、OpenStack Networking デプロイメントの基本的なサニティーチェックは、以下の手順に記載するステップに従って実行することができます。

手順7.14 OpenStack Networking をインストールしたシステムの検証

1. **すべてのノードで以下を実行します。**

   a. 次のコマンドを実行して、Red Hat OpenStack Platform で使用する予定のカスタマイズされた Red Hat Enterprise Linux カーネルを検証します。

   ```
   # uname --kernel-release
   2.6.32-358.6.2.openstack.el6.x86_64
   ```

   返されたカーネルリリースの値に **openstack** の文字列が含まれていない場合には、カーネルを更新してシステムを再起動します。

   b. インストールされている IP ユーティリティーがネットワーク名前空間をサポートしていることを確認します。

   ```
   # ip netns
   ```

   引数が認識されない、またはサポートされていないというエラーが表示された場合には、**yum** コマンドでシステムを更新します。

2. **サービスノードで以下を実行します。**

   a. **neutron-server** サービスが稼働中であることを確認します。

   ```
   # openstack-status | grep neutron-server
   neutron-server:                                 active
   ```

3. **ネットワークノードで以下を実行します。**
   以下のサービスが稼働していることを確認します。

   - DHCP エージェント (**neutron-dhcp-agent**)

- L3 エージェント (**neutron-l3-agent**)
- 該当する場合はプラグインエージェント (**neutron-openvswitch-agent** または **neutron-linuxbridge-agent**)
- Metadata エージェント (**neutron-metadata-agent**)

```
# openstack-status | grep SERVICENAME
```

7.7.1. OpenStack Networking に関する問題のトラブルシューティング

本項では、OpenStack Networking に関する問題のトラブルシューティングに使用可能なコマンドと手順について説明します。

ネットワークデバイスのデバッグ

- **ip a** コマンドで、全物理/仮想デバイスを表示します。
- **ovs-vsctl show** コマンドで、仮想スイッチ内のインターフェースとブリッジを表示します。
- **ovs-dpctl show** コマンドで、スイッチ上のデータパスを表示します。

ネットワークパケットの追跡

- パケットが通過しない場所を確認します。

  ```
  # tcpdump -n -i INTERFACE -e -w FILENAME
  ```

  INTERFACE は、確認するネットワークインターフェースの名前に置き換えます。このインターフェース名には、ブリッジまたはホストのイーサネットデバイスの名前を使用することができます。

  -e フラグで、リンクレベルのヘッダーが出力されるようにします (その場合には **vlan** タグが表示されます)。

  -w フラグはオプションです。出力をファイルに書き込む場合にのみ使用することができます。使用しない場合には、その出力は標準出力 (**stdout**) に書き込まれます。

  tcpdump に関する詳しい情報は man ページを参照してください。

ネットワーク名前空間のデバッグ

- **ip netns list** コマンドで、既知のネットワーク名前空間をすべて一覧表示します。
- 特定の名前空間内のルーティングテーブルを表示します。

  ```
  # ip netns exec NAMESPACE_ID bash
  # route -n
  ```

  bash シェルで **ip netns exec** コマンドを起動し、それ以降に実行するコマンドが **ip netns exec** コマンドを実行しなくても呼び出されるようにします。

第8章 COMPUTE サービスのインストール

8.1. COMPUTE の VNC プロキシーのインストール

8.1.1. Compute の VNC プロキシーパッケージのインストール

VNC プロキシーは、Compute サービスのユーザーが利用できます。VNC プロキシーサーバーのパッケージには 2 つの種別があります。openstack-nova-novncproxy パッケージは、(Websocket を使用して) Web ブラウザーを介した VNC サポートをインスタンスに提供する一方、openstack-nova-console パッケージは、(**openstack-nova-xvpvncproxy** サービス経由で) 従来の VNC クライアントを介したインスタンスへのアクセスを提供します。

openstack-nova-console パッケージにより、コンソール認証サービスも提供されます。このサービスは、VNC 接続の認証に使用されます。通常、コンソール認証サービスおよびプロキシーユーティリティーは、Compute API サービスと同じホストにインストールされます。

以下の手順は、**root** ユーザーとしてログインして実行する必要があります。

手順8.1 Compute の VNC プロキシーパッケージのインストール

- VNC プロキシーのユーティリティーとコンソール認証サービスをインストールします。
  - **yum** コマンドで openstack-nova-novncproxy パッケージをインストールします。

    ```
    # yum install -y openstack-nova-novncproxy
    ```

  - **yum** コマンドで openstack-nova-console パッケージをインストールします。

    ```
    # yum install -y openstack-nova-console
    ```

VNC プロキシーのパッケージとコンソール認証サービスがインストールされ、設定の準備が整いました。

8.1.2. Compute の VNC プロキシーのトラフィックを許可するためのファイアウォール設定

インスタンスへの VNC アクセスをホストするノードは、ファイアウォールを介した VNC トラフィックを許可するように設定する必要があります。デフォルトでは、**openstack-nova-novncproxy** サービスは TCP ポート 6080 をリッスンし、**openstack-nova-xvpvncproxy** サービスは TCP ポート 6081 をリッスンします。

TCP ポート 6080 上のトラフィックがファイアウォールを通過するように許可して openstack-nova-novncproxy パッケージが使用できるようにするには、以下の手順に従って設定します。

以下の手順は、**root** ユーザーとしてログインして実行する必要があります。

手順8.2 Compute の VNC プロキシーのトラフィックを許可するためのファイアウォール設定

1. **/etc/sysconfig/iptables** ファイルを編集して、**-A INPUT -i lo -j ACCEPT** の行の下に以下の新しい行を追加します。この行は、**-A INPUT -j REJECT** ルールの上に配置するようにしてください。

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 6080 -j ACCEPT
```

2. ファイルを保存してエディターを終了します。

- 同様に、**openstack-nova-xvpvncproxy** サービスを使用する場合には、以下の新しい行を同じ場所に追加して、TCP ポート 6081 のトラフィックを有効にします。

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 6081 -j ACCEPT
```

ファイルで新規ファイアウォールルールの編集を終了したら、次のコマンドを **root** ユーザーとして実行し、変更を適用します。

```
# service iptables restart
```

```
# iptables-save
```

VNC プロキシーのトラフィックを許可するようにファイアウォールが設定されました。

8.1.3. VNC プロキシーサービスの設定

インスタンスへの VNC アクセスは、Web ブラウザーまたは従来の VNC クライアントを介して提供されます。**/etc/nova/nova.conf** ファイルには、次のような VNC オプションがあります。

- **vnc_enabled**: デフォルトは true
- **vncserver_listen**: VNC サービスをバインドする IP アドレス
- **vncserver_proxyclient_address**: プロキシーがインスタンスに接続するのに使用するコンピュートホストの IP アドレス
- **novncproxy_base_url**: クライアントがインスタンスに接続するブラウザーのアドレス
- **novncproxy_port**: ブラウザーの VNC 接続をリッスンするポート。デフォルトは 6080。
- **xvpvncproxy_port**: 従来の VNC クライアント向けのバインドするポート。デフォルトは 6081。

root ユーザーとして **service** コマンドを使用して、コンソールの認証サービスを起動します。

```
#service openstack-nova-consoleauth start
```

chkconfig コマンドでサービスを永続的に有効にします。

```
#chkconfig openstack-nova-consoleauth on
```

nova ノードで **root** ユーザーとして **service** コマンドを使用して、ブラウザーベースのサービスを起動します。

```
#service openstack-nova-novncproxy start
```

chkconfig コマンドでサービスを永続的に有効にします。

```
#chkconfig openstack-nova-novncproxy on
```

従来のクライアント (非ブラウザーベース) を使用する VNC サービスへのアクセスを制御するには、上記のコマンドで **openstack-nova-xvpvncproxy** を代わりに使用してください。

8.1.4. ライブマイグレーションの設定

Red Hat OpenStack Platform は、共有ストレージマイグレーションまたはブロックマイグレーションのいずれかを使用したライブマイグレーションをサポートします。以下の項では、両タイプの移行における一般的な要件について説明します。両タイプの詳細の設定手順は、「ライブ (実行中) インスタンスの移行」を参照してください。

8.1.4.1. 一般要件

移行における一般的な要件は以下のとおりです。

- 管理者として、クラウド環境にコマンドラインからアクセスできること (以下の手順はすべてコマンドラインで実行されます)。各種コマンドを実行するには、まずユーザーの認証変数を読み込みます。

  ```
  # source ~/keystonerc_admin
  ```

- 移行元/移行先の両ノードは、同じサブネットに配置され、同じプロセッサータイプを使用すること
- コンピュートサーバー (コントローラーおよびノード) はすべて、相互で名前を解決できること
- コンピュートノード間で Compute サービスおよび libvirt ユーザーの UID および GID は同一であること
- コンピュートノードは、libvirt と KVM を使用すること

8.1.4.2. マルチパス機能の要件

マルチパス機能が設定されたインスタンスを移行する場合は、移行元と移行先のノードでマルチパスデバイスの名前が同じである必要があります。移行先のノードで、インスタンスがマルチパスデバイスの名前を解決できない場合には、移行が失敗します。

移行元ノードと移行先ノードの両方でデバイス WWID の使用を強制することで、一貫性のあるマルチパスデバイス名を指定することができます。これには、移行先/移行元の両ノードで以下のコマンドを実行して **ユーザーフレンドリーな名前** を無効にし、**`multipathd`** を再起動する必要があります。

```
# mpathconf --enable --user_friendly_names n
# service multipathd restart
```

詳しい情報は、『DM Multipath ガイド』の「クラスター内では整合性のあるマルチパスデバイス名を維持する」を参照してください。

8.1.5. Compute の VNC プロキシーを使用したインスタンスへのアクセス

`/etc/nova/nova.conf` ファイルに記載されている **novncproxy_base_url** を参照し、インスタンスコンソールにアクセスします。

以下の画像は、Web ブラウザーを使用した、Fedora Linux インスタンスへの VNC アクセスを示しています。これは、例を示す目的のみで記載しており、IP アドレスなどの設定は、お使いの環境とは異なります。

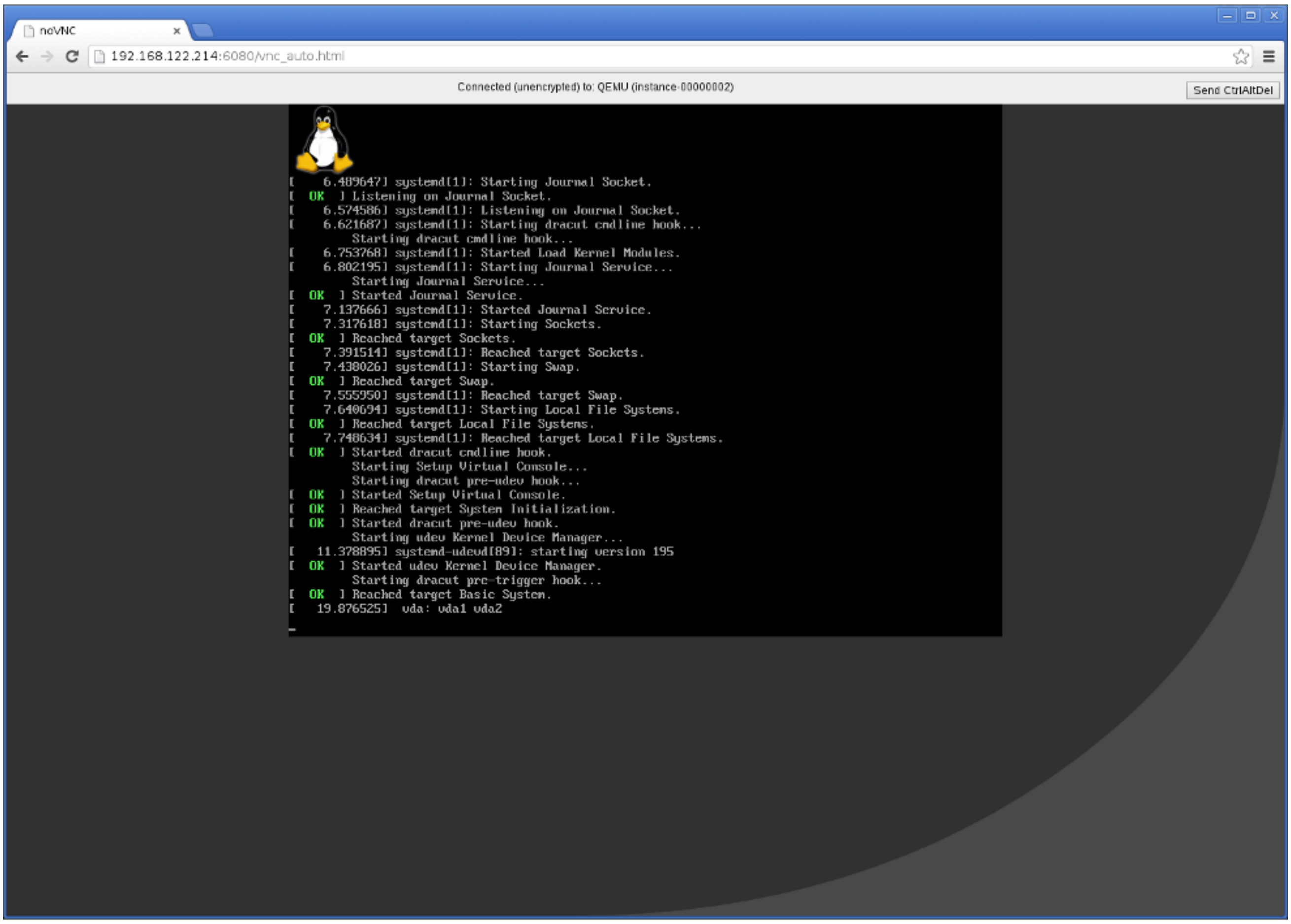

図8.1 インスタンスへの VNC アクセス

8.2. コンピュートノードのインストール

8.2.1. Compute サービスのパッケージのインストール

OpenStack Compute サービスには以下のパッケージが必要です。

openstack-nova-api

OpenStack Compute API サービスを提供します。環境内で少なくとも 1 台のノード API サービスのインスタンスをホストしている必要があります。これは、Identity サービスのエンドポイントの定義によって、Compute サービスにポイントされているノードである必要があります。

openstack-nova-compute

OpenStack Compute サービスを提供します。

openstack-nova-conductor

Compute コンダクターサービスを提供します。コンダクターは、コンピュートノードによって作成されるデータベース要求を処理し、各コンピュートノードがデータベースに直接アクセスする必要がないようにします。各環境で少なくとも 1 台のノードが Compute のコンダクターとして機能する必要があります。

openstack-nova-scheduler

Compute のスケジューラーサービスを提供します。スケジューラーは利用可能な Compute リソース全体にわたり、API に対する要求のスケジューリングを処理します。各環境で、少なくとも 1 台のノードが Compute スケジューラーとして稼働する必要があります。

python-cinderclient

Block Storage サービスによって管理されるストレージにアクセスするためのクライアントユーティリティーを提供します。インスタンスに Block Storage ボリュームを接続しない場合や、Block Storage サービス以外のサービスを使用して Block Storage ボリュームを管理する場合には、このパッケージは必要ありません。

パッケージをインストールします。

```
# yum install -y openstack-nova-api openstack-nova-compute \
   openstack-nova-conductor openstack-nova-scheduler \
   python-cinderclient
```

注記

上記の例では、すべての Compute サービスのパッケージが単一のノードにインストールされます。Red Hat は、実稼働環境にデプロイする場合に、API、コンダクター、スケジューラーのサービスを 1 つのコントローラーノードにインストールするか、それぞれを別々のノードにインストールすることを推奨します。Compute サービス自体は、仮想マシンインスタンスをホストする各ノードにインストールする必要があります。

8.2.2. Compute サービスのデータベースの作成

Compute サービスが使用するデータベースとデータベースユーザーを作成します。以下の手順の全ステップは、データベースサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順8.3 Compute サービスのデータベースの作成

1. データベースサービスに接続します。

   ```
   # mysql -u root -p
   ```

2. **nova** および **nova_api** データベースを作成します。

   ```
   mysql> CREATE DATABASE nova;
   ```

   ```
   mysql> CREATE DATABASE nova_api;
   ```

3. **nova** データベースユーザーを作成して、**nova** および **nova_api** データベースへのユーザーアクセスを許可します。

   ```
   mysql> GRANT ALL ON nova.* TO 'nova'@'localhost' IDENTIFIED BY
   'PASSWORD';
   ```

   ```
   mysql> GRANT ALL ON nova.* TO 'nova'@'%' IDENTIFIED BY 'PASSWORD';
   ```

```
mysql> GRANT ALL ON nova_api.* TO 'nova'@'localhost' IDENTIFIED BY
'PASSWORD';
```

```
mysql> GRANT ALL ON nova_api.* TO 'nova'@'%' IDENTIFIED BY
'PASSWORD';
```

PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. データベースの特権をフラッシュして、設定が即時に反映されるようにします。

```
mysql> FLUSH PRIVILEGES;
```

5. **mysql** クライアントを終了します。

```
mysql> quit
```

8.2.3. Compute サービスのデータベース接続の設定

Compute サービスによって使用されるデータベース接続文字列は、**/etc/nova/nova.conf** ファイルで定義されます。サービスを起動する前に、有効なデータベースサーバーをポイントするように更新しておく必要があります。

データベース接続文字列は、コンダクターサービス (**openstack-nova-conductor**) をホストするノードのみで設定する必要があります。コンピュートノードがメッセージングインフラストラクチャーを使用してコンダクターに通信すると、コンダクターはそれに応えてデータベースとの通信をオーケストレートするので、個別のコンピュートノードは、データベースに直接アクセスする必要がありません。どのようなコンピュート環境においても、コンダクターサービスのインスタンスが少なくとも 1 つ必要です。

以下の手順に記載するステップはすべて、Compute コンダクターサービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順8.4 Compute サービスの SQL データベース接続の設定

- **sql_connection** 設定キーの値を設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT sql_connection mysql://USER:PASS@IP/DB
```

以下の値を置き換えてください。

  - *USER* は、Compute サービスのデータベースのユーザー名 (通常は **nova**) に置き換えます。
  - *PASS* は選択したデータベースユーザーのパスワードに置き換えます。
  - *IP* は、Identity サーバーの IP アドレスまたはホスト名に置き換えます。
  - *DB* は、Compute サービスのデータベースの名前 (通常は **nova**) に置き換えます。

重要

この接続設定キーに指定する IP アドレスまたはホスト名は、Compute サービスのデータベースの作成時に Compute サービスのデータベースユーザーがアクセスを許可された IP アドレスまたはホスト名と一致する必要があります。また、データベースがローカルでホストされ、Compute サービスのデータベースの作成時に「localhost」へのアクセス権を付与した場合には、「localhost」と入力する必要があります。

8.2.4. Compute サービス用のアイデンティティーレコードの作成

Compute サービスで必要な Identity サービスを作成して設定します。これらのエントリーは、Compute サービスによって提供されるボリューム機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identity サービスのサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順8.5 Compute サービス用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **compute** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password PASSWORD
compute
+----------+----------------------------------+
| Field    | Value                            |
+----------+----------------------------------+
| email    | None                             |
| enabled  | True                             |
| id       | 421665cdfaa24f93b4e904c81ff702ad |
| name     | compute                          |
| username | compute                          |
+----------+----------------------------------+
```

