OpenShift Container Platform 4.19

OpenStack へのインストール

OpenStack への OpenShift Container Platform のインストールする

Last Updated: 2025-12-10

OpenShift Container Platform 4.19 OpenStack へのインストール

OpenStack への OpenShift Container Platform のインストールする

Legal Notice

Abstract

このドキュメントでは、OpenStack に OpenShift Container Platform をインストールする方法を説明します。

Table of Contents

第1章 OPENSTACK へのインストールの準備 4
1.1. 前提条件 4
1.2. OPENSTACK に OPENSHIFT CONTAINER PLATFORM をインストールする方法の選択 4
1.3. RHOSP エンドポイントをスキャンしてレガシー HTTPS 証明書を探す 5

第2章 PREPARING TO INSTALL A CLUSTER THAT USES SR-IOV OR OVS-DPDK ON OPENSTACK 9
2.1. SR-IOV または OVS-DPDK のいずれかを使用する RHOSP 上のクラスターの要件 9
2.2. SR-IOV を使用するクラスターのインストールの準備 10
2.3. OVS-DPDK を使用するクラスターのインストールの準備 11
2.4. 次のステップ 11

第3章 カスタマイズによる OPENSTACK へのクラスターのインストール 12
3.1. 前提条件 12
3.2. OPENSHIFT CONTAINER PLATFORM を RHOSP にインストールするリソースのガイドライン 12
3.3. OPENSHIFT CONTAINER PLATFORM のインターネットアクセス 18
3.4. RHOSP での SWIFT の有効化 19
3.5. RHOSP で実行されるクラスター上のカスタムストレージを使用したイメージレジストリーの設定 19
3.6. 外部ネットワークアクセスの確認 22
3.7. インストールプログラムのパラメーターの定義 23
3.8. OPENSTACK CLOUD CONTROLLER MANAGER のオプション設定 24
3.9. インストールプログラムの取得 26
3.10. インストール設定ファイルの作成 27
3.11. クラスターノードの SSH アクセス用のキーペアの生成 45
3.12. 環境へのアクセスの有効化 46
3.13. クラスターのデプロイ 48
3.14. クラスターステータスの確認 50
3.15. CLI の使用によるクラスターへのログイン 50
3.16. OPENSHIFT CONTAINER PLATFORM の TELEMETRY アクセス 51
3.17. 次のステップ 51

第4章 独自のインフラストラクチャーを使用した OPENSTACK へのクラスターのインストール 53
4.1. 前提条件 53
4.2. OPENSHIFT CONTAINER PLATFORM のインターネットアクセス 53
4.3. OPENSHIFT CONTAINER PLATFORM を RHOSP にインストールするリソースのガイドライン 54
4.4. PLAYBOOK 依存関係のダウンロード 56
4.5. インストール PLAYBOOK のダウンロード 57
4.6. インストールプログラムの取得 58
4.7. クラスターノードの SSH アクセス用のキーペアの生成 59
4.8. RED HAT ENTERPRISE LINUX COREOS (RHCOS) イメージの作成 61
4.9. 外部ネットワークアクセスの確認 62
4.10. 環境へのアクセスの有効化 63
4.11. インストールプログラムのパラメーターの定義 65
4.12. RHOSP でのネットワークリソースの作成 67
4.13. インストール設定ファイルの作成 68
4.14. KUBERNETES マニフェストおよび IGNITION 設定ファイルの作成 77
4.15. ブートストラップ IGNITION ファイルの準備 79
4.16. RHOSP でのコントロールプレーンの IGNITION 設定ファイルの作成 82
4.17. RHOSP でのネットワークリソースの更新 83
4.18. RHOSP でのブートストラップマシンの作成 86
4.19. RHOSP でのコントロールプレーンの作成 86
4.20. CLI の使用によるクラスターへのログイン 87
4.21. RHOSP からのブートストラップリソースの削除 88

4.22. RHOSP でのコンピュートマシンの作成 88
4.23. マシンの証明書署名要求の承認 89
4.24. インストールの正常な実行の確認 92
4.25. OPENSHIFT CONTAINER PLATFORM の TELEMETRY アクセス 92
4.26. 次のステップ 92

第5章 非接続環境の OPENSTACK へのクラスターインストール **94**
5.1. 前提条件 94
5.2. ネットワークが制限された環境でのインストールについて 94
5.3. OPENSHIFT CONTAINER PLATFORM を RHOSP にインストールするリソースのガイドライン 95
5.4. OPENSHIFT CONTAINER PLATFORM のインターネットアクセス 97
5.5. RHOSP での SWIFT の有効化 97
5.6. インストールプログラムのパラメーターの定義 98
5.7. OPENSTACK CLOUD CONTROLLER MANAGER のオプション設定 99
5.8. ネットワークが制限されたインストール用の RHCOS イメージの作成 101
5.9. インストール設定ファイルの作成 102
5.10. クラスターノードの SSH アクセス用のキーペアの生成 107
5.11. 環境へのアクセスの有効化 109
5.12. クラスターのデプロイ 111
5.13. クラスターステータスの確認 113
5.14. CLI の使用によるクラスターへのログイン 113
5.15. デフォルトの OPERATORHUB カタログソースの無効化 114
5.16. OPENSHIFT CONTAINER PLATFORM の TELEMETRY アクセス 115
5.17. 次のステップ 115

第6章 OPENSTACK に 3 ノードクラスターをインストールする **116**
6.1. 3 ノードクラスターの設定 116
6.2. 次のステップ 116

第7章 OPENSTACK をインストールした後のネットワーク設定の設定 **117**
7.1. FLOATING IP アドレスを使用したアプリケーションアクセスの設定 117
7.2. OVS ハードウェアオフロードの有効化 118
7.3. OVS ハードウェアオフロードネットワークの接続 120
7.4. RHOSP で POD への IPV6 接続を有効にする 121
7.5. RHOSP で IPV6 接続を持つ POD の作成 121
7.6. RHOSP 上の POD への IPV6 接続の追加 123

第8章 OPENSTACK CLOUD CONTROLLER MANAGER リファレンスガイド **124**
8.1. OPENSTACK CLOUD CONTROLLER MANAGER 124
8.2. OPENSTACK CLOUD CONTROLLER MANAGER (CCM) 設定マップ 125

第9章 ローカルディスク上の ROOTVOLUME および ETCD を使用した OPENSTACK へのデプロイ **131**
9.1. ローカルディスクへの RHOSP のデプロイ 131
9.2. 関連情報 135

第10章 OPENSTACK でのクラスターのアンインストール **136**
10.1. INSTALLER-PROVISIONED INFRASTRUCTURE を使用するクラスターの削除 136

第11章 独自のインフラストラクチャーからの RHOSP のクラスターのアンインストール **138**
11.1. PLAYBOOK 依存関係のダウンロード 138
11.2. 独自のインフラストラクチャーを使用する RHOSP からのクラスターの削除 139

第12章 OPENSTACK のインストール設定パラメーター **140**
12.1. OPENSTACK で使用可能なインストール設定パラメーター 140

第1章 OPENSTACK へのインストールの準備

Red Hat OpenStack Platform (RHOSP) に OpenShift Container Platform をインストールできます。

1.1. 前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスの詳細を確認している。
- クラスターインストール方法の選択およびそのユーザー向けの準備 を確認している。

1.2. OPENSTACK に OPENSHIFT CONTAINER PLATFORM をインストールする方法の選択

OpenShift Container Platform をインストーラーまたは user-provisioned infrastructure にインストールすることができます。デフォルトのインストールタイプは、installer-provisioned infrastructure を使用します。この場合、インストールプログラムがクラスターの基礎となるインフラストラクチャーをプロビジョニングします。OpenShift Container Platform は、ユーザーがプロビジョニングするインフラストラクチャーにインストールすることもできます。インストールプログラムがプロビジョニングするインフラストラクチャーを使用しない場合は、クラスターリソースをユーザー自身で管理し、維持する必要があります。

installer-provisioned installation および user-provisioned installation のプロセスの詳細は、インストールプロセス を参照してください。

1.2.1. installer-provisioned infrastructure へのクラスターのインストール

以下の方法のいずれかを使用して、OpenShift Container Platform インストールプログラムでプロビジョニングされる Red Hat OpenStack Platform (RHOSP) インフラストラクチャーに、クラスターをインストールできます。

- **カスタマイズによる OpenStack へのクラスターのインストール**: カスタマイズされたクラスターを RHOSP にインストールできます。インストールプログラムは、インストールの段階で一部のカスタマイズを適用できるようにします。その他の多くのカスタマイズオプションは、インストール後 に利用できます。
- **ネットワークが制限された環境での OpenStack へのクラスターのインストール**: インストールリリースコンテンツの内部ミラーを作成して、OpenShift Container Platform をネットワークが制限された環境またはネットワークの非接続環境で RHOSP にインストールできます。この方法を使用して、ソフトウェアコンポーネントを取得するためにアクティブなインターネット接続を必要としないクラスターをインストールできます。また、このインストール方法を使用して、クラスターが外部コンテンツに対する組織の制御の条件を満たすコンテナーイメージのみを使用するようにすることもできます。

1.2.2. user-provisioned infrastructure へのクラスターのインストール

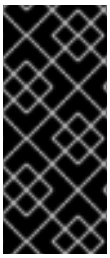
以下の方法のいずれかを使用して、独自にプロビジョニングする RHOSP インフラストラクチャーにクラスターをインストールできます。

- **独自のインフラストラクチャーでの OpenStack へのクラスターのインストール**: user-provisioned RHOSP インフラストラクチャーに OpenShift Container Platform をインストールできます。このインストール方法を使用して、クラスターを既存のインフラストラクチャーおよび変更と統合できます。user-provisioned infrastructure でのインストールの場合、Nova

サーバー、Neutron ポート、セキュリティーグループなどの RHOSP リソースをすべて作成する必要があります。提供される Ansible Playbook を使用してデプロイメントプロセスを支援することができます。

1.3. RHOSP エンドポイントをスキャンしてレガシー HTTPS 証明書を探す

OpenShift Container Platform 4.10 以降、HTTPS 証明書にはサブジェクト代替名 (SAN) フィールドが含まれている必要があります。次のスクリプトを実行して、Red Hat OpenStack Platform (RHOSP) カタログ内の各 HTTPS エンドポイントをスキャンし、**CommonName** フィールドのみを含むレガシー証明書を探します。



重要

OpenShift Container Platform は、インストールまたは更新の前に、基盤となる RHOSP インフラストラクチャーのレガシー証明書をチェックしません。提供されているスクリプトを使用して、これらの証明書をご自身で確認してください。クラスターをインストールまたは更新する前にレガシー証明書を更新しないと、クラスターが機能しなくなります。

前提条件

- スクリプトを実行するマシンに、次のソフトウェアをインストールします。
  - Bash バージョン 4.0 以降
  - **grep**
  - OpenStack クライアント
  - **jq**
  - OpenSSL バージョン 1.1.1l 以降
- ターゲットクラウドの RHOSP クレデンシャルをマシンに入力します。

手順

1. 次のスクリプトをマシンに保存します。

```
#!/usr/bin/env bash

set -Eeuo pipefail

declare catalog san
catalog="$(mktemp)"
san="$(mktemp)"
readonly catalog san

declare invalid=0

openstack catalog list --format json --column Name --column Endpoints \
 | jq -r '.[] | .Name as $name | .Endpoints[] | select(.interface=="public") | [$name, .interface,
.url] | join(" ")' \
 | sort \
 > "$catalog"
```

```
while read -r name interface url; do
 # Ignore HTTP
 if [[ ${url#"http://"} != "$url" ]]; then
  continue
 fi

 # Remove the schema from the URL
 noschema=${url#"https://"}

 # If the schema was not HTTPS, error
 if [[ "$noschema" == "$url" ]]; then
  echo "ERROR (unknown schema): $name $interface $url"
  exit 2
 fi

 # Remove the path and only keep host and port
 noschema="${noschema%%/*}"
 host="${noschema%%:*}"
 port="${noschema##*:}"

 # Add the port if was implicit
 if [[ "$port" == "$host" ]]; then
  port='443'
 fi

 # Get the SAN fields
 openssl s_client -showcerts -servername "$host" -connect "$host:$port" </dev/null
2>/dev/null \
  | openssl x509 -noout -ext subjectAltName \
  > "$san"

 # openssl returns the empty string if no SAN is found.
 # If a SAN is found, openssl is expected to return something like:
 #
 #    X509v3 Subject Alternative Name:
 #        DNS:standalone, DNS:osp1, IP Address:192.168.2.1, IP Address:10.254.1.2
 if [[ "$(grep -c "Subject Alternative Name" "$san" || true)" -gt 0 ]]; then
  echo "PASS: $name $interface $url"
 else
  invalid=$((invalid+1))
  echo "INVALID: $name $interface $url"
 fi
done < "$catalog"

# clean up temporary files
rm "$catalog" "$san"

if [[ $invalid -gt 0 ]]; then
 echo "${invalid} legacy certificates were detected. Update your certificates to include a SAN
field."
 exit 1
else
 echo "All HTTPS certificates for this cloud are valid."
fi
```

2. スクリプトを実行します。
3. スクリプトが **INVALID** と報告する証明書を、SAN フィールドを含む証明書に置き換えます。

重要

OpenShift Container Platform 4.10 をインストールする前、またはクラスターをそのバージョンに更新する前に、すべてのレガシー HTTPS 証明書を置き換える必要があります。レガシー証明書は、次のメッセージで拒否されます。

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

1.3.1. RHOSP エンドポイントをスキャンしてレガシー HTTPS 証明書を手動で探す

OpenShift Container Platform 4.10 以降、HTTPS 証明書にはサブジェクト代替名 (SAN) フィールドが含まれている必要があります。「レガシー HTTPS 証明書の RHOSP エンドポイントのスキャン」にリストされている前提条件ツールにアクセスできない場合は、次の手順を実行して、Red Hat OpenStack Platform (RHOSP) カタログ内の各 HTTPS エンドポイントをスキャンして、**CommonName** フィールドのみを含むレガシー証明書の RHOSP カタログで各 HTTPS エンドポイントをスキャンします。

重要

OpenShift Container Platform は、インストールまたは更新の前に、基盤となる RHOSP インフラストラクチャーのレガシー証明書をチェックしません。これらの証明書を自分で確認するには、次の手順を使用します。クラスターをインストールまたは更新する前にレガシー証明書を更新しないと、クラスターが機能しなくなります。

手順

1. コマンドラインで次のコマンドを実行して、RHOSP パブリックエンドポイントの URL を表示します。

```
$ openstack catalog list
```

   コマンドが返す各 HTTPS エンドポイントの URL を記録します。

2. 各パブリックエンドポイントについて、ホストとポートをメモします。

   ヒント

   スキーム、ポート、およびパスを削除して、エンドポイントのホストを決定します。

3. エンドポイントごとに次のコマンドを実行して、証明書の SAN フィールドを抽出します。

   a. **host** 変数を設定します。

```
$ host=<host_name>
```

   b. **port** 変数を設定します。

```
$ port=<port_number>
```

エンドポイントの URL にポートがない場合は、値 **443** を使用します。

c. 証明書の SAN フィールドを取得します。

```
$ openssl s_client -showcerts -servername "$host" -connect "$host:$port" </dev/null
2>/dev/null \
    | openssl x509 -noout -ext subjectAltName
```

出力例

```
X509v3 Subject Alternative Name:
    DNS:your.host.example.net
```

各エンドポイントについて、前の例に似た出力を探します。エンドポイントの出力がない場合、そのエンドポイントの証明書は無効であるため、再発行する必要があります。

重要

OpenShift Container Platform 4.10 をインストールする前、またはクラスターをそのバージョンに更新する前に、すべてのレガシー HTTPS 証明書を置き換える必要があります。従来の証明書は拒否され、次のメッセージが表示されます。

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

第2章 PREPARING TO INSTALL A CLUSTER THAT USES SR-IOV OR OVS-DPDK ON OPENSTACK

Single Root I/O Virtualization (SR-IOV) または Open vSwitch を使用する OpenShift Container Platform クラスターを Red Hat OpenStack Platform (RHOSP) に Data Plane Development Kit (OVS-DPDK) とともにインストールする前に、テクノロジーごとの要件を理解し、準備タスクを実行する必要があります。

2.1. SR-IOV または OVS-DPDK のいずれかを使用する RHOSP 上のクラスターの要件

デプロイメントで SR-IOV または OVS-DPDK を使用する場合は、次の要件を満たす必要があります。

- RHOSP コンピュートノードは、Huge Page をサポートするフレーバーを使用する必要があります。

2.1.1. SR-IOV を使用する RHOSP 上のクラスターの要件

デプロイメントで Single Root I/O Virtualization (SR-IOV) を使用するには、次の要件を満たす必要があります。

- Red Hat OpenStack Platform (RHOSP) SR-IOV デプロイメントを計画します 。
- OpenShift Container Platform は、使用する NIC をサポートする必要があります。サポートされている NIC のリストは、「ネットワーキング」ドキュメントの「ハードウェアネットワーク」サブセクションにある「Single Root I/O Virtualization (SR-IOV) ハードウェアネットワークについて」を参照してください。
- SR-IOV NIC がアタッチされるノードごとに、RHOSP クラスターに以下が必要です。
  - RHOSP クォータからの 1 インスタンス
  - マシンのサブネットにアタッチされた 1 つのポート
  - SR-IOV 仮想機能ごとに 1 つのポート
  - 少なくとも 16 GB のメモリー、4 つの vCPU および 25 GB のストレージ領域があるフレーバー
- SR-IOV デプロイメントでは、多くの場合、専用の CPU や分離された CPU などのパフォーマンスの最適化が駆使されます。パフォーマンスを最大化するには、基礎となる RHOSP デプロイメントをこれらの最適化機能を使用するように設定してから、OpenShift Container Platform コンピュートマシンを最適化されたインフラストラクチャーで実行するように設定します。
  - パフォーマンスの良い RHOSP コンピュートノードの設定の詳細は、パフォーマンスを向上させるためのコンピュートノードの設定 を参照してください。

2.1.2. OVS-DPDK を使用する RHOSP 上のクラスターの要件

デプロイメントで、Open vSwitch を Data Plane Development Kit (OVS-DPDK) とともに使用するには、以下の要件を満たす必要があります。

- ネットワーク機能仮想化のプランニングおよび設定ガイドの OVS-DPDK デプロイメントのプランニング を参照して、Red Hat OpenStack Platform (RHOSP) OVS-DPDK デプロイメントを計画します。

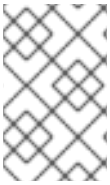
- ネットワーク機能仮想化のプランニングおよび設定ガイドの OVS-DPDK デプロイメントの設定 に従って、RHOSP OVS-DPDK デプロイメントを設定します。

2.2. SR-IOV を使用するクラスターのインストールの準備

SR-IOV を使用するクラスターをインストールする前に、RHOSP を設定する必要があります。

2.2.1. コンピュートマシン用の SR-IOV ネットワークの作成

Red Hat OpenStack Platform (RHOSP) デプロイメントで Single Root I/O Virtualization (SR-IOV) をサポートする場合、コンピュートマシンを実行する SR-IOV ネットワークをプロビジョニングすることができます。



注記

以下の手順では、コンピュートマシンへの接続が可能な外部のフラットネットワークおよび外部の VLAN ベースのネットワークを作成します。RHOSP のデプロイメントによっては、ネットワークの他のタイプが必要になる場合があります。

前提条件

- クラスターは SR-IOV をサポートしている。

注記

クラスターがサポートするかどうかが不明な場合は、OpenShift Container Platform SR-IOV ハードウェアネットワークに関するドキュメントを参照してください。

- RHOSP デプロイメントの一部として、無線とアップリンクのプロバイダーネットワークを作成している。これらのネットワークを表すために **radio** および **uplink** の名前がすべてのコマンド例で使用されています。

手順

1. コマンドラインで、無線の RHOSP ネットワークを作成します。

```
$ openstack network create radio --provider-physical-network radio --provider-network-type flat --external
```

2. アップリンクの RHOSP ネットワークを作成します。

```
$ openstack network create uplink --provider-physical-network uplink --provider-network-type vlan --external
```

3. 無線ネットワーク用のサブネットを作成します。

```
$ openstack subnet create --network radio --subnet-range <radio_network_subnet_range> radio
```

4. アップリンクネットワーク用のサブネットを作成します。

```
$ openstack subnet create --network uplink --subnet-range <uplink_network_subnet_range>
uplink
```

2.3. OVS-DPDK を使用するクラスターのインストールの準備

SR-IOV を使用するクラスターをインストールする前に、RHOSP を設定する必要があります。

- RHOSP にクラスターをインストールする前に、OVS-DPDK 用のフレーバーの作成とインスタンスのデプロイ を完了します。

インストール前のタスクを実行したら、最も関連性の高い OpenShift Container Platform on RHOSP のインストール手順に従ってクラスターをインストールします。次に、このページの「次のステップ」の下にあるタスクを実行します。

2.4. 次のステップ

- いずれかのデプロイメントタイプで、以下を実行します。
  - huge page をサポートする Node Tuning Operator の設定
- クラスターをデプロイした後に SR-IOV 設定を完了するには、以下を実行します。
  - SR-IOV Operator をインストールします。
  - SR-IOV ネットワークデバイスを設定 します。
  - SR-IOV コンピュートマシンを作成 します。
- パフォーマンスを向上させるためにクラスターをデプロイした後、次の参考資料を確認してください。
  - OpenStack で OVS-DPDK を使用するクラスター用のテスト Pod テンプレート
  - OpenStack で SR-IOV を使用するクラスター用のテスト Pod テンプレート
  - OpenStack で OVS-DPDK を使用するクラスター用のパフォーマンスプロファイルテンプレート

第3章 カスタマイズによる OPENSTACK へのクラスターのインストール

OpenShift Container Platform バージョン 4.19 では、Red Hat OpenStack Platform (RHOSP) にカスタマイズしたクラスターをインストールできます。インストールをカスタマイズするには、クラスターをインストールする前に **install-config.yaml** でパラメーターを変更します。

3.1. 前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスの詳細を確認している。
- クラスターインストール方法の選択およびそのユーザー向けの準備 を確認している。
- OpenShift クラスターでサポートされるプラットフォーム セクションを使用して、OpenShift Container Platform 4.19 が RHOSP バージョンと互換性があることを確認した。RHOSP サポートマトリックスの OpenShift Container Platform を参照して、プラットフォームのサポートを異なるバージョン間で比較することもできます。
- ブロックストレージ (Cinder) またはオブジェクトストレージ (Swift) などのストレージサービスが RHOSP にインストールされている。オブジェクトストレージは、OpenShift Container Platform レジストリークラスターデプロイメントに推奨されるストレージ技術です。詳細は、ストレージの最適化 を参照してください。
- クラスターのスケーリング、コントロールプレーンのサイジング、および etcd のパフォーマンスおよびスケーラビリティーに関する理解がある。詳細は、クラスターのスケーリングに関する推奨プラクティス を参照してください。
- RHOSP でメタデータサービスが有効化されている。

3.2. OPENSHIFT CONTAINER PLATFORM を RHOSP にインストールするリソースのガイドライン

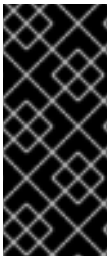
OpenShift Container Platform のインストールをサポートするために、Red Hat OpenStack Platform (RHOSP) クォータは以下の要件を満たす必要があります。

表3.1 RHOSP のデフォルトの OpenShift Container Platform クラスターに関する推奨リソース

リソース	値
Floating IP アドレス	3
ポート	15
ルーター	1
サブネット	1
RAM	88 GB
vCPU	22

リソース	値
ボリュームストレージ	275 GB
インスタンス	7
セキュリティーグループ	3
セキュリティーグループルール	60
サーバーグループ	2 - 各マシンプールの追加のアベイラビリティーゾーンごとに 1 つ追加

クラスターは推奨されるリソースよりもリソースが少ない場合にも機能する場合がありますが、その場合のパフォーマンスは保証されません。



重要

RHOSP オブジェクトストレージ (Swift) が利用可能で、**swiftoperator** ロールを持つユーザーアカウントによって操作されている場合、これは OpenShift Container Platform イメージレジストリーのデフォルトバックエンドとして使用されます。この場合、ボリュームストレージ要件は 175 GB です。Swift 領域要件は、イメージレジストリーのサイズによって異なります。

注記

デフォルトで、セキュリティーグループおよびセキュリティーグループルールのクォータは低く設定される可能性があります。問題が生じた場合には、管理者として **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** を実行して値を増やします。

OpenShift Container Platform デプロイメントは、コントロールプレーンマシン、コンピュートマシン、およびブートストラップマシンで構成されます。

3.2.1. コントロールプレーンマシン

デフォルトでは、OpenShift Container Platform インストールプロセスは 3 つのコントロールプレーンマシンを作成します。

それぞれのマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 16 GB のメモリーと 4 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

3.2.2. コンピュートマシン

デフォルトでは、OpenShift Container Platform インストールプロセスは 3 つのコンピューティングマシンを作成します。

それぞれのマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 8 GB のメモリーと 2 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

ヒント

コンピュートマシンは、OpenShift Container Platform で実行されるアプリケーションをホストします。できるだけ多くのアプリケーションを実行することが意図されています。

3.2.3. ブートストラップマシン

インストール時に、ブートストラップマシンは一時的にプロビジョニングされ、コントロールプレーンを初期化します。実稼働環境用のコントロールプレーンの準備ができた後に、ブートストラップマシンのプロビジョニングは解除されます。

ブートストラップマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 16 GB のメモリーと 4 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

3.2.4. user-provisioned infrastructure の負荷分散要件

OpenShift Container Platform をインストールする前に、デフォルトの内部ロードバランシングソリューションの代わりに使用する独自の API およびアプリケーション Ingress ロードバランシングインフラストラクチャーをプロビジョニングできます。実稼働のシナリオでは、API およびアプリケーション Ingress ロードバランサーを個別にデプロイし、それぞれのロードバランサーインフラストラクチャーを分離してスケーリングすることができます。

注記

Red Hat Enterprise Linux (RHEL) インスタンスを使用して API およびアプリケーション Ingress ロードバランサーをデプロイする場合は、RHEL サブスクリプションを別途購入する必要があります。

負荷分散インフラストラクチャーは以下の要件を満たす必要があります。

1. **API ロードバランサー**: プラットフォームと対話およびプラットフォームを設定するためのユーザー向けの共通のエンドポイントを提供します。以下の条件を設定します。
    - Layer 4 の負荷分散のみ。これは、Raw TCP または SSL パススルーモードと呼ばれます。

- ステートレス負荷分散アルゴリズム。オプションは、ロードバランサーの実装によって異なります。

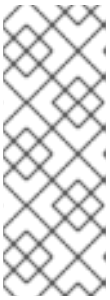
重要

API ロードバランサーのセッションの永続性は設定しないでください。Kubernetes API サーバーのセッション永続性を設定すると、OpenShift Container Platform クラスターとクラスター内で実行される Kubernetes API の過剰なアプリケーショントラフィックによりパフォーマンスの問題が発生する可能性があります。

ロードバランサーのフロントとバックの両方で以下のポートを設定します。

表3.2 API ロードバランサー

ポート	バックエンドマシン (プールメンバー)	内部	外部	説明
6443	ブートストラップおよびコントロールプレーン。ブートストラップマシンがクラスターのコントロールプレーンを初期化した後に、ブートストラップマシンをロードバランサーから削除します。API サーバーのヘルスチェックプローブの **/readyz** エンドポイントを設定する必要があります。	X	X	Kubernetes API サーバー
22623	ブートストラップおよびコントロールプレーン。ブートストラップマシンがクラスターのコントロールプレーンを初期化した後に、ブートストラップマシンをロードバランサーから削除します。	X		マシン設定サーバー



注記

ロードバランサーは、API サーバーが **/readyz** エンドポイントをオフにしてからプールから API サーバーインスタンスを削除するまで最大 30 秒かかるように設定する必要があります。**/readyz** の後の時間枠内でエラーが返されたり、正常になったりする場合は、エンドポイントが削除または追加されているはずです。5 秒または 10 秒ごとのプロービングで、2 回連続成功すると正常、3 回連続失敗すると異常と判断する設定は、十分にテストされた値です。

2. **Application Ingress ロードバランサー**: クラスター外から送られるアプリケーショントラフィックの Ingress ポイントを提供します。Ingress ルーターの作業用の設定が OpenShift Container Platform クラスターに必要です。
以下の条件を設定します。

   - Layer 4 の負荷分散のみ。これは、Raw TCP または SSL パススルーモードと呼ばれます。

   - 選択可能なオプションやプラットフォーム上でホストされるアプリケーションの種類に基づいて、接続ベースの永続化またはセッションベースの永続化が推奨されます。

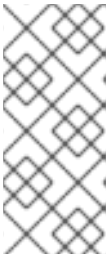
ヒント

クライアントの実際の IP アドレスがアプリケーション Ingress ロードバランサーによって確認できる場合、ソースの IP ベースのセッション永続化を有効にすると、エンドツーエンドの TLS 暗号化を使用するアプリケーションのパフォーマンスを強化できます。