PASSWORD は、Compute サービスが Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **services** テナントのコンテキスト内で、**compute** ユーザーと **admin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
compute admin
```

4. **compute** のサービスエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name compute \
   --description "OpenStack Compute Service" \
   compute
```

5. **compute** エンドポイントエントリーを作成します。

```
[(keystone_admin)]# openstack endpoint create \
   --publicurl "http://IP:8774/v2/%(tenant_id)s" \
   --adminurl "http://IP:8774/v2/%(tenant_id)s" \
   --internalurl "http://IP:8774/v2/%(tenant_id)s" \
   --region RegionOne \
   compute
```

IP は、Compute API サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

8.2.5. Compute サービスの認証の設定

Compute サービスが認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、Compute サービスをホストする各システムに **root** ユーザーとしてログインして実行する必要があります。

手順8.6 Compute サービスが Identity サービスを使用して認証を行うための設定

1. 認証ストラテジーを **keystone** に設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT auth_strategy keystone
```

2. Compute サービスが使用する必要のある Identity サービスのホストを設定します。

```
# openstack-config --set /etc/nova/api-paste.ini \
   filter:authtoken auth_host IP
```

IP は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

3. Compute サービスが正しいテナントとして認証を行うように設定します。

```
# openstack-config --set /etc/nova/api-paste.ini \
   filter:authtoken admin_tenant_name services
```

services は、Compute サービスを使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

4. Compute サービスが **compute** 管理ユーザーアカウントを使用して認証を行うように設定します。

```
# openstack-config --set /etc/nova/api-paste.ini \
   filter:authtoken admin_user compute
```

5. Compute サービスが正しい **compute** 管理ユーザーアカウントのパスワードを使用するように設定します。

```
# openstack-config --set /etc/nova/api-paste.ini \
   filter:authtoken admin_password PASSWORD
```

PASSWORD は、**compute** ユーザーの作成時に設定したパスワードに置き換えます。

8.2.6. Compute サービスのトラフィックを許可するためのファイアウォール設定

仮想マシンコンソールへの接続は、直接またはプロキシー経由に関わらず、**5900** から **5999** までのポートで受信されます。Compute API サービスへは、ポート **8774** 経由で接続されます。サービスノードのファイアウォールは、これらのポートのネットワークトラフィックを許可するように設定する必要があります。以下の手順に記載するステップはすべて、コンピュートノードに **root** ユーザーとしてログインして実行する必要があります。

手順8.7 Copute サービスのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

2. ファイルに以下の行を追記して、**5900** から **5999** までの範囲内のポートで TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールの前に追加する必要があります。

   ```
   -A INPUT -p tcp -m multiport --dports 5900:5999 -j ACCEPT
   ```

3. このファイルに、ポート **8774** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

   ```
   -A INPUT -p tcp -m multiport --dports 8774 -j ACCEPT
   ```

4. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

5. **iptables** サービスを再起動して、変更を有効にします。

   ```
   # systemctl restart iptables.service
   ```

8.2.7. Compute サービスが SSL を使用するための設定

nova.conf ファイルで、以下のオプションを使用して SSL を設定します。

表8.1 Compute の SSL オプション

設定オプション	説明
enabled_ssl_apis	SSL を有効にする API の一覧
ssl_ca_file	クライアントの接続を検証するのに使用する CA 証明書ファイル
ssl_cert_file	API サーバーの SSL 証明書

設定オプション	説明
ssl_key_file	API サーバーの SSL 秘密鍵
tcp_keepidle	サーバーソケットごとに設定する TCP_KEEPIDLE 値 (秒単位)。デフォルトでは 600

8.2.8. Compute サービスのための RabbitMQ メッセージブローカーの設定

RabbitMQ はデフォルト (かつ推奨の) メッセージブローカーです。RabbitMQ メッセージングサービスは、rabbitmq-server パッケージにより提供されます。以下の手順に記載する全ステップは、コンピュートコントローラーサービスおよびコンピュートノードをホストするシステムに **root** ユーザーとしてログインして実行する必要があります。

手順8.8 Compute サービスが RabbitMQ メッセージブローカーを使用するための設定

1. RPC バックエンドとして RabbitMQ を設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT rpc_backend rabbit
```

2. Compute サービスが RabbitMQ ホストに接続するように設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT rabbit_host RABBITMQ_HOST
```

   RABBITMQ_HOST は、メッセージブローカーの IP アドレスまたはホスト名に置き換えます。

3. メッセージブローカーのポートを **5672** に設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT rabbit_port 5672
```

4. RabbitMQ の設定時に Compute サービス用に作成した RabbitMQ ユーザー名とパスワードを設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT rabbit_userid nova
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT rabbit_password NOVA_PASS
```

   nova および *NOVA_PASS* は、Compute サービス用に作成したRabbitMQ ユーザー名とパスワードに置き換えます。

5. RabbitMQ の起動時に、**nova** ユーザーに全リソースに対するパーミッションが付与されます。このアクセスは、特別に仮想ホスト **/** を介して行われます。Compute サービスがこの仮想ホストに接続されるように設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT rabbit_virtual_host /
```

8.2.9. Compute サービスとメッセージブローカーとの間の SSL 通信の有効化

メッセージブローカーで SSL を有効化した場合は、Compute サービスも相応に設定する必要があります。以下の手順では、エクスポートしたクライアントの証明書とキーファイルが必要です。これらのファイルのエクスポートの方法に関する説明は、「クライアント用 SSL 証明書のエクスポート」を参照してください。

手順8.9 Compute サービスと RabbitMQ メッセージブローカーとの間の SSL 通信の有効化

1. メッセージブローカーとの SSL 通信を有効化します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT rabbit_use_ssl True
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT kombu_ssl_certfile /path/to/client.crt
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT kombu_ssl_keyfile /path/to/clientkeyfile.key
```

   以下の値を置き換えてください。

   - */path/to/client.crt* はエクスポートされたクライアント証明書の絶対パスに置き換えます。
   - */path/to/clientkeyfile.key* はエクスポートされたキーファイルの絶対パスに置き換えます。

2. 証明書がサードパーティーの認証局 (CA) によって署名されている場合には、次のコマンドを実行する必要もあります。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT kombu_ssl_ca_certs /path/to/ca.crt
```

   /path/to/ca.crt は、サードパーティー CA によって提供された CA ファイルの絶対パスに置き換えます (詳細は 「RabbitMQ メッセージブローカーでの SSL の有効化」 を参照)。

8.2.10. リソースのオーバーコミットの設定

OpenStack は、コンピュートノード上における CPU およびメモリーリソースのオーバーコミットをサポートしています。オーバーコミットとは、物理リソースを上回る量の仮想 CPU やメモリーを割り当てるテクニックです。

重要

オーバーコミットにより、実行できるインスタンスの数が増えますが、インスタンスのパフォーマンスは低下します。

CPU およびメモリーのオーバーコミット設定は比率で表示されます。OpenStack は、デフォルトで以下のような比率を使用します。

- デフォルトの CPU オーバーコミット率は 16 ですが、これは物理コア 1 つにつき 最大 16 の仮想コアをノードに割り当てることができるという意味です。
- デフォルトのオーバーコミット率は 1.5 ですが、これはインスタンスのメモリー使用量合計が、物理メモリーの空き容量の 1.5 倍未満の場合には、インスタンスを物理ノードに割り当てることができるという意味です。

デフォルト設定を変更するには、**/etc/nova/nova.conf** の **cpu_allocation_ratio** および **ram_allocation_ratio** のディレクティブを使用してください。

8.2.11. ホストのリソースの確保

ホストのメモリーおよびディスクリソースが常に OpenStack で使用できるように確保することができます。一定の容量のメモリーやディスクリソースが仮想マシンでの使用に提供可能と見なされないようにするには、**/etc/nova/nova.conf** で以下のディレクティブを編集します。

- **reserved_host_memory_mb**: デフォルト値は 512 MB
- **reserved_host_disk_mb**: デフォルト値は 0 MB

8.2.12. Compute ネットワークの設定

8.2.12.1. Compute ネットワークの概要

Nova のみのデプロイメントとは異なり、OpenStack Networking を使用している場合には、**nova-network** サービスは**実行してはなりません**。その代わりに、ネットワーク関連の決定事項はすべて OpenStack Networking サービスに委任されます。

このため、ネットワークの設定時には、Nova ネットワークについてのマニュアルや Nova ネットワークでの過去の経験に頼るのではなく、本ガイドを参照していただくことが非常に重要となります。特に、**nova-manage** や **nova** などの CLI ツールを使用したネットワークの管理や IP アドレス指定 (固定 IP および Floating IP の両方を含む) は、OpenStack Networking ではサポートされていません。

重要

物理ノードを使用して OpenStack Network を稼働する前には、**nova-network** をアンインストールして、**nova-network** を実行していた物理ノードを再起動することを強く推奨します。OpenStack Networking サービスの使用中に間違えて **nova-network** プロセスを実行すると、たとえば、以前に実行していた **nova-network** により古い ファイアウォールルールがプッシュダウンされる可能性があり、問題が発生する場合があります。

8.2.12.2. Compute の設定の更新

Compute のインスタンスがプロビジョニングまたはプロビジョニング解除されるたびに、サービスは API を介して OpenStack Networking と通信します。この接続をスムーズに行うには、以下の手順に記載する接続および認証の説明に従って設定を行う必要があります。

以下の手順に記載のステップはすべて、各コンピュートノードに **root** ユーザーとしてログインして実行する必要があります。

手順8.10 コンピュートノードの接続および認証の設定の更新

1. **network_api_class** 設定キーを変更して、OpenStack Networking が使用中であることを示します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT use_neutron true
```

2. Compute サービスが OpenStack Networking API のエンドポイントを使用するように設定します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      neutron url http://IP:9696/
   ```

   IP は、OpenStack Networking API サービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

3. OpenStack Networking サービスが使用するプロジェクトの名前を設定します。本ガイドの例では、*SERVICES* を使用します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      neutron project_name SERVICES
   ```

4. OpenStack Networking の管理ユーザー名を設定します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      neutron username NEUTRON
   ```

5. OpenStack Networking の管理ユーザー名に関連付けられたパスワードを設定します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      neutron password PASSWORD
   ```

6. Identity サービスのエンドポイントに関連付けられた URL を設定します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      neutron auth_url http://IP:35357
   ```

   IP は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

7. メタデータのプロキシーを有効化して、メタデータのプロキシーシークレットを設定します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      neutron service_metadata_proxy true
   # openstack-config --set /etc/nova/nova.conf \
      neutron metadata_proxy_shared_secret METADATA_SECRET
   ```

   METADATA_SECRET は、メタデータプロキシーが通信のセキュリティー保護に使用する文字列に置き換えます。

8. OpenStack Networking セキュリティーグループの使用を有効化します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      DEFAULT security_group_api neutron
   ```

9. ファイアウォールドライバーを **nova.virt.firewall.NoopFirewallDriver** に設定します。

   ```
   # openstack-config --set /etc/nova/nova.conf \
      DEFAULT firewall_driver nova.virt.firewall.NoopFirewallDriver
   ```

この操作は、OpenStack Networking セキュリティーグループが使用中の状態で**実行する必要があります。**

10. テキストエディターで **/etc/sysctl.conf** ファイルを開き、以下のカーネルネットワークパラメーターを追加または編集します。

```
net.ipv4.ip_forward = 1
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
net.bridge.bridge-nf-call-arptables = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
```

11. 更新したカーネルパラメーターを読み込みます。

```
# sysctl -p
```

8.2.12.3. L2 エージェントの設定

各コンピュートノードは、使用中のネットワークプラグインに適したレイヤー 2 (L2) エージェントのインスタンスを実行する必要があります。

- 「Open vSwitch プラグインエージェントの設定」

8.2.12.4. 仮想インターフェース結線の設定

nova-compute がインスタンスを作成する時には、そのインスタンスに関連付ける各 vNIC を、OpenStack Networking によって制御される仮想スイッチに「結線」する必要があります。また Compute が、各 vNIC に関連付けられた OpenStack Networking ポートの識別子を仮想スイッチに通知する必要があります。

Red Hat OpenStack Platform では、仮想インターフェースの汎用ドライバー **nova.virt.libvirt.vif.LibvirtGenericVIFDriver** が提供されます。このドライバーは、必要な仮想インターフェースバインディングの種別を返すことが可能な OpenStack Networking に依存しています。この操作は、以下のプラグインによってサポートされています。

- Linux Bridge
- Open vSwitch
- NEC
- BigSwitch
- CloudBase Hyper-V
- Brocade

汎用ドライバーを使用するには、**openstack-config** コマンドを実行して **vif_driver** 設定キーの値を適切に設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   libvirt vif_driver \
   nova.virt.libvirt.vif.LibvirtGenericVIFDriver
```

重要

Open vSwitch と Linux Bridge のデプロイメントに関する考慮事項:

- セキュリティーグループを有効にした状態で Open vSwitch を実行している場合には、汎用ドライバーではなく、Open vSwitch 固有のドライバー **`nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver`** を使用してください。

- Linux Bridge の場合には、**`/etc/libvirt/qemu.conf`** ファイルに以下の内容を追記して、仮想マシンが適切に起動するようにする必要があります。

```
user = "root"
group = "root"
cgroup_device_acl = [
   "/dev/null", "/dev/full", "/dev/zero",
   "/dev/random", "/dev/urandom",
   "/dev/ptmx", "/dev/kvm", "/dev/kqemu",
   "/dev/rtc", "/dev/hpet", "/dev/net/tun",
]
```

8.2.13. Compute サービスのデータベースへのデータ投入

Compute サービスのデータベース接続文字列を適切に設定した後には、Compute サービスのデータベースにデータを投入します。

重要

この手順は、データベースの初期化とデータ投入するために、1 回のみ実行する必要があります。Compute サービスをホストするシステムの追加時には繰り返す必要はありません。

手順8.11 Compute サービスのデータベースへのデータ投入

1. **`openstack-nova-conductor`** サービスのインスタンスをホストしているシステムにログインします。

2. **nova** ユーザーに切り替えます。

```
# su nova -s /bin/sh
```

3. **`/etc/nova/nova.conf`** で特定されているデータベースを初期化し、データを投入します。

```
$ nova-manage db sync
```

8.2.14. Compute サービスの起動

手順8.12 Compute サービスの起動

1. Libvirt を使用するには、**`messagebus`** を有効化して実行する必要があります。サービスを起動します。

```
# systemctl start messagebus.service
```

2. Compute サービスを使用するには、**libvirtd** サービスを有効化して実行する必要があります。

```
# systemctl start libvirtd.service
# systemctl enable libvirtd.service
```

3. API のインスタンスをホストする各システムで API サービスを起動します。各 API インスタンスは、Identity サービスのデータベースで定義された独自のエンドポイントが設定されているか、エンドポイントとしての機能を果たすロードバランサーによってポイントされる必要がある点に注意してください。サービスを起動してブート時に起動するように設定します。

```
# systemctl start openstack-nova-api.service
# systemctl enable openstack-nova-api.service
```

4. スケジューラーのインスタンスをホストする各システムでスケジューラーを起動します。サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-nova-scheduler.service
# systemctl enable openstack-nova-scheduler.service
```

5. コンダクターのインスタンスをホストする各システムでコンダクターを起動します。コンピュートノードからデータベースへの直接のアクセスを制限することによるセキュリティー上のメリットがなくってしまうので、このサービスは、すべてのコンピュートノードでは実行しないことを推奨している点に注意してください。サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-nova-conductor.service
# systemctl enable openstack-nova-conductor.service
```

6. 仮想マシンインスタンスをホストする予定の全システムで Compute サービスを起動します。サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-nova-compute.service
# systemctl enable openstack-nova-compute.service
```

7. 環境設定によっては、以下のようなサービスも起動する必要がある場合があります。

openstack-nova-cert

Compute サービスに対して EC2 API を使用する場合に必要とされる X509 証明書サービス

注記

Compute サービスに対して EC2 API を使用する場合は、**nova.conf** 設定ファイルでオプションを設定する必要があります。詳しい説明は、『Red Hat OpenStack Platform Configuration Reference Guide』の「Configuring the EC2 API」の項を参照してください。このガイドは以下のリンクから入手できます。

https://access.redhat.com/site/documentation/ja/Red_Hat_Enterprise_Linux_Ope

`openstack-nova-network`

Nova ネットワーキングサービス。OpenStack Networking がインストール/設定済みの場合、またはこれからインストール/設定する予定の場合には、このサービスは**起動してはならない**点に注意してください。

`openstack-nova-objectstore`

Nova オブジェクトストレージサービス。新規デプロイメントには、Object Storage サービス (Swift) の使用が推奨されます。

第9章 ORCHESTRATIONサービスのインストール

9.1. ORCHESTRATION サービスのパッケージのインストール

Orchestration サービスには、以下のパッケージが必要です。

openstack-heat-api
: OpenStack のネイティブの REST API を Orchestration エンジンに提供します。

openstack-heat-api-cfn
: AWS CloudFormation と互換性のある API を Orchestration エンジンサービスに提供します。

openstack-heat-common
: Orchestration の全サービスに共通するコンポーネントを提供します。

openstack-heat-engine
: テンプレートを起動し、イベントを API に送り返すための OpenStack API を提供します。

openstack-heat-api-cloudwatch
: AWS CloudWatch と互換性のある API を Orchestration エンジンサービスに提供します。

heat-cfntools
: **`heat`** によりプロビジョニングされるクラウドインスタンスに必要なツールを提供します。

python-heatclient
: Python API およびコマンドラインスクリプトを提供します。Orchestration API のクライアントは、これらの両方で構成されます。

openstack-utils
: 設定ファイルの編集をはじめとする数々のタスクに役立つサポートユーティリティーを提供します。

パッケージをインストールします。

```
# yum install -y openstack-heat-* python-heatclient openstack-utils
```

9.2. ORCHESTRATION サービスの設定

Orchestration サービスを設定するには、以下のタスクを行う必要があります。

- Orchestration サービス用のデータベースの設定
- 各 Orchestration API サービスと対応する IP アドレスのバインド
- Orchestration サービス用のアイデンティティーレコードの作成/設定
- Orchestration サービスが Identity サービスを使用して認証を行う方法の設定

以下のセクションでは、これらの各手順について詳しく説明します。

9.2.1. Orchestration サービスのデータベース接続の設定

Orchestration サービスにより使用されるデータベースおよびデータベースユーザーを作成します。Orchestration サービスで使用されるデータベース接続文字列は、**/etc/heat/heat.conf** ファイルで定義されます。有効なデータベースサーバーを参照するように更新してから、サービスを起動する必要があります。以下の手順で記載のステップはすべて、データベースサーバーに **root** ユーザーでログインして実行する必要があります。

手順9.1 Orchestration サービスのデータベースの接続の設定

1. データベースサービスに接続します。

   ```
   # mysql -u root -p
   ```

2. **heat** データベースを作成します。

   ```
   mysql> CREATE DATABASE heat;
   ```

3. **heat** という名前のデータベースユーザーを作成して、**heat** データベースへのアクセスを許可します。

   ```
   mysql> GRANT ALL ON heat.* TO 'heat'@'%' IDENTIFIED BY 'PASSWORD';
   mysql> GRANT ALL ON heat.* TO 'heat'@'localhost' IDENTIFIED BY
   'PASSWORD';
   ```

   PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. データベースの特権をフラッシュして、設定が即時に反映されるようにします。

   ```
   mysql> FLUSH PRIVILEGES;
   ```

5. **mysql** クライアントを終了します。

   ```
   mysql> quit
   ```

6. **sql_connection** 設定キーの値を設定します。

   ```
   # openstack-config --set /etc/heat/heat.conf \
      DEFAULT sql_connection mysql://heat:PASSWORD@IP/heat
   ```

   以下の値を置き換えてください。

   - *PASSWORD* は **heat** データベースユーザーのパスワードに置き換えます。
   - *IP* は、Identity サーバーの IP アドレスまたはホスト名に置き換えます。

7. **heat** ユーザーとしてデータベースを同期します。

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

重要

この接続設定キーに指定する IP アドレスまたはホスト名は、Orchestration サービスのデータベースの作成時に Orchestration サービスのデータベースユーザーがアクセスを許可された IP アドレスまたはホスト名と一致する必要があります。また、データベースがローカルでホストされ、Orchestration サービスのデータベースの作成時に「localhost」へのアクセス権を付与した場合には、「localhost」と入力する必要があります。

9.2.2. 各 Orchestration API サービスのバインドアドレスの制限

データベースを設定した後には、各 Orchestration API サービスの **bind_host** 設定を設定します。この設定は、サービスが受信接続に使用する必要のある IP アドレスを制御します。

各 Orchestration API サービスの **bind_host** 設定を設定します。

```
# openstack-config --set /etc/heat/heat.conf
   heat_api bind_host IP
# openstack-config --set /etc/heat/heat.conf
   heat_api_cfn bind_host IP
# openstack-config --set /etc/heat/heat.conf
   heat_api_cloudwatch bind_host IP
```

IP は、対応する API が使用する必要のあるアドレスに置き換えます。

9.2.3. Orchestration サービス用のアイデンティティーレコードの作成

Orchestration サービスで必要な Identity サービスを作成して設定します。これらのエントリーは、Orchestration サービスによって提供されるボリューム機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identity サービスサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順9.2 Orchestration サービス用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **heat** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password PASSWORD heat
+----------+----------------------------------+
| Field    | Value                            |
+----------+----------------------------------+
| email    | None                             |
| enabled  | True                             |
```

```
| id       | 5902673a8d3a4552b813dff31717476b |
| name     | heat                             |
| username | heat                             |
+----------+----------------------------------+
```

PASSWORD は、Orchestration サービスが Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **services** テナントのコンテキスト内で、**heat** ユーザーと **admin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
heat admin
```

4. **heat** および **heat-cfn** のサービスエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name heat
orchestration
[(keystone_admin)]# openstack service create --name heat-cfn
cloudformation
```

5. **heat** および **heat-cfn** のサービス用のエンドポイントエントリーを作成します。

```
[(keystone_admin)]# openstack endpoint create \
   --publicurl 'HEAT_CFN_IP:8000/v1' \
   --adminurl 'HEAT_CFN_IP:8000/v1' \
   --internalurl 'HEAT_CFN_IP:8000/v1' \
   --region RegionOne \
   heat-cfn
[(keystone_admin)]# openstack endpoint create \
   --publicurl 'HEAT_IP:8004/v1/%(tenant_id)s' \
   --adminurl 'HEAT_IP:8004/v1/%(tenant_id)s' \
   --internalurl 'HEAT_IP:8004/v1/%(tenant_id)s' \
   --region RegionOne \
   heat
```

以下の値を置き換えてください。

- *HEAT_CFN_IP* は、**heat-cfn** サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。
- *HEAT_IP* は、**heat** サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

重要

HEAT_CFN_IP および *HEAT_IP* の値には、**http://** プレフィックスを付けます。

9.2.3.1. Orchestration サービス用の必須 Identity ドメインの作成

Orchestration サービスは、独自の Identity ドメインが必要です。このドメインを利用して、ユーザーを作成して、**heat** スタックが所有するインスタンス内にデプロイされた認証情報を関連付けることができます。また、別のドメインを使用することで、インスタンスとスタックをデプロイするユーザーを分

離することができます。これにより、一般ユーザーは管理者権限がなくとも、このような認証情報を必要とする **heat** スタックをデプロイすることができます。

手順9.3 Orchestration サービス用の Identity サービスのドメインの作成

1. **heat** ドメインを作成します。

```
# openstack --os-url=http://IDENTITY_IP:5000/v3 \
   --os-identity-api-version=3 \
   --description "Owns users and projects created by heat"
   domain create heat
```

   IP は、Identity サービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

   このコマンドにより、**heat** ドメインのドメイン ID が返されるはずです。この ID (*HEAT_DOMAIN_ID*) は次のステップで使用します。

2. **heat** ドメイン内で管理者権限を持つことのできる **heat_domain_admin** という名前のユーザーを作成します。

```
# openstack --os-url=http://IDENTITY_IP:5000/v3 \
   --os-identity-api-version=3 user create heat_domain_admin \
   --password PASSWORD \
   --domain HEAT_DOMAIN_ID \
   --description "Manages users and projects created by heat"
```

   PASSWORD は、このユーザーのパスワードに置き換えます。上記のコマンドによりユーザー ID (*DOMAIN_ADMIN_ID*) が返されます。この ID は、次のステップで使用します。

3. **heat_domain_admin** ユーザーに、**heat** ドメイン内の管理者権限を付与します。

```
# openstack --os-url=http://IDENTITY_IP:5000/v3 \
   --os-identity-api-version=3 role add --user DOMAIN_ADMIN_ID \
   --domain HEAT_DOMAIN_ID admin
```

4. Orchestration サービスをホストするサーバー上で、Orchestration サービスが **heat** ドメインとユーザーを使用するように設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT stack_domain_admin_password DOMAIN_PASSWORD
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT stack_domain_admin heat_domain_admin
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT stack_user_domain HEAT_DOMAIN_ID
```

9.2.4. Orchestration サービスの認証設定

Orchestration サービスが認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、Orchestration サービスをホストする各システムに **root** ユーザーとしてログインして実行する必要があります。

手順9.4 Orchestration サービスが Identity サービスを使用して認証を行うための設定

1. Orchestration サービスが正しいテナントとして認証を行うように設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   keystone_authtoken admin_tenant_name services
```

services は、Orchestration サービスを使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

2. Orchestration サービスが **heat** 管理ユーザーアカウントを使用して認証を行うように設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   keystone_authtoken admin_user heat
```

3. Orchestration サービスが正しい **heat** 管理ユーザーアカウントパスワードを使用するように設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   keystone_authtoken admin_password PASSWORD
```

PASSWORD は、**heat** ユーザーの作成時に設定したパスワードに置き換えます。

4. Orchestration サービスが使用する必要のある Identity サービスのホストを設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   keystone_authtoken service_host KEYSTONE_HOST
# openstack-config --set /etc/heat/heat.conf \
   keystone_authtoken auth_host KEYSTONE_HOST
# openstack-config --set /etc/heat/heat.conf \
   keystone_authtoken auth_uri http://KEYSTONE_HOST:35357/v2.0
# openstack-config --set /etc/heat/heat.conf \
   keystone_authtoken keystone_ec2_uri
http://KEYSTONE_HOST:35357/v2.0
```

KEYSTONE_HOST は、Identity サービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。Identity サービスが同じシステムでホストされている場合には、**127.0.0.1** を使用してください。

5. 仮想マシンインスタンスの接続先となる **heat-api-cfn** および **heat-api-cloudwatch** のサービスのホスト名を設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT heat_metadata_server_url HEAT_CFN_HOST:8000
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT heat_waitcondition_server_url
HEAT_CFN_HOST:8000/v1/waitcondition
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT heat_watch_server_url HEAT_CLOUDWATCH_HOST:8003
```

以下の値を置き換えてください。

- *HEAT_CFN_HOST* は、**heat-api-cfn** サービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

- *HEAT_CLOUDWATCH_HOST* は、**`heat-api-cloudwatch`** サービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

重要

全サービスが同じシステム上でホストされている場合でも、**`127.0.0.1`** はどのサービスホスト名にも**使用しないでください**。この IP アドレスは、各インスタンスのローカルホストを参照するので、そのインスタンスが実際のサービスに到達できなくなります。

6. アプリケーションテンプレートは、オーケストレーションに WaitCondition とシグナル送信を使用します。進捗データを受信するユーザーの Identity ロールを定義してください。デフォルトでは、このロールは **`heat_stack_user`** です。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT heat_stack_user_role heat_stack_user
```

9.2.5. Orchestration サービスのための RabbitMQ メッセージブローカーの設定

RabbitMQ はデフォルト (かつ推奨の) メッセージブローカーです。RabbitMQ メッセージングサービスは、rabbitmq-server パッケージにより提供されます。以下の手順で記載する全ステップは、Orchestration コントローラーサービスをホストするシステムに **`root`** ユーザーとしてログインして実行する必要があります。

手順9.5 Orchestration サービスが RabbitMQ メッセージブローカーを使用するための設定

1. RPC バックエンドとして RabbitMQ を設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT rpc_backend heat.openstack.common.rpc.impl_kombu
```

2. Orchestration サービスが RabbitMQ ホストに接続するように設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT rabbit_host RABBITMQ_HOST
```

RABBITMQ_HOST は、メッセージブローカーの IP アドレスまたはホスト名に置き換えます。

3. メッセージブローカーのポートを **`5672`** に設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT rabbit_port 5672
```

4. RabbitMQ の設定時に Orchestration サービス用に作成した RabbitMQ ユーザー名とパスワードを設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT rabbit_userid heat
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT rabbit_password HEAT_PASS
```

heat および *HEAT_PASS* は、Orchestration サービス用に作成した RabbitMQ ユーザー名とパスワードに置き換えます。

5. RabbitMQ の起動時に、**heat** ユーザーに全リソースに対するパーミッションが付与されます。このアクセスは、特別に仮想ホスト **/** を介して行われます。Orchestration サービスがこの仮想ホストに接続されるように設定します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT rabbit_virtual_host /
```

9.2.6. Orchestration サービスとメッセージブローカーとの間の SSL 通信の有効化

メッセージブローカーで SSL を有効化した場合は、Orchestration サービスも相応に設定する必要があります。以下の手順では、エクスポートしたクライアントの証明書とキーファイルが必要です。これらのファイルのエクスポートの方法に関する説明は、「クライアント用 SSL 証明書のエクスポート」を参照してください。

手順9.6 Orchestration サービスと RabbitMQ メッセージブローカーとの間の SSL 通信の有効化

1. メッセージブローカーとの SSL 通信を有効化します。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT rabbit_use_ssl True
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT kombu_ssl_certfile /path/to/client.crt
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT kombu_ssl_keyfile /path/to/clientkeyfile.key
```

以下の値を置き換えてください。

   - */path/to/client.crt* はエクスポートされたクライアント証明書の絶対パスに置き換えます。
   - */path/to/clientkeyfile.key* はエクスポートされたキーファイルの絶対パスに置き換えます。

2. 証明書がサードパーティーの認証局 (CA) によって署名されている場合には、次のコマンドを実行する必要もあります。

```
# openstack-config --set /etc/heat/heat.conf \
   DEFAULT kombu_ssl_ca_certs /path/to/ca.crt
```

/path/to/ca.crt は、サードパーティー CA によって提供された CA ファイルの絶対パスに置き換えます (詳細は 「RabbitMQ メッセージブローカーでの SSL の有効化」 を参照)。

9.3. ORCHESTRATION サービスの起動

手順9.7 Orchestration サービスの起動

1. Orchestration API サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-heat-api.service
# systemctl enable openstack-heat-api.service
```

2. Orchestration AWS CloudFormation と互換性のある API サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-heat-api-cfn.service
# systemctl enable openstack-heat-api-cfn.service
```

3. Orchestration AWS CloudWatch と互換性のある API サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-heat-api-cloudwatch.service
# systemctl enable openstack-heat-api-cloudwatch.service
```

4. テンプレートの起動やイベントを API に送信するために Orchestration API サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-heat-engine.service
# systemctl enable openstack-heat-engine.service
```

9.4. ORCHESTRATION テンプレートを使用したスタックのデプロイ

Orchestration エンジンサービスは、テンプレート (**.template** ファイルで定義) を使用してインスタンス、IP アドレス、ボリューム、およびその他のタイプのスタックを起動します。**heat** ユーティリティーは、スタックの作成/設定/起動を可能にするコマンドラインインターフェースです。

注記

openstack-heat-templates パッケージには、Orchestration のコア機能のテストに使用することができるサンプルテンプレートが含まれています。また、テンプレート関連のスクリプトと変換ツールも同梱されています。このパッケージをインストールするには、以下のコマンドを実行してください。

```
# yum install -y openstack-heat-templates
```

一部の Orchestration テンプレートは、**openstack-heat-api-cfn** サービスへのアクセスが必要なインスタンスを起動します。このようなインスタンスは、**openstack-heat-api-cloudwatch** サービスおよび **openstack-heat-api-cfn** サービスとの通信が可能である必要があります。これらのサービスが使用する IP アドレスおよびポートは、**/etc/heat/heat.conf** ファイルで **heat_metadata_server_url** および **heat_watch_server_url** として設定されている値です。

これらのサービスへのアクセスを許可するには、**openstack-heat-api-cloudwatch** (8003)、**openstack-heat-api-cfn** (8000)、**openstack-api** (8004) が使用するポートを開放する必要があります。

手順9.8 Orchestration テンプレートを使用したスタックのデプロイ

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

2. **8003**、**8000**、**8004** のポートの TCP トラフィックを許可する以下の INPUT ルールを追加します。

```
-A INPUT -i BR -p tcp --dport 8003 -j ACCEPT
-A INPUT -i BR -p tcp --dport 8000 -j ACCEPT
-A INPUT -p tcp -m multiport --dports 8004 -j ACCEPT
```

BR は、Orchestration テンプレートから起動したインスタンスが使用するブリッジのインターフェースに置き換えます。**nova-network** を使用しない場合、または Orcestration サービスおよび **nova-compute** が同じサーバーでホストされない場合には、**INPUT** ルールに **-i *BR*** パラメーターを含めないでください。

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

4. **iptables** サービスを再起動して、ファイアウォールの変更を有効にします。

```
# systemctl restart iptables.service
```

5. アプリケーションを起動します。

```
# heat stack-create STACKNAME \
   --template-file=PATH_TEMPLATE \
 --parameters="PARAMETERS"
```

以下の値を置き換えてください。

- *STACKNAME* は、そのスタックに割り当てる名前に置き換えます。この名前は、**heat stack-list** コマンドを実行する際に表示されます。
- *PATH_TEMPLATE* は、**.template** ファイルへのパスに置き換えます。
- *PARAMETERS* は、使用するスタック作成パラメーターのセミコロン区切りリストです。サポートされているパラメーターは、テンプレートファイル自体で定義されています。

9.5. TELEMETRY および ORCHESTRATION サービスの統合

Orchestration サービスは **heat stack-create** コマンドで作成したスタックのリソース使用状況の監視に Telemetry サービス (およびそのアラーム) を使用することができます。この機能を有効にするには、それに応じて Orchestration サービスのインストールと設定を行う必要があります (詳細は「Telemetry サービスのデプロイメントの概要」を参照)。

Telemetry サービスのアラームは、**autoscaling** 機能で使用されます。この機能により、特定のリソースの使用率が一定のレベルに達すると Orchestration サービスが自動的にスタックを作成できるようになります。Orchestration が Telemetry アラームを使用できるようにするには、**/etc/heat/environment.d/default.yaml** の **resource_registry** セクションで以下の行をコメント解除または追加します。

```
"AWS::CloudWatch::Alarm":
"file:///etc/heat/templates/AWS_CloudWatch_Alarm.yaml"
```

第10章 DASHBOARD のインストール

10.1. DASHBOARD サービスの要件

以下の方法で、Dashboard サービスをホストするシステムを設定する必要があります。

- (セキュリティーの関係上)、httpd、mod_wsgi、mod_ssl パッケージをインストールする必要があります。

  ```
  # yum install -y mod_wsgi httpd mod_ssl
  ```

- システムは、Identity サービスおよびその他の OpenStack API サービス (OpenStack Compute、Block Storage、Object Storage、Image および Networking の各サービス) に接続されている必要があります。

- Identity サービスのエンドポイントの URL を知っておく必要があります。

10.2. DASHBOARD パッケージのインストール

Dashboard サービスに必要なパッケージをインストールします。

注記

Dashboard サービスは設定可能なバックエンドセッションストアを使用します。以下のインストールでは、セッションストアとして **memcached** を使用します。

以下のパッケージが必要です。

openstack-dashboard
: OpenStack Dashboard サービスを提供します。

memcached を使用する場合には、以下のパッケージもインストールする必要があります。

memcached
: データベースの負荷を軽減することにより、動的な Web アプリケーションを高速化するメモリーオブジェクトキャッシングシステム

python-memcached
: **memcached** デーモンへの Python インターフェース

手順10.1 Dashboard パッケージのインストール

1. 必要に応じて **memcached** オブジェクトのキャッシュシステムをインストールします。

   ```
   # yum install -y memcached python-memcached
   ```

2. Dashboard パッケージをインストールします。

   ```
   # yum install -y openstack-dashboard
   ```

10.3. APACHE WEB サービスの起動

Dashboard は Django (Python) Web アプリケーションであるため、**httpd** サービスによってホストされます。サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start httpd.service
# systemctl enable httpd.service
```

10.4. DASHBOARD の設定

10.4.1. 接続とロギングの設定

ユーザーがダッシュボードに初めて接続する前には、**/etc/openstack-dashboard/local_settings** ファイルで以下のパラメーターを設定します (サンプルファイルは、https://access.redhat.com/site/documentation/ja/Red_Hat_Enterprise_Linux_OpenStack_Platform に掲載されている『Configuration Reference Guide』を参照してください)。

手順10.2 Dashboard の接続およびロギングの設定

1. **ALLOWED_HOSTS** パラメーターにアプリケーションがサービス提供可能なホスト/ドメイン名のコンマ区切りリストで指定します。例は以下のとおりです。

   ```
   ALLOWED_HOSTS = ['horizon.example.com', 'localhost',
   '192.168.20.254', ]
   ```

2. **CACHES** の設定は、**memcached**の値を使用して更新します。

   ```
   SESSION_ENGINE = 'django.contrib.sessions.backends.cache'
   CACHES = {
    'default': {
     'BACKEND' : 'django.core.cache.backends.memcached.MemcachedCache',
     'LOCATION' : 'memcacheURL:port',
    }
   }
   ```

   以下の値を置き換えてください。

   - *memcacheURL* は、**memcached** がインストールされたホストの IP アドレスに置き換えます。
   - *port* は **/etc/sysconfig/memcached** ファイルの **PORT** パラメーターからの値に置き換えます。

3. Identity サービスのエンドポイントのホスト URL を以下のように指定します。

   ```
   OPENSTACK_KEYSTONE_URL="127.0.0.1"
   ```

4. Dashboard のタイムゾーンを更新します。

   ```
   TIME_ZONE="UTC"
   ```

   タイムゾーンは、Dashboard GUI を使用して更新することも可能です。

5. 設定の変更を有効にするには、Apache サーバーを再起動します。