ロードバランサーのフロントとバックの両方で以下のポートを設定します。

表3.3 アプリケーション Ingress ロードバランサー

ポート	バックエンドマシン (プールメンバー)	内部	外部	説明
443	デフォルトで Ingress コントローラー Pod、コンピュート、またはワーカーを実行するマシン。	X	X	HTTPS トラフィック
80	デフォルトで Ingress コントローラー Pod、コンピュート、またはワーカーを実行するマシン。	X	X	HTTP トラフィック



注記

ゼロ (0) コンピュートノードで 3 ノードクラスターをデプロイする場合、Ingress コントローラー Pod はコントロールプレーンノードで実行されます。3 ノードクラスターデプロイメントでは、HTTP および HTTPS トラフィックをコントロールプレーンノードにルーティングするようにアプリケーション Ingress ロードバランサーを設定する必要があります。

3.2.4.1. ユーザー管理のロードバランサーを使用してデプロイされたクラスターのロードバランサー設定の例

このセクションでは、ユーザー管理のロードバランサーを使用してデプロイされたクラスターのロードバランシング要件を満たす API およびアプリケーションの Ingress load balancer 設定の例を示します。この例は、HAProxy ロードバランサーの **/etc/haproxy/haproxy.cfg** 設定です。この例では、特定の負荷分散ソリューションを選択するためのアドバイスを提供することを目的としていません。

この例では、同じロードバランサーが Kubernetes API およびアプリケーションの Ingress トラフィックに使用されます。実稼働のシナリオでは、API およびアプリケーション Ingress ロードバランサーを個別にデプロイし、それぞれのロードバランサーインフラストラクチャーを分離してスケーリングすることができます。

注記

HAProxy をロードバランサーとして使用し、SELinux が **enforcing** に設定されている場合は、**setsebool -P haproxy_connect_any=1** を実行して、HAProxy サービスが設定済みの TCP ポートにバインドできることを確認する必要があります。

例3.1 API およびアプリケーション Ingress ロードバランサーの設定例

```
global
  log         127.0.0.1 local2
```

```
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1) ポート **6443** は Kubernetes API トラフィックを処理し、コントロールプレーンマシンを参照します。

2 4 ブートストラップエントリーは、OpenShift Container Platform クラスターのインストール前に有効にし、ブートストラッププロセスの完了後にそれらを削除する必要があります。

3 ポート **22623** はマシン設定サーバートラフィックを処理し、コントロールプレーンマシンを参照します。

5 ポート **443** は HTTPS トラフィックを処理し、Ingress コントローラー Pod を実行するマシンを参照します。Ingress コントローラー Pod はデフォルトでコンピュートマシンで実行されます。

6 ポート **80** は HTTP トラフィックを処理し、Ingress コントローラー Pod を実行するマシンを参照します。Ingress コントローラー Pod はデフォルトでコンピュートマシンで実行されます。

注記

ゼロ (0) コンピュートノードで 3 ノードクラスターをデプロイする場合、Ingress コントローラー Pod はコントロールプレーンノードで実行されます。3 ノードクラスターデプロイメントでは、HTTP および HTTPS トラフィックをコントロールプレーンノードにルーティングするようにアプリケーション Ingress ロードバランサーを設定する必要があります。

ヒント

HAProxy をロードバランサーとして使用する場合は、HAProxy ノードで **netstat -nltupe** を実行して、ポート **6443**、**22623**、**443**、および **80** で **haproxy** プロセスがリッスンしていることを確認することができます。

3.3. OPENSHIFT CONTAINER PLATFORM のインターネットアクセス

OpenShift Container Platform 4.19 では、クラスターをインストールするためにインターネットアクセスが必要です。

次のアクションを実行するには、インターネットにアクセスできる必要があります。

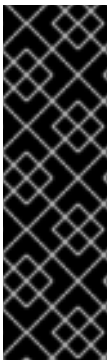
- OpenShift Cluster Manager にアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターがインターネットにアクセスでき、Telemetry を無効にしていない場合、そのサービスによってクラスターのサブスクリプションが自動的に有効化されます。
- クラスターのインストールに必要なパッケージを取得するために Quay.io にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。

重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにインストールパッケージを設定します。インストールタイプに応じて、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

3.4. RHOSP での SWIFT の有効化

Swift は、**swiftoperator** ロールのあるユーザーアカウントによって操作されます。インストールプログラムを実行する前に、ロールをアカウントに追加します。



重要

一般に Swift と呼ばれる Red Hat OpenStack Platform (RHOSP) オブジェクトストレージサービス が利用可能な場合、OpenShift Container Platform はこれをイメージレジストリーストレージとして使用します。利用できない場合、インストールプログラムは Cinder として知られる RHOSP ブロックストレージサービスに依存します。

Swift が存在し、これを使用する必要がある場合は、Swift へのアクセスを有効にする必要があります。これが存在しない場合や使用する必要がない場合は、このセクションを省略してください。

重要

RHOSP 17 では、Ceph RGW の **rgw_max_attr_size** パラメーターが 256 文字に設定されます。この設定は、コンテナーイメージを OpenShift Container Platform レジストリーにアップロードする際に問題を引き起こします。**rgw_max_attr_size** の値は、1024 文字以上に設定する必要があります。

インストールする前に、RHOSP のデプロイメントがこの問題の影響を受けるかどうか確認してください。影響を受ける場合は、Ceph RGW を再設定します。

前提条件

- ターゲット環境に RHOSP 管理者アカウントがあります。
- Swift サービスがインストールされています。
- Ceph RGW で、**account in url** オプションが有効化されています。

手順

RHOSP 上で Swift を有効にするには、以下を実行します。

1. RHOSP CLI の管理者として、**swiftoperator** ロールを Swift にアクセスするアカウントに追加します。

```
$ openstack role add --user <user> --project <project> swiftoperator
```

RHOSP デプロイメントでは、イメージレジストリーに Swift を使用することができます。

3.5. RHOSP で実行されるクラスター上のカスタムストレージを使用したイメージレジストリーの設定

Red Hat OpenStack Platform (RHOSP) にクラスターをインストールした後に、特定のアベイラビリティーゾーンにある Cinder ボリュームをレジストリーストレージとして使用できます。

手順

1. YAML ファイルを作成して、使用するストレージクラスとアベイラビリティーゾーンを指定します。以下に例を示します。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: custom-csi-storageclass
provisioner: cinder.csi.openstack.org
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
parameters:
  availability: <availability_zone_name>
```

注記

OpenShift Container Platform では、選択したアベイラビリティーゾーンが存在するかどうかは確認されません。設定を適用する前に、アベイラビリティーゾーンの名前を確認してください。

2. コマンドラインから設定を適用します。

```
$ oc apply -f <storage_class_file_name>
```

出力例

```
storageclass.storage.k8s.io/custom-csi-storageclass created
```

3. ストレージクラスと **openshift-image-registry** namespace を使用する永続ボリュームクレーム (PVC) を指定する YAML ファイルを作成します。以下に例を示します。

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-pvc-imageregistry
  namespace: openshift-image-registry
  annotations:
    imageregistry.openshift.io: "true"
spec:
  accessModes:
  - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 100Gi
  storageClassName: <your_custom_storage_class>
```

ここでは、以下のようになります。

openshift-image-registry

この namespace を指定すると、クラスターイメージレジストリーオペレーターは PVC を使用できます。

storage

このオプションのフィールドは、ボリュームサイズを調整します。

storageClassName

作成されるストレージクラスの名前を指定します。

4. コマンドラインから設定を適用します。

```
$ oc apply -f <pvc_file_name>
```

出力例

```
persistentvolumeclaim/csi-pvc-imageregistry created
```

5. イメージレジストリー設定の元の永続ボリューム要求は、新しい要求に置き換えます。

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --type 'json' -p='[{"op":
"replace", "path": "/spec/storage/pvc/claim", "value": "csi-pvc-imageregistry"}]'
```

出力例

```
config.imageregistry.operator.openshift.io/cluster patched
```

数分すると、設定が更新されます。

検証

レジストリーが定義したリソースを使用していることを確認するには、以下を実行します。

1. PVC クレーム値が PVC 定義で指定した名前と同じであることを確認します。

```
$ oc get configs.imageregistry.operator.openshift.io/cluster -o yaml
```

出力例

```
...
status:
    ...
    managementState: Managed
    pvc:
      claim: csi-pvc-imageregistry
...
```

2. PVC のステータスが **Bound** であることを確認します。

```
$ oc get pvc -n openshift-image-registry csi-pvc-imageregistry
```

出力例

```
NAME                    STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS            AGE
csi-pvc-imageregistry   Bound    pvc-72a8f9c9-f462-11e8-b6b6-fa163e18b7b5   100Gi
RWO            custom-csi-storageclass   11m
```

3.6. 外部ネットワークアクセスの確認

OpenShift Container Platform インストールプロセスでは、外部ネットワークへのアクセスが必要です。外部ネットワーク値をこれに指定する必要があります。指定しない場合には、デプロイメントは失敗します。このプロセスを実行する前に、外部ルータータイプのネットワークが Red Hat OpenStack Platform (RHOSP) に存在することを確認します。

前提条件

- OpenStack のネットワークサービスを、DHCP エージェントがインスタンスの DNS クエリーを転送できるように設定します。

手順

1. RHOSP CLI を使用して、'External' ネットワークの名前と ID を確認します。

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

出力例

```
+--------------------------------------+----------------+-------------+
| ID                                   | Name           | Router Type |
+--------------------------------------+----------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+----------------+-------------+
```

外部ルータータイプのあるネットワークがネットワークリストに表示されます。1 つ以上のネットワークが表示されない場合は、デフォルトの Floating IP ネットワークの作成 および デフォルトのプロバイダーネットワークの作成 を参照してください。

重要

外部ネットワークの CIDR 範囲がデフォルトのネットワーク範囲のいずれかと重複している場合、インストールプロセスを開始する前に、**install-config.yaml** ファイルで一致するネットワーク範囲を変更する必要があります。

デフォルトのネットワーク範囲は以下のとおりです。

ネットワーク	範囲
machineNetwork	10.0.0.0/16
serviceNetwork	172.30.0.0/16
clusterNetwork	10.128.0.0/14

警告

インストールプログラムにより同じ名前を持つ複数のネットワークが見つかる場合、それらのネットワークのいずれかがランダムに設定されます。この動作を回避するには、RHOSP でリソースの一意の名前を作成します。

注記

Neutron トランクサービスプラグインが有効にされると、トランクポートがデフォルトで作成されます。詳細は、Neutron trunk port を参照してください。

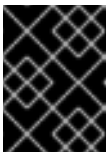
3.7. インストールプログラムのパラメーターの定義

OpenShift Container Platform インストールプログラムは、**clouds.yaml** というファイルを使用します。このファイルは、プロジェクト名、ログイン情報、認可サービスの URL を含む Red Hat OpenStack Platform (RHOSP) 設定パラメーターを説明します。

手順

1. **clouds.yaml** ファイルを作成します。

   - RHOSP ディストリビューションに Horizon Web UI が含まれる場合には、そこに **clouds.yaml** ファイルを生成します。

     重要

     パスワードを必ず **auth** フィールドに追加してください。シークレットは、**clouds.yaml** の 別のファイル に保持できます。

   - RHOSP ディストリビューションに Horizon Web UI が含まれない場合や Horizon を使用する必要がない場合には、このファイルを独自に作成します。**clouds.yaml** の詳細は、RHOSP ドキュメントの Config files を参照してください。

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: <username>
           password: <password>
           user_domain_name: Default
           project_domain_name: Default
       dev-env:
         region_name: RegionOne
         auth:
           username: <username>
           password: <password>
           project_name: 'devonly'
           auth_url: 'https://10.10.14.22:5001/v2.0'
     ```

2. RHOSP インストールでエンドポイント認証用に自己署名認証局 (CA) を使用する場合、以下を実行します。

   a. 認証局ファイルをマシンにコピーします。

   b. **cacerts** キーを **clouds.yaml** ファイルに追加します。この値は、CA 証明書への絶対的な root 以外によるアクセスが可能なパスである必要があります。

   ```
   clouds:
     shiftstack:
       ...
       cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
   ```

   ヒント

   カスタム CA 証明書を使用してインストーラーを実行した後に、**cloud-provider-config** キーマップの **ca-cert.pem** キーの値を編集して証明書を更新できます。コマンドラインで、以下を実行します。

   ```
   $ oc edit configmap -n openshift-config cloud-provider-config
   ```

3. **clouds.yaml** ファイルを以下の場所のいずれかに置きます。

   a. **OS_CLIENT_CONFIG_FILE** 環境変数の値

   b. 現行ディレクトリー

   c. Unix 固有のユーザー設定ディレクトリー (例: **~/.config/openstack/clouds.yaml**)

   d. Unix 固有のサイト設定ディレクトリー (例: **/etc/openstack/clouds.yaml**)
   インストールプログラムはこの順序で **clouds.yaml** を検索します。

3.8. OPENSTACK CLOUD CONTROLLER MANAGER のオプション設定

オプションで、クラスターの OpenStack Cloud Controller Manager (CCM) 設定を編集できます。この設定は、OpenShift Container Platform が Red Hat OpenStack Platform (RHOSP) と対話する方法を制御します。

設定パラメーターの完全なリストは、「OpenStack のインストール」ドキュメントの「OpenStack Cloud Controller Manager リファレンスガイド」を参照してください。

手順

1. クラスター用に生成されたマニフェストファイルがない場合は、以下のコマンドを実行して生成します。

   ```
   $ openshift-install --dir <destination_directory> create manifests
   ```

2. テキストエディターで、cloud-provider 設定マニフェストファイルを開きます。以下に例を示します。

   ```
   $ vi openshift/manifests/cloud-provider-config.yaml
   ```

3. CCM リファレンスガイドに従ってオプションを変更します。
   負荷分散のために Octavia を設定することが一般的です。以下に例を示します。

```
#...
[LoadBalancer]
lb-provider = "amphora" (1)
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" (2)
create-monitor = True (3)
monitor-delay = 10s (4)
monitor-timeout = 10s (5)
monitor-max-retries = 1 (6)
#...
```

(1) このプロパティーは、ロードバランサーが使用する Octavia プロバイダーを設定します。**"ovn"** または **"amphora"** を値として受け入れます。OVN の使用を選択する場合は、**lb-method** を **SOURCE_IP_PORT**。

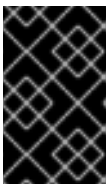
(2) このプロパティーは、複数の外部ネットワークをクラスターで使用する場合に必要です。クラウドプロバイダーは、ここで指定するネットワーク上に Floating IP アドレスを作成します。

(3) このプロパティーは、クラウドプロバイダーが Octavia ロードバランサーのヘルスモニターを作成するかどうかを制御します。ヘルスモニターを作成するには、値を **True** に設定します。RHOSP 16.2 の時点で、この機能は Amphora プロバイダーでのみ利用できます。

(4) このプロパティーは、監視されるエンドポイントの頻度を設定します。値は **time.ParseDuration()** 形式である必要があります。このプロパティーは、**create-monitor** プロパティーの値が **True** の場合に必要です。

(5) このプロパティーは、タイムアウトする前に監視要求が開く時間を設定します。値は **time.ParseDuration()** 形式である必要があります。このプロパティーは、**create-monitor** プロパティーの値が **True** の場合に必要です。

(6) このプロパティーは、ロードバランサーがオンラインとしてマークされる前に必要なモニタリング要求の数を定義します。値は整数でなければなりません。このプロパティーは、**create-monitor** プロパティーの値が **True** の場合に必要です。



重要

変更を保存する前に、ファイルが正しく構造化されていることを確認します。プロパティーが適切なセクションに置かれていないと、クラスターが失敗することがあります。

重要

.spec.externalTrafficPolicy プロパティーの値が **Local** に設定されたサービスを使用する場合は、**create-monitor** プロパティーの値を **True** に設定する必要があります。RHOSP 16.2 の OVN Octavia プロバイダーは、ヘルスモニターをサポートしません。そのため、**lb-provider** の値が **"ovn"** に設定されている場合、**ETP** パラメーターの値が **Local** に設定されたサービスは応答しない可能性があります。

4. 変更をファイルに保存し、インストールを続行します。

ヒント

インストーラーの実行後に、クラウドプロバイダー設定を更新できます。コマンドラインで、以下を実行します。

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

変更を保存した後、クラスターの再設定には多少時間がかかります。ノードが **SchedulingDisabled** のままの場合は、プロセスが完了します。

3.9. インストールプログラムの取得

OpenShift Container Platform をインストールする前に、インストールに使用しているホストにインストールファイルをダウンロードします。

前提条件

- 500 MB のローカルディスク領域がある Linux または macOS を実行するコンピューターが必要です。

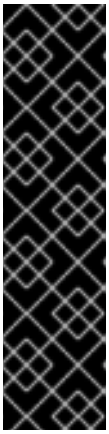
手順

1. Red Hat Hybrid Cloud Console の Cluster Type ページに移動します。Red Hat アカウントがある場合は、認証情報を使用してログインします。アカウントがない場合はこれを作成します。

ヒント

特定の OpenShift Container Platform リリースのバイナリーをダウンロード することもできます。

2. ページの **Run it yourself** セクションからインフラストラクチャープロバイダーを選択します。
3. **OpenShift Installer** のドロップダウンメニューからホストオペレーティングシステムとアーキテクチャーを選択し、**Download Installer** をクリックします。
4. ダウンロードしたファイルを、インストール設定ファイルを保存するディレクトリーに配置します。



重要

- インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターのインストール完了後は、インストールプログラムおよびインストールプログラムが作成するファイルを保持する必要があります。クラスターを削除するには、両方のファイルが必要です。
- インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。クラスターを削除するには、特定のクラウドプロバイダー用の OpenShift Container Platform のアンインストール手順を実行します。

5. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar -xvf openshift-install-linux.tar.gz
```

6. Red Hat OpenShift Cluster Manager からインストールプルシークレット をダウンロードします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する Quay.io など、組み込まれた各種の認証局によって提供されるサービスで認証できます。

ヒント

Red Hat カスタマーポータル からインストールプログラムを取得することもできます。このページでは、ダウンロードするインストールプログラムのバージョンを指定できます。ただし、このページにアクセスするには、有効なサブスクリプションが必要です。

3.10. インストール設定ファイルの作成

Red Hat OpenStack Platform (RHOSP) にインストールする OpenShift Container Platform クラスターをカスタマイズできます。

前提条件

- OpenShift Container Platform インストールプログラムおよびクラスターのプルシークレットがある。

手順

1. **install-config.yaml** ファイルを作成します。
   a. インストールプログラムが含まれるディレクトリーに切り替え、以下のコマンドを実行します。

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 **<installation_directory>** の場合、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

   ディレクトリーを指定する場合:

   - ディレクトリーに **execute** 権限があることを確認します。この権限は、インストールディレクトリーで Terraform バイナリーを実行するために必要です。
   - 空のディレクトリーを使用します。ブートストラップ X.509 証明書などの一部のインストールアセットは有効期限が短いため、インストールディレクトリーを再利用しないでください。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してください。

   b. プロンプト時に、クラウドの設定の詳細情報を指定します。
      i. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

ii. ターゲットに設定するプラットフォームとして **openstack** を選択します。

iii. クラスターのインストールに使用する Red Hat OpenStack Platform (RHOSP) の外部ネットワーク名を指定します。

iv. OpenShift API への外部アクセスに使用する floating IP アドレスを指定します。

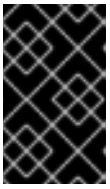
v. コントロールプレーンノードに使用する少なくとも 16 GB の RAM とコンピュートノードに使用する 8 GB の RAM を持つ RHOSP フレーバーを指定します。

vi. クラスターをデプロイするベースドメインを選択します。すべての DNS レコードはこのベースのサブドメインとなり、クラスター名も含まれます。

vii. クラスターの名前を入力します。名前は 14 文字以下でなければなりません。

2. **install-config.yaml** ファイルを変更します。利用可能なパラメーターの詳細は、「インストール設定パラメーター」のセクションを参照してください。

3. **install-config.yaml** ファイルをバックアップし、複数のクラスターをインストールするのに使用できるようにします。



重要

install-config.yaml ファイルはインストールプロセス時に使用されます。このファイルを再利用する必要がある場合は、この段階でこれをバックアップしてください。

関連情報

- OpenStack のインストール設定パラメーター

3.10.1. インストール時のクラスター全体のプロキシーの設定

実稼働環境では、インターネットへの直接アクセスを拒否し、代わりに HTTP または HTTPS プロキシーを使用することができます。プロキシー設定を **install-config.yaml** ファイルで行うことにより、新規の OpenShift Container Platform クラスターをプロキシーを使用するように設定できます。

前提条件

- 既存の **install-config.yaml** ファイルがある。
- クラスターがアクセスする必要のあるサイトを確認済みで、それらのいずれかがプロキシーをバイパスする必要があるかどうかを判別している。デフォルトで、すべてのクラスター Egress トラフィック (クラスターをホストするクラウドに関するクラウドプロバイダー API に対する呼び出しを含む) はプロキシーされます。プロキシーを必要に応じてバイパスするために、サイトを **Proxy** オブジェクトの **spec.noProxy** フィールドに追加している。

注記

Proxy オブジェクトの **status.noProxy** フィールドには、インストール設定の **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr**、および **networking.serviceNetwork[]** フィールドの値が設定されます。

Amazon Web Services (AWS)、Microsoft Azure、および Red Hat OpenStack Platform (RHOSP) へのインストールの場合、**Proxy** オブジェクトの **status.noProxy** フィールドには、インスタンスメタデータのエンドポイント (**169.254.169.254**) も設定されます。

手順

1. **install-config.yaml** ファイルを編集し、プロキシー設定を追加します。以下に例を示します。

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. クラスター外の HTTP 接続を作成するために使用するプロキシー URL。URL スキームは **http** である必要があります。
2. クラスター外で HTTPS 接続を作成するために使用するプロキシー URL。
3. プロキシーから除外するための宛先ドメイン名、IP アドレス、または他のネットワーク CIDR のコンマ区切りのリスト。サブドメインのみと一致するように、ドメインの前に **.** を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。***** を使用し、すべての宛先のプロキシーをバイパスします。
4. 指定されている場合、インストールプログラムは HTTPS 接続のプロキシーに必要な 1 つ以上の追加の CA 証明書が含まれる **user-ca-bundle** という名前の設定マップを **openshift-config** namespace に生成します。次に Cluster Network Operator は、これらのコンテンツを Red Hat Enterprise Linux CoreOS (RHCOS) 信頼バンドルにマージする **trusted-ca-bundle** config map を作成し、この config map は **Proxy** オブジェクトの **trustedCA** フィールドで参照されます。**additionalTrustBundle** フィールドは、プロキシーのアイデンティティー証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。
5. オプション: **trustedCA** フィールドの **user-ca-bundle** 設定マップを参照する **Proxy** オブジェクトの設定を決定するポリシー。許可される値は **Proxyonly** および **Always** です。**Proxyonly** を使用して、**http/https** プロキシーが設定されている場合にのみ **user-ca-bundle** 設定マップを参照します。**Always** を使用して、常に **user-ca-bundle** 設定マップを参照します。デフォルト値は **Proxyonly** です。

注記

インストールプログラムは、プロキシーの **readinessEndpoints** フィールドをサポートしません。

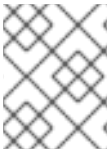
注記

インストーラーがタイムアウトした場合は、インストーラーの **wait-for** コマンドを使用してデプロイメントを再起動してからデプロイメントを完了します。以下に例を示します。

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. ファイルを保存し、OpenShift Container Platform のインストール時にこれを参照します。

インストールプログラムは、指定の **install-config.yaml** ファイルのプロキシー設定を使用する **cluster** という名前のクラスター全体のプロキシーを作成します。プロキシー設定が指定されていない場合、**cluster Proxy** オブジェクトが依然として作成されますが、これには **spec** がありません。



注記

cluster という名前の **Proxy** オブジェクトのみがサポートされ、追加のプロキシーを作成することはできません。

3.10.2. RHOSP デプロイメントでのカスタムサブネット

オプションで、選択する Red Hat OpenStack Platform (RHOSP) サブネットにクラスターをデプロイすることができます。サブネットの GUID は、**install-config.yaml** ファイルの **platform.openstack.machinesSubnet** の値として渡されます。

このサブネットはクラスターのプライマリーサブネットとして使用されます。デフォルトで、ノードおよびポートはこの上に作成されます。**platform.openstack.machinesSubnet** プロパティーの値をサブネットの UUID に設定すると、異なる RHOSP サブネットにノードおよびポートを作成することができます。

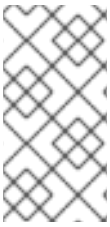
カスタムサブネットを使用して OpenShift Container Platform インストーラーを実行する前に、設定が以下の要件を満たしていることを確認してください。

- **platform.openstack.machinesSubnet** で使用されるサブネットで DHCP が有効にされている。
- **platform.openstack.machinesSubnet** の CIDR は **networking.machineNetwork** の CIDR に一致する。
- インストールプログラムのユーザーには、固定 IP アドレスを持つポートなど、このネットワークでポートを作成するパーミッションがある。

カスタムサブネットを使用するクラスターには、以下の制限があります。

- Floating IP アドレスを使用するクラスターをインストールする予定の場合には、**platform.openstack.machinesSubnet** サブネットを **externalNetwork** ネットワークに接続されているルーターに接続する必要があります。

- **platform.openstack.machinesSubnet** の値が **install-config.yaml** ファイルに設定されている場合、インストールプログラムは RHOSP マシンのプライベートネットワークまたはサブネットを作成しません。
- **platform.openstack.externalDNS** プロパティーは、カスタムサブネットと同時に使用することはできません。カスタムサブネットを使用するクラスターに DNS を追加するには、RHOSP ネットワークで DNS を設定します。



注記

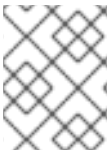
デフォルトでは、API VIP は x.x.x.5 を取得し、Ingress VIP はネットワークの CIDR ブロックから x.x.x.7 を取得します。これらのデフォルト値を上書きするには、DHCP 割り当てプール外の **platform.openstack.apiVIPs** および **platform.openstack.ingressVIPs** の値を設定します。

重要

ネットワークの CIDR 範囲は、クラスターのインストール後に調整できません。Red Hat は、namespace ごとに作成される Pod の数を慎重に検討する必要があるため、クラスターのインストール時に範囲を決定するための直接的なガイダンスを提供していません。

3.10.3. ベアメタルマシンを使用したクラスターのデプロイ

クラスターでベアメタルマシンを使用する必要がある場合は、**install-config.yaml** ファイルを変更します。クラスターには、ベアメタル上で実行されるコンピュートマシンを含めることができます。



注記

install-config.yaml ファイルで、ベアメタルワーカーに使用する RHOSP ネットワークが Floating IP アドレスをサポートするかどうかが反映されていることを確認します。

前提条件

- RHOSP Bare Metal サービス (Ironic) が有効になっており、RHOSP Compute API 経由でアクセスできる。
- ベアメタルを RHOSP フレーバー として利用できる。
- クラスターが 16.1.6 以降、16.2.4 未満の RHOSP バージョンで実行している場合は、メタデータサービスが OpenShift Container Platform ノード上のサービスで使用できなくなる 既知の問題 により、ベアメタルワーカーは機能しません。
- RHOSP ネットワークが、仮想マシンとベアメタルサーバーの両方の接続をサポートしている。
- 既存のネットワークにマシンをデプロイする場合、RHOSP サブネットがプロビジョニングされている。
- インストーラーによってプロビジョニングされるネットワークにマシンをデプロイする場合、RHOSP Bare Metal サービス (Ironic) が、テナントネットワークで実行される Preboot eXecution Environment (PXE) ブートマシンをリッスンして通信することができる。
- OpenShift Container Platform インストールプロセスの一環として、**install-config.yaml** ファイルを作成した。

手順

1. **install-config.yaml** ファイルで、マシンのフレーバーを編集します。

   a. **compute.platform.openstack.type** の値をベアメタルフレーバーに変更します。

   b. 既存のネットワークにマシンをデプロイする場合は、**platform.openstack.machinesSubnet** の値をネットワークの RHOSP サブネット UUID に変更します。

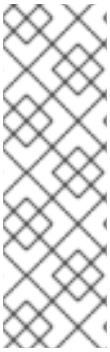
   ベアメタルの install-config.yaml のサンプルファイル

```
compute:
  - architecture: amd64
    hyperthreading: Enabled
    name: worker
    platform:
      openstack:
        type: <bare_metal_compute_flavor> 1
    replicas: 3
...