```
# systemctl restart httpd.service
```

注記

HORIZON_CONFIG ディレクトリーには Dashboard のすべての設定が含まれます。サービスが Dashboard に含まれているかどうかは、Identity サービスの Service Catalog configuration によって決定します。

注記

django-secure モジュールを使用して、推奨プラクティスと最新のブラウザー保護メカニズムの大半を有効にすることをお勧めします。詳しい情報は http://django-secure.readthedocs.org/en/latest/ (『django-secure』) を参照してください。

10.4.2. HTTPS で使用するための Dashboard の設定

デフォルトのインストールでは、暗号化されていないチャンネル (HTTP) を使用していますが、Dashboard の SSL サポートを有効にすることが可能です。

手順10.3 HTTPS を使用するためのダッシュボードの設定

1. テキストエディターで **/etc/openstack-dashboard/local_settings** ファイルを開き、以下のパラメーターをアンコメントします。

```
SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTOCOL', 'https')
CSRF_COOKIE_SECURE = True
SESSION_COOKIE_SECURE = True
```

後の 2 つの設定は、Dashboard のクッキーを HTTPS 接続のみで送信するようにブラウザーに指示して、セッションが HTTP 上では機能しないようにします。

2. テキストエディターで **/etc/httpd/conf/httpd.conf** ファイルを開き、以下の行を追加します。

```
NameVirtualHost *:443
```

3. テキストエディターで **/etc/httpd/conf.d/openstack-dashboard.conf** ファイルを開きます。

a. 以下の行を削除します。

```
WSGIDaemonProcess dashboard
WSGIProcessGroup dashboard
WSGISocketPrefix run/wsgi

WSGIScriptAlias /dashboard /usr/share/openstack-
dashboard/openstack_dashboard/wsgi/django.wsgi
Alias /static /usr/share/openstack-dashboard/static/

<Directory /usr/share/openstack-
dashboard/openstack_dashboard/wsgi>
```

```
    <IfModule mod_deflate.c>
      SetOutputFilter DEFLATE
      <IfModule mod_headers.c>
        # Make sure proxies donât deliver the wrong content
        Header append Vary User-Agent env=!dont-vary
      </IfModule>
    </IfModule>

  Order allow,deny
  Allow from all
</Directory>
<Directory /usr/share/openstack-dashboard/static>
  <IfModule mod_expires.c>
    ExpiresActive On
    ExpiresDefault "access 6 month"
  </IfModule>
  <IfModule mod_deflate.c>
    SetOutputFilter DEFLATE
  </IfModule>

  Order allow,deny
  Allow from all
</Directory>

  RedirectMatch permanent ^/$
https://xxx.xxx.xxx.xxx:443/dashboard
```

b. 以下の行を追加します。

```
WSGIDaemonProcess dashboard
WSGIProcessGroup dashboard
WSGISocketPrefix run/wsgi
LoadModule ssl_module modules/mod_ssl.so

<VirtualHost *:80>
  ServerName openstack.example.com
  RedirectPermanent / https://openstack.example.com/
</VirtualHost>

<VirtualHost *:443>
    ServerName openstack.example.com
    SSLEngine On
    SSLCertificateFile /etc/httpd/SSL/openstack.example.com.crt
    SSLCACertificateFile /etc/httpd/SSL/openstack.example.com.crt
    SSLCertificateKeyFile
/etc/httpd/SSL/openstack.example.com.key
    SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-
shutdown
    WSGIScriptAlias / /usr/share/openstack-
dashboard/openstack_dashboard/wsgi/django.wsgi
    WSGIDaemonProcess horizon user=apache group=apache
processes=3 threads=10
    RedirectPermanent /dashboard https://openstack.example.com
    Alias /static /usr/share/openstack-dashboard/static/
    <Directory /usr/share/openstack-
dashboard/openstack_dashboard/wsgi>
```

```
      Order allow,deny
      Allow from all
    </Directory>
</VirtualHost>

<Directory /usr/share/openstack-dashboard/static>
  <IfModule mod_expires.c>
    ExpiresActive On
    ExpiresDefault "access 6 month"
  </IfModule>
  <IfModule mod_deflate.c>
    SetOutputFilter DEFLATE
  </IfModule>

Order allow,deny
Allow from all
</Directory>

RedirectMatch permanent ^/$ /dashboard/
```

新しい設定では、Apache がポート 443 をリッスンし、セキュリティーで保護されていない要求をすべて HTTPS プロトコルにリダイレクトします。**<VirtualHost *:443>** のセクションでは、秘密鍵、公開鍵、証明書など、このプロトコルに必要なオプションを定義します。

4. Apache サービスと **memcached** サービスを再起動します。

```
# systemctl restart httpd.service
# systemctl restart memcached.service
```

ブラウザーで HTTP バージョンの Dashboard を使用すると、ユーザーは HTTPS バージョンのページにリダイレクトされるようになりました。

10.4.3. Dashboard のデフォルトロールの変更

デフォルトでは、Dashboard サービスは Identity によって自動的に作成される **_member_** という Identity ロールを使用します。これは、一般ユーザーに適切なロールです。別のロールを作成することを選択し、Dashboard がそのロールを使用するように設定する場合には、このロールは、Dashboard を使用する前に Identity サービスで作成してから Dashboard が使用するように設定しておく必要があります。

以下の手順は、Identity サービスのサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順10.4 Dashboard のデフォルトロールの変更

1. Keystone 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 ~/keystonerc_admin
```

2. 新しいロールを作成します。

```
[(keystone_admin)]# keystone role-create --name NEW_ROLE
+----------+----------------------------------+
| Property |              Value               |
```

```
+----------+----------------------------------+
| id       | 8261ac4eabcc4da4b01610dbad6c038a |
| name     |             NEW_ROLE             |
+----------+----------------------------------+
```

NEW_ROLE は、そのロールの名前に置き換えます。

3. テキストエディターで **/etc/openstack-dashboard/local_settings** ファイルを開き、以下のパラメーターの値を変更します。

```
OPENSTACK_KEYSTONE_DEFAULT_ROLE = 'NEW_ROLE'
```

NEW_ROLE は、前のステップで作成したロールの名前に置き換えます。

4. Apache サービスを再起動し、変更を有効にします。

```
# systemctl restart httpd.service
```

10.4.4. SELinux の設定

SELinux は、アクセス制御を提供する Red Hat Enterprise Linux のセキュリティー機能です。SELinux のステータスの値は、「Enforcing」、「Permissive」、および「Disabled」です。SELinux が「Enforcing」モードに設定されている場合には、SELinux ポリシーを変更して、**httpd** サービスから Identity サーバーへの通信を許可する必要があります。この設定変更は、SELinux が「Permissive」モードに設定されている場合にも推奨されます。

手順10.5 SELinux が Apache サービスからの接続を許可するように設定

1. システム上の SELinux のステータスを確認します。

```
# getenforce
```

2. 出力された値が「Enforcing」、「Permissive」の場合には、**httpd** サービスと Identity サービスの間の接続が可能です。

```
# setsebool -P httpd_can_network_connect on
```

10.4.5. Dashboard のファイアウォール設定

ユーザーが Dashboard にアクセスできるようにするには、接続を許可するようにシステムのファイアウォールを設定する必要があります。**httpd** サービスと Dashboard は、HTTP 接続と HTTPS 接続の両方をサポートします。以下の手順に記載するステップはすべて、**httpd** サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

注記

認証情報およびその他の情報を保護するには、HTTPS 接続のみを有効化することを推奨します。

手順10.6 Dashboard のトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** の設定ファイルを開きます。

- HTTPS のみを使用した受信接続を許可するには、以下のファイアウォールルールを追加します。

  ```
  -A INPUT -p tcp --dport 443 -j ACCEPT
  ```

- HTTP および HTTPS の両方を使用した受信接続を許可するには、以下のファイアウォールルールを追加します。

  ```
  -A INPUT -p tcp -m multiport --dports 80,443 -j ACCEPT
  ```

2. **iptables** サービスを再起動して、ファイアウォールの変更を有効にします。

   ```
   # systemctl restart iptables.service
   ```

重要

上記のルールにより、全リモートホストから Dashboard サービスを実行するサーバーへの通信がポート 80 または 443 で 許可されます。より制限の厳しいファイアウォールルールの作成についての説明は、以下のリンクで 『Red Hat Enterprise Linux セキュリティーガイド』を参照してください。

https://access.redhat.com/site/documentation/ja-JP/Red_Hat_Enterprise_Linux/

10.5. DASHBOARD のインストールの検証

Dashboard のインストールと設定が正常に完了すると、Web ブラウザーでユーザーインターフェースにアクセスすることができます。*HOSTNAME* は、Dashboard サービスをインストールしたサーバーのホスト名または IP アドレスに置き換えてください。

- HTTPS

  ```
  https://HOSTNAME/dashboard/
  ```

- HTTP

  ```
  http://HOSTNAME/dashboard/
  ```

ログイン画面が表示されたら、OpenStack ユーザーの認証情報を使用してログインします。

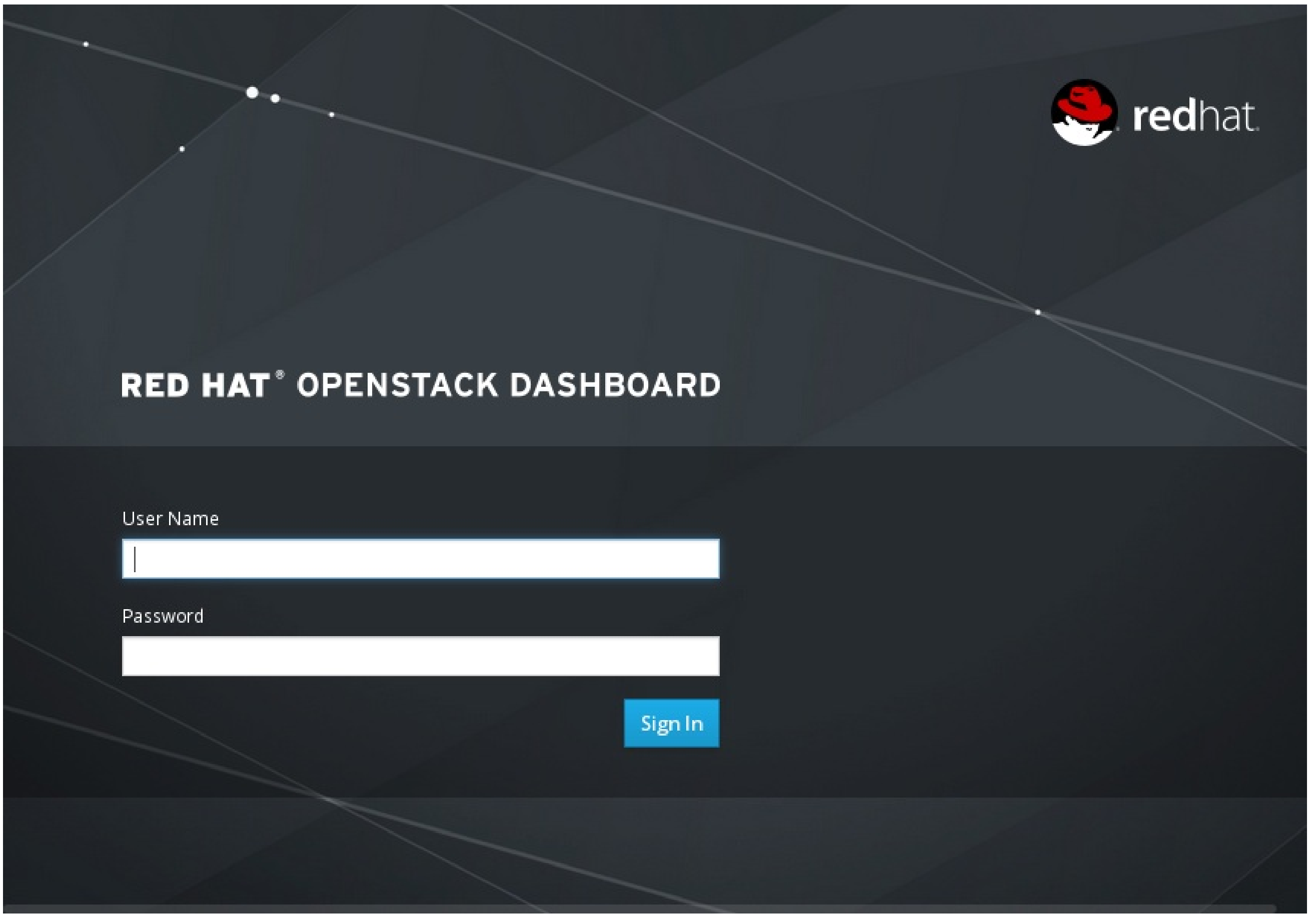

図10.1 Dashboard のログイン画面

第11章 DATA PROCESSING サービスのインストール

11.1. DATA PROCESSING サービスのパッケージのインストール

Data Processing サービスをホストするサーバーで、openstack-sahara-api および openstack-sahara-engine パッケージをインストールします。

```
# yum install openstack-sahara-api openstack-sahara-engine
```

このパッケージは、Data Processing CLI クライアント (**sahara** および **sahara-db-manage**) と **`openstack-sahara-api`** サービスを提供します。

11.2. DATA PROCESSING サービスの設定

Data Processing サービス (Sahara) を設定するには、以下のタスクを行う必要があります。

- Data Processing サービスのデータベース接続の設定
- Data Processing API サーバーが Identity サービスで認証を行うための設定
- ファイアウォールが Data Processing サービスのサービストラフィックを (ポート **`8386`** で) 許可する設定

11.2.1. Data Processing サービスデータベースの作成

Data Processing API サービスで使用するデータベースおよびデータベースのユーザーを作成します。Data Processing サービスのデータベースの接続文字列は、**`/etc/sahara/sahara.conf`** ファイルで定義します。 Data Processing API サービス (**`openstack-sahara-api`**) を起動する前に、有効なデータベースサーバーを参照するように更新する必要があります。

手順11.1 Data Processing API サービスのデータベースの作成および設定

1. データベースサービスに接続します。

   ```
   # mysql -u root -p
   ```

2. **`sahara`** データベースを作成します。

   ```
   mysql> CREATE DATABASE sahara;
   ```

3. **`sahara`** データベースユーザーを作成し、**`sahara`** データベースへのアクセスを許可します。

   ```
   mysql> GRANT ALL ON sahara.* TO 'sahara'@'%' IDENTIFIED BY
   'PASSWORD';
   mysql> GRANT ALL ON sahara.* TO 'sahara'@'localhost' IDENTIFIED BY
   'PASSWORD';
   ```

   PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. **`mysql`** クライアントを終了します。

```
mysql> quit
```

5. **sql_connection** 設定キーの値を設定します。

```
# openstack-config --set /etc/sahara/sahara.conf \
    database connection mysql://sahara:PASSWORD@IP/sahara
```

以下の値を置き換えてください。

- *PASS* は選択したデータベースユーザーのパスワードに置き換えます。
- *IP* は、データベースサービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

6. **sahara** データベースのスキーマを設定します。

```
# sahara-db-manage --config-file /etc/sahara/sahara.conf upgrade
head
```

重要

この接続設定キーに指定する IP アドレスまたはホスト名は、Data Processing サービスのデータベースの作成時に Data Processing サービスのデータベースユーザーがアクセスを許可された IP アドレスまたはホスト名と一致する必要があります。また、データベースがローカルでホストされ、Data Processing サービスのデータベースの作成時に「localhost」へのアクセス権を付与した場合には、「localhost」と入力する必要があります。

11.2.2. Data Processing サービス用のアイデンティティーレコードの作成

Data Processing サービスで必要な Identity サービスを作成して設定します。これらのエントリーは、Data Processing サービスによって提供されるボリューム機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identity サービスのサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順11.2 Data Processing サービス用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **sahara** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password PASSWORD sahara
```

```
+----------+----------------------------------+
| Field    | Value                            |
+----------+----------------------------------+
| email    | None                             |
| enabled  | True                             |
| id       | 1fc5b7ac48b646ab850854e565ac1cfc |
| name     | sahara                           |
| username | sahara                           |
+----------+----------------------------------+
```

PASSWORD は、Data Processing サービスが Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **services** テナントのコンテキスト内で、**sahara** ユーザーと **admin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
sahara admin
```

4. **sahara** のサービスエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name sahara \
--description "OpenStack Data Processing" \
data-processing
```

5. **sahara** エンドポイントエントリーを作成します。

```
[(keystone_admin)]# openstack endpoint create \
   --publicurl 'http://SAHARA_HOST:8386/v1.1/%(tenant_id)s' \
   --adminurl 'http://SAHARA_HOST:8386/v1.1/%(tenant_id)s' \
   --internalurl 'http://SAHARA_HOST:8386/v1.1/%(tenant_id)s' \
   --region RegionOne \
   sahara
```

SAHARA_HOST は Data Processing サービスをホストするサーバーの IP アドレスまたは完全修飾ドメイン名に置き換えます。

注記

デフォルトでは、エンドポイントはデフォルトのリージョンである **RegionOne** で作成されます (この値は大文字小文字の区別があります)。エンドポイントの作成時に異なるリージョンを指定するには、***--region*** 引数を使用して指定してください。

詳しい情報は 「サービスのリージョン」 を参照してください。

11.2.3. Data Processing サービスの認証設定

Data Processing API サービス (**openstack-sahara-api**) が認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、Data Processing API サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順11.3 Data Processing API サービスが Identity サービスを使用して認証を行うための設定

1. Data Processing API サービスが使用する必要のある Identity サービスホストを設定します。

```
# openstack-config --set /etc/sahara/sahara.conf \
   keystone_authtoken auth_uri http://IP:5000/v2.0/
# openstack-config --set /etc/sahara/sahara.conf \
   keystone_authtoken identity_uri http://IP:35357
```

   IP は、Identity サービスをホストするサーバーの IP アドレスに置き換えます。

2. Data Processing API サービスが正しいテナントとして認証を行うように設定します。

```
# openstack-config --set /etc/sahara/sahara.conf \
   keystone_authtoken admin_tenant_name services
```

   services は、Data Processing サービスを使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

3. Data Processing API サービスが **sahara** 管理ユーザーアカウントを使用して認証を行うように設定します。

```
# openstack-config --set /etc/sahara/sahara.conf \
   keystone_authtoken admin_user sahara
```

4. Data Processing API サービスが正しい **sahara** 管理ユーザーアカウントのパスワードを使用するように設定します。

```
# openstack-config --set /etc/sahara/sahara.conf \
   keystone_authtoken admin_password PASSWORD
```

   PASSWORD は、**sahara** ユーザーの作成時に設定したパスワードに置き換えます。

11.2.4. OpenStack Data Processing サービスのトラフィックを許可するためのファイアウォール設定

Data Processing サービスは、ポート **8386** で接続を受信します。このサービスノードのファイアウォールは、このポートのネットワークトラフィックを許可するように設定する必要があります。以下の手順に記載するステップはすべて、Data Processing サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順11.4 Data Processing サービスのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。

2. ポート **8386** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

```
-A INPUT -p tcp -m multiport --dports 8386 -j ACCEPT
```

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。

4. **iptables** サービスを再起動して、変更を有効にします。