platform:
    openstack:
      machinesSubnet: <subnet_UUID> 2
...
```

1. この値を、コンピュートマシンに使用するベアメタルのフレーバーに変更します。
2. 既存のネットワークを使用する必要がある場合は、この値を RHOSP サブネットの UUID に変更します。

更新された **install-config.yaml** ファイルを使用してインストールプロセスを完了します。デプロイメント時に作成されるコンピュートマシンは、ファイルに追加したフレーバーを使用します。



注記

インストーラーは、ベアメタルマシンの起動中にタイムアウトする可能性があります。

インストーラーがタイムアウトした場合は、インストーラーの **wait-for** コマンドを使用してデプロイメントを再起動してからデプロイメントを完了します。以下に例を示します。

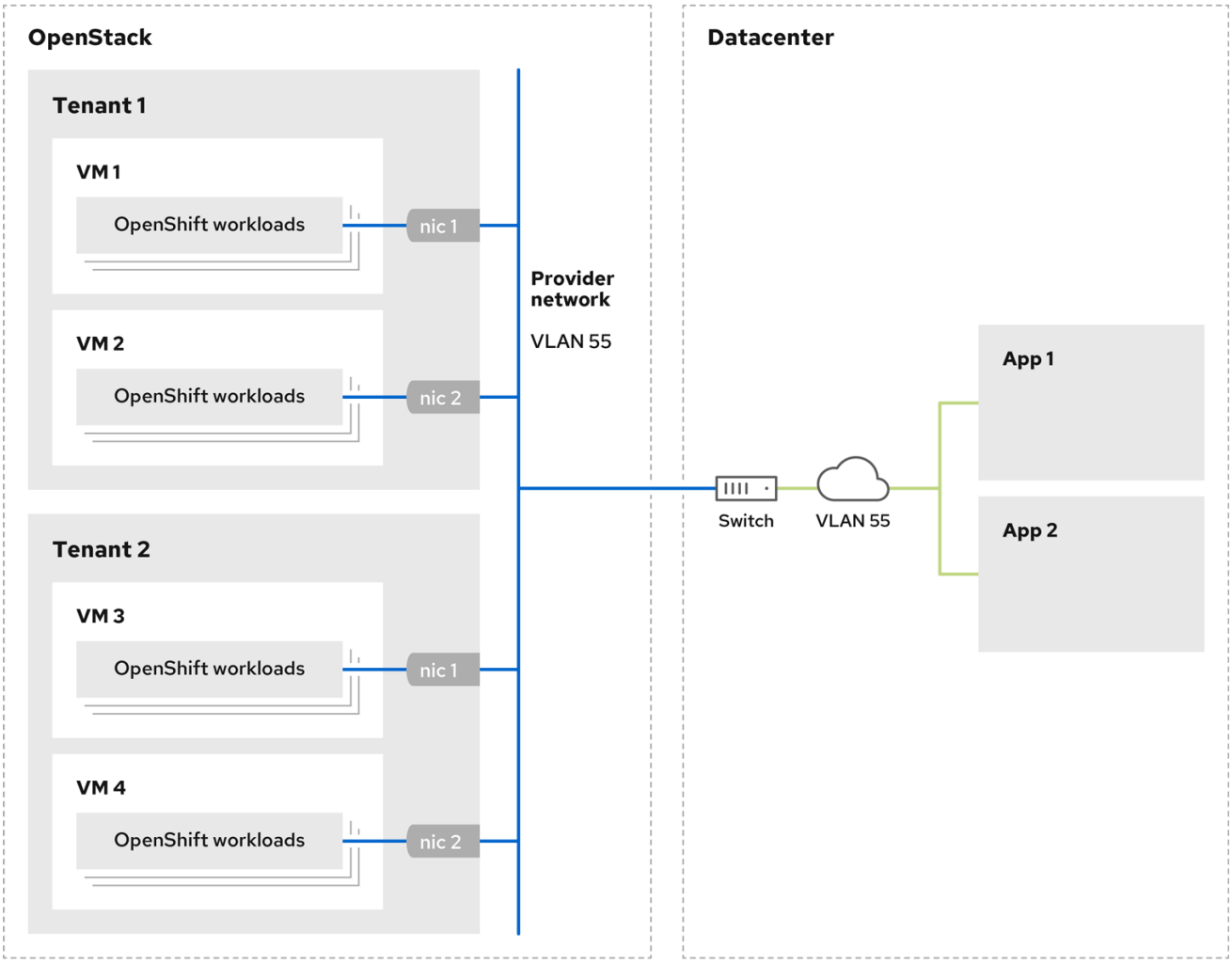
```
$ ./openshift-install wait-for install-complete --log-level debug
```

3.10.4. RHOSP プロバイダーネットワーク上のクラスターデプロイメント

プロバイダーネットワーク上のプライマリーネットワークインターフェイスを使用して、OpenShift Container Platform クラスターを Red Hat OpenStack Platform (RHOSP) にデプロイできます。プロバイダーネットワークは一般的に、インターネットへの到達に使用可能なパブリックネットワークに、プロジェクトが直接アクセスできるように使用します。ネットワーク作成プロセスの一環として、プロバイダーネットワークをプロジェクト間で共有することもできます。

RHOSP プロバイダーネットワークは、データセンター内の既存の物理ネットワークに直接マップします。RHOSP 管理者はこれらを作成する必要があります。

以下の例では、OpenShift Container Platform ワークロードはプロバイダーネットワークを使用してデータセンターに接続されます。



プロバイダーネットワークにインストールされている OpenShift Container Platform クラスターは、テナントネットワークまたは Floating IP アドレスを必要としません。インストーラーは、インストール中にこれらのリソースを作成しません。

プロバイダーネットワークタイプの例には、フラット (タグなし) および VLAN (802.1Q タグ付き) が含まれます。

注記

クラスターは、ネットワークタイプが許可する限り多くのプロバイダーネットワーク接続をサポートできます。たとえば、VLAN ネットワークは、通常最大 4096 の接続をサポートします。

プロバイダーネットワークおよびテナントネットワークの詳細は、RHOSP のドキュメント を参照してください。

3.10.4.1. クラスターのインストールにおける RHOSP プロバイダーネットワーク要件

OpenShift Container Platform クラスターをインストールする前に、Red Hat OpenStack Platform (RHOSP) のデプロイメントおよびプロバイダーネットワークは、さまざまな条件を満たす必要があります。

- RHOSP ネットワークサービス (Neutron) が有効化され、RHOSP ネットワーク API 経由でアクセス可能であること。
- RHOSP ネットワークサービスでは、ポートセキュリティーと許可するアドレスペアの機能拡張が有効化 されていること。
- プロバイダーネットワークは他のテナントと共有できます。

  ヒント

  --share フラグを指定して **openstack network create** コマンドを使用して、共有できるネットワークを作成します。

- クラスターのインストールに使用する RHOSP プロジェクトは、プロバイダーネットワークと適切なサブネットを所有する必要があります。

  ヒント

  "openshift" という名前のプロジェクトのネットワークを作成するには、以下のコマンドを入力します。

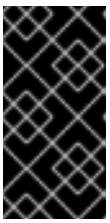
  ```
  $ openstack network create --project openshift
  ```

  "openshift" という名前のプロジェクトのサブネットを作成するには、以下のコマンドを入力します。

  ```
  $ openstack subnet create --project openshift
  ```

  RHOSP でのネットワークの作成に関する詳細は、プロバイダーネットワークに関するドキュメント を参照してください。

  クラスターが **admin** ユーザーによって所有されている場合、そのユーザーとしてインストーラーを実行してネットワーク上でポートを作成する必要があります。

  重要

  プロバイダーネットワークは、クラスターの作成に使用される RHOSP プロジェクトによって所有されている必要があります。所有されていない場合は、RHOSP Compute サービス (Nova) はそのネットワークからポートを要求できません。

- プロバイダーネットワークが、デフォルトで **169.254.169.254** である RHOSP メタデータサービスの IP アドレスに到達できることを確認します。
  RHOSP SDN とネットワークサービス設定によっては、サブネットを作成する際に、ルートを提供しなければならない場合があります。以下に例を示します。

  ```
  $ openstack subnet create --dhcp --host-route
  destination=169.254.169.254/32,gateway=192.0.2.2 ...
  ```

- オプション: ネットワークのセキュリティーを保護するには、単一のプロジェクトへのネットワークアクセスを制限する ロールベースのアクセス制御 (RBAC) ルールを作成します。

3.10.4.2. プロバイダーネットワークにプライマリーインターフェイスを持つクラスターのデプロイ

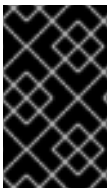
Red Hat OpenStack Platform (RHOSP) プロバイダーネットワーク上にプライマリーネットワークインターフェイスを持つ OpenShift Container Platform クラスターをデプロイすることができます。

前提条件

- 「クラスターのインストールにおける RHOSP プロバイダーネットワーク要件」に記載されているとおりに、お使いの Red Hat OpenStack Platform (RHOSP) のデプロイメントが設定されています。

手順

1. テキストエディターで **install-config.yaml** ファイルを開きます。
2. **platform.openstack.apiVIPs** プロパティーの値を API VIP の IP アドレスに設定します。
3. **platform.openstack.ingressVIPs** プロパティーの値を Ingress VIP の IP アドレスに設定します。
4. **platform.openstack.machinesSubnet** プロパティーの値をプロバイダーネットワークサブネットの UUID に設定します。
5. **networking.machineNetwork.cidr** プロパティーの値をプロバイダーネットワークサブネットの CIDR ブロックに設定します。



重要

platform.openstack.apiVIPs プロパティーおよび **platform.openstack.ingressVIPs** プロパティーはいずれも、**networking.machineNetwork.cidr** ブロックから割り当てられていない IP アドレスである必要があります。

RHOSP プロバイダーネットワークに依存するクラスターのインストール設定ファイルのセクション

```
...
platform:
  openstack:
    apiVIPs: 1
      - 192.0.2.13
    ingressVIPs: 2
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
networking:
  machineNetwork:
  - cidr: 192.0.2.0/24
```

1 2 OpenShift Container Platform 4.12 以降では、**apiVIP** および **ingressVIP** 設定は非推奨です。代わりに、リスト形式を使用して、**apiVIPs** および **ingressVIPs** 設定に値を入力します。

警告

プライマリーネットワークインターフェイスにプロバイダーネットワークを使用している間は、**platform.openstack.externalNetwork** パラメーターまたは **platform.openstack.externalDNS** パラメーターを設定することはできません。

クラスターをデプロイする際に、インストーラーは **install-config.yaml** ファイルを使用してプロバイダーネットワークにクラスターをデプロイします。

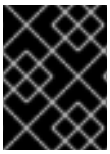
ヒント

プロバイダーネットワークを含むネットワークを **platform.openstack.additionalNetworkIDs** リストに追加できます。

クラスターのデプロイ後に、Pod を追加のネットワークに接続することができます。詳細は、複数ネットワークについて を参照してください。

3.10.5. RHOSP のカスタマイズされた install-config.yaml ファイルのサンプル

次の **install-config.yaml** ファイルの例は、すべての可能な Red Hat OpenStack Platform (RHOSP) カスタマイズオプションを示しています。



重要

このサンプルファイルは参照用にのみ提供されます。インストールプログラムを使用して **install-config.yaml** ファイルを取得する必要があります。

例3.2 シングルスタックの install-config.yaml ファイルの例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
```

```
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

例3.3 デュアルスタックの install-config.yaml ファイルの例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: 192.168.25.0/24
  - cidr: fd2e:6f44:5dd8:c956::/64
  serviceNetwork:
  - 172.30.0.0/16
  - fd02::/112
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiVIPs:
    - 192.168.25.10
    - fd2e:6f44:5dd8:c956:f816:3eff:fec3:5955
```

```
    ingressVIPs:
    - 192.168.25.132
    - fd2e:6f44:5dd8:c956:f816:3eff:fe40:aecb
    controlPlanePort:
      fixedIPs:
      - subnet:
          name: openshift-dual4
      - subnet:
          name: openshift-dual6
      network:
        name: openshift-dual
  fips: false
  pullSecret: '{"auths": ...}'
  sshKey: ssh-ed25519 AAAA...
```

3.10.6. デュアルスタックネットワークを使用したクラスターの設定

RHOSP 上にデュアルスタッククラスターを作成できます。ただし、デュアルスタック設定は、IPv4 および IPv6 サブネットを持つ RHOSP ネットワークを使用している場合にのみ有効になります。

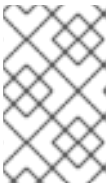
注記

RHOSP は、IPv4 シングルスタッククラスターからデュアルスタッククラスターネットワークへの変換をサポートしていません。

3.10.6.1. デュアルスタッククラスターの導入

手順

1. IPv4 および IPv6 サブネットを含むネットワークを作成します。**ipv6-ra-mode** および **ipv6-address-mode** フィールドで使用可能なアドレスモードは、**dhcpv6-stateful**、**dhcpv6-stateless**、および **slaac** です。



注記

デュアルスタックネットワーク MTU は、IPv6 の最小 MTU (1280) と OVN-Kubernetes カプセル化オーバーヘッド (100) の両方に対応する必要があります。

注記

DHCP はサブネット上で有効にする必要があります。

2. API ポートと Ingress VIP ポートを作成します。
3. IPv6 サブネットをルーターに追加して、ルーターのアドバタイズメントを有効にします。プロバイダーネットワークを使用している場合は、ネットワークを外部ゲートウェイとして追加することでルーターのアドバタイズメントを有効にすることができ、これにより外部接続も有効になります。
4. クラスターノードの IPv4 および IPv6 アドレスエンドポイントを設定するには、**install-config.yaml** ファイルを編集します。以下は、**install-config.yaml** ファイルの例です。

install-config.yaml の例

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  machineNetwork: 1
  - cidr: "192.168.25.0/24"
  - cidr: "fd2e:6f44:5dd8:c956::/64"
  clusterNetwork: 2
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  serviceNetwork: 3
  - 172.30.0.0/16
  - fd02::/112
platform:
  openstack:
    ingressVIPs: ['192.168.25.79', 'fd2e:6f44:5dd8:c956:f816:3eff:fef1:1bad'] 4
    apiVIPs: ['192.168.25.199', 'fd2e:6f44:5dd8:c956:f816:3eff:fe78:cf36'] 5
    controlPlanePort: 6
      fixedIPs: 7
      - subnet: 8
          name: subnet-v4
          id: subnet-v4-id
      - subnet: 9
          name: subnet-v6
          id: subnet-v6-id
      network: 10
        name: dualstack
        id: network-id
```

1 2 3 IPv4 と IPv6 の両方のアドレスファミリーの IP アドレス範囲を指定する必要があります。

4 クラスターにインターフェイスを提供するための Ingress VIP サービスの仮想 IP (VIP) アドレスエンドポイントを指定します。

5 API VIP サービスの仮想 IP (VIP) アドレスエンドポイントを指定して、クラスターにインターフェイスを提供します。

6 クラスター内のすべてのノードで使用されるデュアルスタックネットワークの詳細を指定します。

7 このフィールドで指定するサブネットの CIDR は、**networks.machineNetwork** にリストされている CIDR と一致する必要があります。

8 9 **name** または **id** のいずれか、または両方の値を指定できます。

10 **ControlPlanePort** フィールドでの **network** の指定は任意です。

あるいは、IPv6 プライマリーデュアルスタッククラスターが必要な場合は、以下の例に従って **install-config.yaml** ファイルを編集します。

install-config.yaml の例

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  machineNetwork: 1
  - cidr: "fd2e:6f44:5dd8:c956::/64"
  - cidr: "192.168.25.0/24"
  clusterNetwork: 2
  - cidr: fd01::/48
    hostPrefix: 64
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork: 3
  - fd02::/112
  - 172.30.0.0/16
platform:
  openstack:
    ingressVIPs: ['fd2e:6f44:5dd8:c956:f816:3eff:fef1:1bad', '192.168.25.79'] 4
    apiVIPs: ['fd2e:6f44:5dd8:c956:f816:3eff:fe78:cf36', '192.168.25.199'] 5
    controlPlanePort: 6
      fixedIPs: 7
      - subnet: 8
          name: subnet-v6
          id: subnet-v6-id
      - subnet: 9
          name: subnet-v4
```

```
    id: subnet-v4-id
  network: 10
    name: dualstack
    id: network-id
```

1 2 3 IPv4 と IPv6 の両方のアドレスファミリーの IP アドレス範囲を指定する必要があります。

4 クラスターにインターフェイスを提供するための Ingress VIP サービスの仮想 IP (VIP) アドレスエンドポイントを指定します。

5 API VIP サービスの仮想 IP (VIP) アドレスエンドポイントを指定して、クラスターにインターフェイスを提供します。

6 クラスター内のすべてのノードで使用されるデュアルスタックネットワークの詳細を指定します。

7 このフィールドで指定するサブネットの CIDR は、**networks.machineNetwork** にリストされている CIDR と一致する必要があります。

8 9 **name** または **id** のいずれか、または両方の値を指定できます。

10 **ControlPlanePort** フィールドでの **network** の指定は任意です。

注記

隔離されたデュアルスタックネットワークでインストールホストを使用する場合、再起動時に IPv6 アドレスが正しく再割り当てされない場合があります。

Red Hat Enterprise Linux (RHEL) 8 でこの問題を解決するには、次の設定を含む **/etc/NetworkManager/system-connections/required-rhel8-ipv6.conf** という名前のファイルを作成します。

```
[connection]
type=ethernet
[ipv6]
addr-gen-mode=eui64
method=auto
```

RHEL 9 でこの問題を解決するには、次の設定を含む **/etc/NetworkManager/conf.d/required-rhel9-ipv6.conf** という名前のファイルを作成します。

```
[connection]
ipv6.addr-gen-mode=0
```

ファイルを作成して編集したら、インストールホストを再起動します。

注記

すべてのノードに設定される **ip=dhcp,dhcp6** カーネル引数により、複数のインターフェイスで同時にアクティブ化される単一の Network Manager 接続プロファイルが生成されます。このような動作のため、追加のネットワークには、同一の UUID を持つ同じ接続が適用されます。インターフェイス固有の設定が必要な場合は、そのインターフェイス用の新しい接続プロファイルを作成して、デフォルトの接続が適用されないようにしてください。

3.10.7. シングルスタック IPv6 ネットワークを使用したクラスターの設定

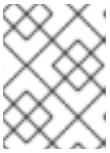
RHOSP デプロイメントを設定した後、Red Hat OpenStack Platform (RHOSP) 上にシングルスタック IPv6 クラスターを作成できます。

重要

デュアルスタッククラスターをシングルスタック IPv6 クラスターに変換することはできません。

前提条件

- RHOSP デプロイメントに、マシンネットワークとして使用する DHCPv6 ステートフル IPv6 サブネットを持つ既存のネットワークがある。
- 既存の IPv6 サブネットに対して DNS が設定されている。
- IPv6 サブネットが RHOSP ルーターに追加され、ルーターがルーター広告 (RA) を送信するように設定されている。
- クラスターで使用される追加の IPv6 サブネットを RHOSP ルーターに追加し、ルーター広告を有効にした。



注記

IPv6 SLAAC サブネットの使用はサポートされていません。RHOSP Neutron によって **dns_nameservers** アドレスが適用されないためです。

- IPv6 インターフェイスを備えたミラーレジストリーがある。
- RHOSP ネットワークが最小 MTU サイズ (1442 バイト) を受け入れる。
- マシンネットワーク上の RHOSP ポートとして API および Ingress 仮想 IP アドレス (VIP) を作成し、それらのアドレスを **install-config.yaml** ファイルに含めた。

手順

1. 次のコマンドを実行して、ネットワーク上に API 仮想 IP ポートを作成します。

```
$ openstack port create api --network <v6_machine_network>
```

2. 次のコマンドを実行して、ネットワーク上に Ingress 仮想 IP ポートを作成します。

```
$ openstack port create ingress --network <v6_machine_network>
```

3. ネットワークリソースの事前作成が完了したら、IPv6 ネットワーク設定を反映した **install-config.yaml** ファイルを使用してクラスターをデプロイします。たとえば、以下のようになります。

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  machineNetwork:
  - cidr: "fd2e:6f44:5dd8:c956::/64" (1)
  clusterNetwork:
  - cidr: fd01::/48
    hostPrefix: 64
  serviceNetwork:
  - fd02::/112
platform:
  openstack:
    ingressVIPs: ['fd2e:6f44:5dd8:c956::383'] (2)
    apiVIPs: ['fd2e:6f44:5dd8:c956::9a'] (3)
    controlPlanePort:
      fixedIPs: (4)
      - subnet:
          name: subnet-v6
      network: (5)
        name: v6-network
imageContentSources: (6)
- mirrors:
  - <mirror>
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
- mirrors:
  - <mirror>
  source: registry.ci.openshift.org/ocp/release
additionalTrustBundle: |
<certificate_of_the_mirror>
```

(1) このフィールドで指定したサブネットの CIDR が、**controlPlanePort** セクションで指定したサブネットの CIDR と一致する必要があります。

(2) (3) 前のステップで生成したポートのアドレスを、パラメーター **platform.openstack.ingressVIPs** および **platform.openstack.apiVIPs** の値として使用します。

4 5 **platform.openstack.controlPlanePort.fixedIPs** キーと **platform.openstack.controlPlanePort.network** キーの下の項目には、ID、名前、または

6 **imageContentSources** セクションにはミラーの詳細を含めます。ローカルイメージレジストリーの設定の詳細は、「mirror registry for Red Hat OpenShift を使用したミラーレジストリーの作成」を参照してください。

関連情報

- mirror registry for Red Hat OpenShift を使用したミラーレジストリーの作成 を参照してください。

3.10.8. ユーザー管理のロードバランサーを使用した OpenStack 上のクラスターのインストール設定

次の **install-config.yaml** ファイルの例は、デフォルトの内部ロードバランサーではなく、ユーザー管理の外部ロードバランサーを使用するクラスターを設定する方法を示しています。

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.10.0/24
platform:
  openstack:
    cloud: mycloud
    machinesSubnet: 8586bf1a-cc3c-4d40-bdf6-c243decc603a 1
    apiVIPs:
    - 192.168.10.5
    ingressVIPs:
    - 192.168.10.7
    loadBalancer:
      type: UserManaged 2
```

1 どのロードバランサーを使用するかに関係なく、ロードバランサーはこのサブネットにデプロイされます。

2 **UserManaged** 値は、ユーザー管理のロードバランサーを使用していることを示します。

3.11. クラスターノードの SSH アクセス用のキーペアの生成

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストールプログラムに指定できます。キーは、Ignition 設定ファイルを介して Red Hat Enterprise Linux CoreOS (RHCOS) ノードに渡され、ノードへの SSH アクセスを認証するために使用されます。このキーは各ノードの **core** ユーザーの ~/**.ssh/authorized_keys** リストに追加され、パスワードなしの認証が可能になります。

キーがノードに渡されると、キーペアを使用して RHCOS ノードにユーザー **core** として SSH を実行できます。SSH 経由でノードにアクセスするには、秘密鍵のアイデンティティーをローカルユーザーの SSH で管理する必要があります。

インストールのデバッグまたは障害復旧を実行するためにクラスターノードに対して SSH を実行する場合は、インストールプロセスの間に SSH 公開鍵を指定する必要があります。**./openshift-install gather** コマンドでは、SSH 公開鍵がクラスターノードに配置されている必要もあります。

重要

障害復旧およびデバッグが必要な実稼働環境では、この手順を省略しないでください。

手順

1. クラスターノードへの認証に使用するローカルマシンに既存の SSH キーペアがない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 新しい SSH キーのパスとファイル名 (~/**.ssh/id_ed25519** など) を指定します。既存のキーペアがある場合は、公開鍵が ~/**.ssh** ディレクトリーにあることを確認します。

注記

x86_64、**ppc64le**、および **s390x** アーキテクチャーのみで FIPS 140-2/140-3 検証のために NIST に提出された RHEL 暗号化ライブラリーを使用する OpenShift Container Platform クラスターをインストールする予定がある場合は、**ed25519** アルゴリズムを使用するキーを作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

2. 公開 SSH キーを表示します。

```
$ cat <path>/<file_name>.pub
```

たとえば、次のコマンドを実行して ~/**.ssh/id_ed25519.pub** 公開鍵を表示します。

```
$ cat ~/.ssh/id_ed25519.pub
```

3. ローカルユーザーの SSH エージェントに SSH 秘密鍵 ID が追加されていない場合は、それを追加します。キーの SSH エージェント管理は、クラスターノードへのパスワードなしの SSH 認証、または **./openshift-install gather** コマンドを使用する場合は必要になります。

注記

一部のディストリビューションでは、~/**.ssh/id_rsa** および ~/**.ssh/id_dsa** などのデフォルトの SSH 秘密鍵のアイデンティティーは自動的に管理されます。

a. **ssh-agent** プロセスがローカルユーザーに対して実行されていない場合は、バックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"
```

出力例

```
Agent pid 31874
```

注記

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

4. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> 1
```

1 ~/**.ssh/id_ed25519** などの、SSH プライベートキーのパスおよびファイル名を指定します。

出力例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

次のステップ

- OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストールプログラムに指定します。

3.12. 環境へのアクセスの有効化

デプロイ時に、OpenShift Container Platform マシンはすべて Red Hat OpenStack Platform (RHOSP) テナントネットワークに作成されます。したがって、ほとんどの RHOSP デプロイメントでは直接アクセスできません。

インストール時に Floating IP アドレス (FIP) を使用して OpenShift Container Platform API およびアプリケーションのアクセスを設定できます。FIP を設定せずにインストールを完了することもできますが、インストーラーは API またはアプリケーションを外部からアクセスする方法を設定しません。

3.12.1. floating IP アドレスを使用したアクセスの有効化

OpenShift Container Platform API およびクラスターアプリケーションへの外部アクセス用に Floating IP (FIP) アドレスを作成します。

手順

1. Red Hat OpenStack Platform (RHOSP) CLI を使用して、API FIP を作成します。

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Red Hat OpenStack Platform (RHOSP) CLI を使用して、apps (アプリ)、または Ingress、FIP を作成します。

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. API および Ingress FIP の DNS サーバーに、これらのパターンに準拠するレコードを追加します。

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注記

DNS サーバーを制御していない場合は、次のようなクラスタードメイン名を **/etc/hosts** ファイルに追加することで、クラスターにアクセスできます。

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts ファイル内のクラスタードメイン名により、クラスターの Web コンソールおよび監視インターフェイスへのローカルアクセスが許可されます。**kubectl** または **oc** を使用することもできます。<application_floating_ip> を指す追加のエントリーを使用して、ユーザーアプリケーションにアクセスできます。このアクションにより、API およびアプリケーションは他のユーザーがアクセスできない状態になり、この状態は実稼働デプロイメントには適していませんが、開発およびテスト目的のインストールが可能になります。

4. FIP を、以下のパラメーターの値として **install-config.yaml** ファイルに追加します。

    - **platform.openstack.ingressFloatingIP**
    - **platform.openstack.apiFloatingIP**

これらの値を使用する場合には、**install-config.yaml** ファイルの **platform.openstack.externalNetwork** パラメーターの値として外部ネットワークを入力する必要もあります。

ヒント

Floating IP アドレスを割り当て、ファイアウォール設定を更新することで、OpenShift Container Platform リソースがクラスター外で利用できる状態にすることができます。

3.12.2. Floating IP アドレスなしでのインストールの完了

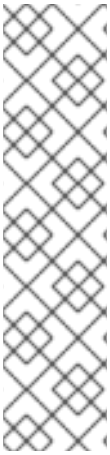
Floating IP アドレスを指定せずに OpenShift Container Platform を Red Hat OpenStack Platform (RHOSP) にインストールすることができます。

install-config.yaml ファイルで以下のパラメーターを定義しないでください。

- **platform.openstack.ingressFloatingIP**
- **platform.openstack.apiFloatingIP**

外部ネットワークを提供できない場合は、**platform.openstack.externalNetwork** を空白のままにすることもできます。**platform.openstack.externalNetwork** の値を指定しない場合はルーターが作成されず、追加のアクションがない場合は、インストーラーは Glance からのイメージの取得に失敗します。外部接続を独自に設定する必要があります。

Floating IP アドレスまたは名前解決がないために、クラスター API に到達できないシステムからインストーラーを実行すると、インストールに失敗します。このような場合にインストールが失敗するのを防ぐために、プロキシーネットワークを使用するか、マシンと同じネットワークにあるシステムからインストーラーを実行できます。



注記

API および Ingress ポートの DNS レコードを作成して、名前解決を有効にできます。以下に例を示します。

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

DNS サーバーを制御しない場合は、**/etc/hosts** ファイルにレコードを追加できます。このアクションにより、API は他者のアクセスできない状態になり、この状態は実稼働デプロイメントには適していませんが、開発およびテスト目的のインストールが可能になります。