```
# systemctl restart iptables.service
```

11.3. DATA PROCESSING サービスの設定と起動

手順11.5 Data Processing サービスの起動

1. OpenStack のデプロイメントで OpenStack Networking (**neutron**) を使用する場合は、Data Processing サービスを適切に設定する必要があります。

```
# openstack-config --set /etc/sahara/sahara.conf \
   DEFAULT use_neutron true
```

2. Data Processing サービスを起動して、ブート時に起動するように設定します。

```
# systemctl start openstack-sahara-api.service
# systemctl start openstack-sahara-engine.service
# systemctl enable openstack-sahara-api.service
# systemctl enable openstack-sahara-engine.service
```

第12章 TELEMETRY サービスのインストール

12.1. TELEMETRY サービスのデプロイメントの概要

Telemetry サービスは、API サーバー 1 つ、**openstack-ceilometer** エージェント 3 つで構成されます。API サーバー (openstack-ceilometer-api パッケージによって提供) は、単一または複数の中央管理サーバー上で実行され、Telemetry データベースへのアクセスを提供します。

注記

現在、**mongod** は、Telemetry サービスがサポートする唯一のデータベースサービスです。

3 つの Telemetry エージェント (およびそれぞれに対応するパッケージ) は以下のとおりです。

- 中央エージェント (openstack-ceilometer-central により提供): 集中サーバー上で実行され、パブリックの REST API をポーリングして (通知を介して、またはハイパーバイザーレイヤーから) 表示が可能でないリソースの統計を活用します。
- コレクター (openstack-ceilometer-collector により提供): 単一または複数の中央管理サーバーで実行され、リソースの使用状況に関する通知を受信します。またコレクターは、リソース使用状況の統計の解析も行って、Telemetry データベースにデータポイントとして保存します。
- コンピュートエージェント (openstack-ceilometer-compute により提供): 各 Compute サービスノードで実行され、インスタンスの使用状況の統計をポーリングします。openstack-ceilometer-compute パッケージをノードにインストールする前には、あらかじめ Compute サービスのインストールと設定を済ませておく必要があります。

各コンポーネントに対して以下の設定を行います。

- Identity サービスのトークンおよび Telemetry シークレットなどの認証
- Telemetry データベースに接続するためのデータベース接続文字列

上記のコンポーネント認証設定およびデータベース接続文字列はすべて **/etc/ceilometer/ceilometer.conf** で設定されます。このため、同じホストにデプロイされたコンポーネントは、同じ設定を共有することになります。Telemetry コンポーネントが複数のホストにデプロイされる場合には、認証の変更をこれらのホストに複製する必要があります。これは、新規設定を適用した後に **ceilometer.conf** ファイルを全ホストにコピーすることによって対応することができます。

Telemetry サービス (それぞれのホスト先に関わらず、その全コンポーネント) がデプロイされ設定された後には、Telemetry サービスにデータを送信するように各監視対象サービス (Image、Networking、Object Storage、Block Storage、および各コンピュートノード) を設定する必要があります。これに関連する設定は、各サービスの設定ファイルで行います。

12.2. TELEMETRY サービスのパッケージのインストール

Telemetry サービスには以下のパッケージが必要です。

mongodb

MongoDB データベースサーバーを提供します。Telemetry サービスは MongoDB をバックエンドデータリポジトリーとして使用します。

openstack-ceilometer-api

ceilometer API サーバーを提供します。

openstack-ceilometer-central

中央 **ceilometer** エージェントを提供します。

openstack-ceilometer-collector

ceilometer コレクターエージェントを提供します。

openstack-ceilometer-common

全 **ceilometer** サービスに共通のコンポーネントを提供します。

openstack-ceilometer-compute

各コンピュートノードで実行する必要のある **ceilometer** エージェントを提供します。

openstack-ceilometer-notification

ceilometer 通知エージェントを提供します。このエージェントは、別の OpenStack サービスからコレクターエージェントにメトリックを提供します。

python-ceilometer

ceilometer python ライブラリーを提供します。

python-ceilometerclient

ceilometer コマンドラインツールと Python API (具体的には **ceilometerclient** モジュール) を提供します。

API サーバー、中央エージェント、MongoDB データベースサービス、コレクターは異なるホストにデプロイすることが可能です。また、各コンピュートノードにコンピュートエージェントをインストールする必要もあります。このエージェントは、コンピュートノード上で実行されているインスタンスの詳しい使用状況メトリックを収集します。

同じホストに、必要なパッケージをインストールします。

```
# yum install -y mongodb openstack-ceilometer-* python-ceilometer python-ceilometerclient
```

12.3. MONGODB バックエンドの設定および TELEMETRY データベースの作成

Telemetary サービスはバックログのデータベースリポジトリーとして MongoDB サービスを使用します。**mongod** サービスを起動する前に、オプションで **mongod** が **--smallfiles** パラメーター使用して実行するように設定する必要がある場合があります。このパラメーターは、MongoDB がより小さなデフォルトのデータファイルとジャーナルサイズを使用するように設定します。これにより、MongoDB は各データファイルのサイズを制限し、512 MB に達すると新規ファイルを作成して書き込みます。

手順12.1 MongoDB バックエンドの設定および Telemetry データベースの作成

1. オプションで、**mongod** が **--smallfiles** パラメーターを指定して実行するように設定します。テキストエディターで **/etc/sysconfig/mongod** ファイルを開き、以下の行を追加します。

   ```
   OPTIONS="--smallfiles /etc/mongodb.conf"
   ```

   MongoDB は、**mongod** の起動時に **OPTIONS** セクションで指定したパラメーターを使用します。

2. MongoDB サービスを起動します。

   ```
   # systemctl start mongod.service
   ```

3. ローカルホスト以外のサーバーからデータベースにアクセスする必要がある場合には、テキストエディターで **/etc/mongod.conf** ファイルを開き、**bind_ip** を MongoDB サーバーの IP アドレスに更新します。

   ```
   bind_ip = MONGOHOST
   ```

4. テキストエディターで **/etc/sysconfig/iptables** ファイルを開き、ポート **27017** の TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載するようにしてください。

   ```
   -A INPUT -p tcp -m multiport --dports 27017 -j ACCEPT
   ```

5. **iptables** サービスを再起動して、変更を有効にします。

   ```
   # systemctl restart iptables.service
   ```

6. Telemetry サービス用のデータベースを作成します。

   ```
   # mongo --host MONGOHOST --eval '
      db = db.getSiblingDB("ceilometer");
      db.addUser({user: "ceilometer",
       pwd: "MONGOPASS",
       roles: [ "readWrite", "dbAdmin" ]})'
   ```

   これにより、**ceilometer** という名前のデータベースユーザーも作成されます。*MONGOHOST* は、MongoDB データベースをホストするサーバーの IP アドレスまたはホスト名に、*MONGOPASS* は **ceilometer** ユーザーのパスワードに置き換えます。

12.4. TELEMETRY サービスのデータベース接続の設定

Telemetry サービスが使用するデータベース接続 URL は、**/etc/ceilometer/ceilometer.conf** ファイルで定義されています。この URL は、Telemetry の API サービス (**openstack-ceilometer-api**)、通知エージェント (**openstack-ceilometer-notification**)、コレクターエージェント (**openstack-ceilometer-collector**) を起動する前に、有効なデータベースサーバーをポイントするように設定する必要があります。

以下の手順に記載するステップはすべて、**openstack-ceilometer-api** サービスおよび **openstack-ceilometer-collector** サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順12.2 Telemetry サービスのデータベース接続の設定

- データベース接続文字列の設定

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   database connection
mongodb://ceilometer:MONGOPASS@MONGOHOST/ceilometer
```

以下の値を置き換えてください。

  - *MONGOPASS* は、データベースサーバーにログインするために Telemetry サービスが必要とする **ceilometer** ユーザーのパスワードに置き換えます。データベースサーバーが必要とする場合のみ、これらの認証情報を提示してください (例: データベースサーバーが別のシステムまたはノードでホストされている場合)。
  - *MONGOHOST* は、データベースサービスをホストするサーバーの IP アドレスまたはホスト名およびポートに置き換えます。

MongoDB が同じホスト上のローカルでホストされている場合には、必要となるデータベース接続文字列は以下のとおりです。

```
mongodb://localhost:27017/ceilometer
```

12.5. TELEMETRY アイデンティティーレコードの作成

Telemetry サービスで必要な Identity サービスを作成して設定します。これらのエントリーは、Telemetry サービスによって提供されるボリューム機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

以下の手順では、管理ユーザーと **services** テナントが作成済みであることを前提としています。詳しい説明は、以下のリンクを参照してください。

- 「管理者アカウントおよび Identity サービスエンドポイントの作成」
- 「サービステナントの作成」

以下の手順は、Identity サービスのサーバーまたは **keystonerc_admin** ファイルをコピーして **keystone** コマンドラインユーティリティーをインストールした任意のマシンで実行してください。

手順12.3 Telemetry サービス用のアイデンティティーレコードの作成

1. Keystone 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 ~/keystonerc_admin
```

2. **ceilometer** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password PASSWORD --
email CEILOMETER_EMAIL ceilometer
```

以下の値を置き換えてください。

  - *PASSWORD* は、Telemetry サービスが Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

- *CEILOMETER_EMAIL* は、Telemetry サービスの使用するメールアドレスに置き換えます。

3. **ResellerAdmin** ロールを作成します。

```
[(keystone_admin)]# openstack role create ResellerAdmin
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 9276cfe40bca485bacc1775d10ef98f6 |
| name      | ResellerAdmin                    |
+-----------+----------------------------------+
```

4. **services** テナントのコンテキスト内で、**ceilometer** ユーザーと **ResellerAdmin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
ceilometer ResellerAdmin
```

5. **services** テナントのコンテキスト内で、**ceilometer** ユーザーと **admin** ロールを関連付けます。

```
[(keystone_admin)]# openstack role add --project services --user
ceilometer admin
```

6. **ceilometer** のサービスエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name ceilometer \
   --description "OpenStack Telemetry Service" \
    metering
```

7. **ceilometer** エンドポイントエントリーを作成します。

```
[(keystone_admin)]# openstack endpoint create \
   --publicurl 'IP:8777' \
   --adminurl 'IP:8777' \
   --internalurl 'IP:8777' \
   --region RegionOne \
   ceilometer
```

IP は、Telemetry サービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

注記

デフォルトでは、エンドポイントはデフォルトのリージョンである **RegionOne** で作成されます (この値は大文字小文字の区別があります)。エンドポイントの作成時に異なるリージョンを指定するには、***--region*** 引数を使用して指定してください。

詳しい情報は 「サービスのリージョン」 を参照してください。

12.6. TELEMETRY サービスの認証の設定

Telemetry API サービス (**openstack-ceilometer-api**) が認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、Telemetry API サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順12.4 Telemetry サービスが Identity サービスを使用して認証を行うための設定

1. Telemetry API サービスが使用する必要のある Identity サービスのホストを設定します。

   ```
   # openstack-config --set /etc/ceilometer/ceilometer.conf \
      keystone_authtoken auth_host IP
   ```

   IP は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。

2. Telemetry API サービスが使用する必要のある Identity サービスのホストを設定します。

   ```
   # openstack-config --set /etc/ceilometer/ceilometer.conf \
      keystone_authtoken auth_port PORT
   ```

   PORT は、Identity サービスが使用する認証ポート (通常は **35357**) に置き換えます。

3. Telemetry API サービスで認証に **http** プロトコルを使用するように設定します。

   ```
   # openstack-config --set /etc/ceilometer/ceilometer.conf \
      keystone_authtoken auth_protocol http
   ```

4. Telemetry API サービス が正しいテナントとして認証を行うように設定します。

   ```
   # openstack-config --set /etc/ceilometer/ceilometer.conf \
      keystone_authtoken admin_tenant_name services
   ```

   services は、Telemetry サービスを使用するために作成したテナントの名前に置き換えます。本ガイドの例では、**services** を使用しています。

5. Telemetry サービスが **ceilometer** 管理ユーザーアカウントを使用して認証を行うように設定します。

   ```
   # openstack-config --set /etc/ceilometer/ceilometer.conf \
      keystone_authtoken admin_user ceilometer
   ```

6. Telemetry サービスが正しい **ceilometer** 管理ユーザーアカウントパスワードを使用するように設定します。

   ```
   # openstack-config --set /etc/ceilometer/ceilometer.conf \
      keystone_authtoken admin_password PASSWORD
   ```

   PASSWORD は、**ceilometer** ユーザーの作成時に設定したパスワードに置き換えます。

7. Telemetry シークレットは、複数のホスト全体にわたって Telemetry サービスの全コンポーネント間の通信 (例: コレクターエージェントとコンピュートノードエージェントの間の通信など) のセキュリティー保護を支援するために使用する文字列です。この Telemetry シークレットを設定するには、以下のコマンドを実行します。

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   publisher_rpc metering_secret SECRET
```

SECRET は、全 Telemetry サービスのコンポーネントが AMQP で送受信されたメッセージの署名および検証に使用する必要のある文字列に置き換えます。

8. 各コンポーネントをデプロイするホストで、中央エージェント、コンピュートエージェント、アラームエバリュエーターが使用するサービスエンドポイントを設定します。

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT os_auth_url http://IP:35357/v2.0
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT os_username ceilometer
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT os_tenant_name services
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT os_password PASSWORD
```

以下の値を置き換えてください。

- *IP* は、Identity サービスをホストするシステムの IP アドレスまたはホスト名に置き換えます。
- *PASSWORD* は、**ceilometer** ユーザーの作成時に設定したパスワードに置き換えます。

12.7. TELEMETRY サービスのトラフィックを許可するためのファイアウォール設定

Telemetry サービスは、ポート **8777** で接続を受信します。このサービスノードのファイアウォールは、このポートのネットワークトラフィックを許可するように設定する必要があります。以下の手順に記載するステップはすべて、Telemetry サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順12.5 Telemetry サービスのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。
2. このファイルに、ポート **8777** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

```
-A INPUT -p tcp -m multiport --dports 8777 -j ACCEPT
```

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。
4. **iptables** サービスを再起動して、変更を有効にします。

```
# systemctl restart iptables.service
```

12.8. TELEMETRY サービスのための RABBITMQ メッセージブローカーの設定

RabbitMQ はデフォルト (かつ推奨の) メッセージブローカーです。RabbitMQ メッセージングサービスは、rabbitmq-server パッケージにより提供されます。以下の手順で記載する全ステップは、Telemetry サービスをホストするシステムに **root** ユーザーとしてログインして実行する必要があります。

手順12.6 Telemetry サービスが RabbitMQ メッセージブローカーを使用するための設定

1. RPC バックエンドとして RabbitMQ を設定します。

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT rpc_backend rabbit
```

2. Telemetry サービスが RabbitMQ ホストに接続するように設定します。

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT rabbit_host RABBITMQ_HOST
```

   RABBITMQ_HOST は、メッセージブローカーの IP アドレスまたはホスト名に置き換えます。

3. メッセージブローカーのポートを **5672** に設定します。

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT rabbit_port 5672
```

4. RabbitMQ の設定時に Telemetry サービス用に作成した RabbitMQ ユーザー名とパスワードを設定します。

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT rabbit_userid ceilometer
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT rabbit_password CEILOMETER_PASS
```

   ceilometer および *CEILOMETER_PASS* は、Telemetry サービス用に作成したRabbitMQ ユーザー名とパスワードに置き換えます。

5. RabbitMQ の起動時に、**ceilometer** ユーザーに全リソースに対するパーミッションが付与されます。このアクセスは、特別に仮想ホスト **/** を介して行われます。Telemetry サービスがこの仮想ホストに接続されるように設定します。

```
# openstack-config --set /etc/ceilometer/ceilometer.conf \
   DEFAULT rabbit_virtual_host /
```

12.9. コンピュートノードの設定

Telemetry サービスは、各ノードにインストールされたコンピュートエージェント (**openstack-ceilometer-compute**) から使用状況データを収集することにより、そのノードを監視します。ノードのコンピュートエージェントは、Telemetry コンポーネントをすでに設定済みの別のホストから **/etc/ceilometer/ceilometer.conf** ファイルを複製することで設定できます。

コンピュートノード自体が通知を有効化するように設定する必要があります。

手順12.7 コンピュートノード上での通知の有効化

1. ノードで openstack-ceilometer-compute、python-ceilometer、python-ceilometerclient をインストールします。

```
# yum install openstack-ceilometer-compute python-ceilometer python-
ceilometerclient
```

2. ノード上で監査を有効にします。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT instance_usage_audit True
```

3. 監査の頻度を設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT instance_usage_audit_period hour
```

4. 通知をトリガーする状態変更の種別を設定します。

```
# openstack-config --set /etc/nova/nova.conf \
   DEFAULT notify_on_state_change vm_and_task_state
```

5. ノードが正しい通知ドライバーを使用するように設定します。テキストエディターで **/etc/nova/nova.conf** ファイルを開き、**DEFAULT** セクションに以下の表を追加します。

```
notification_driver = messagingv2
notification_driver = ceilometer.compute.nova_notifier
```

   コンピュートノードには、2 種の通知ドライバーが必要です。これらのドライバーは、同じ設定キーを使用して定義されます。**openstack-config** を使用して、この値を設定することはできません。

6. コンピュートエージェントを開始します。

```
# systemctl start openstack-ceilometer-compute.service
```

7. エージェントがブート時に起動するように設定します。

```
# systemctl enable openstack-ceilometer-compute.service
```

8. **openstack-nova-compute** サービスを再起動して、**/etc/nova/nova.conf** に加えた変更をすべて適用します。

```
# systemctl restart openstack-nova-compute.service
```

12.10. 監視対象サービスの設定

Telemetry サービスは、Image サービス、OpenStack Networking、Object Storage サービス、および Block Storage サービスの各サービスを監視することも可能です。この機能を有効にするには、各サービスがコレクターサービスにサンプルを送信するように設定する必要があります。これらのサービスを設定する前には、このサービスをホストするノードに python-ceilometer および python-ceilometerclient のパッケージをあらかじめインストールする必要があります。

```
# yum install python-ceilometer python-ceilometerclient
```

注記

サービスを Telemetry サービスの監視対象に設定した後には、各サービスを再起動します。

Image サービス (glance)

```
# openstack-config --set /etc/glance/glance-api.conf \
   DEFAULT notifier_strategy NOTIFYMETHOD
```

NOTIFYMETHOD は通知キュー **rabbit** (**rabbitmq** キューを使用する場合) または **qpid** (**qpid** メッセージキューを使用する場合) に置き換えます。

Block Storage サービス (cinder)

```
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT notification_driver messagingv2
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT rpc_backend cinder.openstack.common.rpc.impl_kombu
# openstack-config --set /etc/cinder/cinder.conf \
   DEFAULT control_exchange cinder
```

Object Storage サービス (swift)

Telemetry サービスは、Telemetry に必要なアイデンティティーレコードの設定時に作成した **ResellerAdmin** ロールを使用して Object Storage サービス (**swift**) からサンプルを収集します。また、Object Storage サービスが **ceilometer** からのトラフィックを処理するように設定する必要があります。

1. テキストエディターで **/etc/swift/proxy-server.conf** ファイルを開いて以下の行を追加します。

```
[filter:ceilometer]
use = egg:ceilometer#swift

[pipeline:main]
pipeline = healthcheck cache authtoken keystoneauth ceilometer
proxy-server
```