3.13. クラスターのデプロイ

互換性のあるクラウドプラットフォームに OpenShift Container Platform をインストールできます。

重要

インストールプログラムの **create cluster** コマンドは、初期インストール時に 1 回だけ実行できます。

前提条件

- OpenShift Container Platform インストールプログラムおよびクラスターのプルシークレットがある。
- ホスト上のクラウドプロバイダーアカウントに、クラスターをデプロイするための適切な権限があることを確認している。アカウントの権限が正しくないと、インストールプロセスが失敗し、不足している権限を示すエラーメッセージが表示されます。

手順

- インストールプログラムが格納されているディレクトリーで、次のコマンドを実行して、クラスターのデプロイメントを初期化します。

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 **<installation_directory>** に、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。

2 異なるインストールの詳細情報を表示するには、**info** ではなく、**warn**、**debug**、または **error** を指定します。

検証

クラスターのデプロイが正常に完了すると、次のようになります。

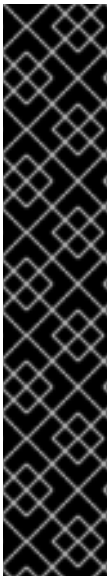
- ターミナルには、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報など、クラスターにアクセスするための指示が表示されます。
- 認証情報は **<installation_directory>/.openshift_install.log** にも出力されます。

重要

インストールプログラム、またはインストールプログラムが作成するファイルを削除することはできません。これらはいずれもクラスターを削除するために必要になります。

出力例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになり、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、**コントロールプレーン証明書の期限切れの状態からのリカバリー** に関するドキュメントを参照してください。
- 24 時間証明書はクラスターのインストール後 16 時間から 22 時間にローテーションするため、Ignition 設定ファイルは、生成後 12 時間以内に使用することを推奨します。12 時間以内に Ignition 設定ファイルを使用することにより、インストール中に証明書の更新が実行された場合のインストールの失敗を回避できます。

3.14. クラスターステータスの確認

インストール時またはインストール後に OpenShift Container Platform クラスターのステータスを確認することができます。

手順

1. クラスター環境で、管理者の kubeconfig ファイルをエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。

kubeconfig ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターに関する情報が含まれます。

2. デプロイメント後に作成されたコントロールプレーンおよびコンピュートマシンを表示します。

```
$ oc get nodes
```

3. クラスターのバージョンを表示します。

```
$ oc get clusterversion
```

4. Operator のステータスを表示します。

```
$ oc get clusteroperator
```

5. クラスター内のすべての実行中の Pod を表示します。

```
$ oc get pods -A
```

3.15. CLI の使用によるクラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターに関する情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターをデプロイしていること。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. 次のコマンドを実行して、**kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。

2. 次のコマンドを実行し、エクスポートされた設定を使用して **oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

関連情報

- OpenShift Container Platform Web コンソールへのアクセスと理解に関する詳細は、Web コンソールへのアクセス を参照してください。

3.16. OPENSHIFT CONTAINER PLATFORM の TELEMETRY アクセス

OpenShift Container Platform 4.19 では、Telemetry サービスにもインターネットアクセスが必要です。Telemetry サービスは、クラスターの健全性と更新の成功に関するメトリクスを提供するためにデフォルトで実行されます。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは OpenShift Cluster Manager に登録されます。

OpenShift Cluster Manager インベントリーが正常である (Telemetry によって自動的に維持、または OpenShift Cluster Manager を使用して手動で維持) ことを確認した後に、subscription watch を使用 して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

関連情報

- Telemetry サービスの詳細は、リモートヘルスモニタリングについて を参照してください。

3.17. 次のステップ

- クラスターのカスタマイズ
- 必要に応じて、リモートヘルスレポート を作成できます。
- ノードポートへの外部アクセスを有効にする必要がある場合は、ノードポートを使用して Ingress クラストラフィックを設定 します。
- Floating IP アドレス上でアプリケーショントラフィックを受け入れるように RHOSP を設定していない場合は、Floating IP アドレスを使用して RHOSP アクセスを設定 します。

第4章 独自のインフラストラクチャーを使用した OPENSTACK へのクラスターのインストール

OpenShift Container Platform バージョン 4.19 では、user-provisioned infrastructure 上で実行される Red Hat OpenStack Platform (RHOSP) にクラスターをインストールできます。

独自のインフラストラクチャーを使用することで、クラスターを既存のインフラストラクチャーおよび変更と統合できます。このプロセスでは、installer-provisioned installation の場合よりも多くの手作業が必要になります。Nova サーバー、Neutron ポート、セキュリティーグループなどのすべての RHOSP リソースを作成する必要があるためです。ただし、Red Hat では、デプロイメントプロセスを支援する Ansible Playbook を提供しています。

4.1. 前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスの詳細を確認している。
- クラスターインストール方法の選択およびそのユーザー向けの準備 を確認している。
- OpenShift クラスターでサポートされるプラットフォーム セクションを使用して、OpenShift Container Platform 4.19 が RHOSP バージョンと互換性があることを確認した。RHOSP サポートマトリックスの OpenShift Container Platform を参照して、プラットフォームのサポートを異なるバージョン間で比較することもできます。
- OpenShift Container Platform のインストール先に RHOSP アカウントがある。
- クラスターのスケーリング、コントロールプレーンのサイジング、および etcd のパフォーマンスおよびスケーラビリティーに関する理解がある。詳細は、クラスターのスケーリングに関する推奨プラクティス を参照してください。
- インストールプログラムを実行するマシンには、以下が含まれる。
  - インストールプロセス時に作成したファイルを保持できる単一ディレクトリー
  - Python 3

4.2. OPENSHIFT CONTAINER PLATFORM のインターネットアクセス

OpenShift Container Platform 4.19 では、クラスターをインストールするためにインターネットアクセスが必要です。

次のアクションを実行するには、インターネットにアクセスできる必要があります。

- OpenShift Cluster Manager にアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターがインターネットにアクセスでき、Telemetry を無効にしていない場合、そのサービスによってクラスターのサブスクリプションが自動的に有効化されます。
- クラスターのインストールに必要なパッケージを取得するために Quay.io にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。

重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにインストールパッケージを設定します。インストールタイプに応じて、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

4.3. OPENSHIFT CONTAINER PLATFORM を RHOSP にインストールするリソースのガイドライン

OpenShift Container Platform のインストールをサポートするために、Red Hat OpenStack Platform (RHOSP) クォータは以下の要件を満たす必要があります。

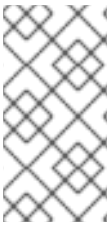
表4.1 RHOSP のデフォルトの OpenShift Container Platform クラスターに関する推奨リソース

リソース	値
Floating IP アドレス	3
ポート	15
ルーター	1
サブネット	1
RAM	88 GB
vCPU	22
ボリュームストレージ	275 GB
インスタンス	7
セキュリティーグループ	3
セキュリティーグループルール	60
サーバーグループ	2 - 各マシンプールの追加のアベイラビリティーゾーンごとに 1 つ追加

クラスターは推奨されるリソースよりもリソースが少ない場合にも機能する場合がありますが、その場合のパフォーマンスは保証されません。

重要

RHOSP オブジェクトストレージ (Swift) が利用可能で、**swiftoperator** ロールを持つユーザーアカウントによって操作されている場合、これは OpenShift Container Platform イメージレジストリーのデフォルトバックエンドとして使用されます。この場合、ボリュームストレージ要件は 175 GB です。Swift 領域要件は、イメージレジストリーのサイズによって異なります。



注記

デフォルトで、セキュリティーグループおよびセキュリティーグループルールのクォータは低く設定される可能性があります。問題が生じた場合には、管理者として **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** を実行して値を増やします。

OpenShift Container Platform デプロイメントは、コントロールプレーンマシン、コンピュートマシン、およびブートストラップマシンで構成されます。

4.3.1. コントロールプレーンマシン

デフォルトでは、OpenShift Container Platform インストールプロセスは 3 つのコントロールプレーンマシンを作成します。

それぞれのマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 16 GB のメモリーと 4 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

4.3.2. コンピュートマシン

デフォルトでは、OpenShift Container Platform インストールプロセスは 3 つのコンピューティングマシンを作成します。

それぞれのマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 8 GB のメモリーと 2 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

ヒント

コンピュートマシンは、OpenShift Container Platform で実行されるアプリケーションをホストします。できるだけ多くのアプリケーションを実行することが意図されています。

4.3.3. ブートストラップマシン

インストール時に、ブートストラップマシンは一時的にプロビジョニングされ、コントロールプレーンを初期化します。実稼働環境用のコントロールプレーンの準備ができた後に、ブートストラップマシンのプロビジョニングは解除されます。

ブートストラップマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 16 GB のメモリーと 4 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

4.4. PLAYBOOK 依存関係のダウンロード

user-provisioned infrastructure でのインストールプロセスを簡素化する Ansible Playbook には、いくつかの Ansible コレクションと Python モジュールが必要です。インストールプログラムを実行するマシンに Red Hat OpenStack Platform (RHOSP) リポジトリーを追加し、パッケージをインストールします。

次の依存関係が必要です。

- Python モジュール:
  - **openstackclient**
  - **openstacksdk**
  - **netaddr**
  - **pip**
- Ansible コレクション:
  - **ansible-collections-openstack** は Ansible Core をインストールします
  - **ansible-collection-community-general**
  - **ansible-collection-ansible-netcommon**

注記

この手順では、Red Hat Enterprise Linux (RHEL) 8 を使用していることを前提としています。

前提条件

- Python 3 がマシンにインストールされている。

手順

1. コマンドラインで、リポジトリーを追加します。
   a. Red Hat Subscription Manager に登録します。

```
$ sudo subscription-manager register # If not done already
```

b. 最新のサブスクリプションデータをプルします。

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

c. 現在のリポジトリーを無効にします。

```
$ sudo subscription-manager repos --disable=* # If not done already
```

d. 必要なリポジトリーを追加します。

```
$ sudo subscription-manager repos \
  --enable=rhel-9-for-x86_64-appstream-rpms \
  --enable=rhel-9-for-x86_64-baseos-rpms \
  --enable=openstack-17.1-for-rhel-9-x86_64-rpms
```

2. モジュールをインストールします。

```
$ sudo dnf install ansible-collection-ansible-netcommon \
    ansible-collection-community-general \
    ansible-collections-openstack \
    python3-netaddr \
    python3-openstackclient \
    python3-openstacksdk \
    python3-pip
```

3. **python** コマンドが **python3** を参照していることを確認します。

```
$ sudo alternatives --set python /usr/bin/python3
```

4.5. インストール PLAYBOOK のダウンロード

OpenShift Container Platform を独自の Red Hat OpenStack Platform (RHOSP) インフラストラクチャーにインストールするために使用できる Ansible Playbook をダウンロードします。

前提条件

- curl コマンドラインツールがマシンで利用できる。

手順

- Playbook を作業ディレクトリーにダウンロードするには、コマンドラインから以下のスクリプトを実行します。

```
$ xargs -n 1 curl -O <<< '
      https://raw.githubusercontent.com/openshift/installer/release-
4.19/upi/openstack/bootstrap.yaml
      https://raw.githubusercontent.com/openshift/installer/release-
4.19/upi/openstack/common.yaml
      https://raw.githubusercontent.com/openshift/installer/release-
4.19/upi/openstack/compute-nodes.yaml
```

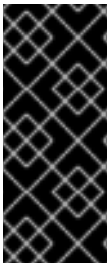
```
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/down-bootstrap.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/down-compute-nodes.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/down-control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/down-network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/down-security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/down-containers.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/inventory.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-4.19/upi/openstack/update-network-resources.yaml'
```

Playbook はマシンにダウンロードされます。

重要

インストールプロセス時に、Playbook を変更してデプロイメントを設定できます。

クラスターの有効期間中に、すべての Playbook を保持します。OpenShift Container Platform クラスターを RHOSP から削除するには Playbook が必要です。



重要

bootstrap.yaml、**compute-nodes.yaml**、**control-plane.yaml**、**network.yaml**、および **security-groups.yaml** ファイルに加えた編集内容は、**down-** の接頭辞が付けられた対応する Playbook に一致している必要があります。たとえば、**bootstrap.yaml** ファイルへの編集は、**down-bootstrap.yaml** ファイルにも反映される必要があります。両方のファイルを編集しない場合、サポートされるクラスターの削除プロセスは失敗します。

4.6. インストールプログラムの取得

OpenShift Container Platform をインストールする前に、インストールに使用しているホストにインストールファイルをダウンロードします。

前提条件

- 500 MB のローカルディスク領域がある Linux または macOS を実行するコンピューターが必要です。

手順

1. Red Hat Hybrid Cloud Console の Cluster Type ページに移動します。Red Hat アカウントがある場合は、認証情報を使用してログインします。アカウントがない場合はこれを作成します。

ヒント

特定の OpenShift Container Platform リリースのバイナリーをダウンロード することもできます。

2. ページの **Run it yourself** セクションからインフラストラクチャープロバイダーを選択します。
3. **OpenShift Installer** のドロップダウンメニューからホストオペレーティングシステムとアーキテクチャーを選択し、**Download Installer** をクリックします。
4. ダウンロードしたファイルを、インストール設定ファイルを保存するディレクトリーに配置します。

重要

- インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターのインストール完了後は、インストールプログラムおよびインストールプログラムが作成するファイルを保持する必要があります。クラスターを削除するには、両方のファイルが必要です。
- インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。クラスターを削除するには、特定のクラウドプロバイダー用の OpenShift Container Platform のアンインストール手順を実行します。

5. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar -xvf openshift-install-linux.tar.gz
```

6. Red Hat OpenShift Cluster Manager からインストールプルシークレット をダウンロードします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する Quay.io など、組み込まれた各種の認証局によって提供されるサービスで認証できます。

ヒント

Red Hat カスタマーポータル からインストールプログラムを取得することもできます。このページでは、ダウンロードするインストールプログラムのバージョンを指定できます。ただし、このページにアクセスするには、有効なサブスクリプションが必要です。

4.7. クラスターノードの SSH アクセス用のキーペアの生成

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストールプログラムに指定できます。キーは、Ignition 設定ファイルを介して Red Hat Enterprise Linux CoreOS (RHCOS) ノードに渡され、ノードへの SSH アクセスを認証するために使用されます。このキーは各ノードの **core** ユーザーの **~/.ssh/authorized_keys** リストに追加され、パスワードなしの認証が可能になります。

キーがノードに渡されると、キーペアを使用して RHCOS ノードにユーザー **core** として SSH を実行できます。SSH 経由でノードにアクセスするには、秘密鍵のアイデンティティーをローカルユーザーの SSH で管理する必要があります。

インストールのデバッグまたは障害復旧を実行するためにクラスターノードに対して SSH を実行する場合は、インストールプロセスの間に SSH 公開鍵を指定する必要があります。**./openshift-install gather** コマンドでは、SSH 公開鍵がクラスターノードに配置されている必要もあります。

重要

障害復旧およびデバッグが必要な実稼働環境では、この手順を省略しないでください。

注記

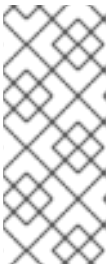
プラットフォーム固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. クラスターノードへの認証に使用するローカルマシンに既存の SSH キーペアがない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 新しい SSH キーのパスとファイル名 (~/**.ssh/id_ed25519** など) を指定します。既存のキーペアがある場合は、公開鍵が ~/**.ssh** ディレクトリーにあることを確認します。



注記

x86_64、**ppc64le**、および **s390x** アーキテクチャーのみで FIPS 140-2/140-3 検証のために NIST に提出された RHEL 暗号化ライブラリーを使用する OpenShift Container Platform クラスターをインストールする予定がある場合は、**ed25519** アルゴリズムを使用するキーを作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

2. 公開 SSH キーを表示します。

```
$ cat <path>/<file_name>.pub
```

たとえば、次のコマンドを実行して ~/**.ssh/id_ed25519.pub** 公開鍵を表示します。

```
$ cat ~/.ssh/id_ed25519.pub
```

3. ローカルユーザーの SSH エージェントに SSH 秘密鍵 ID が追加されていない場合は、それを追加します。キーの SSH エージェント管理は、クラスターノードへのパスワードなしの SSH 認証、または **./openshift-install gather** コマンドを使用する場合は必要になります。

注記

一部のディストリビューションでは、~/**.ssh/id_rsa** および ~/**.ssh/id_dsa** などのデフォルトの SSH 秘密鍵のアイデンティティーは自動的に管理されます。

a. **ssh-agent** プロセスがローカルユーザーに対して実行されていない場合は、バックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"
```

出力例

```
Agent pid 31874
```

注記

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

4. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> 1
```

1 ~/**.ssh/id_ed25519** などの、SSH プライベートキーのパスおよびファイル名を指定します。

出力例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

次のステップ

- OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストールプログラムに指定します。

4.8. RED HAT ENTERPRISE LINUX COREOS (RHCOS) イメージの作成

OpenShift Container Platform インストールプログラムでは、Red Hat Enterprise Linux CoreOS (RHCOS) イメージが Red Hat OpenStack Platform (RHOSP) クラスターに存在する必要があります。最新の RHCOS イメージを取得した後、RHOSP CLI を使用してこれをアップロードします。

前提条件

- RHOSP CLI がインストールされています。

手順

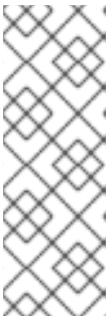
1. Red Hat カスタマーポータルの 製品ダウンロードページ にログインします。
2. **Version** で、Red Hat Enterprise Linux (RHEL) 8 用の OpenShift Container Platform 4.19 の最新リリースを選択します。

重要

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。

3. **Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)**をダウンロードします。

4. イメージを展開します。



注記

クラスターが使用する前に RHOSP イメージを圧縮解除する必要があります。ダウンロードしたファイルの名前に、**.gz** または **.tgz** などの圧縮拡張子が含まれていない場合があります。ファイルを圧縮するか、どのように圧縮するかを確認するには、コマンドラインで以下を入力します。

```
$ file <name_of_downloaded_file>
```

5. ダウンロードしたイメージから、RHOSP CLI を使用して **rhcos** という名前のイメージをクラスターに作成します。

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-${RHCOS_VERSION}-openstack.qcow2 rhcos
```

重要

RHOSP 環境によっては、**.raw** または **.qcow2** 形式 のいずれかでイメージをアップロードできる場合があります。Ceph を使用する場合は、**.raw** 形式を使用する必要があります。

警告

インストールプログラムが同じ名前を持つ複数のイメージを見つける場合、それらのイメージのいずれかがランダムに選択されます。この動作を回避するには、RHOSP でリソースの一意の名前を作成します。

RHOSP にイメージをアップロードした後は、インストールプログラムでイメージを利用できます。

4.9. 外部ネットワークアクセスの確認

OpenShift Container Platform インストールプロセスでは、外部ネットワークへのアクセスが必要です。外部ネットワーク値をこれに指定する必要があります。指定しない場合には、デプロイメントは失敗します。このプロセスを実行する前に、外部ルータータイプのネットワークが Red Hat OpenStack

Platform (RHOSP) に存在することを確認します。

前提条件

- OpenStack のネットワークサービスを、DHCP エージェントがインスタンスの DNS クエリーを転送できるように設定します。

手順

1. RHOSP CLI を使用して、'External' ネットワークの名前と ID を確認します。

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

出力例

```
+--------------------------------------+----------------+-------------+
| ID                                   | Name           | Router Type |
+--------------------------------------+----------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+----------------+-------------+
```

外部ルータータイプのあるネットワークがネットワークリストに表示されます。1 つ以上のネットワークが表示されない場合は、デフォルトの Floating IP ネットワークの作成 および デフォルトのプロバイダーネットワークの作成 を参照してください。

注記

Neutron トランクサービスプラグインが有効にされると、トランクポートがデフォルトで作成されます。詳細は、Neutron trunk port を参照してください。

4.10. 環境へのアクセスの有効化

デプロイ時に、OpenShift Container Platform マシンはすべて Red Hat OpenStack Platform (RHOSP) テナントネットワークに作成されます。したがって、ほとんどの RHOSP デプロイメントでは直接アクセスできません。

インストール時に Floating IP アドレス (FIP) を使用して OpenShift Container Platform API およびアプリケーションのアクセスを設定できます。FIP を設定せずにインストールを完了することもできますが、インストーラーは API またはアプリケーションを外部からアクセスする方法を設定しません。

4.10.1. floating IP アドレスを使用したアクセスの有効化

OpenShift Container Platform API、クラスターアプリケーション、およびブートストラッププロセスへの外部アクセス用に Floating IP (FIP) アドレスを作成します。

手順

1. Red Hat OpenStack Platform (RHOSP) CLI を使用して、API FIP を作成します。

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Red Hat OpenStack Platform (RHOSP) CLI を使用して、apps (アプリ)、または Ingress、FIP を作成します。

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. Red Hat OpenStack Platform (RHOSP) CLI を使用して、ブートストラップ FIP を作成します。

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. API および Ingress FIP の DNS サーバーに、これらのパターンに準拠するレコードを追加します。

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注記

DNS サーバーを制御していない場合は、次のようなクラスタードメイン名を **/etc/hosts** ファイルに追加することで、クラスターにアクセスできます。

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts ファイル内のクラスタードメイン名により、クラスターの Web コンソールおよび監視インターフェイスへのローカルアクセスが許可されます。**kubectl** または **oc** を使用することもできます。<application_floating_ip> を指す追加のエントリーを使用して、ユーザーアプリケーションにアクセスできます。このアクションにより、API およびアプリケーションは他のユーザーがアクセスできない状態になり、この状態は実稼働デプロイメントには適していませんが、開発およびテスト目的のインストールが可能になります。

5. FIP を以下の変数の値として **inventory.yaml** ファイルに追加します。

- **os_api_fip**
- **os_bootstrap_fip**
- **os_ingress_fip**

これらの値を使用する場合には、**inventory.yaml** ファイルの **os_external_network** 変数の値として外部ネットワークを入力する必要もあります。

ヒント

Floating IP アドレスを割り当て、ファイアウォール設定を更新することで、OpenShift Container Platform リソースがクラスター外で利用できる状態にすることができます。

4.10.2. Floating IP アドレスなしでのインストールの完了

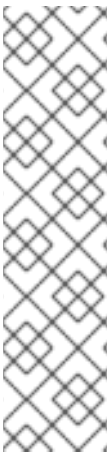
Floating IP アドレスを指定せずに OpenShift Container Platform を Red Hat OpenStack Platform (RHOSP) にインストールすることができます。

inventory.yaml ファイルで、以下の変数を定義しないでください。

- **os_api_fip**
- **os_bootstrap_fip**
- **os_ingress_fip**

外部ネットワークを提供できない場合は、**os_external_network** を空白のままにすることもできます。**os_external_network** の値を指定しない場合はルーターが作成されず、追加のアクションがない場合は、インストーラーは Glance からのイメージの取得に失敗します。インストールプロセスで、ネットワークリソースを作成する際に、独自の外部接続を設定する必要があります。

Floating IP アドレスまたは名前解決がないために、クラスター API に到達できないシステムから **wait-for** コマンドでインストーラーを実行すると、インストールに失敗します。このような場合にインストールが失敗するのを防ぐために、プロキシーネットワークを使用するか、マシンと同じネットワークにあるシステムからインストーラーを実行できます。



注記

API および Ingress ポートの DNS レコードを作成して、名前解決を有効にできます。以下に例を示します。

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

DNS サーバーを制御しない場合は、**/etc/hosts** ファイルにレコードを追加できます。このアクションにより、API は他者のアクセスできない状態になり、この状態は実稼働デプロイメントには適していませんが、開発およびテスト目的のインストールが可能になります。

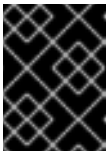
4.11. インストールプログラムのパラメーターの定義

OpenShift Container Platform インストールプログラムは、**clouds.yaml** というファイルを使用します。このファイルは、プロジェクト名、ログイン情報、認可サービスの URL を含む Red Hat OpenStack Platform (RHOSP) 設定パラメーターを説明します。

手順

1. **clouds.yaml** ファイルを作成します。

- RHOSP ディストリビューションに Horizon Web UI が含まれる場合には、そこに **clouds.yaml** ファイルを生成します。



重要

パスワードを必ず **auth** フィールドに追加してください。シークレットは、**clouds.yaml** の 別のファイル に保持できます。

- RHOSP ディストリビューションに Horizon Web UI が含まれない場合や Horizon を使用する必要がない場合には、このファイルを独自に作成します。**clouds.yaml** の詳細は、RHOSP ドキュメントの Config files を参照してください。

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. RHOSP インストールでエンドポイント認証用に自己署名認証局 (CA) を使用する場合、以下を実行します。

   a. 認証局ファイルをマシンにコピーします。

   b. **cacerts** キーを **clouds.yaml** ファイルに追加します。この値は、CA 証明書への絶対的な root 以外によるアクセスが可能なパスである必要があります。

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

ヒント

カスタム CA 証明書を使用してインストーラーを実行した後に、**cloud-provider-config** キーマップの **ca-cert.pem** キーの値を編集して証明書を更新できます。コマンドラインで、以下を実行します。

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. **clouds.yaml** ファイルを以下の場所のいずれかに置きます。

   a. **OS_CLIENT_CONFIG_FILE** 環境変数の値

b. 現行ディレクトリー

c. Unix 固有のユーザー設定ディレクトリー (例: ~/**.config**/**openstack**/**clouds.yaml**)

d. Unix 固有のサイト設定ディレクトリー (例: /**etc**/**openstack**/**clouds.yaml**)
インストールプログラムはこの順序で **clouds.yaml** を検索します。

4.12. RHOSP でのネットワークリソースの作成

独自のインフラストラクチャーを使用する Red Hat OpenStack Platform (RHOSP) インストールの OpenShift Container Platform に必要なネットワークリソースを作成します。時間を節約するには、セキュリティーグループ、ネットワーク、サブネット、ルーター、およびポートを生成する指定された Ansible Playbook を実行します。

前提条件

- 「Playbook 依存関係のダウンロード」でモジュールをダウンロードしている。
- 「インストール Playbook のダウンロード」で Playbook をダウンロードしている。

手順

1. デュアルスタッククラスターデプロイメントの場合は、**inventory.yaml** ファイルを編集し、**os_subnet6** 属性のコメントを解除します。
2. RHOSP デプロイメントでネットワークリソースに一意の名前が付けられるように、Ansible Playbook で使用する環境変数と JSON ファイルを作成します。
   a. 次のコマンドを実行して、一意の名前の値を持つ環境変数を作成します。

```
$ export OS_NET_ID="openshift-$(dd if=/dev/urandom count=4 bs=1 2>/dev/null
|hexdump -e '"%02x"')"
```

   b. コマンドラインで次のコマンドを実行して、変数が設定されていることを確認します。

```
$ echo $OS_NET_ID
```

   c. 次のコマンドを実行して、**netid.json** というファイルに、変数を含む JSON オブジェクトを作成します。

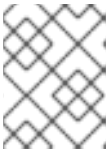
```
$ echo "{\"os_net_id\": \"$OS_NET_ID\"}" | tee netid.json
```

3. コマンドラインで次のコマンドを実行して、ネットワークリソースを作成します。

```
$ ansible-playbook -i inventory.yaml network.yaml
```

注記

inventory.yaml Playbook の API VIP フィールドと Ingress VIP フィールドは、ネットワークポートに割り当てられた IP アドレスで上書きされます。



注記

network.yaml Playbook によって作成されたリソースは、**down-network.yaml** Playbook によって削除されます。

4.13. インストール設定ファイルの作成

Red Hat OpenStack Platform (RHOSP) にインストールする OpenShift Container Platform クラスターをカスタマイズできます。

前提条件

- OpenShift Container Platform インストールプログラムおよびクラスターのプルシークレットがある。

手順

1. **install-config.yaml** ファイルを作成します。

   a. インストールプログラムが含まれるディレクトリーに切り替え、以下のコマンドを実行します。

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 **<installation_directory>** の場合、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

   ディレクトリーを指定する場合:

   - ディレクトリーに **execute** 権限があることを確認します。この権限は、インストールディレクトリーで Terraform バイナリーを実行するために必要です。
   - 空のディレクトリーを使用します。ブートストラップ X.509 証明書などの一部のインストールアセットは有効期限が短いため、インストールディレクトリーを再利用しないでください。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してください。

   b. プロンプト時に、クラウドの設定の詳細情報を指定します。

      i. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。

      注記

      インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

      ii. ターゲットに設定するプラットフォームとして **openstack** を選択します。

      iii. クラスターのインストールに使用する Red Hat OpenStack Platform (RHOSP) の外部ネットワーク名を指定します。

iv. OpenShift API への外部アクセスに使用する floating IP アドレスを指定します。

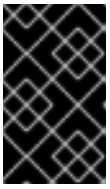
v. コントロールプレーンノードに使用する少なくとも 16 GB の RAM とコンピュートノードに使用する 8 GB の RAM を持つ RHOSP フレーバーを指定します。

vi. クラスターをデプロイするベースドメインを選択します。すべての DNS レコードはこのベースのサブドメインとなり、クラスター名も含まれます。

vii. クラスターの名前を入力します。名前は 14 文字以下でなければなりません。

2. **install-config.yaml** ファイルを変更します。利用可能なパラメーターの詳細は、「インストール設定パラメーター」のセクションを参照してください。