2. **swift** ユーザーを **ceilometer** グループに追加します。

```
# usermod -a -G ceilometer swift
```

3. Object Storage サービスが **/var/log/ceilometer/swift-proxy-server.log** にログを出力できるようにします。

```
# touch /var/log/ceilometer/swift-proxy-server.log
# chown ceilometer:ceilometer /var/log/ceilometer/swift-proxy-
server.log
# chmod 664 /var/log/ceilometer/swift-proxy-server.log
```

OpenStack Networking (`neutron`)

Telemetry は IP アドレスの範囲を区別するためのラベルの使用をサポートしています。OpenStack Networking と Telemetry との 統合を有効化します。

```
# openstack-config --set /etc/neutron/neutron.conf \
   DEFAULT notification_driver messagingv2
```

12.11. TELEMETRY の API およびエージェントの起動

Telemetry サービスの各コンポーネントに対応するサービスを起動して、各サービスがブート時に起動するように設定します。

```
# systemctl start SERVICENAME.service
# systemctl enable SERVICENAME.service
```

SERVICENAME は、対応する各 Telemetry コンポーネントサービス名に置き換えます。

- openstack-ceilometer-compute
- openstack-ceilometer-central
- openstack-ceilometer-collector
- openstack-ceilometer-api
- openstack-ceilometer-notification

第13章 TELEMETRY ALARMING サービスのインストール

Telemetry Alarming service (**Aodh**) は、Telemetry サービスで収集されたデータに基づいて定義済みのアクションをトリガーします。

13.1. TELEMETRY ALARMING サービスのパッケージのインストール

以下のパッケージにより、Telemetry Alarming サービスのコンポーネントが提供されます。

openstack-aodh-api
: OpenStack Telemetry Alarming サービスのメインの API を提供します。

openstack-aodh-common
: OpenStack Telemetry Alarming サービスのコンポーネントすべてに共通するファイルを提供します。

openstack-aodh-evaluator
: アラームをいつトリガーするかを判断するアラームエバリュエーターを提供します。

openstack-aodh-expirer
: 失効したアラームの履歴データを消去する、エクスパイアラーコンポーネントを提供します。

openstack-aodh-listener
: 定義済みのアクションを実行するリスナーデーモンを提供します。

openstack-aodh-notifier
: アラームの設定が可能なノーティファイアーデーモンを提供します。

python-aodh
: OpenStack Telemetry Alarming サービスの Python ライブラリーを提供します。

python-aodhclient
: OpenStack Telemetry Alarming サービスのコマンドラインツールおよび Python API (具体的には aodhclient モジュール) を提供します。

必要なパッケージをすべてインストールします。

```
# yum install openstack-aodh-api openstack-aodh-evaluator openstack-aodh-expirer openstack-aodh-listener openstack-aodh-notifier python-aodhclient
```

これで、openstack-aodh-common および python-aodh パッケージは依存関係にあるので、これらのパッケージが自動的にインストールされます。

13.2. TELEMETRY ALARMING サービスデータベースの作成

TTelemetry Alarming サービスが使用するデータベースとデータベースユーザーを作成します。以下の手順のステップはすべて、データベースサーバーに `root` ユーザーとしてログインして実行する必要があります。

手順13.1 Telemetry Alarming サービスデータベースの作成

1. データベースサービスに接続します。

```
# mysql -u root -p
```

2. **aodh** データベースを作成します。

```
mysql> CREATE DATABASE aodh;
```

3. **aodh** データベースユーザーを作成し、**aodh** データベースへのアクセスを許可します。

```
mysql> GRANT ALL ON aodh.* TO 'aodh'@'%' IDENTIFIED BY
'AODH_PASSWORD';
mysql> GRANT ALL ON aodh.* TO 'aodh'@'localhost' IDENTIFIED BY
'AODH_PASSWORD';
```

AODH_PASSWORD は、このユーザーとしてデータベースサーバーとの認証を行う際に使用するセキュアなパスワードに置き換えます。

4. **mysql** クライアントを終了します。

```
mysql> quit
```

13.3. TELEMETRY ALARMING サービスの設定

Telemetry Alarming サービスを設定してから個別のデーモンを起動する必要があります。

13.3.1. Telemetry Alarming サービスのデータベース接続の設定

Telemetry Alarming サービスが使用するデータベース接続の URL は、**/etc/aodh/aodh.conf** ファイルで定義されます。以下のコマンドを実行して、URL を設置してください。

```
# openstack-config --set /etc/aodh/aodh.conf \
database connection mysql+pymysql://aodh:AODH_PASSWORD@IP/aodh
```

AODH_PASSWORD は Telemetry Alarming サービスのパスワードに、*IP* はデータベースサービスをホストするサーバーの IP アドレスまたはホスト名に置き換えます。

13.3.2. Telemetry Alarming サービスのアイデンティティーレコードの作成

Telemetry Alarming サービスで必要な Identity サービスを作成して設定します。これらのエントリーは、Telemetry Alarming サービスによって提供されるボリューム機能を検索してアクセスを試みる他の OpenStack サービスを補助します。

手順13.2 Telemetry Alarming サービス用のアイデンティティーレコードの作成

1. Identity サービス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 ~/keystonerc_admin
```

2. **aodh** ユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password AODH_PASSWORD
aodh
+----------+----------------------------------+
| Field    | Value                            |
+----------+----------------------------------+
| email    | None                             |
| enabled  | True                             |
| id       | f55915b5ca2d451d8b4109251976a4bc |
| name     | aodh                             |
| username | aodh                             |
+----------+----------------------------------+
```

AODH_PASSWORD は Telemetry Alarming サービスのパスワードに置き換えます。

3. **admin** ロールのメンバーとして、**aodh** ユーザーを **services** プロジェクトに追加します。

```
[(keystone_admin)]# openstack role add --project services --user
aodh admin
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 63aa6177a61b44aca25dd88a917353bc |
| name      | admin                            |
+-----------+----------------------------------+
```

4. **aodh** のサービスエントリーを作成します。

```
[(keystone_admin)]# openstack service create --name aodh \
--description "Telemetry Alarming Service" \
alarming
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Telemetry Alarming Service       |
| enabled     | True                             |
| id          | 67bb52266ae84c1f88877bbb4bf5d587 |
| name        | aodh                             |
| type        | alarming                         |
+-------------+----------------------------------+
```

5. Telemetry Alarming サービスのエンドポイントを作成します。

```
[(keystone_admin)]# openstack endpoint create \
--publicurl "http://IP:8042" \
--adminurl "http://IP:8042" \
--internalurl "http://IP:8042" \
--region RegionOne \
alarming
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
```

```
| adminurl     | http://IP:8042                   |
| id           | ac2735777336400baa38e2d408d26392 |
| internalurl  | http://IP:8042                   |
| publicurl    | http://IP:8042                   |
| region       | RegionOne                        |
| service_id   | 67bb52266ae84c1f88877bbb4bf5d587 |
| service_name | aodh                             |
| service_type | alarming                         |
+--------------+----------------------------------+
```

IP は、Telemetry Alarming サーバーの IP アドレスまたはホスト名に置き換えます。

13.3.3. Telemetry Alarming サービスのトラフィックを許可するためのファイアウォール設定

Telemetry Alarming サービスは、ポート **8042** で接続を受信します。このサービスノードのファイアウォールは、このポートのネットワークトラフィックを許可するように設定する必要があります。以下の手順に記載するステップはすべて、Telemetry Alarming サービスをホストするサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順13.3 Telemetry Alarming サービスのトラフィックを許可するためのファイアウォール設定

1. テキストエディターで **/etc/sysconfig/iptables** ファイルを開きます。
2. このファイルに、ポート **8042** で TCP トラフィックを許可する INPUT ルールを追加します。新規ルールは、トラフィックを REJECT する INPUT ルールよりも前に記載する必要があります。

```
-A INPUT -p tcp -m multiport --dports 8042 -j ACCEPT
```

3. **/etc/sysconfig/iptables** ファイルへの変更を保存します。
4. **iptables** サービスを再起動して、変更を有効にします。

```
# systemctl restart iptables.service
```

13.3.4. Telemetry Alarming サービスの認証の設定

Telemetry Alarming サービスが認証に Identity サービスを使用するように設定します。以下の手順に記載するステップはすべて、Telemetry Alarming サービスをホストする各サーバーに **root** ユーザーとしてログインして実行する必要があります。

手順13.4 Telemetry Alarming サービス が Identity サービスを使用して認証を行うための設定

1. サービス認証を設定します。

```
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials auth_type password
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials auth_url = http://CONTROLLER:5000/v3
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials interface internalURL
# openstack-config --set /etc/aodh/aodh.conf \
```

```
service_credentials password AODH_PASSWORD
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials project_domain_name default
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials project_name service
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials region_name RegionOne
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials user_domain_name default
# openstack-config --set /etc/aodh/aodh.conf \
service_credentials username aodh
```

CONTROLLER は Identity サーバーのホスト名または IP アドレスに、*AODH_PASSWORD* は Telemetry Alarming サービスのパスワードに置き換えます。

2. Identity サービスへのアクセスを設定します。

```
# openstack-config --set /etc/aodh/aodh.conf \
DEFAULT auth_strategy keystone
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken auth_type password
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken auth_uri http://CONTROLLER:5000
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken auth_url http://CONTROLLER:35357
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken memcached_servers CONTROLLER:11211
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken password AODH_PASSWORD
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken project_domain_name default
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken project_name service
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken user_domain_name default
# openstack-config --set /etc/aodh/aodh.conf \
keystone_authtoken username aodh
```

こちらでも、*CONTROLLER* は Identity サーバーのホスト名または IP アドレスに、*AODH_PASSWORD* は Telemetry Alarming サービスのパスワードに置き換えます。

13.3.5. Telemetry Alarming サービスのための RabbitMQ メッセージブローカーの設定

RabbitMQ メッセージキューへのアクセスを設定します。

```
# openstack-config --set /etc/aodh/aodh.conf \
DEFAULT rpc_backend rabbit
# openstack-config --set /etc/aodh/aodh.conf \
oslo_messaging_rabbit rabbit_host CONTROLLER
# openstack-config --set /etc/aodh/aodh.conf \
oslo_messaging_rabbit rabbit_userid USER
# openstack-config --set /etc/aodh/aodh.conf \
oslo_messaging_rabbit rabbit_password RABBITMQ_PASSWORD
```

CONTROLLER は、RabbitMQ ブローカーのホスト名または IP アドレスに、*USER* は RabbitMQ ユーザー ID に、*RABBITMQ_PASSWORD* は RabbitMQ パスワードに置き換えます。

13.4. TELEMETRY ALARMING サービスの起動

Telemetry Alarming サービスデーモンを起動して、このデーモンがブート時に起動するように設定します。

```
# systemctl start openstack-aodh-api.service \
openstack-aodh-evaluator.service \
openstack-aodh-notifier.service \
openstack-aodh-listener.service
# systemctl enable openstack-aodh-api.service \
openstack-aodh-evaluator.service \
openstack-aodh-notifier.service \
openstack-aodh-listener.service
```

第14章 TIME-SERIES-DATABASE-AS-A-SERVICE のインストール

Time-Series-Database-as-a-Service (**gnocchi**) はマルチテナントのメトリックおよびリソースのデータベースです。大規模なメトリックを格納する一方でオペレーターやユーザーにメトリックおよびリソースの情報へのアクセスを提供します。

Time-Series-Database-as-a-Service についての詳しい情報は、『Logging, Monitoring, and Troubleshooting Guide』の「Time-Series-Database-as-a-Service の使用」の章を参照してください。

Time-Series-as-a-Service は、以下のドライバーを中心に構築されています。

storage

: **storage** ドライバーは、作成したメトリックのメジャーを保管する役割を果たします。タイムスタンプと値を受け取り、定義済みのアーカイブポリシーに従ってアグリゲーションを算出します。

indexer

: **indexer** ドライバーは、全リソースのインデックスとそれらのタイプおよびプロパティーを保管します。Time-Series-as-a-Service は OpenStack プロジェクトからのリソースの種別のみを認識しますが、一般的なタイプも提供するので、基本的なリソースを作成して、リソースのプロパティーを自分で処理することができます。**indexer** はリソースをメトリックとリンクする役割も果たします。

ユーザーに公開される REST API は、正しいインフラストラクチャーのメジャーを提供するのに必要な全機能を提供するドライバーを操作します。

14.1. TIME-SERIES-DATABASE-AS-A-SERVICE パッケージのインストール

Time-Series-Database-as-a-Service のコンポーネントは、以下のパッケージにより提供されます。

openstack-gnocchi-api

: OpenStack Time-Series-Database-as-a-Service のメインの API を提供します。

openstack-gnocchi-carbonara

: OpenStack Time-Series-Database-as-a-Service の carbonara を提供します。

openstack-gnocchi-doc

: OpenStack Time-Series-Database-as-a-Service のドキュメントを提供します。

openstack-gnocchi-indexer-sqlalchemy

: OpenStack Time-Series-Database-as-a-Service の indexer SQLAlchemy を提供します。

openstack-gnocchi-statsd

: OpenStack Time-Series-Database-as-a-Service の統計デーモンを提供します。

python-gnocchi

: OpenStack Time-Series-Database-as-a-Service の Python ライブラリーを提供します。

コントローラーノードに全パッケージをインストールします。

```
# yum install openstack-gnocchi\* -y
```

14.2. TIME-SERIES-DATABASE-AS-A-SERVICE の初期化

indexer を初期化します。

```
# gnocchi-upgrade
```

14.3. TIME-SERIES-DATABASE-AS-A-SERVICE の設定

手動で Time-Series-Database-as-a-Service パッケージをインストールする場合には、サービスの設定ファイル (**/etc/gnocchi/gnocchi.conf**) には何も設定が指定されていません。必要に応じて各設定を手動で追加/設定する必要があります。

- **[DEFAULT]**セクションで、ロギングと詳細な出力を有効にします。

```
[DEFAULT]
debug = true
verbose = true
```

- **[API]** のセクションに、ワーカーの数を記載します。

```
[api]
workers = 1
```

- **[database]** セクションで、バックエンドを **sqlalchemy** に設定します。

```
[database]
backend = sqlalchemy
```

- **[indexer]** セクションに、ユーザー名、パスワード、IP アドレスを渡して、SQL データベースを設定します。

```
[indexer]
url = mysql://USER_NAME:PASSWORD@192.0.2.10/gnocchi2?charset=utf8
```

注記

データベースは、**gnocchi-api** を起動する前に作成しておく必要があります。

- **[keystone_authtoken]** セクションで、認証パラメーターを更新します。以下に例を示します。

```
[keystone_authtoken]
auth_uri = http://192.0.2.7:5000/v2.0
signing_dir = /var/cache/gnocchi
auth_host = 192.0.2.7
auth_port = 35357
```

```
auth_protocol = http
identity_uri = http://192.0.2.7:35357/
admin_user = admin
admin_password = 5179f4d3c5b1a4c51269cad2a23dbf336513efeb
admin_tenant_name = admin
```

- **[statsd]** セクションに以下の値を追加します。

  ```
  [statsd]
  resource_id = RESOURCE_ID
  user_id = USER_ID
  project_id = PROJECT_ID
  archive_policy_name = low
  flush_delay = 5
  ```

  RESOURCE_ID、**USER_ID**、および **PROJECT_ID** の値は、お使いのデプロイメントの値に置き換えてください。

- **[storage]** セクションで、**coordination_url** および **file_basepath**を手動で追加してから、**driver** の値を file に設定します。

  ```
  [storage]
  coordination_url = file:///var/lib/gnocchi/locks
  driver = file
  file_basepath = /var/lib/gnocchi
  ```

- **gnocchi** サービスを再起動して、変更を有効にします。

  ```
  # systemctl restart openstack-gnocchi-api.service
  # systemctl restart openstack-gnocchi-metricd.service
  # systemctl restart openstack-gnocchi-statsd.service
  ```

14.4. TIME-SERIES-DATABASE-AS-A-SERVICE データベースの作成

Time-Series-Database-as-a-Service サービスが使用するデータベースとデータベースユーザーを作成します。以下の手順のステップはすべて、データベースサーバーに **root** ユーザーとしてログインして実行する必要があります。

- データベースサービスに接続します。

  ```
  # mysql -u root -p
  ```

- Time-Series-Database-as-a-Service データベースを作成します。

  ```
  mysql> CREATE DATABASE gnocchi;
  ```

- Time-Series-Database-as-a-Service データベースのユーザーを作成し、Time-Series-Database-as-a-Service データベースへのアクセス権を付与します。

  ```
  mysql> GRANT ALL ON gnocchi.* TO 'gnocchi'@'%' IDENTIFIED BY
  'PASSWORD';
  mysql> GRANT ALL ON gnocchi.* TO 'gnocchi'@'localhost' IDENTIFIED BY
  ```

```
'PASSWORD';
```

PASSWORD は、このユーザーとしてデータベースと認証を行う際に使用するセキュアなパスワードに置き換えます。

- データベースの特権をフラッシュして、設定が即時に反映されるようにします。

```
mysql> FLUSH PRIVILEGES;
```

- **mysql** クライアントを終了します。

```
mysql> quit
```

14.5. TELEMETRY サービスのバックエンドとしての TIME-SERIES-DATABASE-AS-A-SERVICE の設定

Telemetry サービスは、デフォルトではデータベース内でのみ計測データを保存します。Telemetry が計測データをデータベース以外の他のシステムにも送信できるようにするには、複数のディスパッチャーを開発し、Telemetry の設定ファイルを変更して有効化することができます。**gnocchi** のディスパッチャーは計測データを TDSaaS バックエンドに送信します。

gnocchi ディスパッチャーの場合には、以下の設定を **/etc/ceilometer/ceilometer.conf** ファイルに追加します。

```
[DEFAULT]
dispatcher = gnocchi

[dispatcher_gnocchi]
filter_project = gnocchi_swift
filter_service_activity = True
archive_policy = low
url = http://localhost:8041
```

上記の設定の **url** は TDSaaS エンドポイントの URL で、デプロイメントによって異なります。

注記

gnocchi ディスパッチャーが有効化されている場合には、Ceilometer API コールは、空の結果で **410** が返されます。データにアクセスするには、代わりに TDSaaS API を使用する必要があります。

gnocchi サービスを再起動して、変更を有効にします。

```
# systemctl restart openstack-ceilometer-api.service
```

第15章 SHARED FILE SYSTEM サービスのインストール

OpenStack の Shared File System サービスは、複数のインスタンスにより消費可能な共有ファイルシステムを簡単にプロビジョニングにする方法を提供します。これらの共有ファイルシステムは、既存のバックエンドボリュームからプロビジョニングします。

15.1. SHARED FILE SYSTEM サービスのバックエンドの要件

OpenStack Shared File System サービスにより、共有ファイルシステムをオンデマンドで作成することができます。共有のためのバックエンドリソースがすでに存在している必要があります。本章では、このサービスを NetApp の統合された **manila** ドライバー (**manila.share.drivers.netapp.common.NetAppDriver**) とともにデプロイする方法について説明します。