3. **install-config.yaml** ファイルをバックアップし、複数のクラスターをインストールするのに使用できるようにします。



重要

install-config.yaml ファイルはインストールプロセス時に使用されます。このファイルを再利用する必要がある場合は、この段階でこれをバックアップしてください。

これで、指定したディレクトリーに **install-config.yaml** ファイルが作成されます。

関連情報

- OpenStack のインストール設定パラメーター

4.13.1. RHOSP デプロイメントでのカスタムサブネット

オプションで、選択する Red Hat OpenStack Platform (RHOSP) サブネットにクラスターをデプロイすることができます。サブネットの GUID は、**install-config.yaml** ファイルの **platform.openstack.machinesSubnet** の値として渡されます。

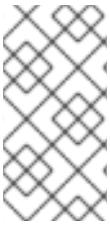
このサブネットはクラスターのプライマリーサブネットとして使用されます。デフォルトで、ノードおよびポートはこの上に作成されます。**platform.openstack.machinesSubnet** プロパティーの値をサブネットの UUID に設定すると、異なる RHOSP サブネットにノードおよびポートを作成することができます。

カスタムサブネットを使用して OpenShift Container Platform インストーラーを実行する前に、設定が以下の要件を満たしていることを確認してください。

- **platform.openstack.machinesSubnet** で使用されるサブネットで DHCP が有効にされている。
- **platform.openstack.machinesSubnet** の CIDR は **networking.machineNetwork** の CIDR に一致する。
- インストールプログラムのユーザーには、固定 IP アドレスを持つポートなど、このネットワークでポートを作成するパーミッションがある。

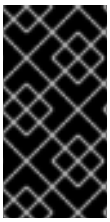
カスタムサブネットを使用するクラスターには、以下の制限があります。

- Floating IP アドレスを使用するクラスターをインストールする予定の場合には、**platform.openstack.machinesSubnet** サブネットを **externalNetwork** ネットワークに接続されているルーターに接続する必要があります。
- **platform.openstack.machinesSubnet** の値が **install-config.yaml** ファイルに設定されている場合、インストールプログラムは RHOSP マシンのプライベートネットワークまたはサブネットを作成しません。
- **platform.openstack.externalDNS** プロパティーは、カスタムサブネットと同時に使用することはできません。カスタムサブネットを使用するクラスターに DNS を追加するには、RHOSP ネットワークで DNS を設定します。



注記

デフォルトでは、API VIP は x.x.x.5 を取得し、Ingress VIP はネットワークの CIDR ブロックから x.x.x.7 を取得します。これらのデフォルト値を上書きするには、DHCP 割り当てプール外の **platform.openstack.apiVIPs** および **platform.openstack.ingressVIPs** の値を設定します。



重要

ネットワークの CIDR 範囲は、クラスターのインストール後に調整できません。Red Hat は、namespace ごとに作成される Pod の数を慎重に検討する必要があるため、クラスターのインストール時に範囲を決定するための直接的なガイダンスを提供していません。

4.13.2. RHOSP のカスタマイズされた install-config.yaml ファイルのサンプル

次の **install-config.yaml** ファイルの例は、すべての可能な Red Hat OpenStack Platform (RHOSP) カスタマイズオプションを示しています。

重要

このサンプルファイルは参照用にのみ提供されます。インストールプログラムを使用して **install-config.yaml** ファイルを取得する必要があります。

例4.1 シングルスタックの install-config.yaml ファイルの例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
```

```
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

例4.2 デュアルスタックの install-config.yaml ファイルの例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: 192.168.25.0/24
  - cidr: fd2e:6f44:5dd8:c956::/64
  serviceNetwork:
  - 172.30.0.0/16
  - fd02::/112
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiVIPs:
    - 192.168.25.10
    - fd2e:6f44:5dd8:c956:f816:3eff:fec3:5955
```

```
    ingressVIPs:
    - 192.168.25.132
    - fd2e:6f44:5dd8:c956:f816:3eff:fe40:aecb
    controlPlanePort:
      fixedIPs:
      - subnet:
          name: openshift-dual4
      - subnet:
          name: openshift-dual6
      network:
        name: openshift-dual
  fips: false
  pullSecret: '{"auths": ...}'
  sshKey: ssh-ed25519 AAAA...
```

4.13.3. マシンのカスタムサブネットの設定

インストールプログラムがデフォルトで使用する IP 範囲は、OpenShift Container Platform のインストール時に作成する Neutron サブネットと一致しない可能性があります。必要な場合は、インストール設定ファイルを編集して、新規マシンの CIDR 値を更新します。

前提条件

- OpenShift Container Platform インストールプログラムで生成された **install-config.yaml** ファイルがあります。
- Python 3 がインストールされている。

手順

1. コマンドラインで、**install-config.yaml** ファイルと **inventory.yaml** ファイルが含まれるディレクトリーを参照します。
2. そのディレクトリーからスクリプトを実行して **install-config.yaml** ファイルを編集するか、手動でファイルを更新します。
   - スクリプトを使用して値を設定するには、次のコマンドを実行します。

```
$ python -c 'import os
import sys
import yaml
import re
re_os_net_id = re.compile(r"{{\s*os_net_id\s*}}")
os_net_id = os.getenv("OS_NET_ID")
path = "common.yaml"
facts = None
for _dict in yaml.safe_load(open(path))[0]["tasks"]:
    if "os_network" in _dict.get("set_fact", {}):
        facts = _dict["set_fact"]
        break
if not facts:
    print("Cannot find `os_network` in common.yaml file. Make sure OpenStack resource
names are defined in one of the tasks.")
    sys.exit(1)
```

```
os_network = re_os_net_id.sub(os_net_id, facts["os_network"])
os_subnet = re_os_net_id.sub(os_net_id, facts["os_subnet"])
path = "install-config.yaml"
data = yaml.safe_load(open(path))
inventory = yaml.safe_load(open("inventory.yaml"))["all"]["hosts"]["localhost"]
machine_net = [{"cidr": inventory["os_subnet_range"]}]
api_vips = [inventory["os_apiVIP"]]
ingress_vips = [inventory["os_ingressVIP"]]
ctrl_plane_port = {"network": {"name": os_network}, "fixedIPs": [{"subnet": {"name":
os_subnet}}]}
if inventory.get("os_subnet6_range"): (1)
    os_subnet6 = re_os_net_id.sub(os_net_id, facts["os_subnet6"])
    machine_net.append({"cidr": inventory["os_subnet6_range"]})
    api_vips.append(inventory["os_apiVIP6"])
    ingress_vips.append(inventory["os_ingressVIP6"])
    data["networking"]["networkType"] = "OVNKubernetes"
    data["networking"]["clusterNetwork"].append({"cidr": inventory["cluster_network6_cidr"],
"hostPrefix": inventory["cluster_network6_prefix"]})
    data["networking"]["serviceNetwork"].append(inventory["service_subnet6_range"])
    ctrl_plane_port["fixedIPs"].append({"subnet": {"name": os_subnet6}})
data["networking"]["machineNetwork"] = machine_net
data["platform"]["openstack"]["apiVIPs"] = api_vips
data["platform"]["openstack"]["ingressVIPs"] = ingress_vips
data["platform"]["openstack"]["controlPlanePort"] = ctrl_plane_port
del data["platform"]["openstack"]["externalDNS"]
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

(1) デュアルスタック (IPv4/IPv6) 環境に適用されます。

4.13.4. コンピュートマシンプールを空にする

独自のインフラストラクチャーを使用するインストールを実行するには、インストール設定ファイルのコンピュートマシンの数をゼロに設定します。その後、これらのマシンを手動で作成します。

前提条件

- OpenShift Container Platform インストールプログラムで生成された **install-config.yaml** ファイルがあります。

手順

1. コマンドラインで、**install-config.yaml** が含まれるディレクトリーを参照します。
2. そのディレクトリーからスクリプトを実行して **install-config.yaml** ファイルを編集するか、手動でファイルを更新します。
   - スクリプトを使用して値を設定するには、以下を実行します。

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["compute"][0]["replicas"] = 0;
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

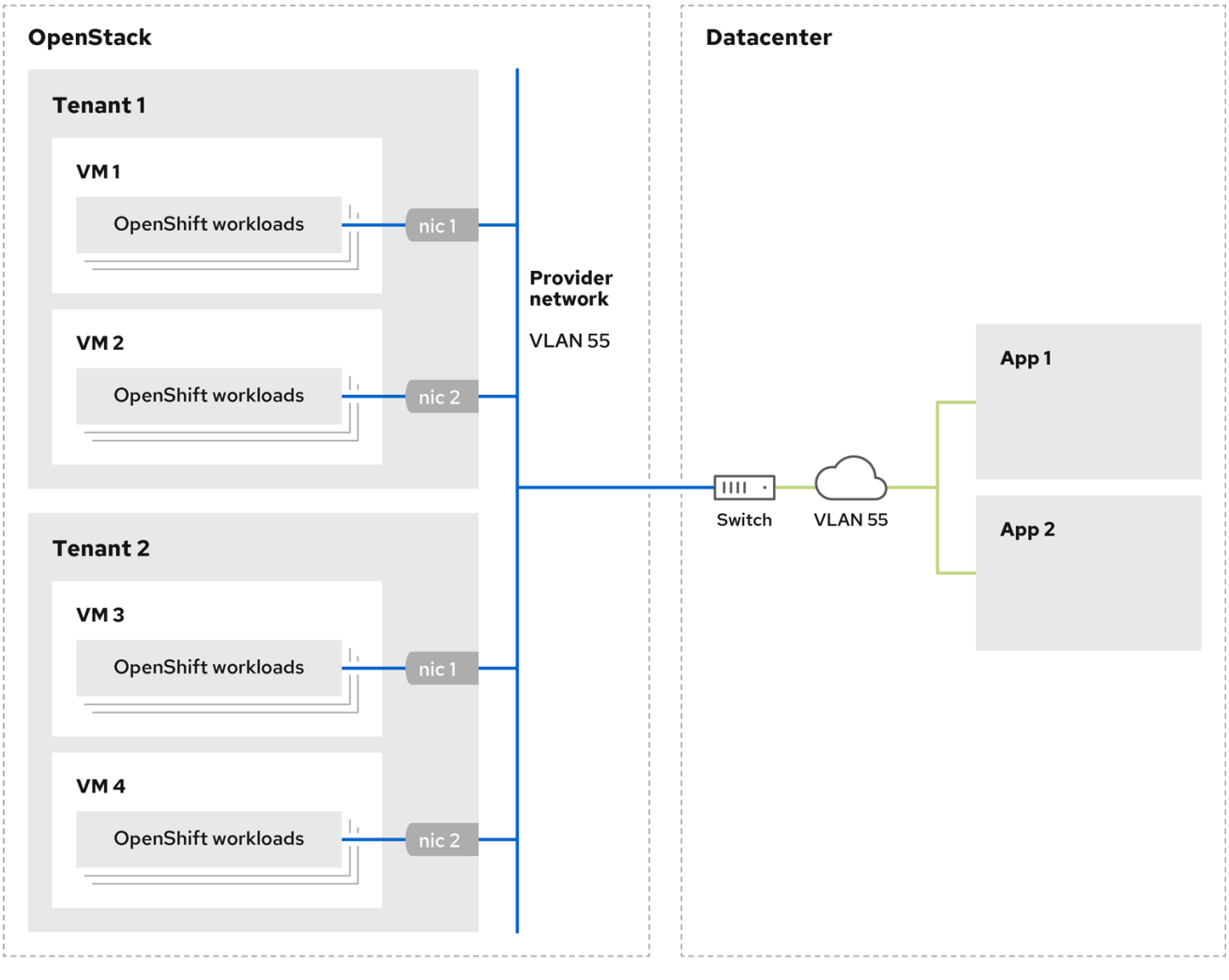
- 値を手動で設定するには、ファイルを開き、**compute.<first entry>.replicas** の値を **0** に設定します。

4.13.5. RHOSP プロバイダーネットワーク上のクラスターデプロイメント

プロバイダーネットワーク上のプライマリーネットワークインターフェイスを使用して、OpenShift Container Platform クラスターを Red Hat OpenStack Platform (RHOSP) にデプロイできます。プロバイダーネットワークは一般的に、インターネットへの到達に使用可能なパブリックネットワークに、プロジェクトが直接アクセスできるように使用します。ネットワーク作成プロセスの一環として、プロバイダーネットワークをプロジェクト間で共有することもできます。

RHOSP プロバイダーネットワークは、データセンター内の既存の物理ネットワークに直接マップします。RHOSP 管理者はこれらを作成する必要があります。

以下の例では、OpenShift Container Platform ワークロードはプロバイダーネットワークを使用してデータセンターに接続されます。



プロバイダーネットワークにインストールされている OpenShift Container Platform クラスターは、テナントネットワークまたは Floating IP アドレスを必要としません。インストーラーは、インストール中にこれらのリソースを作成しません。

プロバイダーネットワークタイプの例には、フラット (タグなし) および VLAN (802.1Q タグ付き) が含まれます。

注記

クラスターは、ネットワークタイプが許可する限り多くのプロバイダーネットワーク接続をサポートできます。たとえば、VLAN ネットワークは、通常最大 4096 の接続をサポートします。

プロバイダーネットワークおよびテナントネットワークの詳細は、RHOSP のドキュメント を参照してください。

4.13.5.1. クラスターのインストールにおける RHOSP プロバイダーネットワーク要件

OpenShift Container Platform クラスターをインストールする前に、Red Hat OpenStack Platform (RHOSP) のデプロイメントおよびプロバイダーネットワークは、さまざまな条件を満たす必要があります。

- RHOSP ネットワークサービス (Neutron) が有効化され、RHOSP ネットワーク API 経由でアクセス可能であること。
- RHOSP ネットワークサービスでは、ポートセキュリティーと許可するアドレスペアの機能拡張が有効化 されていること。
- プロバイダーネットワークは他のテナントと共有できます。

  ヒント

  --share フラグを指定して **openstack network create** コマンドを使用して、共有できるネットワークを作成します。

- クラスターのインストールに使用する RHOSP プロジェクトは、プロバイダーネットワークと適切なサブネットを所有する必要があります。

  ヒント

  "openshift" という名前のプロジェクトのネットワークを作成するには、以下のコマンドを入力します。

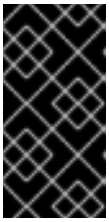
  ```
  $ openstack network create --project openshift
  ```

  "openshift" という名前のプロジェクトのサブネットを作成するには、以下のコマンドを入力します。

  ```
  $ openstack subnet create --project openshift
  ```

  RHOSP でのネットワークの作成に関する詳細は、プロバイダーネットワークに関するドキュメント を参照してください。

  クラスターが **admin** ユーザーによって所有されている場合、そのユーザーとしてインストーラーを実行してネットワーク上でポートを作成する必要があります。



重要

プロバイダーネットワークは、クラスターの作成に使用される RHOSP プロジェクトによって所有されている必要があります。所有されていない場合は、RHOSP Compute サービス (Nova) はそのネットワークからポートを要求できません。

- プロバイダーネットワークが、デフォルトで **169.254.169.254** である RHOSP メタデータサービスの IP アドレスに到達できることを確認します。
  RHOSP SDN とネットワークサービス設定によっては、サブネットを作成する際に、ルートを提供しなければならない場合があります。以下に例を示します。

```
$ openstack subnet create --dhcp --host-route
destination=169.254.169.254/32,gateway=192.0.2.2 ...
```

- オプション: ネットワークのセキュリティーを保護するには、単一のプロジェクトへのネットワークアクセスを制限する ロールベースのアクセス制御 (RBAC) ルールを作成します。

4.13.5.2. プロバイダーネットワークにプライマリーインターフェイスを持つクラスターのデプロイ

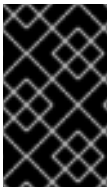
Red Hat OpenStack Platform (RHOSP) プロバイダーネットワーク上にプライマリーネットワークインターフェイスを持つ OpenShift Container Platform クラスターをデプロイすることができます。

前提条件

- 「クラスターのインストールにおける RHOSP プロバイダーネットワーク要件」に記載されているとおりに、お使いの Red Hat OpenStack Platform (RHOSP) のデプロイメントが設定されています。

手順

1. テキストエディターで **install-config.yaml** ファイルを開きます。
2. **platform.openstack.apiVIPs** プロパティーの値を API VIP の IP アドレスに設定します。
3. **platform.openstack.ingressVIPs** プロパティーの値を Ingress VIP の IP アドレスに設定します。
4. **platform.openstack.machinesSubnet** プロパティーの値をプロバイダーネットワークサブネットの UUID に設定します。
5. **networking.machineNetwork.cidr** プロパティーの値をプロバイダーネットワークサブネットの CIDR ブロックに設定します。



重要

platform.openstack.apiVIPs プロパティーおよび **platform.openstack.ingressVIPs** プロパティーはいずれも、**networking.machineNetwork.cidr** ブロックから割り当てられていない IP アドレスである必要があります。

RHOSP プロバイダーネットワークに依存するクラスターのインストール設定ファイルのセクション

```
...
platform:
  openstack:
    apiVIPs: 1
      - 192.0.2.13
    ingressVIPs: 2
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
networking:
  machineNetwork:
  - cidr: 192.0.2.0/24
```

1 2 OpenShift Container Platform 4.12 以降では、**apiVIP** および **ingressVIP** 設定は非推奨です。代わりに、リスト形式を使用して、**apiVIPs** および **ingressVIPs** 設定に値を入力します。

警告

プライマリーネットワークインターフェイスにプロバイダーネットワークを使用している間は、**platform.openstack.externalNetwork** パラメーターまたは **platform.openstack.externalDNS** パラメーターを設定することはできません。

クラスターをデプロイする際に、インストーラーは **install-config.yaml** ファイルを使用してプロバイダーネットワークにクラスターをデプロイします。

ヒント

プロバイダーネットワークを含むネットワークを **platform.openstack.additionalNetworkIDs** リストに追加できます。

クラスターのデプロイ後に、Pod を追加のネットワークに接続することができます。詳細は、複数ネットワークについて を参照してください。

4.14. KUBERNETES マニフェストおよび IGNITION 設定ファイルの作成

一部のクラスター定義ファイルを変更し、クラスターマシンを手動で起動する必要があるため、クラスターがマシンを設定するために必要な Kubernetes マニフェストと Ignition 設定ファイルを生成する必要があります。

インストール設定ファイルは Kubernetes マニフェストに変換されます。マニフェストは Ignition 設定ファイルにラップされます。これはクラスターマシンを設定するために後で使用されます。

重要

- OpenShift Container Platform のインストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになり、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、**コントロールプレーン証明書の期限切れの状態からのリカバリー** に関するドキュメントを参照してください。

- 24 時間証明書はクラスターのインストール後 16 時間から 22 時間にローテーションするため、Ignition 設定ファイルは、生成後 12 時間以内に使用することを推奨します。12 時間以内に Ignition 設定ファイルを使用することにより、インストール中に証明書の更新が実行された場合のインストールの失敗を回避できます。

前提条件

- OpenShift Container Platform インストールプログラムを取得していること。
- **install-config.yaml** インストール設定ファイルを作成していること。

手順

1. OpenShift Container Platform のインストールプログラムが含まれるディレクトリーに切り替え、クラスターの Kubernetes マニフェストを生成します。

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 **<installation_directory>** には、作成した **install-config.yaml** ファイルが含まれるインストールディレクトリーを指定します。

2. コントロールプレーンマシン、コンピュートマシンセット、およびコントロールプレーンマシンセットを定義する Kubernetes マニフェストファイルを削除します。

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

これらのリソースを独自に作成および管理するため、それらを初期化する必要はありません。

- コンピュートマシンセットファイルを保存して、マシン API を使用してコンピュートマシンを作成することができますが、環境に合わせてそれらへの参照を更新する必要があります。

3. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes マニフェストファイルの **mastersSchedulable** パラメーターが **false** に設定されていることを確認します。この設定により、Pod がコントロールプレーンマシンにスケジュールされなくなります。

   a. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** ファイルを開きます。

b. **mastersSchedulable** パラメーターを見つけ、これが **false** に設定されていることを確認します。

c. ファイルを保存し、終了します。

4. Ignition 設定ファイルを作成するには、インストールプログラムが含まれるディレクトリーから以下のコマンドを実行します。

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 **<installation_directory>** には、同じインストールディレクトリーを指定します。

Ignition 設定ファイルは、インストールディレクトリー内のブートストラップ、コントロールプレーン、およびコンピュートノード用に作成されます。**kubeadmin-password** および **kubeconfig** ファイルが **./<installation_directory>/auth** ディレクトリーに作成されます。

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5. メタデータファイルの **infraID** キーを環境変数としてエクスポートします。

```
$ export INFRA_ID=$(jq -r .infraID metadata.json)
```

ヒント

metadata.json から **infraID** キーを抽出し、作成するすべての RHOSP リソースの接頭辞として使用します。これを実行することで、同じプロジェクトで複数のデプロイメントを実行する際に名前の競合が発生しないようにします。

4.15. ブートストラップ IGNITION ファイルの準備

OpenShift Container Platform インストールプロセスは、ブートストラップ Ignition 設定ファイルから作成されるブートストラップマシンに依存します。

ファイルを編集し、アップロードします。次に、Red Hat OpenStack Platform (RHOSP) がプライマリーファイルをダウンロードする際に使用するセカンダリーブートストラップ Ignition 設定ファイルを作成します。

前提条件

- インストーラープログラムが生成するブートストラップ Ignition ファイル **bootstrap.ign** があります。
- インストーラーのメタデータファイルのインフラストラクチャー ID は環境変数 (**$INFRA_ID**) として設定されます。
  - 変数が設定されていない場合は、**Kubernetes マニフェストおよび Ignition 設定ファイルの**

作成 を参照してください。

- HTTP(S) でアクセス可能な方法でブートストラップ Ignition ファイルを保存できます。
  - 記載された手順では RHOSP イメージサービス (Glance) を使用しますが、RHOSP ストレージサービス (Swift)、Amazon S3、内部 HTTP サーバー、またはアドホックの Nova サーバーを使用することもできます。

手順

1. 以下の Python スクリプトを実行します。スクリプトはブートストラップ Ignition ファイルを変更して、ホスト名および利用可能な場合は、実行時の CA 証明書ファイルを設定します。

```
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap\n').decode().strip()
files.append(
{
    'path': '/etc/hostname',
    'mode': 420,
    'contents': {
        'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
    }
})

ca_cert_path = os.environ.get('OS_CACERT', '')
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
        ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()

    files.append(
    {
        'path': '/opt/openshift/tls/cloud-ca-cert.pem',
        'mode': 420,
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + ca_cert_b64
        }
    })

ignition['storage']['files'] = files;

with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```

2. RHOSP CLI を使用して、ブートストラップ Ignition ファイルを使用するイメージを作成します。

```
$ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign
<image_name>
```

3. イメージの詳細を取得します。

```
$ openstack image show <image_name>
```

file 値をメモします。これは **v2/images/<image_ID>/file** パターンをベースとしています。

注記

作成したイメージがアクティブであることを確認します。

4. イメージサービスのパブリックアドレスを取得します。

```
$ openstack catalog show image
```

5. パブリックアドレスとイメージ **file** 値を組み合わせ、結果を保存場所として保存します。この場所は、**<image_service_public_URL>/v2/images/<image_ID>/file** パターンをベースとしています。

6. 認証トークンを生成し、トークン ID を保存します。

```
$ openstack token issue -c id -f value
```

7. **$INFRA_ID-bootstrap-ignition.json** というファイルに以下のコンテンツを挿入し、独自の値に一致するようにプレースホルダーを編集します。

```
{
  "ignition": {
    "config": {
      "merge": [{
        "source": "<storage_url>", 1
        "httpHeaders": [{
          "name": "X-Auth-Token", 2
          "value": "<token_ID>" 3
        }]
      }]
    },
    "security": {
      "tls": {
        "certificateAuthorities": [{
          "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" 4
        }]
      }
    },
    "version": "3.2.0"
  }
}
```

1 **ignition.config.merge.source** の値をブートストラップ Ignition ファイルのストレージ URL に置き換えます。

2 **httpHeaders** の **name** を **"X-Auth-Token"** に設定します。

3 **httpHeaders** の **value** をトークンの ID に設定します。

4 ブートストラップ Ignition ファイルサーバーが自己署名証明書を使用する場合は、base64 でエンコードされた証明書を含めます。

8. セカンダリー Ignition 設定ファイルを保存します。

ブートストラップ Ignition データはインストール時に RHOSP に渡されます。

警告

ブートストラップ Ignition ファイルには、**clouds.yaml** 認証情報などの機密情報が含まれます。これを安全な場所に保存し、インストールプロセスの完了後に削除します。

4.16. RHOSP でのコントロールプレーンの IGNITION 設定ファイルの作成

OpenShift Container Platform を独自のインフラストラクチャーの Red Hat OpenStack Platform (RHOSP) にインストールするには、コントロールプレーンの Ignition 設定ファイルが必要です。複数の設定ファイルを作成する必要があります。

注記

ブートストラップ Ignition 設定と同様に、各コントロールプレーンマシンのホスト名を明示的に定義する必要があります。

前提条件

- インストールプログラムのメタデータファイルのインフラストラクチャー ID は環境変数 (**$INFRA_ID**) として設定されます。
  - 変数が設定されていない場合は、「Kubernetes マニフェストおよび Ignition 設定ファイルの作成」を参照してください。

手順

- コマンドラインで、以下の Python スクリプトを実行します。

```
$ for index in $(seq 0 2); do
    MASTER_HOSTNAME="$INFRA_ID-master-$index\n"
    python -c "import base64, json, sys;
ignition = json.load(sys.stdin);
storage = ignition.get('storage', {});
files = storage.get('files', []);
files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
'data:text/plain;charset=utf-8;base64,' +
base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(), 'verification': {}},
```

```
'filesystem': 'root'});
storage['files'] = files;
ignition['storage'] = storage
json.dump(ignition, sys.stdout)" <master.ign >"$INFRA_ID-master-$index-ignition.json"
done
```

以下の 3 つのコントロールプレーン Ignition ファイルが作成されます。**<INFRA_ID>-master-0-ignition.json**、**<INFRA_ID>-master-1-ignition.json**、および **<INFRA_ID>-master-2-ignition.json**。

4.17. RHOSP でのネットワークリソースの更新

独自のインフラストラクチャーを使用する Red Hat OpenStack Platform (RHOSP) インストールの OpenShift Container Platform に必要なネットワークリソースを更新します。

前提条件

- Python 3 がマシンにインストールされている。
- 「Playbook 依存関係のダウンロード」でモジュールをダウンロードしている。
- 「インストール Playbook のダウンロード」で Playbook をダウンロードしている。

手順

1. オプション: 外部ネットワークの値を **inventory.yaml** Playbook に追加します。

   inventory.yaml Ansible Playbook の外部ネットワーク値の例

```
...
    # The public network providing connectivity to the cluster. If not
    # provided, the cluster external connectivity must be provided in another
    # way.

    # Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
    os_external_network: 'external'
...
```

重要

inventory.yaml ファイルの **os_external_network** の値を指定しなかった場合は、仮想マシンが Glance および外部接続にアクセスできるようにする必要があります。

2. オプション: 外部ネットワークおよび Floating IP (FIP) アドレスの値を **inventory.yaml** Playbook に追加します。

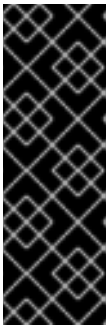
   inventory.yaml Ansible Playbook の FIP 値の例

```
...
    # OpenShift API floating IP address. If this value is non-empty, the
    # corresponding floating IP will be attached to the Control Plane to
    # serve the OpenShift API.
```

```
os_api_fip: '203.0.113.23'

# OpenShift Ingress floating IP address. If this value is non-empty, the
# corresponding floating IP will be attached to the worker nodes to serve
# the applications.
os_ingress_fip: '203.0.113.19'

# If this value is non-empty, the corresponding floating IP will be
# attached to the bootstrap machine. This is needed for collecting logs
# in case of install failure.
os_bootstrap_fip: '203.0.113.20'
```



重要

os_api_fip および **os_ingress_fip** の値を定義しない場合、インストール後のネットワーク設定を実行する必要があります。

os_bootstrap_fip の値を定義しなかった場合、インストールプログラムが失敗したインストールからデバッグ情報をダウンロードできません。

詳細は、「環境へのアクセスの有効化」を参照してください。

3. コマンドラインで、**security-groups.yaml** Playbook を実行してセキュリティーグループを作成します。

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

4. コマンドラインで、**update-network-resources.yaml** Playbook を実行してネットワークリソースを更新します。

```
$ ansible-playbook -i inventory.yaml update-network-resources.yaml 1
```

1 この Playbook は、ネットワーク、サブネット、ポート、ルーターにタグを追加します。また、Floating IP アドレスを API ポートと Ingress ポートに接続し、それらのポートのセキュリティーグループを設定します。

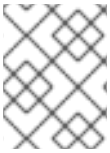
5. オプション: Nova サーバーが使用するデフォルトのリゾルバーを制御する必要がある場合は、RHOSP CLI コマンドを実行します。

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2>
"$INFRA_ID-nodes"
```

6. オプション: 作成した **inventory.yaml** ファイルを使用して、インストールをカスタマイズできます。たとえば、ベアメタルマシンを使用するクラスターをデプロイすることができます。

4.17.1. ベアメタルマシンを使用したクラスターのデプロイ

クラスターがベアメタルマシンを使用する必要がある場合は、**inventory.yaml** ファイルを変更します。クラスターには、ベアメタル上で実行されるコンピュートマシンを含めることができます。



注記

install-config.yaml ファイルで、ベアメタルワーカーに使用する RHOSP ネットワークが Floating IP アドレスをサポートするかどうかが反映されていることを確認します。

前提条件

- RHOSP Bare Metal サービス (Ironic) が有効になっており、RHOSP Compute API 経由でアクセスできる。
- ベアメタルを RHOSP フレーバー として利用できる。
- クラスターが 16.1.6 以降、16.2.4 未満の RHOSP バージョンで実行している場合は、メタデータサービスが OpenShift Container Platform ノード上のサービスで使用できなくなる 既知の問題 により、ベアメタルワーカーは機能しません。
- RHOSP ネットワークが、仮想マシンとベアメタルサーバーの両方の接続をサポートしている。
- 既存のネットワークにマシンをデプロイする場合、RHOSP サブネットがプロビジョニングされている。
- インストーラーによってプロビジョニングされるネットワークにマシンをデプロイする場合、RHOSP Bare Metal サービス (Ironic) が、テナントネットワークで実行される Preboot eXecution Environment (PXE) ブートマシンをリッスンして通信することができる。
- **inventory.yaml** ファイルを OpenShift Container Platform インストールプロセスの一部として作成している。

手順

1. **inventory.yaml** ファイルで、マシンのフレーバーを編集します。
   a. **os_flavor_worker** の値をベアメタルフレーバーに変更します。

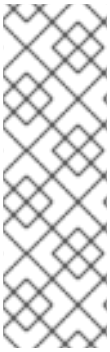
ベアメタルの inventory.yaml のサンプルファイル

```
all:
  hosts:
    localhost:
      ansible_connection: local
      ansible_python_interpreter: "{{ansible_playbook_python}}"

      # User-provided values
      os_subnet_range: '10.0.0.0/16'
      os_flavor_master: 'my-vm-flavor'
      os_flavor_worker: 'my-bare-metal-flavor' (1)
      os_image_rhcos: 'rhcos'
      os_external_network: 'external'
...
```

(1) この値を、コンピュートマシンに使用するベアメタルのフレーバーに変更します。

更新された **inventory.yaml** ファイルを使用してインストールプロセスを完了します。デプロイメント時に作成されるマシンは、ファイルに追加したフレーバーを使用します。



注記

インストーラーは、ベアメタルマシンの起動中にタイムアウトする可能性があります。

インストーラーがタイムアウトした場合は、インストーラーの **wait-for** コマンドを使用してデプロイメントを再起動してからデプロイメントを完了します。以下に例を示します。

```
$ ./openshift-install wait-for install-complete --log-level debug
```

4.18. RHOSP でのブートストラップマシンの作成

ブートストラップマシンを作成し、これに Red Hat OpenStack Platform (RHOSP) で実行するために必要なネットワークアクセスを付与します。Red Hat は、このプロセスを単純化するために実行する Ansible Playbook を提供しています。

前提条件

- 「Playbook 依存関係のダウンロード」でモジュールをダウンロードしている。
- 「インストール Playbook のダウンロード」で Playbook をダウンロードしている。
- **inventory.yaml**、**common.yaml**、および **bootstrap.yaml** Ansible Playbook は共通ディレクトリーにある。
- インストールプログラムが作成した **metadata.json** ファイルが Ansible Playbook と同じディレクトリーにあります。

手順

1. コマンドラインで、作業ディレクトリーを Playbook の場所に切り替えます。
2. コマンドラインで、**bootstrap.yaml** Playbook を実行します。

   ```
   $ ansible-playbook -i inventory.yaml bootstrap.yaml
   ```

3. ブートストラップサーバーがアクティブになった後に、ログを表示し、Ignition ファイルが受信されたことを確認します。

   ```
   $ openstack console log show "$INFRA_ID-bootstrap"
   ```

4.19. RHOSP でのコントロールプレーンの作成

生成した Ignition 設定ファイルを使用して 3 つのコントロールプレーンマシンを作成します。Red Hat は、このプロセスを単純化するために実行する Ansible Playbook を提供しています。

前提条件

- 「Playbook 依存関係のダウンロード」でモジュールをダウンロードしている。

- 「インストール Playbook のダウンロード」で Playbook をダウンロードしている。
- インストールプログラムのメタデータファイルのインフラストラクチャー ID は環境変数 (**$INFRA_ID**) として設定されます。
- **inventory.yaml**、**common.yaml**、および **control-plane.yaml** Ansible Playbook は共通ディレクトリーにあります。
- 「コントロールプレーンの Ignition 設定ファイルの作成」で作成された 3 つの Ignition ファイルがある。

手順

1. コマンドラインで、作業ディレクトリーを Playbook の場所に切り替えます。
2. コントロールプレーンの Ignition 設定ファイルが作業ディレクトリーにまだない場合は、ファイルを作業ディレクトリーにコピーします。
3. コマンドラインで、**control-plane.yaml** Playbook を実行します。

```
$ ansible-playbook -i inventory.yaml control-plane.yaml
```

4. 以下のコマンドを実行してブートストラッププロセスをモニターします。

```
$ openshift-install wait-for bootstrap-complete
```

コントロールプレーンマシンが実行され、クラスターに参加していることを確認できるメッセージが表示されます。

```
INFO API v1.32.3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

4.20. CLI の使用によるクラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターに関する情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターをデプロイしていること。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. 次のコマンドを実行して、**kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。

2. 次のコマンドを実行し、エクスポートされた設定を使用して **oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

4.21. RHOSP からのブートストラップリソースの削除

不要になったブートストラップリソースを削除します。

前提条件

- 「Playbook 依存関係のダウンロード」でモジュールをダウンロードしている。
- 「インストール Playbook のダウンロード」で Playbook をダウンロードしている。
- **inventory.yaml**、**common.yaml**、および **down-bootstrap.yaml** Ansible Playbook が共通ディレクトリーにある。
- コントロールプレーンマシンが実行中である。
  - マシンのステータスが不明な場合は、「クラスターステータスの確認」を参照してください。

手順

1. コマンドラインで、作業ディレクトリーを Playbook の場所に切り替えます。
2. コマンドラインで、**down-bootstrap.yaml** Playbook を実行します。

```
$ ansible-playbook -i inventory.yaml down-bootstrap.yaml
```

ブートストラップポート、サーバー、および Floating IP アドレスが削除されます。

警告

ブートストラップ Ignition ファイル URL をまだ無効にしていない場合は、無効にしてください。

4.22. RHOSP でのコンピュートマシンの作成

コントロールプレーンの起動後、コンピュートマシンを作成します。Red Hat は、このプロセスを単純化するために実行する Ansible Playbook を提供しています。

前提条件

- 「Playbook 依存関係のダウンロード」でモジュールをダウンロードしている。
- 「インストール Playbook のダウンロード」で Playbook をダウンロードしている。
- **inventory.yaml**、**common.yaml**、および **compute-nodes.yaml** Ansible Playbook が共通ディレクトリーにある。
- インストールプログラムが作成した **metadata.json** ファイルが Ansible Playbook と同じディレクトリーにあります。
- コントロールプレーンが有効である。

手順

1. コマンドラインで、作業ディレクトリーを Playbook の場所に切り替えます。
2. コマンドラインで Playbook を実行します。

```
$ ansible-playbook -i inventory.yaml compute-nodes.yaml
```

次のステップ

- マシンの証明書署名要求を承認します。

4.23. マシンの証明書署名要求の承認

マシンをクラスターに追加する際に、追加したそれぞれのマシンに対して 2 つの保留状態の証明書署名要求 (CSR) が生成されます。これらの CSR が承認されていることを確認するか、必要な場合はそれらを承認してください。最初にクライアント要求を承認し、次にサーバー要求を承認する必要があります。

前提条件

- マシンがクラスターに追加されています。

手順

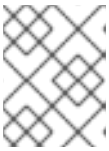
1. クラスターがマシンを認識していることを確認します。

```
$ oc get nodes
```

出力例

```
NAME     STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.32.3
master-1  Ready     master  63m  v1.32.3
master-2  Ready     master  64m  v1.32.3
```

出力には作成したすべてのマシンがリスト表示されます。



注記

上記の出力には、一部の CSR が承認されるまで、ワーカーノード (ワーカーノードとも呼ばれる) が含まれない場合があります。

2. 保留中の証明書署名要求 (CSR) を確認し、クラスターに追加したそれぞれのマシンのクライアントおよびサーバー要求に **Pending** または **Approved** ステータスが表示されていることを確認します。

```
$ oc get csr
```

出力例

```
NAME        AGE     REQUESTOR                                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

この例では、2 つのマシンがクラスターに参加しています。このリストにはさらに多くの承認された CSR が表示される可能性があります。

3. 追加したマシンの保留中の CSR すべてが **Pending** ステータスになった後に CSR が承認されない場合には、クラスターマシンの CSR を承認します。

注記

CSR のローテーションは自動的に実行されるため、クラスターにマシンを追加後 1 時間以内に CSR を承認してください。1 時間以内に承認しない場合には、証明書のローテーションが行われ、各ノードに 3 つ以上の証明書が存在するようになります。これらの証明書すべてを承認する必要があります。クライアントの CSR が承認された後に、Kubelet は提供証明書のセカンダリー CSR を作成します。これには、手動の承認が必要になります。次に、後続の提供証明書の更新要求は、Kubelet が同じパラメーターを持つ新規証明書を要求する場合に **machine-approver** によって自動的に承認されます。

注記

ベアメタルおよび他の user-provisioned infrastructure などのマシン API ではないプラットフォームで実行されているクラスターの場合、kubelet 提供証明書要求 (CSR) を自動的に承認する方法を実装する必要があります。要求が承認されない場合、API サーバーが kubelet に接続する際に提供証明書が必須であるため、**oc exec**、**oc rsh**、および **oc logs** コマンドは正常に実行できません。Kubelet エンドポイントにアクセスする操作には、この証明書の承認が必要です。この方法は新規 CSR の有無を監視し、CSR が **system:node** または **system:admin** グループの **node-bootstrapper** サービスアカウントによって提出されていることを確認し、ノードの ID を確認します。

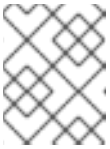
- それらを個別に承認するには、それぞれの有効な CSR に以下のコマンドを実行します。

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** は、現行の CSR のリストからの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注記

一部の Operator は、一部の CSR が承認されるまで利用できない可能性があります。

4. クライアント要求が承認されたら、クラスターに追加した各マシンのサーバー要求を確認する必要があります。

```
$ oc get csr
```

出力例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 残りの CSR が承認されず、それらが **Pending** ステータスにある場合、クラスターマシンの CSR を承認します。

- それらを個別に承認するには、それぞれの有効な CSR に以下のコマンドを実行します。

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** は、現行の CSR のリストからの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. すべてのクライアントおよびサーバーの CSR が承認された後に、マシンのステータスが **Ready** になります。以下のコマンドを実行して、これを確認します。

```
$ oc get nodes
```

出力例

```
NAME      STATUS    ROLES   AGE  VERSION
```

```
master-0  Ready    master  73m  v1.32.3
master-1  Ready    master  73m  v1.32.3
master-2  Ready    master  74m  v1.32.3
worker-0  Ready    worker  11m  v1.32.3
worker-1  Ready    worker  11m  v1.32.3
```

注記

サーバー CSR の承認後にマシンが **Ready** ステータスに移行するまでに数分の時間がかかる場合があります。

関連情報

- 証明書署名要求

4.24. インストールの正常な実行の確認

OpenShift Container Platform のインストールが完了していることを確認します。

前提条件

- インストールプログラム (**openshift-install**) があります。

手順

- コマンドラインで、以下を入力します。

```
$ openshift-install --log-level debug wait-for install-complete
```

プログラムはコンソール URL と管理者のログイン情報を出力します。

4.25. OPENSHIFT CONTAINER PLATFORM の TELEMETRY アクセス

OpenShift Container Platform 4.19 では、Telemetry サービスにもインターネットアクセスが必要です。Telemetry サービスは、クラスターの健全性と更新の成功に関するメトリクスを提供するためにデフォルトで実行されます。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは OpenShift Cluster Manager に登録されます。

OpenShift Cluster Manager インベントリーが正常である (Telemetry によって自動的に維持、または OpenShift Cluster Manager を使用して手動で維持) ことを確認した後に、subscription watch を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

関連情報

- Telemetry サービスの詳細は、リモートヘルスモニタリングについて を参照してください。

4.26. 次のステップ

- クラスターのカスタマイズ
- 必要に応じて、リモートヘルスレポート を作成できます。

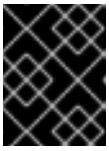
- ノードポートへの外部アクセスを有効にする必要がある場合は、ノードポートを使用して Ingress クラストラフィックを設定 します。
- Floating IP アドレス上でアプリケーショントラフィックを受け入れるように RHOSP を設定していない場合は、Floating IP アドレスを使用して RHOSP アクセスを設定 します。

第5章 非接続環境の OPENSTACK へのクラスターインストール

OpenShift Container Platform 4.19 では、インストールリリースコンテンツの内部ミラーを作成して、制限されたネットワーク内で Red Hat OpenStack Platform (RHOSP) にクラスターをインストールできます。

5.1. 前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスの詳細を確認している。
- クラスターインストール方法の選択およびそのユーザー向けの準備 を確認している。
- OpenShift クラスターでサポートされるプラットフォーム セクションを使用して、OpenShift Container Platform 4.19 が RHOSP バージョンと互換性があることを確認した。RHOSP サポートマトリックスの OpenShift Container Platform を参照して、プラットフォームのサポートを異なるバージョン間で比較することもできます。
- ミラーホストにレジストリーを作成 し、使用しているバージョンの OpenShift Container Platform の **imageContentSources** データを取得している。



重要

インストールメディアはミラーホストにあるため、そのコンピューターを使用してすべてのインストール手順を完了することができます。

- クラスターのスケーリング、コントロールプレーンのサイジング、および etcd のパフォーマンスおよびスケーラビリティーに関する理解がある。詳細は、クラスターのスケーリングに関する推奨プラクティス を参照してください。
- RHOSP でメタデータサービスが有効化されている。

5.2. ネットワークが制限された環境でのインストールについて

OpenShift Container Platform 4.19 では、ソフトウェアコンポーネントを取得するためにインターネットへのアクティブな接続を必要としないインストールを実行できます。ネットワークが制限された環境のインストールは、クラスターのインストール先となるクラウドプラットフォームに応じて、installer-provisioned infrastructure または user-provisioned infrastructure を使用して実行できます。

クラウドプラットフォーム上でネットワークが制限されたインストールの実行を選択した場合でも、そのクラウド API へのアクセスが必要になります。Amazon Web Service の Route 53 DNS や IAM サービスなどの一部のクラウド機能には、インターネットアクセスが必要です。ネットワークによっては、ベアメタルハードウェア、Nutanix、または VMware vSphere へのインストールに必要なインターネットアクセスが少なくて済む場合があります。

ネットワークが制限されたインストールを完了するには、OpenShift イメージレジストリーのコンテンツをミラーリングし、インストールメディアを含むレジストリーを作成する必要があります。このミラーは、インターネットと制限されたネットワークの両方にアクセスできるミラーホストで、または制限に対応する他の方法を使用して作成できます。

5.2.1. その他の制限

ネットワークが制限された環境のクラスターには、以下の追加の制限および制約があります。

- **ClusterVersion** ステータスには **Unable to retrieve available updates** エラーが含まれます。

- デフォルトでは、必要なイメージストリームタグにアクセスできないため、開発者カタログのコンテンツは使用できません。

5.3. OPENSHIFT CONTAINER PLATFORM を RHOSP にインストールするリソースのガイドライン

OpenShift Container Platform のインストールをサポートするために、Red Hat OpenStack Platform (RHOSP) クォータは以下の要件を満たす必要があります。

表5.1 RHOSP のデフォルトの OpenShift Container Platform クラスターに関する推奨リソース

リソース	値
Floating IP アドレス	3
ポート	15
ルーター	1
サブネット	1
RAM	88 GB
vCPU	22
ボリュームストレージ	275 GB
インスタンス	7
セキュリティーグループ	3
セキュリティーグループルール	60
サーバーグループ	2 - 各マシンプールの追加のアベイラビリティーゾーンごとに 1 つ追加

クラスターは推奨されるリソースよりもリソースが少ない場合にも機能する場合がありますが、その場合のパフォーマンスは保証されません。

重要

RHOSP オブジェクトストレージ (Swift) が利用可能で、**swiftoperator** ロールを持つユーザーアカウントによって操作されている場合、これは OpenShift Container Platform イメージレジストリーのデフォルトバックエンドとして使用されます。この場合、ボリュームストレージ要件は 175 GB です。Swift 領域要件は、イメージレジストリーのサイズによって異なります。

注記

デフォルトで、セキュリティーグループおよびセキュリティーグループルールのクォータは低く設定される可能性があります。問題が生じた場合には、管理者として **openstack quota set --secgroups 3 --secgroup-rules 60 <project>** を実行して値を増やします。

OpenShift Container Platform デプロイメントは、コントロールプレーンマシン、コンピュートマシン、およびブートストラップマシンで構成されます。

5.3.1. コントロールプレーンマシン

デフォルトでは、OpenShift Container Platform インストールプロセスは 3 つのコントロールプレーンマシンを作成します。

それぞれのマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 16 GB のメモリーと 4 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

5.3.2. コンピュートマシン

デフォルトでは、OpenShift Container Platform インストールプロセスは 3 つのコンピューティングマシンを作成します。

それぞれのマシンには以下が必要です。

- RHOSP クォータからのインスタンス
- RHOSP クォータからのポート
- 少なくとも 8 GB のメモリーと 2 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

ヒント

コンピュートマシンは、OpenShift Container Platform で実行されるアプリケーションをホストします。できるだけ多くのアプリケーションを実行することが意図されています。

5.3.3. ブートストラップマシン

インストール時に、ブートストラップマシンは一時的にプロビジョニングされ、コントロールプレーンを初期化します。実稼働環境用のコントロールプレーンの準備ができた後に、ブートストラップマシンのプロビジョニングは解除されます。

ブートストラップマシンには以下が必要です。

- RHOSP クォータからのインスタンス

- RHOSP クォータからのポート
- 少なくとも 16 GB のメモリーと 4 つの vCPU を備えたフレーバー
- RHOSP クォータから少なくとも 100 GB のストレージ容量

5.4. OPENSHIFT CONTAINER PLATFORM のインターネットアクセス

OpenShift Container Platform 4.19 では、クラスターのインストールに必要なイメージを取得するために、インターネットにアクセスする必要があります。

次のアクションを実行するには、インターネットにアクセスできる必要があります。

- OpenShift Cluster Manager にアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターがインターネットにアクセスでき、Telemetry を無効にしていない場合、そのサービスによってクラスターのサブスクリプションが自動的に有効化されます。
- クラスターのインストールに必要なパッケージを取得するために Quay.io にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。

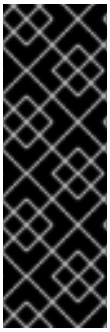
5.5. RHOSP での SWIFT の有効化

Swift は、**swiftoperator** ロールのあるユーザーアカウントによって操作されます。インストールプログラムを実行する前に、ロールをアカウントに追加します。

重要

一般に Swift と呼ばれる Red Hat OpenStack Platform (RHOSP) オブジェクトストレージサービス が利用可能な場合、OpenShift Container Platform はこれをイメージレジストリーストレージとして使用します。利用できない場合、インストールプログラムは Cinder として知られる RHOSP ブロックストレージサービスに依存します。

Swift が存在し、これを使用する必要がある場合は、Swift へのアクセスを有効にする必要があります。これが存在しない場合や使用する必要がない場合は、このセクションを省略してください。



重要

RHOSP 17 では、Ceph RGW の **rgw_max_attr_size** パラメーターが 256 文字に設定されます。この設定は、コンテナーイメージを OpenShift Container Platform レジストリーにアップロードする際に問題を引き起こします。**rgw_max_attr_size** の値は、1024 文字以上に設定する必要があります。

インストールする前に、RHOSP のデプロイメントがこの問題の影響を受けるかどうか確認してください。影響を受ける場合は、Ceph RGW を再設定します。

前提条件

- ターゲット環境に RHOSP 管理者アカウントがあります。
- Swift サービスがインストールされています。
- Ceph RGW で、**account in url** オプションが有効化されています。

手順

RHOSP 上で Swift を有効にするには、以下を実行します。

1. RHOSP CLI の管理者として、**swiftoperator** ロールを Swift にアクセスするアカウントに追加します。

```
$ openstack role add --user <user> --project <project> swiftoperator
```

RHOSP デプロイメントでは、イメージレジストリーに Swift を使用することができます。

5.6. インストールプログラムのパラメーターの定義

OpenShift Container Platform インストールプログラムは、**clouds.yaml** というファイルを使用します。このファイルは、プロジェクト名、ログイン情報、認可サービスの URL を含む Red Hat OpenStack Platform (RHOSP) 設定パラメーターを説明します。

手順

1. **clouds.yaml** ファイルを作成します。
   - RHOSP ディストリビューションに Horizon Web UI が含まれる場合には、そこに **clouds.yaml** ファイルを生成します。

重要

パスワードを必ず **auth** フィールドに追加してください。シークレットは、**clouds.yaml** の 別のファイル に保持できます。

   - RHOSP ディストリビューションに Horizon Web UI が含まれない場合や Horizon を使用する必要がない場合には、このファイルを独自に作成します。**clouds.yaml** の詳細は、RHOSP ドキュメントの Config files を参照してください。

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. RHOSP インストールでエンドポイント認証用に自己署名認証局 (CA) を使用する場合、以下を実行します。
   a. 認証局ファイルをマシンにコピーします。

b. **cacerts** キーを **clouds.yaml** ファイルに追加します。この値は、CA 証明書への絶対的な root 以外によるアクセスが可能なパスである必要があります。

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

ヒント

カスタム CA 証明書を使用してインストーラーを実行した後に、**cloud-provider-config** キーマップの **ca-cert.pem** キーの値を編集して証明書を更新できます。コマンドラインで、以下を実行します。

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. **clouds.yaml** ファイルを以下の場所のいずれかに置きます。

a. **OS_CLIENT_CONFIG_FILE** 環境変数の値

b. 現行ディレクトリー

c. Unix 固有のユーザー設定ディレクトリー (例: ~/**.config**/**openstack**/**clouds.yaml**)

d. Unix 固有のサイト設定ディレクトリー (例: /**etc**/**openstack**/**clouds.yaml**)
インストールプログラムはこの順序で **clouds.yaml** を検索します。

5.7. OPENSTACK CLOUD CONTROLLER MANAGER のオプション設定

オプションで、クラスターの OpenStack Cloud Controller Manager (CCM) 設定を編集できます。この設定は、OpenShift Container Platform が Red Hat OpenStack Platform (RHOSP) と対話する方法を制御します。

設定パラメーターの完全なリストは、「OpenStack のインストール」ドキュメントの「OpenStack Cloud Controller Manager リファレンスガイド」を参照してください。

手順

1. クラスター用に生成されたマニフェストファイルがない場合は、以下のコマンドを実行して生成します。

```
$ openshift-install --dir <destination_directory> create manifests
```

2. テキストエディターで、cloud-provider 設定マニフェストファイルを開きます。以下に例を示します。

```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. CCM リファレンスガイドに従ってオプションを変更します。
負荷分散のために Octavia を設定することが一般的です。以下に例を示します。

```
#...
```

```
[LoadBalancer]
lb-provider = "amphora" (1)
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" (2)
create-monitor = True (3)
monitor-delay = 10s (4)
monitor-timeout = 10s (5)
monitor-max-retries = 1 (6)
#...
```

(1) このプロパティーは、ロードバランサーが使用する Octavia プロバイダーを設定します。**"ovn"** または **"amphora"** を値として受け入れます。OVN の使用を選択する場合は、**lb-method** を **SOURCE_IP_PORT**。

(2) このプロパティーは、複数の外部ネットワークをクラスターで使用する場合に必要です。クラウドプロバイダーは、ここで指定するネットワーク上に Floating IP アドレスを作成します。

(3) このプロパティーは、クラウドプロバイダーが Octavia ロードバランサーのヘルスモニターを作成するかどうかを制御します。ヘルスモニターを作成するには、値を **True** に設定します。RHOSP 16.2 の時点で、この機能は Amphora プロバイダーでのみ利用できます。

(4) このプロパティーは、監視されるエンドポイントの頻度を設定します。値は **time.ParseDuration()** 形式である必要があります。このプロパティーは、**create-monitor** プロパティーの値が **True** の場合に必要です。

(5) このプロパティーは、タイムアウトする前に監視要求が開く時間を設定します。値は **time.ParseDuration()** 形式である必要があります。このプロパティーは、**create-monitor** プロパティーの値が **True** の場合に必要です。

(6) このプロパティーは、ロードバランサーがオンラインとしてマークされる前に必要なモニタリング要求の数を定義します。値は整数でなければなりません。このプロパティーは、**create-monitor** プロパティーの値が **True** の場合に必要です。

重要

変更を保存する前に、ファイルが正しく構造化されていることを確認します。プロパティーが適切なセクションに置かれていないと、クラスターが失敗することがあります。

重要

.spec.externalTrafficPolicy プロパティーの値が **Local** に設定されたサービスを使用する場合は、**create-monitor** プロパティーの値を **True** に設定する必要があります。RHOSP 16.2 の OVN Octavia プロバイダーは、ヘルスモニターをサポートしません。そのため、**lb-provider** の値が **"ovn"** に設定されている場合、**ETP** パラメーターの値が **Local** に設定されたサービスは応答しない可能性があります。

4. 変更をファイルに保存し、インストールを続行します。

ヒント

インストーラーの実行後に、クラウドプロバイダー設定を更新できます。コマンドラインで、以下を実行します。

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

変更を保存した後、クラスターの再設定には多少時間がかかります。ノードが **SchedulingDisabled** のままの場合は、プロセスが完了します。

5.8. ネットワークが制限されたインストール用の RHCOS イメージの作成

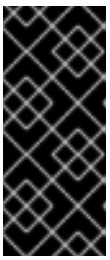
Red Hat Enterprise Linux CoreOS (RHCOS) イメージをダウンロードし、OpenShift Container Platform をネットワークが制限された Red Hat OpenStack Platform (RHOSP) 環境にインストールします。

前提条件

- OpenShift Container Platform インストールプログラムを取得します。ネットワークが制限されたインストールでは、プログラムはミラーレジストリースト上に置かれます。

手順

1. Red Hat カスタマーポータルの 製品ダウンロードページ にログインします。
2. **Version** で、RHEL 8 用の OpenShift Container Platform 4.19 の最新リリースを選択します。



重要

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。

3. **Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)**イメージをダウンロードします。
4. イメージを展開します。

注記

クラスターが使用する前にイメージを圧縮解除する必要があります。ダウンロードしたファイルの名前に、**.gz** または **.tgz** などの圧縮拡張子が含まれていない場合があります。ファイルを圧縮するか、どのように圧縮するかを確認するには、コマンドラインで以下を入力します。

```
$ file <name_of_downloaded_file>
```

5. 圧縮解除したイメージを、Glance などの bastion サーバーからアクセス可能な場所にアップロードします。以下に例を示します。

```
$ openstack image create --file rhcos-44.81.202003110027-0-openstack.x86_64.qcow2 --
```

```
disk-format qcow2 rhcos-${RHCOS_VERSION}
```

重要

RHOSP 環境によっては、**.raw** または **.qcow2** 形式 のいずれかでイメージをアップロードできる場合があります。Ceph を使用する場合は、**.raw** 形式を使用する必要があります。

警告

インストールプログラムが同じ名前を持つ複数のイメージを見つける場合、それらのイメージのいずれかがランダムに選択されます。この動作を回避するには、RHOSP でリソースの一意の名前を作成します。

これで、イメージが制限されたインストールで利用可能になります。OpenShift Container Platform デプロイメントで使用するイメージの名前または場所をメモします。

5.9. インストール設定ファイルの作成

Red Hat OpenStack Platform (RHOSP) にインストールする OpenShift Container Platform クラスターをカスタマイズできます。

前提条件

- OpenShift Container Platform インストールプログラムおよびクラスターのプルシークレットがある。ネットワークが制限されたインストールでは、これらのファイルがミラーホスト上に置かれます。
- ミラーレジストリーの作成時に生成された **imageContentSources** 値がある。
- ミラーレジストリーの証明書の内容を取得している。
- Red Hat Enterprise Linux CoreOS (RHCOS) イメージを取得し、アクセス可能な場所にアップロードしている。

手順

1. **install-config.yaml** ファイルを作成します。
   a. インストールプログラムが含まれるディレクトリーに切り替え、以下のコマンドを実行します。

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 **<installation_directory>** の場合、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

ディレクトリーを指定する場合:

- ディレクトリーに **execute** 権限があることを確認します。この権限は、インストールディレクトリーで Terraform バイナリーを実行するために必要です。
- 空のディレクトリーを使用します。ブートストラップ X.509 証明書などの一部のインストールアセットは有効期限が短いため、インストールディレクトリーを再利用しないでください。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してください。

b. プロンプト時に、クラウドの設定の詳細情報を指定します。

i. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

ii. ターゲットに設定するプラットフォームとして **openstack** を選択します。

iii. クラスターのインストールに使用する Red Hat OpenStack Platform (RHOSP) の外部ネットワーク名を指定します。

iv. OpenShift API への外部アクセスに使用する floating IP アドレスを指定します。

v. コントロールプレーンノードに使用する少なくとも 16 GB の RAM とコンピュートノードに使用する 8 GB の RAM を持つ RHOSP フレーバーを指定します。

vi. クラスターをデプロイするベースドメインを選択します。すべての DNS レコードはこのベースのサブドメインとなり、クラスター名も含まれます。

vii. クラスターの名前を入力します。名前は 14 文字以下でなければなりません。

2. **install-config.yaml** ファイルで、**platform.openstack.clusterOSImage** の値をイメージの場所または名前に設定します。以下に例を示します。

```
platform:
  openstack:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
openstack.x86_64.qcow2.gz?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. **install-config.yaml** ファイルを編集し、ネットワークが制限された環境でのインストールに必要な追加の情報を提供します。

a. **pullSecret** の値を更新して、レジストリーの認証情報を追加します。

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

<mirror_host_name> の場合、ミラーレジストリーの証明書で指定したレジストリードメイン名を指定し、**<credentials>** の場合は、ミラーレジストリーの base64 でエンコードされたユーザー名およびパスワードを指定します。

b. **additionalTrustBundle** パラメーターおよび値を追加します。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

この値は、ミラーレジストリーに使用した証明書ファイルの内容である必要があります。証明書ファイルは、既存の信頼できる認証局、またはミラーレジストリー用に生成した自己署名証明書のいずれかです。

c. 次の YAML の抜粋のようなイメージコンテンツリソースを追加します。

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

これらの値には、ミラーレジストリーの作成時に記録された **imageContentSources** を使用します。

d. 公開ストラテジーを **Internal** に設定します:

```
publish: Internal
```

このオプションを設定すると、内部 Ingress コントローラーおよびプライベートロードバランサーを作成します。

4. 必要な **install-config.yaml** ファイルに他の変更を加えます。
パラメーターの詳細は、「インストール設定パラメーター」を参照してください。

5. **install-config.yaml** ファイルをバックアップし、複数のクラスターをインストールするのに使用できるようにします。

重要

install-config.yaml ファイルはインストールプロセス時に使用されます。このファイルを再利用する必要がある場合は、この段階でこれをバックアップしてください。

関連情報

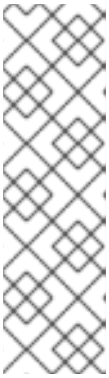
- OpenStack のインストール設定パラメーター

5.9.1. インストール時のクラスター全体のプロキシーの設定

実稼働環境では、インターネットへの直接アクセスを拒否し、代わりに HTTP または HTTPS プロキシーを使用することができます。プロキシー設定を **install-config.yaml** ファイルで行うことにより、新規の OpenShift Container Platform クラスターをプロキシーを使用するように設定できます。

前提条件

- 既存の **install-config.yaml** ファイルがある。
- クラスターがアクセスする必要のあるサイトを確認済みで、それらのいずれかがプロキシーをバイパスする必要があるかどうかを判別している。デフォルトで、すべてのクラスター Egress トラフィック (クラスターをホストするクラウドに関するクラウドプロバイダー API に対する呼び出しを含む) はプロキシーされます。プロキシーを必要に応じてバイパスするために、サイトを **Proxy** オブジェクトの **spec.noProxy** フィールドに追加している。



注記

Proxy オブジェクトの **status.noProxy** フィールドには、インストール設定の **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr**、および **networking.serviceNetwork[]** フィールドの値が設定されます。

Amazon Web Services (AWS)、Microsoft Azure、および Red Hat OpenStack Platform (RHOSP) へのインストールの場合、**Proxy** オブジェクトの **status.noProxy** フィールドには、インスタンスメタデータのエンドポイント (**169.254.169.254**) も設定されます。

手順

1. **install-config.yaml** ファイルを編集し、プロキシー設定を追加します。以下に例を示します。

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. クラスター外の HTTP 接続を作成するために使用するプロキシー URL。URL スキームは **http** である必要があります。
2. クラスター外で HTTPS 接続を作成するために使用するプロキシー URL。
3. プロキシーから除外するための宛先ドメイン名、IP アドレス、または他のネットワーク CIDR のコンマ区切りのリスト。サブドメインのみと一致するように、ドメインの前に **.** を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。***** を使用し、すべての宛先のプロキシーをバイパスします。
4. 指定されている場合、インストールプログラムは HTTPS 接続のプロキシーに必要な 1 つ以上の追加の CA 証明書が含まれる **user-ca-bundle** という名前の設定マップを **openshift-config** namespace に生成します。次に Cluster Network Operator は、これら

のコンテンツを Red Hat Enterprise Linux CoreOS (RHCOS) 信頼バンドルにマージする **trusted-ca-bundle** config map を作成し、この config map は **Proxy** オブジェクトの **trustedCA** フィールドで参照されます。**additionalTrustBundle** フィールドは、プロキシーのアイデンティティー証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。

5 オプション: **trustedCA** フィールドの **user-ca-bundle** 設定マップを参照する **Proxy** オブジェクトの設定を決定するポリシー。許可される値は **Proxyonly** および **Always** です。**Proxyonly** を使用して、**http/https** プロキシーが設定されている場合にのみ **user-ca-bundle** 設定マップを参照します。**Always** を使用して、常に **user-ca-bundle** 設定マップを参照します。デフォルト値は **Proxyonly** です。

注記

インストールプログラムは、プロキシーの **readinessEndpoints** フィールドをサポートしません。

注記

インストーラーがタイムアウトした場合は、インストーラーの **wait-for** コマンドを使用してデプロイメントを再起動してからデプロイメントを完了します。以下に例を示します。

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. ファイルを保存し、OpenShift Container Platform のインストール時にこれを参照します。

インストールプログラムは、指定の **install-config.yaml** ファイルのプロキシー設定を使用する **cluster** という名前のクラスター全体のプロキシーを作成します。プロキシー設定が指定されていない場合、**cluster Proxy** オブジェクトが依然として作成されますが、これには **spec** がありません。

注記

cluster という名前の **Proxy** オブジェクトのみがサポートされ、追加のプロキシーを作成することはできません。

5.9.2. 制限された OpenStack インストールのカスタマイズされた install-config.yaml ファイルのサンプル

このサンプル **install-config.yaml** は、すべての可能な Red Hat OpenStack Platform (RHOSP) カスタマイズオプションを示しています。

重要

このサンプルファイルは参照用にのみ提供されます。インストールプログラムを使用して **install-config.yaml** ファイルを取得する必要があります。

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
```

```
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    region: region1
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
additionalTrustBundle: |

  -----BEGIN CERTIFICATE-----

  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ

  -----END CERTIFICATE-----

imageContentSources:
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

5.10. クラスターノードの SSH アクセス用のキーペアの生成

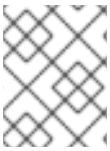
OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストールプログラムに指定できます。キーは、Ignition 設定ファイルを介して Red Hat Enterprise Linux CoreOS (RHCOS) ノードに渡され、ノードへの SSH アクセスを認証するために使用されます。このキーは各ノードの **core** ユーザーの ~/**.ssh/authorized_keys** リストに追加され、パスワードなしの認証が可能になります。

キーがノードに渡されると、キーペアを使用して RHCOS ノードにユーザー **core** として SSH を実行できます。SSH 経由でノードにアクセスするには、秘密鍵のアイデンティティーをローカルユーザーの SSH で管理する必要があります。

インストールのデバッグまたは障害復旧を実行するためにクラスターノードに対して SSH を実行する場合は、インストールプロセスの間に SSH 公開鍵を指定する必要があります。**./openshift-install gather** コマンドでは、SSH 公開鍵がクラスターノードに配置されている必要もあります。

重要

障害復旧およびデバッグが必要な実稼働環境では、この手順を省略しないでください。



注記

プラットフォーム固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. クラスターノードへの認証に使用するローカルマシンに既存の SSH キーペアがない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 新しい SSH キーのパスとファイル名 (~/**.ssh/id_ed25519** など) を指定します。既存のキーペアがある場合は、公開鍵が ~/**.ssh** ディレクトリーにあることを確認します。

注記

x86_64、**ppc64le**、および **s390x** アーキテクチャーのみで FIPS 140-2/140-3 検証のために NIST に提出された RHEL 暗号化ライブラリーを使用する OpenShift Container Platform クラスターをインストールする予定がある場合は、**ed25519** アルゴリズムを使用するキーを作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

2. 公開 SSH キーを表示します。

```
$ cat <path>/<file_name>.pub
```

たとえば、次のコマンドを実行して ~/**.ssh/id_ed25519.pub** 公開鍵を表示します。

```
$ cat ~/.ssh/id_ed25519.pub
```

3. ローカルユーザーの SSH エージェントに SSH 秘密鍵 ID が追加されていない場合は、それを追加します。キーの SSH エージェント管理は、クラスターノードへのパスワードなしの SSH 認証、または **./openshift-install gather** コマンドを使用する場合は必要になります。

注記

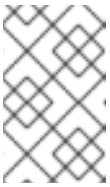
一部のディストリビューションでは、~/**.ssh/id_rsa** および ~/**.ssh/id_dsa** などのデフォルトの SSH 秘密鍵のアイデンティティーは自動的に管理されます。

a. **ssh-agent** プロセスがローカルユーザーに対して実行されていない場合は、バックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"
```

出力例

```
Agent pid 31874
```



注記

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

4. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> 1
```

1 ~/**.ssh/id_ed25519** などの、SSH プライベートキーのパスおよびファイル名を指定します。

出力例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

次のステップ

- OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストールプログラムに指定します。

5.11. 環境へのアクセスの有効化

デプロイ時に、OpenShift Container Platform マシンはすべて Red Hat OpenStack Platform (RHOSP) テナントネットワークに作成されます。したがって、ほとんどの RHOSP デプロイメントでは直接アクセスできません。

インストール時に Floating IP アドレス (FIP) を使用して OpenShift Container Platform API およびアプリケーションのアクセスを設定できます。FIP を設定せずにインストールを完了することもできますが、インストーラーは API またはアプリケーションを外部からアクセスする方法を設定しません。

5.11.1. floating IP アドレスを使用したアクセスの有効化

OpenShift Container Platform API およびクラスターアプリケーションへの外部アクセス用に Floating IP (FIP) アドレスを作成します。

手順

1. Red Hat OpenStack Platform (RHOSP) CLI を使用して、API FIP を作成します。

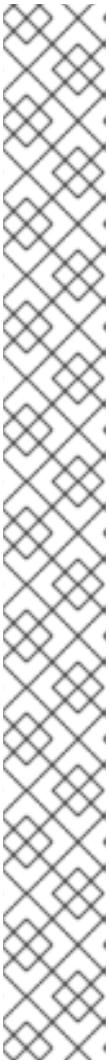
```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. Red Hat OpenStack Platform (RHOSP) CLI を使用して、apps (アプリ)、または Ingress、FIP を作成します。

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. API および Ingress FIP の DNS サーバーに、これらのパターンに準拠するレコードを追加します。

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```



注記

DNS サーバーを制御していない場合は、次のようなクラスタードメイン名を **/etc/hosts** ファイルに追加することで、クラスターにアクセスできます。

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts ファイル内のクラスタードメイン名により、クラスターの Web コンソールおよび監視インターフェイスへのローカルアクセスが許可されます。**kubectl** または **oc** を使用することもできます。<application_floating_ip> を指す追加のエントリーを使用して、ユーザーアプリケーションにアクセスできます。このアクションにより、API およびアプリケーションは他のユーザーがアクセスできない状態になり、この状態は実稼働デプロイメントには適していませんが、開発およびテスト目的のインストールが可能になります。

4. FIP を、以下のパラメーターの値として **install-config.yaml** ファイルに追加します。
   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.apiFloatingIP**

これらの値を使用する場合には、**install-config.yaml** ファイルの **platform.openstack.externalNetwork** パラメーターの値として外部ネットワークを入力する必要もあります。

ヒント

Floating IP アドレスを割り当て、ファイアウォール設定を更新することで、OpenShift Container Platform リソースがクラスター外で利用できる状態にすることができます。

5.11.2. Floating IP アドレスなしでのインストールの完了

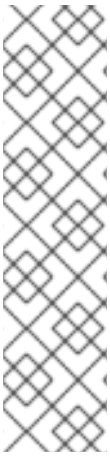
Floating IP アドレスを指定せずに OpenShift Container Platform を Red Hat OpenStack Platform (RHOSP) にインストールすることができます。

install-config.yaml ファイルで以下のパラメーターを定義しないでください。

- **platform.openstack.ingressFloatingIP**
- **platform.openstack.apiFloatingIP**

外部ネットワークを提供できない場合は、**platform.openstack.externalNetwork** を空白のままにすることもできます。**platform.openstack.externalNetwork** の値を指定しない場合はルーターが作成されず、追加のアクションがない場合は、インストーラーは Glance からのイメージの取得に失敗します。外部接続を独自に設定する必要があります。

Floating IP アドレスまたは名前解決がないために、クラスター API に到達できないシステムからインストーラーを実行すると、インストールに失敗します。このような場合にインストールが失敗するのを防ぐために、プロキシーネットワークを使用するか、マシンと同じネットワークにあるシステムからインストーラーを実行できます。



注記

API および Ingress ポートの DNS レコードを作成して、名前解決を有効にできます。以下に例を示します。

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

DNS サーバーを制御しない場合は、**/etc/hosts** ファイルにレコードを追加できます。このアクションにより、API は他者のアクセスできない状態になり、この状態は実稼働デプロイメントには適していませんが、開発およびテスト目的のインストールが可能になります。

5.12. クラスターのデプロイ

互換性のあるクラウドプラットフォームに OpenShift Container Platform をインストールできます。

重要

インストールプログラムの **create cluster** コマンドは、初期インストール時に 1 回だけ実行できます。

前提条件

- OpenShift Container Platform インストールプログラムおよびクラスターのプルシークレットがある。
- ホスト上のクラウドプロバイダーアカウントに、クラスターをデプロイするための適切な権限があることを確認している。アカウントの権限が正しくないと、インストールプロセスが失敗し、不足している権限を示すエラーメッセージが表示されます。

手順

- インストールプログラムが格納されているディレクトリーで、次のコマンドを実行して、クラスターのデプロイメントを初期化します。

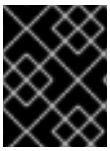
```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. **<installation_directory>** に、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。
2. 異なるインストールの詳細情報を表示するには、**info** ではなく、**warn**、**debug**、または **error** を指定します。

検証

クラスターのデプロイが正常に完了すると、次のようになります。

- ターミナルには、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報など、クラスターにアクセスするための指示が表示されます。
- 認証情報は **<installation_directory>/.openshift_install.log** にも出力されます。



重要

インストールプログラム、またはインストールプログラムが作成するファイルを削除することはできません。これらはいずれもクラスターを削除するために必要になります。

出力例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになり、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、**コントロールプレーン証明書の期限切れの状態からのリカバリー** に関するドキュメントを参照してください。
- 24 時間証明書はクラスターのインストール後 16 時間から 22 時間にローテーションするため、Ignition 設定ファイルは、生成後 12 時間以内に使用することを推奨します。12 時間以内に Ignition 設定ファイルを使用することにより、インストール中に証明書の更新が実行された場合のインストールの失敗を回避できます。

5.13. クラスターステータスの確認

インストール時またはインストール後に OpenShift Container Platform クラスターのステータスを確認することができます。

手順

1. クラスター環境で、管理者の kubeconfig ファイルをエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。

kubeconfig ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターに関する情報が含まれます。

2. デプロイメント後に作成されたコントロールプレーンおよびコンピュートマシンを表示します。

```
$ oc get nodes
```

3. クラスターのバージョンを表示します。

```
$ oc get clusterversion
```

4. Operator のステータスを表示します。

```
$ oc get clusteroperator
```

5. クラスター内のすべての実行中の Pod を表示します。

```
$ oc get pods -A
```

5.14. CLI の使用によるクラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターに関する情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターをデプロイしていること。
- OpenShift CLI (**oc**) がインストールされている。

手順

1. 次のコマンドを実行して、**kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。

2. 次のコマンドを実行し、エクスポートされた設定を使用して **oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

関連情報

- OpenShift Container Platform Web コンソールへのアクセスと理解に関する詳細は、Web コンソールへのアクセス を参照してください。

5.15. デフォルトの OPERATORHUB カタログソースの無効化

Red Hat によって提供されるコンテンツを調達する Operator カタログおよびコミュニティープロジェクトは、OpenShift Container Platform のインストール時にデフォルトで OperatorHub に設定されます。ネットワークが制限された環境では、クラスター管理者としてデフォルトのカタログを無効にする必要があります。

手順

- **disableAllDefaultSources: true** を **OperatorHub** オブジェクトに追加して、デフォルトカタログのソースを無効にします。

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

ヒント

または、Web コンソールを使用してカタログソースを管理できます。**Administration → Cluster Settings → Configuration → OperatorHub** ページから、**Sources** タブをクリックして、個別のソースを作成、更新、削除、無効化、有効化できます。

5.16. OPENSHIFT CONTAINER PLATFORM の TELEMETRY アクセス

OpenShift Container Platform 4.19 では、Telemetry サービスにもインターネットアクセスが必要です。Telemetry サービスは、クラスターの健全性と更新の成功に関するメトリクスを提供するためにデフォルトで実行されます。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは OpenShift Cluster Manager に登録されます。

OpenShift Cluster Manager インベントリーが正常である (Telemetry によって自動的に維持、または OpenShift Cluster Manager を使用して手動で維持) ことを確認した後に、subscription watch を使用 して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

関連情報

- Telemetry サービスの詳細は、リモートヘルスモニタリングについて を参照してください。

5.17. 次のステップ

- クラスターのカスタマイズ
- クラスターのインストールに使用したミラーレジストリーに信頼された CA がある場合は、追加のトラストストアを設定 してクラスターに追加します。
- 必要に応じて、リモートヘルスレポート を作成できます。
- 必要に応じて、非接続クラスターの登録 を参照してください。
- Cluster Samples Operator および **must-gather** ツールの イメージストリームを設定 します。
- 非接続環境で Operator Lifecycle Manager を使用 する方法を確認します。
- Floating IP アドレス上でアプリケーショントラフィックを受け入れるように RHOSP を設定していない場合は、Floating IP アドレスを使用して RHOSP アクセスを設定 します。

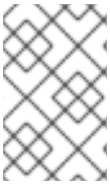
第6章 OPENSTACK に 3 ノードクラスターをインストールする

OpenShift Container Platform バージョン 4.19 では、Red Hat OpenStack Platform (RHOSP) に 3 ノードクラスターをインストールできます。3 ノードクラスターは、コンピュートマシンとしても機能する 3 つのコントロールプレーンマシンで構成されます。このタイプのクラスターは、クラスター管理者および開発者がテスト、開発、および実稼働に使用するためのより小さくリソース効率の高いクラスターを提供します。

3 ノードクラスターは installer-provisioned infrastructure にのみインストールできます。

6.1. 3 ノードクラスターの設定

クラスターをデプロイする前に、**install-config.yaml** ファイルでワーカーノードの数を **0** に設定して、3 ノードクラスターを設定します。ワーカーノードの数を **0** に設定すると、コントロールプレーンマシンがスケジュール可能になります。これにより、アプリケーションワークロードをコントロールプレーンノードから実行するようにスケジュールできます。



注記

アプリケーションワークロードはコントロールプレーンノードから実行され、コントロールプレーンノードはコンピュートノードと見なされるため、追加のサブスクリプションが必要です。

前提条件

- 既存の **install-config.yaml** ファイルがある。

手順

- 次の **compute** スタンザに示すように、**install-config.yaml** ファイルでコンピューティングレプリカの数を **0** に設定します。

  3 ノードクラスターの install-config.yaml ファイルの例

```
apiVersion: v1
baseDomain: example.com
compute:
- name: worker
  platform: {}
  replicas: 0
# ...
```

6.2. 次のステップ

- カスタマイズによる OpenStack へのクラスターのインストール

第7章 OPENSTACK をインストールした後のネットワーク設定の設定

インストール後に、Red Hat OpenStack Platform (RHOSP) クラスター上の OpenShift Container Platform のネットワーク設定を設定できます。

7.1. FLOATING IP アドレスを使用したアプリケーションアクセスの設定

OpenShift Container Platform をインストールした後に、アプリケーションネットワークトラフィックを許可するように Red Hat OpenStack Platform (RHOSP) を設定します。

注記

インストール中に、**install-config.yaml** ファイルの **platform.openstack.apiFloatingIP** および **platform.openstack.ingressFloatingIP** に値を指定した場合、または **inventory.yaml** Playbook の **os_api_fip** および **os_ingress_fip** に値を指定した場合は、この手順を実行する必要はありません。Floating IP アドレスはすでに設定されています。

前提条件

- OpenShift Container Platform クラスターがインストールされている必要があります。
- OpenShift Container Platform の RHOSP へのインストールに関するドキュメントで説明されているように、Floating IP アドレスが有効にされます。

手順

OpenShift Container Platform クラスターをインストールした後に、Floating IP アドレスを Ingress ポートに割り当てます。

1. ポートを表示します。

```
$ openstack port show <cluster_name>-<cluster_ID>-ingress-port
```

2. ポートを IP アドレスに接続します。

```
$ openstack floating ip set --port <ingress_port_ID> <apps_FIP>
```

3. ***apps.** のワイルドカード **A** レコードを DNS ファイルに追加します。

```
*.apps.<cluster_name>.<base_domain>  IN  A  <apps_FIP>
```

注記

DNS サーバーを制御せず、非実稼働環境でアプリケーションアクセスを有効にする必要がある場合は、これらのホスト名を **/etc/hosts** に追加できます。

```
<apps_FIP> console-openshift-console.apps.<cluster name>.<base domain>
<apps_FIP> integrated-oauth-server-openshift-authentication.apps.<cluster name>.
<base domain>
<apps_FIP> oauth-openshift.apps.<cluster name>.<base domain>
<apps_FIP> prometheus-k8s-openshift-monitoring.apps.<cluster name>.<base
domain>
<apps_FIP> <app name>.apps.<cluster name>.<base domain>
```

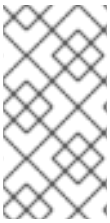
7.2. OVS ハードウェアオフロードの有効化

Red Hat OpenStack Platform (RHOSP) で実行されるクラスターの場合、Open vSwitch(OVS) ハードウェアオフロードを有効にすることができます。

OVS は、大規模なマルチサーバーネットワークの仮想化を可能にするマルチレイヤー仮想スイッチです。

前提条件

- Single-root Input/Output Virtualization (SR-IOV) 用に設定された RHOSP にクラスターをインストールしている。
- SR-IOV Network Operator がクラスターにインストールされている。
- クラスターに 2 つの **hw-offload** タイプの Virtual Function (VF) インターフェイスを作成している。



注記

アプリケーション層のゲートウェイフローは、OpenShift Container Platform バージョン 4.10、4.11、および 4.12 では機能しません。また、OpenShift Container Platform バージョン 4.13 のアプリケーション層のゲートウェイフローをオフロードすることはできません。

手順

1. クラスターにある 2 つの **hw-offload** タイプの VF インターフェイスの **SriovNetworkNodePolicy** ポリシーを作成します。

   1 番目の Virtual Function インターフェイス

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy (1)
metadata:
  name: "hwoffload9"
  namespace: openshift-sriov-network-operator
spec:
  deviceType: netdevice
  isRdma: true
  nicSelector:
```

```
    pfNames: 2
    - ens6
  nodeSelector:
    feature.node.kubernetes.io/network-sriov.capable: 'true'
  numVfs: 1
  priority: 99
  resourceName: "hwoffload9"
```

1 **SriovNetworkNodePolicy** の値をここに挿入します。

2 どちらのインターフェイスにも Physical Function (PF) 名が含まれている必要があります。

2 番目の Virtual Function インターフェイス

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy 1
metadata:
  name: "hwoffload10"
  namespace: openshift-sriov-network-operator
spec:
  deviceType: netdevice
  isRdma: true
  nicSelector:
    pfNames: 2
    - ens5
  nodeSelector:
    feature.node.kubernetes.io/network-sriov.capable: 'true'
  numVfs: 1
  priority: 99
  resourceName: "hwoffload10"
```

1 **SriovNetworkNodePolicy** の値をここに挿入します。

2 どちらのインターフェイスにも Physical Function (PF) 名が含まれている必要があります。

2. 2 つのインターフェイス用に **NetworkAttachmentDefinition** リソースを作成します。

1 番目のインターフェイス用 NetworkAttachmentDefinition リソース

```
apiVersion: k8s.cni.cncf.io/v1
kind: NetworkAttachmentDefinition
metadata:
  annotations:
    k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload9
  name: hwoffload9
  namespace: default
spec:
    config: '{ "cniVersion":"0.3.1", "name":"hwoffload9","type":"host-device","device":"ens6"
    }'
```

2 番目のインターフェイス用 NetworkAttachmentDefinition リソース

```
apiVersion: k8s.cni.cncf.io/v1
kind: NetworkAttachmentDefinition
metadata:
  annotations:
    k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload10
  name: hwoffload10
  namespace: default
spec:
    config: '{ "cniVersion":"0.3.1", "name":"hwoffload10","type":"host-device","device":"ens5"
    }'
```

3. Pod で作成したインターフェイスを使用します。以下に例を示します。

2 つの OVS オフロードインターフェイスを使用する Pod

```
apiVersion: v1
kind: Pod
metadata:
  name: dpdk-testpmd
  namespace: default
  annotations:
    irq-load-balancing.crio.io: disable
    cpu-quota.crio.io: disable
    k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload9
    k8s.v1.cni.cncf.io/resourceName: openshift.io/hwoffload10
spec:
  restartPolicy: Never
  containers:
  - name: dpdk-testpmd
    image: quay.io/krister/centos8_nfv-container-dpdk-testpmd:latest
```

7.3. OVS ハードウェアオフロードネットワークの接続

Open vSwitch (OVS) ハードウェアオフロードネットワークをクラスターに接続できます。

前提条件

- クラスターがインストールされ、実行されている。
- クラスターで使用するために、Red Hat OpenStack Platform (RHOSP) で OVS ハードウェアオフロードネットワークをプロビジョニングしている。

手順

1. 次のテンプレートから **network.yaml** という名前のファイルを作成します。

```
spec:
  additionalNetworks:
  - name: hwoffload1
    namespace: cnf
```

```
    rawCNIConfig: '{ "cniVersion": "0.3.1", "name": "hwoffload1", "type": "host-
device","pciBusId": "0000:00:05.0", "ipam": {}}' 1
    type: Raw
```

ここでは、以下のようになります。

pciBusId

オフロードネットワークに接続されているデバイスを指定します。この値がわからない場合は、次のコマンドを実行してこの値を見つけることができます。

```
$ oc describe SriovNetworkNodeState -n openshift-sriov-network-operator
```

2. コマンドラインから次のコマンドを入力して、ファイルを使用してクラスターにパッチを適用します。

```
$ oc apply -f network.yaml
```