15.2. SHARED FILE SYSTEM サービスのパッケージのインストール

以下のパッケージは、Shared File System サービスのコンポーネントを提供します。

openstack-manila

OpenStack Shared File System の主要なサービスを提供します。

openstack-manila-share

プロビジョニングした共有のエクスポートに必要なサービスを提供します。

python-manilaclient

Shared File System サービスのクライアントライブラリーおよび CLI を提供します。

コントローラーノードにパッケージをインストールします。

```
# yum install -y openstack-manila openstack-manila-share python-
manilaclient
```

15.3. SHARED FILE SYSTEM サービス用のアイデンティティーレコードの作成

必要なパッケージをインストールした後に、Shared File System サービスに必要な Identity のレコードを作成します。Identity サービスのホストまたは **keystonerc_admin** ファイルをコピーした先のマシン上で、以下の手順を実行してください。

注記

keystonerc_admin ファイルに関する詳しい情報は 「管理者アカウントおよび Identity サービスエンドポイントの作成」 を参照してください。

手順15.1 Shared File System サービス用のアイデンティティーレコードの作成

1. Identity サービス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 ~/keystonerc_admin
```

2. **manila** サービスユーザーを作成します。

```
[(keystone_admin)]# openstack user create --password MANILAPASS --
email manila@localhost manila
```

MANILAPASS は、Shared File System サービスが Identity サービスとの認証を行う際に使用するセキュアなパスワードに置き換えます。

3. **admin** ロールを **manila** ユーザーに追加します。

```
[(keystone_admin)]# openstack role add --project services --user
manila admin
```

4. **manila** サービスエンティティーを作成します。

```
[(keystone_admin)]# openstack service create --name manila --
description "OpenStack Shared Filesystems" share
```

5. **manila** エンドポイントエントリーを作成します。

```
[(keystone_admin)]# openstack endpoint create \
--publicurl 'http://MANILAIP:8786/v1/%(tenant_id)s' \
--internalurl 'http://MANILAIP:8786/v1/%(tenant_id)s' \
--adminurl 'http://MANILAIP:8786/v1/%(tenant_id)s' \
--region RegionOne \
manila
```

MANILAIP は、コントローラーノードの IP に置き換えます。

15.4. 基本的な SHARED FILE SYSTEM サービスの設定

Shared File System サービスのパッケージを手動でインストールする場合には、サービスの設定ファイル (**/etc/manila/manila.conf**) には何も設定が指定されていません。必要に応じて各設定をアンコメントしたり、追加/設定したりする必要があります。

以下のコードスニペットは、Shared File System サービスのデプロイに必要な基本設定です。この内容を **/etc/manila/manila.conf** にコピーしてください。コピーする際には、必要な変数を置き換えてください。

```
[DEFAULT]

osapi_share_listen=0.0.0.0

sql_connection=mysql://manila:MANILADBPASS@CONTROLLERIP/manila # (1)
(2)

api_paste_config=/etc/manila/api-paste.ini
state_path=/var/lib/manila
sql_idle_timeout=3600
storage_availability_zone=nova
rootwrap_config=/etc/manila/rootwrap.conf
auth_strategy=keystone
```

```
nova_catalog_info=compute:nova:publicURL
nova_catalog_admin_info=compute:nova:adminURL
nova_api_insecure=False
nova_admin_username=nova

nova_admin_password=NOVAADMINPASS # 3
nova_admin_tenant_name=services
nova_admin_auth_url=http://localhost:5000/v2.0
network_api_class=manila.network.neutron.neutron_network_plugin.NeutronNet
workPlugin
debug=False
verbose=True
log_dir=/var/log/manila
use_syslog=False
rpc_backend=manila.openstack.common.rpc.impl_kombu
control_exchange=openstack
amqp_durable_queues=False

[oslo_messaging_rabbit]
rabbit_ha_queues=False
rabbit_userid=guest
rabbit_password=guest
rabbit_port=5672
rabbit_use_ssl=False
rabbit_virtual_host=/

rabbit_host=CONTROLLERIP # 4

rabbit_hosts=CONTROLLERIP:5672 # 5

[oslo_concurrency]
lock_path=/tmp/manila/manila_locks
```

以下の値を置き換えてください。

1	*MANILADBPASS* は、「Shared File System サービスのデータベースの作成」 で使用した Shared File System サービスのデータベースパスワードに置き換えます。
2 4 5	*CONTROLLERIP* は、コントローラーノードの IP アドレスに置き換えます。
3	*NOVAADMINPASS* は、Compute サービスの管理者パスワードに置き換えます。これは、**/etc/neutron/neutron.conf** の **nova_admin_password** と同じです。 **注記** director を使用して OpenStack をデプロイした場合には、このパスワードは、アンダークラウドの **/home/stack/tripleo-overcloud-passwords** ファイルでも確認できます。

本リリースの時点では、Shared File System サービスの設定の一部はまだ **/etc/manila/api-**

paste.ini で定義されている場合もあります。以下のコードスニペットを使用して、このファイルを更新してください。

```
[filter:keystonecontext]
paste.filter_factory =
manila.api.middleware.auth:ManilaKeystoneContext.factory

[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
service_protocol = http
service_host = localhost
service_port = 5000
auth_host = localhost
auth_port = 35357
auth_protocol = http
admin_tenant_name = services
admin_user = manila

admin_password = MANILAPASS # (1)
signing_dir = /var/lib/manila

auth_uri=http://CONTROLLERIP:5000/v2.0 # (2)

identity_uri=http://CONTROLLERIP:35357 # (3)
```

(1)	*MANILAPASS* は (「Shared File System サービス用のアイデンティティーレコードの作成」 で使用した) **manila** サーバーのユーザーパスワードに置き換えます。
(2) (3)	*CONTROLLERIP* は、コントローラーノードの IP アドレスに置き換えます。

15.5. SHARED FILE SYSTEM サービスのデータベースの作成

Shared File System サービスが使用するデータベースとデータベースユーザーを作成します。以下の手順はすべて、データベースサーバーに **root** ユーザーとしてログインして実行する必要があります。

手順15.2 Shared File System サービスのデータベースの作成

1. データベースサービスに接続します。

   ```
   # mysql -u root
   ```

2. **manila** データベースを作成します。

   ```
   mysql> CREATE DATABASE manila;
   ```

3. **manila** データベースユーザーを作成し、**manila** データベースへのアクセスを許可します。

   ```
   mysql> GRANT ALL ON manila.* TO 'manila'@'%' IDENTIFIED BY
   'MANILADBPASS';
   ```

```
mysql> GRANT ALL ON manila.* TO 'manila'@'localhost' IDENTIFIED BY
'MANILADBPASS';
```

MANILADBPASS は、データベースサーバーとの認証を行う際に使用する **manila** サービスのセキュアなパスワードに置き換えます。これと同じパスワードは、「基本的な Shared File System サービスの設定」 で後ほど使用そします。

4. データベースの特権をフラッシュして、設定が即時に反映されるようにします。

```
mysql> FLUSH PRIVILEGES;
```

5. **mysql** クライアントを終了します。

```
mysql> quit
```

6. Shared File System サービスのテーブルを作成して、必要な移行をすべて適用します。

```
# manila-manage db sync
```

15.6. SHARED FILE SYSTEM サービスのバックエンドの定義

Shared File System サービスにはバックエンドが必要です。このバックエンドは **/etc/manila/manila.conf** 内の独自のセクションで定義されます。NetApp は Shared File System サービスのバックエンドを定義する方法についての詳しい情報を提供しています。NetApp の『OpenStack Deployment and Operations Guide』の「OpenStack Shared File System Service (Manila)」の章を参照してください。

15.7. SHARED FILE SYSTEM サービスの起動

この時点では、Shared File System サービスは完全に設定されているはずです。これらの必須設定を適用するには、全 Shared File System サービスを再起動します。

```
# systemctl start openstack-manila-api
# systemctl start openstack-manila-share
# systemctl start openstack-manila-scheduler
```

その後には、これらのサービスを有効化します。

```
# systemctl enable openstack-manila-api
# systemctl enable openstack-manila-share
# systemctl enable openstack-manila-scheduler
```

各サービスが正常に起動して有効になっていることを確認するには、以下を実行します。

```
# systemctl status openstack-manila-api
openstack-manila-api.service - OpenStack Manila API Server
Loaded: loaded (/usr/lib/systemd/system/openstack-manila-api.service;
enabled)
Active: active (running) since Mon 2015-07-27 17:02:49 AEST; 1 day 18h ago
[...]

# systemctl status openstack-manila-share
```

```
openstack-manila-share.service - OpenStack Manila Share Service
Loaded: loaded (/usr/lib/systemd/system/openstack-manila-share.service;
enabled)
Active: active (running) since Mon 2015-07-27 17:02:49 AEST; 1 day 18h ago
[...]

# systemctl status openstack-manila-scheduler
openstack-manila-scheduler.service - OpenStack Manila Scheduler
Loaded: loaded (/usr/lib/systemd/system/openstack-manila-
scheduler.service; enabled)
Active: active (running) since Mon 2015-07-27 17:02:49 AEST; 1 day 18h ago
[...]
```

15.8. 定義済みのバックエンドに対する共有種別の作成

Shared File System サービスでは、固有の設定で共有を作成する際に利用可能な **共有種別** を定義できます。共有種別は、Block Storage のボリューム種別と全く同じように機能します。各種別には設定が関連付けられており (**追加スペック**)、共有の作成中に種別を呼び出して、これらの設定を適用します。

デフォルト以外のバックエンドで共有を作成する場合は、使用するバックエンドを明示的に指定する必要があります。プロセスがユーザーにとってシームレスになるように、共有種別を作成して、(「Shared File System サービスのバックエンドの定義」 で選択した) バックエンドの **share_backend_name** の値と関連付けます。

TYPENAME という名前の共有種別を作成するには、OpenStack の admin ユーザーとして以下を実行します。

```
# manila type-create TYPENAME SHAREHANDLING
```

SHAREHANDLING は、共有種別がドライバーを使用して共有のライフサイクルの処理するかどうかを指定します。これは、バックエンド定義の **driver_handles_share_servers** で設定した値になるはずです。**driver_handles_share_servers=False** では、*SHAREHANDLING* も **false** に設定する必要があります。**share1** という共有種別を作成するには、以下のコマンドを実行します。

```
# manila type-create share1 false
```

次に **share1** 種別を特定のバックエンドに関連付けます。**share_backend_name** の値を使用して、バックエンドを指定します。たとえば、共有種別 **share1** をバックエンド **GENERIC** に関連付けるには、以下のコマンドを実行します。

```
# manila type-key share1 set share_backend_name='GENERIC'
```

SHARE1 種別を呼び出すと、**GENERIC** バックエンドから共有を作成できるようになっているはずです。

第16章 DATABASE-AS-A-SERVICE のインストール (テクノロジープレビュー)

OpenStack Database-as-a-Service (trove) により、ユーザーは単一テナントのデーターベースを容易にプロビジョニングして、データベースのデプロイ、使用、管理、モニタリング、スケーリングに伴う従来の管理オーバーヘッドの多くを回避することができます。

> **警告**
>
> **非推奨機能のお知らせ**: Red Hat OpenStack Platform 11 以降では、OpenStack Trove サービスは Red Hat OpenStack Platform ディストリビューションには同梱されません。現在、信頼できるパートナーと連携して実稼動環境ですぐに使用できる DBaaS サービスをお客様に提供できるように取り組んでいます。このオプションに関する情報は、担当のセールスアカウントマネージャーにお問い合わせください。

> **警告**
>
> 本リリースでは、OpenStack Database-as-a-Service は **テクノロジープレビュー** として提供されているため、Red Hat では全面的にはサポートしていません。これは、テスト目的のみでご利用いただく機能で、実稼働環境にデプロイすべきではありません。テクノロジープレビューについての詳しい情報は 「対象範囲の詳細」を参照してください。

16.1. DATABASE-AS-A-SERVICE の要件

Database-as-a-Service を使用するにあたっては、以下のステップを実行しておく必要があります。

1. admin ユーザーのパスワードを更新します。

```
# keystone user-password-update --pass ADMIN_PASSWORD admin
```

2. **/root/keystonerc_admin** を新しいパスワードで更新します。

```
export OS_USERNAME=admin
export OS_TENANT_NAME=admin
export OS_PASSWORD=ADMIN_PASSWORD
export OS_AUTH_URL=http://keystone IP:5000/v2.0/
export PS1='[\u@\h \W(keystone_admin)]\$ '
```

3. 環境変数を読み込み、**admin** ユーザーに **services** テナント内の **admin** ロールが割り当てられていることを確認します。

```
# source keystonerc_admin
~(keystone_admin)]# keystone user-role-add --user admin --tenant
services --role admin
~(keystone_admin)]# keystone user-role-list --user admin --tenant
services
+----------------------------------+-------+--------------------
-------------+----------------------------------+
|                id                |  name |             user_id
|            tenant_id             |
+----------------------------------+-------+--------------------
-------------+----------------------------------+
| 4501ce8328324ef5bf1ed93ceb5494e6 | admin |
4db867e819ad40e4bf79681bae269084 | 70cd02c84f86471b8dd934db46fb484f
|
+----------------------------------+-------+--------------------
-------------+----------------------------------+
```

16.2. DATABASE-AS-A-SERVICE パッケージのインストール

以下のパッケージは、Database-as-a-Service のコンポーネントを提供します。

openstack-trove-api
: OpenStack Database-as-a-Service のメインの API を提供します。

openstack-trove-conductor
: OpenStack Database-as-a-Service のコンダクターサービスを提供します。

openstack-trove-guestagent
: OpenStack Database-as-a-Service のゲストエージェントサービスを提供します。

openstack-trove-taskmanager
: OpenStack Database-as-a-Service のタスクマネージャーサービスを提供します。

openstack-trove-images
: OpenStack Database-as-a-Service のイメージ作成ツールを提供します。

python-trove
: OpenStack Database-as-a-Service の Python ライブラリーを提供します。

python-troveclient
: Database-as-a-Service API のクライアントを提供します。

コントローラーノードに Database-as-a-Service の全パッケージをインストールします。

```
# yum install openstack-trove\*
```

16.3. DATABASE-AS-A-SERVICE の設定

1. keystone ユーザーを作成して、Database-as-a-Service 用のロールを追加します。

```
[root@rhosp-trove ~(keystone_admin)]# openstack user create --
password trove --email trove@localhost --project services trove
+------------+----------------------------------+
| Field      | Value                            |
+------------+----------------------------------+
| email      | trove@localhost                  |
| enabled    | True                             |
| id         | 8740fd0cba314fe68cf0ca95144d2766 |
| name       | trove                            |
| project_id | 42e1efb4bd5e49a49cb2b346078d6325 |
| username   | trove                            |
+------------+----------------------------------+
[root@rhosp-trove ~(keystone_admin)]# openstack role add --project
services --user trove admin
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 63aa6177a61b44aca25dd88a917353bc |
| name      | admin                            |
+-----------+----------------------------------+
[root@rhosp-trove ~(keystone_admin)]# openstack user role list --
project services trove
+----------------------------------+----------+----------+-------
+
| ID                               | Name     | Project  | User  |
+----------------------------------+----------+----------+-------
+
| 63aa6177a61b44aca25dd88a917353bc | admin    | services | trove |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ | services | trove |
+----------------------------------+----------+----------+-------
+
```

2. オプションで、全設定ファイルの詳細なデバッグ情報を設定します。

```
[root@rhosp-trove ~(keystone_admin)]# for conf_file in {trove,trove-
conductor,trove-taskmanager,trove-guestagent}; do
> openstack-config --set /etc/trove/$conf_file.conf DEFAULT verbose
True;
> openstack-config --set /etc/trove/$conf_file.conf DEFAULT debug
True;
> done
```

3. **`api-paste.ini`** ファイルを作成します (存在していない場合)。

```
[root@rhosp-trove ~(keystone_admin)]# cp /usr/share/trove/trove-
dist-paste.ini /etc/trove/api-paste.ini
```

4. **`api-paste.ini`** の keystone 認証トークンを更新します。

```
[filter:authtoken]
paste.filter_factory =
keystoneclient.middleware.auth_token:filter_factory
auth_uri = http://127.0.0.1:35357/
```

```
identity_uri = http://127.0.0.1:35357/
admin_password = TROVE_PASSWORD
admin_user = trove
admin_tenant_name = services
```

```
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove.conf DEFAULT api_paste_config /etc/trove/api-paste.ini
```

5. **api-paste.ini** と同じ情報で、**trove.conf** を更新します。

```
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove.conf keystone_authtoken auth_uri http://127.0.0.1:35357/
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove.conf keystone_authtoken identity_uri http://127.0.0.1:35357/
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove.conf keystone_authtoken admin_password TROVE_PASSWORD
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove.conf keystone_authtoken admin_user trove
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove.conf keystone_authtoken admin_tenant_name = services
```

6. **trove-taskmanager.conf** の **nova_proxy** 情報を更新します。Database-as-a-Service は、admin ユーザーの認証情報で nova コマンドを発行するので、この設定は実際の admin ユーザーにする必要があります。

```
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove-taskmanager.conf DEFAULT nova_proxy_admin_user admin
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove-taskmanager.conf DEFAULT nova_proxy_admin_password
ADMIN_PASSWORD
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
trove-taskmanager.conf DEFAULT nova_proxy_admin_tenant_name services
```

7. RabbitMQ ホストの情報で設定ファイルを更新します。

```
[root@rhosp-trove trove(keystone_admin)]# cat
/etc/rabbitmq/rabbitmq.config
% This file managed by Puppet
% Template Path: rabbitmq/templates/rabbitmq.config
[
  {rabbit, [
    {default_user, <<"guest">>},
    {default_pass, <<"RABBITMQ_GUEST_PASSWORD">>}
  ]},
```

```
[root@rhosp-trove trove(keystone_admin)]# for conf_file in
trove.conf trove-taskmanager.conf trove-conductor.conf ; do
> openstack-config --set /etc/trove/$conf_file DEFAULT rabbit_host
127.0.0.1;
> openstack-config --set /etc/trove/$conf_file DEFAULT
rabbit_password RABBITMQ_GUEST_PASSWORD;
> done
```

8. 設定ファイルにサービスの URL を追加します。

```
[root@rhosp-trove trove(keystone_admin)]# for conf_file in
trove.conf trove-taskmanager.conf trove-conductor.conf ; do
> openstack-config --set /etc/trove/$conf_file DEFAULT
trove_auth_url http://127.0.0.1:5000/v2.0
> openstack-config --set /etc/trove/$conf_file DEFAULT
nova_compute_url http://127.0.0.1:8774/v2
> openstack-config --set /etc/trove/$conf_file DEFAULT cinder_url
http://127.0.0.1:8776/v1
> openstack-config --set /etc/trove/$conf_file DEFAULT swift_url
http://127.0.0.1:8080/v1/AUTH_
> openstack-config --set /etc/trove/$conf_file DEFAULT
sql_connection mysql://trove:trove@127.0.0.1/trove
> openstack-config --set /etc/trove/$conf_file DEFAULT
notifier_queue_hostname 127.0.0.1
> done
```

上記のコマンドは、MySQL の接続を追加しますが、これはまだ機能しません。これらのパーミッションは次のステップで付与します。

9. **cloud-init** の情報を使用してタスクマネージャーの設定を更新します。

```
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
/etc/trove/trove-taskmanager.conf DEFAULT cloud-init_loaction
/etc/trove/cloudinit
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
/etc/trove/trove-taskmanager.conf DEFAULT taskmanager_manager
trove.taskmanager.manager.Manager
[root@rhosp-trove trove(keystone_admin)]# mkdir /etc/trove/cloudinit
```

10. デフォルトのデータストア (database type) で **`trove.conf`** を更新し、インスタンスをアタッチする OpenStack Networking ネットワークの名前を設定します。この場合は、ネットワークは **`private`** という名前です。

```
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
/etc/trove/trove.conf DEFAULT default_datastore mysql
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
/etc/trove/trove.conf DEFAULT add_addresses True
[root@rhosp-trove trove(keystone_admin)]# openstack-config --set
/etc/trove/trove.conf DEFAULT network_label_regex ^private$
```

11. Database-as-a-Service のデータベースを作成し、**`trove`** ユーザーにパーミッションを付与します。

```
[root@rhosp-trove trove(keystone_admin)]# mysql -u root
MariaDB [(none)]> create database trove;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> grant all on trove.* to trove@'localhost'
identified by 'TROVE_PASSWORD';
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [(none)]> grant all on trove.* to trove@'%' identified by
'TROVE_PASSWORD';
Query OK, 0 rows affected (0.00 sec)
```

12. 新規データベースにデータを読み込み、初期データストアを作成します。

```
[root@rhosp-trove trove(keystone_admin)]# trove-manage db_sync
[root@rhosp-trove trove(keystone_admin)]# trove-manage
datastore_update mysql ''
```

13. イメージとともに使用する **cloud-init** ファイルを作成します。

注記

Database-as-a-Service によりインスタンスが作成される時には、そのインスタンスをビルドするためにデータベースに設定した **image_id** が使用されます。また、指定したデータストアに基づいて、ユーザーデータにアタッチする **.cloudinit** ファイルを **/etc/trove/cloudinit/** 内で検索します。たとえば、新規インスタンスのデータストアに **mysql** を選択した場合には、nova が **/etc/trove/cloudinit/** 内で **mysql.cloudinit** ファイルを検索して、ユーザーデータスクリプトとしてアタッチします。これは、ビルド時に MySQL を登録/インストールするのに使用されます。

以下の内容で **/etc/trove/cloudinit/mysql.cloudinit** ファイルを作成します。各 *PASSWORD* は適切なパスワードに、*RHN_USERNAME*、*RHN_PASSWORD*、*POOL_ID* はお使いの Red Hat 認証情報とサブスクリプションプール ID に、*host SSH public key* はパスワードなしの SSH ログイン用のキーに置き換えます。

```
#!/bin/bash

sed -i'.orig' -e's/without-password/yes/' /etc/ssh/sshd_config
echo "PASSWORD" | passwd --stdin cloud-user
echo "PASSWORD" | passwd --stdin root
systemctl restart sshd

subscription-manager register --username=RHN_USERNAME --
password=RHN_PASSWORD
subscription-manager attach --pool POOL_ID
subscription-manager repos --disable=*
subscription-manager repos --enable=rhel-7-server-optional-rpms
subscription-manager repos --enable=rhel-7-server-rpms
subscription-manager repos --enable=rhel-server-rhscl-7-rpms
yum install -y openstack-trove-guestagent mysql55

cat << EOF > /etc/trove/trove-guestagent.conf
rabbit_host = 172.1.0.12
rabbit_password = RABBITMQ_GUEST_PASSWORD
nova_proxy_admin_user = admin
nova_proxy_admin_pass = ADMIN_PASSWORD
nova_proxy_admin_tenant_name = services
trove_auth_url = http://172.1.0.12:35357/v2.0
control_exchange = trove
EOF
```

```
echo "host SSH public key" >> /root/.ssh/authorized_keys

echo "host SSH public key" >> /home/cloud-user/.ssh/authorized_keys

systemctl stop trove-guestagent
systemctl enable trove-guestagent
systemctl start trove-guestagent
```

注記

上記は bash スクリプトとして記述されており、**cloud-init** に対応しています。これは、**could-init** の YAML 形式のレイアウトを使用して記述することもできます。

14. glance を使用して、**--file** オプションで指定したクラウドイメージをアップロードします。

```
[root@rhosp-trove trove(keystone_admin)]# glance image-create --name
rhel7 \
> --file image.qcow2 \
> --disk_format qcow2 \
> --container_format bare \
> --is-public True \
> --owner trove

[root@rhosp-trove trove(keystone_admin)]# glance image-list
+--------------------------------------+--------+-------------+--
----------------+-----------+--------+
| ID                                   | Name   | Disk Format |
Container Format | Size      | Status |
+--------------------------------------+--------+-------------+--
----------------+-----------+--------+
| b88fa633-7219-4b80-87fa-300840575f91 | cirros | qcow2       | bare
| 13147648  | active |
| 9bd48cdf-52b4-4463-8ce7-ce81f44205ae | rhel7  | qcow2       | bare
| 435639808 | active |
+--------------------------------------+--------+-------------+--
----------------+-----------+--------+
```

15. Red Hat Enterprise Linux 7 イメージへの参照で Database-as-a-Service データベースを更新します。前のコマンドの出力に表示された ID を使用してください。

```
[root@rhosp-trove trove(keystone_admin)]# trove-manage --config-
file=/etc/trove/trove.conf datastore_version_update \
> mysql mysql-5.5 mysql 9bd48cdf-52b4-4463-8ce7-ce81f44205ae mysql55
1
```

注記

構文は「**trove-manage datastore_version_update *datastore version_name manager image_id packages active***」です。

16. keystone を使用して Database-as-a-Service サービスを作成し、OpenStack がこのサービスの存在を認識するようにします。

```
[root@rhosp-trove trove(keystone_admin)]# openstack service create -
-name trove \
> --description "OpenStack DBaaS" \
> database
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack DBaaS                  |
| enabled     | True                             |
| id          | b05b564d5ac049f49984a827d820c5a5 |
| name        | trove                            |
| type        | database                         |
+-------------+----------------------------------+
```

17. Database-as-a-Service API 用の URL エンドポイントを追加します。

```
[root@rhosp-trove trove(keystone_admin)]# openstack endpoint create
\
> --publicurl 'http://127.0.0.1:8779/v1.0/%(tenant_id)s' \
> --internalurl 'http://127.0.0.1:8779/v1.0/%(tenant_id)s' \
> --adminurl 'http://127.0.0.1:8779/v1.0/%(tenant_id)s' \
> --region RegionOne \
> database
```

18. 3 つの Database-as-a-Service サービスを起動して、ブート時に有効になるように設定します。

```
[root@rhosp-trove trove(keystone_admin)]# systemctl start openstack-
trove-{api,taskmanager,conductor}
[root@rhosp-trove trove(keystone_admin)]# systemctl enable
openstack-trove-{api,taskmanager,conductor}
ln -s '/usr/lib/systemd/system/openstack-trove-api.service'
'/etc/systemd/system/multi-user.target.wants/openstack-trove-
api.service'
ln -s '/usr/lib/systemd/system/openstack-trove-taskmanager.service'
'/etc/systemd/system/multi-user.target.wants/openstack-trove-
taskmanager.service'
ln -s '/usr/lib/systemd/system/openstack-trove-conductor.service'
'/etc/systemd/system/multi-user.target.wants/openstack-trove-
conductor.service'
```

重要

systemctl status openstack-trove-{api,taskmanager,conductor} のコマンドを実行して、サービスが起動しているかどうかを確認します。操作が失敗して、**/var/log/trove** にエラーが出力されが場合には、以下のコマンドを実行して問題を解決します。

```
[root@rhosp-trove trove(keystone_admin)]# chown -R
trove:trove /var/log/trove
[root@rhosp-trove trove(keystone_admin)]# systemctl
restart openstack-trove-{api,taskmanager,conductor}
```

第17章 OPENSTACK INTEGRATION TEST SUITE のインストール

OpenStack Integration Test Suite (tempest) は、ライブの OpenStack クラスターに対して実行される統合テストセットです。Integration Test Suite には、一連の OpenStack API の検証テスト、シナリオ、Red Hat OpenStack Platform のデプロイメントに役立つその他の固有テストが含まれます。

OpenStack Integration Test Suite のインストールおよび設定に関する詳しい情報は、『OpenStack Integration Test Suite Guide』を参照してください。

第18章 OPENSTACK BENCHMARKING サービスのインストール

OpenStack Benchmarking サービスを使用すると、「OpenStack が大規模な環境でどのように機能するか」が分かります。OpenStack デプロイメント、クラウドの検証、ベンチマーキング、プロファイリングなどのプロセスを自動化することで実現できます。Benchmarking サービスは、OpenStack クラウドをテスト/検証するための各種アクションを提供できますが、本章では、Benchmarking サービスのインストールと設定を行い、OpenStack デプロイメントと連携させることに焦点を当てます。

注記

本リリースでは、OpenStack Benchmarking サービスは **テクノロジープレビュー** として提供されているため、Red Hat では全面的にはサポートしていません。これは、テスト目的のみでご利用いただく機能で、実稼働環境にデプロイすべきではありません。 テクノロジープレビューについての詳しい情報は 「対象範囲の詳細」を参照してください。

18.1. OPENSTACK BENCHMARKING サービスパッケージのインストール

開始する前に:

- **`root`** ユーザーとして、**`virt-manager`** を使用して **`rally`** という名前の仮想マシンを作成して、Red Hat Enterprise Linux 7.2 をインストールします。詳しい情報は 「VIRT-MANAGER を使用したゲスト作成」を参照してください。

- Benchmarking サービスの仮想マシンには、内部 API ネットワーク、テナントネットワーク、コントローラーネットワークへのアクセスが必要です。

  注記

  詳しい情報は 「VIRT-MANAGER を使用したゲスト作成」を参照してください。

- Benchmarking サービスの仮想マシンで必要なチャンネルにサブスクライブされていることを確認します。

手順18.1 Benchmarking サービスのインストール

1. Benchmarking サービスの仮想マシンで **`root`** ユーザーとして作業ディレクトリーを作成します。

```
# mkdir /path/to/myrally
```

2. リポジトリーをクローンします。

```
# git clone https://github.com/redhat-openstack/rally.git
```

3. 作業ディレクトリーに移動します。

```
# cd /path/to/myrally/rally
```

4. **`root`** ユーザーで **`install_rally.sh`** スクリプトを実行して、プロンプトに従い必要な依存関係をインストールします。

```
# //path/to/myrally/rally/install_rally.sh

[ ... Output Abbreviated ...]
Installing rally-manage script to /usr/bin
Installing rally script to /usr/bin
====================================================================
==
Information about your Rally installation:
* Method: system
* Database at: /var/lib/rally/database
* Configuration file at: /etc/rally
===================================================
```

18.2. OPENSTACK BENCHMARKING サービスの設定

Benchmarking サービスを正常にインストールしたら、OpenStack デプロイメントへのサービスアクセスを提供して、Benchmarking サービスの環境を設定する必要があります。

手順18.2 OpenStack Benchmarking サービスの設定

1. 適切な認証情報を使用して、OpenStack の環境変数をエクスポートします。この環境の認証情報は以下のとおりです。

```
# export OS_USERNAME=admin
# export OS_TENANT_NAME=admin
# export OS_PASSWORD=
# export OS_AUTH_URL=http://IP:35357/v2
```

2. 既存の Red Hat OpenStack 環境を Benchmarking サービスのデータベースに追加します。**osp-deployment.json** という名前の Benchmarking サービスのデプロイメント json ファイルを作成します。

```
{
  "type": "ExistingCloud",
    "auth_url": "http://DEPLOYMENT_IP:35357/v2.0/",
    "region_name": "RegionOne",
  "endpoint_type": "public",
    "admin": {
        "username": "admin",
        "password": "********",
        "tenant_name": "admin"
    },
    "users": [
        {
            "username": "admin",
            "password": "*****",
            "tenant_name": "admin"
        }
    ]
}
```

3. (オプション) **rally** 外で追加のユーザーを作成して、**osp-deployment.json** ファイルに追加する場合は、既知の競合状態に関する情報は 1175432 を参照してください。

4. 既存の OpenStack デプロイメントを **rally** データベースに追加します。以下の例では、エントリー *RA_Rally* に名前を指定して、**osp-deployment.json** ファイルで必要とされる適切な認証情報すべてを取得します。

```
# rally deployment create --name RA-Rally --file osp-deployment.json
--------------------------+--------------------------+---------
-+-----------------+--------+
| uuid                     | created_at                 | name
| status           | active |
+--------------------------+--------------------------+---------
-+-----------------+--------+
| 496e3be9-6020-4543-82f6-e7eb90517153 | 2015-05-12 15:18:13.817669
| RA-Rally | deploy->finished |    |
+--------------------------+--------------------------+---------
-+-----------------+--------+

Using deployment: 496e3be9-6020-4543-82f6-e7eb90517153
~/.rally/openrc was updated
[ ... Output Abbreviated ... ]
```

18.3. BENCHMARKING サービスのテナントネットワークの拡張

以下の手順は **オプション** で、実行する Benchmarking サービスのシナリオでゲストに (**ssh**) で直接アクセスできる必要がある場合のみに必要です。**NovaServers.boot_server** シナリオを実行する場合は、テナントネットワークを拡張する必要はありません。理由は、この固有なシナリオではゲストに **ssh** アクセスする必要がなく、ゲストを起動するだけで良いためです。

他に指定がない限り、コントローラーノードで以下の手順を実行します。

1. Red Hat OpenStack Platform のデプロイメントで Floating IP を利用できない場合には、**テナント** ネットワークを **rally** ホストに拡張します。以下のコマンドを実行して、OpenStack Networking OpenvSwitch エージェントパッケージをインストールします。

```
# yum install openstack-neutron-openvswitch
```

2. OpenStack Networking 設定ファイルをコンピュートノードからコントローラーノードにコピーします。

```
# scp COMPUTE_NODE_IP:/etc/neutron/* /etc/neutron
```

3. OpenStack クラスターの一部であった場合は、**Tenant** ネットワークに関連付けられるはずのインターフェースに、IP アドレスを設定します。

4. **ovs_neutron_plugin.ini** ファイルを編集して、**Tenant** ネットワークに配置されている IP アドレスに **local_ip** を変更します。

```
[OVS]
vxlan_udp_port=4789
network_vlan_ranges=
tunnel_type=vxlan
tunnel_id_ranges=1:1000
tenant_network_type=vxlan
local_ip=IP_WITHIN_TENANT_NETWORK
```

```
enable_tunneling=True
integration_bridge=br-int
tunnel_bridge=br-tu
```

5. **/etc/neutron/plugin.ini** ファイルを編集して、**Tenant** ネットワークに配置されている IP アドレスに **local_ip** を変更します。

```
[OVS]
vxlan_udp_port=4789
network_vlan_ranges=
tunnel_type=vxlan
tunnel_id_ranges=1:1000
tenant_network_type=vxlan
local_ip=IP_WITHIN_TENANT_NETWORK
enable_tunneling=True
integration_bridge=br-int
tunnel_bridge=br-tun
```

6. **neutron-openvswitch-agent** を再起動します。

```
# systemctl restart openvswitch
# systemctl restart neutron-openvswitch-agent
```

7. コントローラーノード上で、**neutron agent-list** コマンドを使用してエージェントが正しく設定されていることを確認します。

```
# neutron agent-list
... Output Abbreviated ... ]
| 8578499b-0873-47a9-9cae-9884d4abf768 | Open vSwitch agent |
Controller_Host | :-) | True | neutron-openvswitch-agent
```

8. コントローラーノードで、以下の情報を使用して **netid** と **hostid** の変数を作成します。

```
# netid=PRIVATE_NETWORK_ID
# echo $netid
# hostid=CONTROLLER_NODE_HOSTNAME
# echo $hostid
```

9. コントローラーノードで **rally-port** という名前の **neutron** ポートを作成して、**host_id** にバインドします。このポートは、**netid** に関連付けられたネットワーク内に作成するようにしてください。

```
# neutron port-create --name rally-port --binding:host_id=$hostid
$netid
Created a new port:
+-----------------------
+----------------------------------------------------------------
-------------------+
| Field                 | Value
|
+-----------------------
+----------------------------------------------------------------
-------------------+
```

```
| admin_state_up        | True
|
| allowed_address_pairs |
|
| binding:host_id       | VISIONING_HOST
|
| binding:profile       | {}
|
| binding:vif_details   | {"port_filter": true, "ovs_hybrid_plug":
true}
|
| binding:vif_type      | ovs
|
| binding:vnic_type     | normal
|
| device_id             |
|
| device_owner          |
|
| extra_dhcp_opts       |
|
| fixed_ips             | {"subnet_id": "5279a66d-a5f5-4639-b664-
163c39f26838", "ip_address": "10.1.0.2"}
|
| id                    | 0a9e54da-6588-42ce-9011-d637c3387670
|
| mac_address           | fa:16:3e:44:c3:d9
|
| name                  | rally-port
|
| network_id            | 1cd7ae4f-d057-41bd-8c56-557771bf9f73
|
| security_groups       | 76fe37c7-debb-46b4-a57a-d7dbb9dfd0ed
|
| status                | DOWN
|
| tenant_id             | 6f98906137a2421da579d4e70714cac6
|
+-----------------------
+----------------------------------------------------------------
------------------+
```

10. コントローラーノード内で、**rally** 仮想マシンの XML を以下のように変更します。

```
# virsh edit rally
...
   <interface type='bridge'>
     <mac address='fa:16:3e:1a:3b:f1'/>
     <source bridge='br-int'/>
     <virtualport type='openvswitch'>
       <parameters interfaceid='neutron-port-id'/>
     </virtualport>
     <target dev='vnet4'/>
     <model type='virtio'/>
     <alias name='net1'/>
     <address type='pci' domain='0x0000' bus='0x00' slot='0x0f'
```

```
function='0x0'/>
   </interface>
...
```

注記

rally-port を作成した先ほどの手順で特定した **id** に **neutron-port-id** が更新されていることを確認してください。

11. **rally** ゲストに XML ファイルの変更を適用した後には、ゲストを再起動します。

18.4. OPENSTACK BENCHMARKING サービスの検証

rally の仮想マシンで、以下のコマンドを実行します。

1. Red Hat OpenStack Platform のデプロイメントの現在の状況を表示します。

```
# rally deployment list
--------------------------+---------------------------+---------
-+-----------------+--------+
| uuid                                 | created_at
| name     | status           | active |
+--------------------------------------+------------------------
---+----------+----------------+--------+
| 496e3be9-6020-4543-82f6-e7eb90517153 | 2015-05-12 15:18:13.817669
| RA-Rally | deploy->finished | * |
+--------------------------------------+------------------------
---+----------+----------------+--------+
```

2. **create-user.json** という名前の同梱サンプル **keystone** のシナリオを使用して **rally** 環境をテストし、Benchmarking サービスが正しく機能していることを確認します。以下の **create-user.json** シナリオでは、1 度に 10 個の名前を最大 100 回作成し、ユーザーの作成時間を 10 個からなるサブセットで、合計ユーザー生成数が最大 (100 個まで) に達するまで表示されます。このサンプルテストでは、Benchmarking サービスが OpenStack 環境にアクセスできるかどうかを検証します。

```
# rally task start
/path/to/myrally/rally/samples/tasks/scenarios/keystone/create-
user.json
```