7.4. RHOSP で POD への IPV6 接続を有効にする

異なるノード上にある追加のネットワークを持つ Pod 間の IPv6 接続を有効にするには、サーバーの IPv6 ポートのポートセキュリティーを無効にします。ポートセキュリティーを無効にすると、Pod に割り当てられた IPv6 アドレスごとに許可されたアドレスペアを作成する必要がなくなり、セキュリティーグループのトラフィックが有効になります。

重要

次の IPv6 追加ネットワーク設定のみがサポートされています。

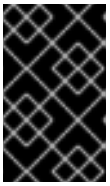
- SLAAC とホストデバイス
- SLAAC と MACVLAN
- DHCP ステートレスおよびホストデバイス
- DHCP ステートレスおよび MACVLAN

手順

- コマンドラインで、次のコマンドを入力します。

```
$ openstack port set --no-security-group --disable-port-security <compute_ipv6_port> 1
```

1 1 コンピュートサーバーの IPv6 ポートを指定します。



重要

このコマンドは、ポートからセキュリティーグループを削除し、ポートセキュリティーを無効にします。トラフィックの制限は、ポートから完全に削除されます。

7.5. RHOSP で IPV6 接続を持つ POD の作成

Pod の IPv6 接続を有効にして Pod に追加したら、セカンダリー IPv6 接続を持つ Pod を作成します。

手順

1. IPv6 namespace とアノテーション **k8s.v1.cni.cncf.io/networks: <additional_network_name>** を使用する Pod を定義します。ここで、**<additional_network_name** は追加のネットワークの名前になります。たとえば、**Deployment** オブジェクトの一環として、以下を行います。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-openshift
  namespace: ipv6
spec:
  affinity:
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        - labelSelector:
            matchExpressions:
            - key: app
              operator: In
              values:
              - hello-openshift
  replicas: 2
  selector:
    matchLabels:
      app: hello-openshift
  template:
    metadata:
      labels:
        app: hello-openshift
      annotations:
        k8s.v1.cni.cncf.io/networks: ipv6
    spec:
      securityContext:
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      containers:
      - name: hello-openshift
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
            - ALL
        image: quay.io/openshift/origin-hello-openshift
        ports:
        - containerPort: 8080
```

2. Pod を作成します。たとえば、コマンドラインで次のコマンドを入力します。

```
$ oc create -f <ipv6_enabled_resource> 1
```

1 リソース定義を含むファイルを指定します。

7.6. RHOSP 上の POD への IPV6 接続の追加

Pod で IPv6 接続を有効にしたら、Container Network Interface (CNI) 設定を使用して Pod に接続を追加します。

手順

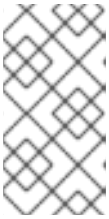
1. Cluster Network Operator (CNO) を編集するには、次のコマンドを入力します。

```
$ oc edit networks.operator.openshift.io cluster
```

2. **spec** フィールドで CNI 設定を指定します。たとえば、次の設定では、MACVLAN で SLAAC アドレスモードを使用します。

```
...
spec:
  additionalNetworks:
  - name: ipv6
    namespace: ipv6 1
    rawCNIConfig: '{ "cniVersion": "0.3.1", "name": "ipv6", "type": "macvlan", "master": "ens4"}'
2
    type: Raw
```

1. 必ず同じ namespace に Pod を作成してください。
2. より多くのネットワークが設定されている場合、または別のカーネルドライバーが使用されている場合、ネットワークアタッチメントの **"master"** フィールドのインターフェイスは **"ens4"** とは異なる場合があります。



注記

ステートフルアドレスモードを使用している場合は、CNI 設定に IP アドレス管理 (IPAM) を含めます。

Multus は DHCPv6 をサポートしていません。

3. 変更を保存し、テキストエディターを終了して、変更をコミットします。

検証

- コマンドラインで、次のコマンドを入力します。

```
$ oc get network-attachment-definitions -A
```

出力例

```
NAMESPACE      NAME          AGE
ipv6           ipv6          21h
```

セカンダリー IPv6 接続を持つ Pod を作成できるようになりました。

第8章 OPENSTACK CLOUD CONTROLLER MANAGER リファレンスガイド

8.1. OPENSTACK CLOUD CONTROLLER MANAGER

OpenShift Container Platform 4.12 以降、Red Hat OpenStack Platform (RHOSP) で実行されるクラスターは、従来の OpenStack クラウドプロバイダーから外部の OpenStack Cloud Controller Manager (CCM) に切り替えられました。これは、Kubernetes がツリー内の従来のクラウドプロバイダーから、Cloud Controller Manager を使用して実装される外部クラウドプロバイダーに移行したことに伴う変更です。

従来のクラウドプロバイダーのユーザー定義の設定を保持するために、移行プロセスの一環として、既存の設定が新しい設定にマップされます。**openshift-config** namespace で **cloud-provider-config** と呼ばれる設定を検索します。

注記

設定マップ名 **cloud-provider-config** は静的に設定されていません。これは、**infrastructure/cluster** CRD の **spec.cloudConfig.name** 値から派生します。

見つかった設定は、**openshift-cloud-controller-manager** namespace の **cloud-conf** config map に同期されます。

この同期の一環として、OpenStack CCM Operator は新しい config map を変更して、そのプロパティーが外部クラウドプロバイダーと互換性を持つようにします。ファイルは次の方法で変更されます。

- **[Global] secret-name**、**[Global] secret-namespace**、**[Global] kubeconfig-path** オプションは削除されました。これらは、外部のクラウドプロバイダーには適用されません。
- **[Global] use-clouds**、**[Global] clouds-file**、**[Global] cloud** オプションが追加されました。
- **[BlockStorage]** セクション全体が削除されます。外部のクラウドプロバイダーは、ストレージ操作を実行しなくなりました。ブロックストレージの設定は、Cinder CSI ドライバーによって管理されます。

さらに、CCM Operator は多くのデフォルトオプションを適用します。これらのオプションの値は、次のように常にオーバーライドされます。

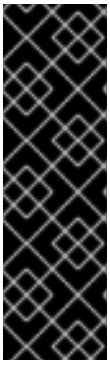
```
[Global]
use-clouds = true
clouds-file = /etc/openstack/secret/clouds.yaml
cloud = openstack
...

[LoadBalancer]
enabled = true
```

clouds-value 値 (**/etc/openstack/secret/clouds.yaml**) は、**openshift-cloud-controller-manager** namespace の **openstack-cloud-credentials** 設定にマッピングされます。他の **clouds.yaml** ファイルと同様に、このファイルで RHOSP クラウドを変更できます。

8.2. OPENSTACK CLOUD CONTROLLER MANAGER (CCM) 設定マップ

OpenStack CCM config map は、クラスターが RHOSP クラウドと対話する方法を定義します。デフォルトでは、この設定は **openshift-cloud-controller-manager** namespace にある **cloud-conf** config map の **cloud.conf** キーの下に保存されます。



重要

cloud-conf config map は、**openshift-config** namespace の **cloud-provider-config** config map から生成されます。

cloud-conf config map で記述されている設定を変更するには、**cloud-provider-config** config map を変更します。

この同期の一環として、CCM Operator はいくつかのオプションをオーバーライドします。詳細は、「RHOSP Cloud Controller Manager」を参照してください。

以下に例を示します。

cloud-conf config map の例

```
apiVersion: v1
data:
  cloud.conf: |
    [Global] 1
    secret-name = openstack-credentials
    secret-namespace = kube-system
    region = regionOne
    [LoadBalancer]
    enabled = True
kind: ConfigMap
metadata:
  creationTimestamp: "2022-12-20T17:01:08Z"
  name: cloud-conf
  namespace: openshift-cloud-controller-manager
  resourceVersion: "2519"
  uid: cbbeedaf-41ed-41c2-9f37-4885732d3677
```

1 config map を変更するのではなく、**clouds.yaml** ファイルを使用してグローバルオプションを設定します。

次のオプションが config map に存在します。特に明記されている場合を除き、RHOSP で実行されるクラスターには必須です。

8.2.1. ロードバランサー (オプション)

CCM は、Octavia を使用するデプロイメント用にいくつかのロードバランサーオプションをサポートしています。

注記

Neutron-LBaaS のサポートは非推奨です。

オプション	説明
enabled	**LoadBalancer** タイプのサービス統合を有効にするかどうか。デフォルト値は **true** です。
floating-network-id	オプション: ロードバランサーの仮想 IP アドレス (VIP) のフローティング IP アドレスを作成するために使用される外部ネットワーク。クラウドに複数の外部ネットワークがある場合、このオプションを設定するか、ユーザーがサービスアノテーションで **loadbalancer.openstack.org/floating-network-id** を指定する必要があります。
floating-subnet-id	オプション: ロードバランサー VIP のフローティング IP アドレスを作成するために使用される外部ネットワークサブネット。サービスアノテーション **loadbalancer.openstack.org/floating-subnet-id** でオーバーライドできます。
floating-subnet	オプション: ロードバランサー VIP のフローティング IP アドレスを作成するために使用される外部ネットワークサブネットの名前パターン (~ で始まる場合はグロブまたは正規表現)。サービスアノテーション **loadbalancer.openstack.org/floating-subnet** でオーバーライドできます。複数のサブネットがパターンに一致する場合、使用可能な IP アドレスを持つ最初のサブネットが使用されます。
floating-subnet-tags	オプション: ロードバランサー VIP のフローティング IP アドレスを作成するために使用される外部ネットワークサブネットのタグ。サービスアノテーション **loadbalancer.openstack.org/floating-subnet-tags** でオーバーライドできます。複数のサブネットがこれらのタグに一致する場合、使用可能な IP アドレスを持つ最初のサブネットが使用されます。 RHOSP ネットワークが共有を無効にして設定されている場合 (たとえば、作成時に **--no-share** フラグが使用されている場合)、このオプションはサポートされません。このオプションを使用するには、共有するネットワークを設定します。

オプション	説明
lb-method	ロードバランサープールの作成に使用される負荷分散アルゴリズム。Amphora プロバイダーの場合、値は **ROUND_ROBIN**、**LEAST_CONNECTIONS**、または **SOURCE_IP** です。デフォルト値は **ROUND_ROBIN** です。 OVN プロバイダーでは、**SOURCE_IP_PORT** アルゴリズムのみがサポートされています。 Amphora プロバイダの場合、**LEAST_CONNECTIONS** または **SOURCE_IP** メソッドを使用する場合、**openshift-config** namespace の **cloud-provider-config** config map で **create-monitor** オプションを **true** として設定します。また、ロードバランサータイプサービスの **ETP:Local** で、クライアントからサービスのエンドポイント接続でバランスアルゴリズムの実行ができるよう設定します。
lb-provider	オプション: **amphora** や **octavia** など、ロードバランサーのプロバイダーを指定するために使用されます。Amphora および Octavia プロバイダーのみがサポートされています。
lb-version	オプション: ロードバランサー API のバージョン。**"v2"** のみがサポートされています。
subnet-id	ロードバランサーの仮想 IP が作成される Networking サービスのサブネットの ID。デュアルスタックデプロイメントの場合は、このオプションを未設定のままにしておきます。OpenStack クラウドプロバイダーによって、ロードバランサーに使用するサブネットが自動的に選択されます。
network-id	ロードバランサーの仮想 IP が作成される Networking サービスのネットワークの ID。**subnet-id** が設定されている場合は不要です。このプロパティーが設定されていない場合、ネットワークはクラスターノードが使用するネットワークに基づいて自動的に選択されます。
create-monitor	サービスロードバランサーのヘルスモニターを作成するかどうか。**externalTrafficPolicy: Local** を宣言するサービスには、ヘルスモニターが必要です。デフォルト値は **false** です。 **ovn** プロバイダーでバージョン 17 より前の RHOSP を使用する場合、このオプションはサポートされません。

オプション	説明
monitor-delay	プローブがロードバランサーのメンバーに送信される間隔 (秒単位)。デフォルト値は **5** です。
monitor-max-retries	ロードバランサーメンバーの動作ステータスを **ONLINE** に変更するために必要なチェックの成功回数。有効な範囲は **1** - **10** で、デフォルト値は **1** です。
monitor-timeout	モニターがタイムアウトする前にバックエンドへの接続を待機する時間 (秒単位)。デフォルト値は **3** です。
internal-lb	フローティング IP アドレスなしで内部ロードバランサーを作成するかどうか。デフォルト値は **false** です。
LoadBalancerClass "ClassName"	これは、一連のオプションで構成される設定セクションです。 • **floating-network-id** • **floating-subnet-id** • **floating-subnet** • **floating-subnet-tags** • **network-id** • **subnet-id** これらのオプションの動作は、CCM 設定ファイルのロードバランサーセクションにある同じ名前のオプションの動作と同じです。 サービスアノテーション **loadbalancer.openstack.org/class** を指定することで **ClassName** 値を設定できます。
max-shared-lb	ロードバランサーを共有できるサービスの最大数。デフォルト値は **2** です。

8.2.2. Operator がオーバーライドするオプション

CCM Operator は、RHOSP の設定で認識できる次のオプションをオーバーライドします。自分で設定しないでください。これらは、情報提供のみを目的としてこのドキュメントに含まれています。

オプション	説明

オプション	説明
auth-url	RHOSP ID サービスの URL。たとえば、**http://128.110.154.166/identity** です。
os-endpoint-type	サービスカタログから使用するエンドポイントのタイプ。
username	Identity サービスのユーザー名。
password	Identity サービスのユーザーパスワード。
domain-id	Identity サービスのユーザードメイン ID。
domain-name	Identity サービスのユーザードメイン名。
tenant-id	Identity サービスのプロジェクト ID。Identity サービスアプリケーションの認証情報を使用している場合は、このオプションを未設定のままにします。 識別子 **tenant** を **project** に変更したバージョン 3 の Identity API では、**tenant-id** の値が API の project コンストラクトに自動的にマップされます。
tenant-name	Identity サービスのプロジェクト名。
tenant-domain-id	Identity サービスプロジェクトのドメイン ID。
tenant-domain-name	Identity サービスプロジェクトのドメイン名。
user-domain-id	Identity サービスのユーザードメイン ID。
user-domain-name	Identity サービスのユーザードメイン名。

オプション	説明
use-clouds	**clouds.yaml** ファイルから認証認証情報をフェッチするかどうか。このセクションで設定されたオプションは、**clouds.yaml** ファイルから読み取られた値よりも優先されます。 CCM は、次の場所でファイルを検索します。 1. **clouds-file** オプションの値。 2. 環境変数 **OS_CLIENT_CONFIG_FILE** に格納されているファイルパス。 3. ディレクトリー **pkg/openstack**。 4. ディレクトリー **~/.config/openstack**。 5. ディレクトリー **/etc/openstack**。
clouds-file	**clouds.yaml** ファイルのファイルパス。**use-clouds** オプションが **true** に設定されている場合に使用されます。
cloud	使用する **clouds.yaml** ファイル内の名前付きクラウド。**use-clouds** オプションが **true** に設定されている場合に使用されます。

第9章 ローカルディスク上の ROOTVOLUME および ETCD を使用した OPENSTACK へのデプロイ

Day 2 オペレーション中に、etcd をルートボリューム (OpenStack Cinder によって提供されるもの) から専用の一時ローカルディスクに移動することで、Red Hat OpenStack Platform (RHOSP) インストールのパフォーマンスの問題を解決および防止できます。

9.1. ローカルディスクへの RHOSP のデプロイ

既存の RHOSP クラウドがある場合は、そのクラウドから etcd を専用の一時ローカルディスクに移動できます。

前提条件

- Cinder が動作している OpenStack クラウドがある。
- OpenStack クラウドに、OpenShift コントロールプレーンの 3 つのルートボリュームを収容するために、少なくとも 75 GB の利用可能なストレージがある。
- OpenStack クラウドが、**rbd** ではなくローカルストレージバックエンドを使用する Nova 一時ストレージを使用してデプロイされている。

手順

1. 次のコマンドを実行して、少なくとも 10 GB の一時ディスクを備えたコントロールプレーンの Nova フレーバーを作成します。環境に応じて **--ram**、**--disk**、および <flavor_name> の値を置き換えます。

```
$ openstack flavor create --<ram 16384> --<disk 0> --ephemeral 10 --vcpus 4
<flavor_name>
```

2. コントロールプレーンのルートボリュームを含むクラスターをデプロイします。以下に例を示します。

サンプル YAML ファイル

```
# ...
controlPlane:
  name: master
  platform:
    openstack:
      type: ${CONTROL_PLANE_FLAVOR}
      rootVolume:
        size: 25
        types:
        - ${CINDER_TYPE}
  replicas: 3
# ...
```

3. 次のコマンドを実行して、作成したクラスターをデプロイします。

```
$ openshift-install create cluster --dir <installation_directory> 1
```

1 **<installation_directory>** には、以前に作成したカスタマイズ済みの **./install-config.yaml** ファイルの場所を指定します。

4. 次の手順に進む前に、次のコマンドを実行して、デプロイしたクラスターが正常であることを確認します。

```
$ oc wait clusteroperators --all --for=condition=Progressing=false 1
```

1 クラスター Operator の進行が完了しており、クラスターがデプロイまたは更新中でないことを確認します。

5. 次の YAML ファイルを使用して、**98-var-lib-etcd.yaml** という名前のファイルを作成します。

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 98-var-lib-etcd
spec:
  config:
    ignition:
      version: 3.5.0
    systemd:
      units:
      - contents: |
          [Unit]
          Description=Mount local-etcd to /var/lib/etcd

          [Mount]
          What=/dev/disk/by-label/local-etcd 1
          Where=/var/lib/etcd
          Type=xfs
          Options=defaults,prjquota

          [Install]
          WantedBy=local-fs.target
        enabled: true
        name: var-lib-etcd.mount
      - contents: |
          [Unit]
          Description=Create local-etcd filesystem
          DefaultDependencies=no
          After=local-fs-pre.target
          ConditionPathIsSymbolicLink=!/dev/disk/by-label/local-etcd 2

          [Service]
          Type=oneshot
          RemainAfterExit=yes
          ExecStart=/bin/bash -c "[ -L /dev/disk/by-label/ephemeral0 ] || ( >&2 echo Ephemeral
disk does not exist; /usr/bin/false )"
          ExecStart=/usr/sbin/mkfs.xfs -f -L local-etcd /dev/disk/by-label/ephemeral0 3
```

```
      [Install]
      RequiredBy=dev-disk-by\x2dlabel-local\x2detcd.device
    enabled: true
    name: create-local-etcd.service
  - contents: |
      [Unit]
      Description=Migrate existing data to local etcd
      After=var-lib-etcd.mount
      Before=crio.service 4

      Requisite=var-lib-etcd.mount
      ConditionPathExists=!/var/lib/etcd/member
      ConditionPathIsDirectory=/sysroot/ostree/deploy/rhcos/var/lib/etcd/member 5

      [Service]
      Type=oneshot
      RemainAfterExit=yes

      ExecStart=/bin/bash -c "if [ -d /var/lib/etcd/member.migrate ]; then rm -rf
/var/lib/etcd/member.migrate; fi" 6

      ExecStart=/usr/bin/cp -aZ /sysroot/ostree/deploy/rhcos/var/lib/etcd/member/
/var/lib/etcd/member.migrate
      ExecStart=/usr/bin/mv /var/lib/etcd/member.migrate /var/lib/etcd/member 7

      [Install]
      RequiredBy=var-lib-etcd.mount
    enabled: true
    name: migrate-to-local-etcd.service
  - contents: |
      [Unit]
      Description=Relabel /var/lib/etcd

      After=migrate-to-local-etcd.service
      Before=crio.service
      Requisite=var-lib-etcd.mount

      [Service]
      Type=oneshot
      RemainAfterExit=yes

      ExecCondition=/bin/bash -c "[ -n \"$(restorecon -nv /var/lib/etcd)\" ]" 8

      ExecStart=/usr/sbin/restorecon -R /var/lib/etcd

      [Install]
      RequiredBy=var-lib-etcd.mount
    enabled: true
    name: relabel-var-lib-etcd.service
```

1 etcd データベースは、ラベルではなくデバイスによってマウントする必要があります。この設定で使用されるデバイスの依存関係を **systemd** に生成させ、ファイルシステムの作成をトリガーするためです。

2

ファイルシステム **dev/disk/by-label/local-etcd** がすでに存在する場合は実行しないでください。

3 **/dev/disk/by-label/ephemeral0** が存在しない場合は、警告メッセージが表示されて失敗します。

4 既存のデータをローカル etcd データベースに移行します。この設定では、**/var/lib/etcd** がマウントされた後、CRI-O が起動する前に、つまり etcd がまだ実行されていないときにデータを移行します。

5 etcd をマウントすること、etcd にメンバーディレクトリーを含めないこと、および ostree にメンバーディレクトリーを含めることを必須とします。

6 以前の移行状態をクリーンアップします。

7 コピーと移動を別々のステップで実行し、完全なメンバーディレクトリーの作成をアトミック操作として実行します。

8 完全な再帰的なラベルの再設定を実行する前に、マウントポイントディレクトリーの簡単なチェックを実行します。ファイルパス **/var/lib/etcd** 内の restorecon がディレクトリーの名前を変更できない場合、再帰的な名前変更が実行されません。

警告

98-var-lib-etcd.yaml ファイルをシステムに適用した後は、削除しないでください。このファイルを削除すると、etcd メンバーが壊れ、システムが不安定になります。

ロールバックが必要な場合は、**ControlPlaneMachineSet** オブジェクトを変更して、一時ディスクを含まないフレーバーを使用します。この変更により、etcd パーティションに一時ディスクを使用せずにコントロールプレーンノードが再生成され、**98-var-lib-etcd.yaml** ファイルに関連する問題が回避されます。**ControlPlaneMachineSet** オブジェクトの更新が完了し、コントロールプレーンノードが一時ディスクを使用していない場合にのみ、**98-var-lib-etcd.yaml** ファイルを安全に削除できます。

6. 次のコマンドを実行して、新しい **MachineConfig** オブジェクトを作成します。

```
$ oc create -f 98-var-lib-etcd.yaml
```

注記

etcd データベースを各コントロールプレーンマシンのローカルディスクに移動するには時間がかかります。

7. 次のコマンドを実行して、etcd データベースが各コントロールプレーンのローカルディスクに転送されたことを確認します。

   a. 次のコマンドを実行して、クラスターがまだ更新中であることを確認します。

```
$ oc wait --timeout=45m --for=condition=Updating=false machineconfigpool/master
```

b. 次のコマンドを実行して、クラスターの準備ができていることを確認します。

```
$ oc wait node --selector='node-role.kubernetes.io/master' --for condition=Ready --
timeout=30s
```

c. 次のコマンドを実行して、クラスター Operators がクラスター内で実行されていることを確認します。

```
$ oc wait clusteroperators --timeout=30m --all --for=condition=Progressing=false
```

9.2. 関連情報

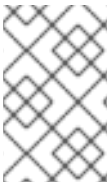
- etcd に関する推奨されるプラクティス
- バックアップおよび復元オプションの概要

第10章 OPENSTACK でのクラスターのアンインストール

Red Hat OpenStack Platform (RHOSP) にデプロイしたクラスターを削除できます。

10.1. INSTALLER-PROVISIONED INFRASTRUCTURE を使用するクラスターの削除

クラウドプラットフォームからプロビジョニングした installer-provisioned infrastructure を使用するクラスターを削除できます。



注記

クラスターを AWS C2S Secret Region にデプロイした場合、インストールプログラムはクラスターの破棄をサポートしません。クラスターリソースは手動で削除する必要があります。

注記

アンインストール後、特に user-provisioned infrastructure クラスターの場合、適切に削除されなかったリソースがないかクラウドプロバイダーを確認してください。インストールプログラムがリソースを作成しなかったか、リソースにアクセスできなかったために、一部のリソースが存在する可能性があります。たとえば、一部の Google Cloud リソースには共有 VPC ホストプロジェクトで IAM パーミッション が必要になるか、削除する必要のあるヘルスチェック が使用されていない可能性があります。

前提条件

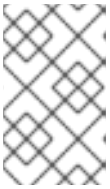
- クラスターをデプロイするために使用したインストールプログラムのコピーがある。
- クラスター作成時にインストールプログラムが生成したファイルがあります。
- CLI で **sudo dnf install coreos-installer** コマンドを入力して、**core-installer** ツールをインストールしている。

手順

1. クラスターのインストールに使用したコンピューターのインストールプログラムがあるディレクトリーから、次のコマンドを実行します。

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。
2. 異なる詳細情報を表示するには、**info** ではなく、**warn**、**debug**、または **error** を指定します。



注記

クラスターのクラスター定義ファイルが含まれるディレクトリーを指定する必要があります。クラスターを削除するには、インストールプログラムでこのディレクトリーにある **metadata.json** ファイルが必要になります。

2. オプション: **coreos-installer** ツールを使用して、**coreos.inst.wipe=yes** フラグを Preboot Execution Environment (PXE) ブート設定に追加します。この操作により、システム上のディスクが消去されるため、新しいクラスターを作成する場合は、クリーンなインストール環境が得られます。詳細な手順については、How to wipe OpenStack disks in OpenShift Container Platform 4 reinstallation (ナレッジベース記事) を参照してください。

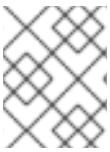
3. オプション: **<installation_directory>** ディレクトリーおよび OpenShift Container Platform インストールプログラムを削除します。

第11章 独自のインフラストラクチャーからの RHOSP のクラスターのアンインストール

user-provisioned infrastructure の Red Hat OpenStack Platform (RHOSP) にデプロイしたクラスターを削除することができます。

11.1. PLAYBOOK 依存関係のダウンロード

user-provisioned infrastructure での削除プロセスを簡素化する Ansible Playbook には、複数の Python モジュールが必要です。プロセスを実行するマシンで、モジュールのリポジトリーを追加し、それらをダウンロードします。



注記

この手順では、Red Hat Enterprise Linux (RHEL) 8 を使用していることを前提としています。

前提条件

- Python 3 がマシンにインストールされている。

手順

1. コマンドラインで、リポジトリーを追加します。
   a. Red Hat Subscription Manager に登録します。

```
$ sudo subscription-manager register # If not done already
```

   b. 最新のサブスクリプションデータをプルします。

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

   c. 現在のリポジトリーを無効にします。

```
$ sudo subscription-manager repos --disable=* # If not done already
```

   d. 必要なリポジトリーを追加します。

```
$ sudo subscription-manager repos \
  --enable=rhel-9-for-x86_64-appstream-rpms \
  --enable=rhel-9-for-x86_64-baseos-rpms \
  --enable=openstack-17.1-for-rhel-9-x86_64-rpms
```

2. モジュールをインストールします。

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk
```

3. **python** コマンドが **python3** を参照していることを確認します。

```
$ sudo alternatives --set python /usr/bin/python3
```

11.2. 独自のインフラストラクチャーを使用する RHOSP からのクラスターの削除

独自のインフラストラクチャーを使用する Red Hat OpenStack Platform (RHOSP) の OpenShift Container Platform クラスターを削除できます。削除プロセスを迅速に完了するには、複数の Ansible Playbook を実行します。

前提条件

- Python 3 がマシンにインストールされている。
- 「Playbook 依存関係のダウンロード」でモジュールをダウンロードしている。
- クラスターのインストールに使用した Playbook がある。
- 対応するインストール Playbook に加えた変更を反映するように **down-** の接頭辞が付けられた Playbook を変更している。たとえば、**bootstrap.yaml** ファイルへの変更は **down-bootstrap.yaml** ファイルに反映されます。
- すべての Playbook は共通ディレクトリーにある。

手順

1. コマンドラインで、ダウンロードした Playbook を実行します。

```
$ ansible-playbook -i inventory.yaml  \
 down-bootstrap.yaml      \
 down-control-plane.yaml  \
 down-compute-nodes.yaml  \
 down-load-balancers.yaml \
 down-network.yaml        \
 down-security-groups.yaml
```

2. OpenShift Container Platform インストールに対して加えた DNS レコードの変更を削除します。

OpenShift Container Platform はお使いのインフラストラクチャーから削除されます。

第12章 OPENSTACK のインストール設定パラメーター

OpenShift Container Platform クラスターを Red Hat OpenStack Platform (RHOSP) にデプロイする前に、パラメーターを指定してクラスターとそれをホストするプラットフォームをカスタマイズします。**install-config.yaml** ファイルを作成するときは、コマンドラインを使用して必要なパラメーターの値を指定します。その後、**install-config.yaml** ファイルを変更して、クラスターをさらにカスタマイズできます。

12.1. OPENSTACK で使用可能なインストール設定パラメーター

以下の表に、インストールプロセスの一部として設定できる必須、オプション、および OpenStack 固有のインストール設定パラメーターを示します。

重要

インストール後は、これらのパラメーターを **install-config.yaml** ファイルで変更することはできません。

12.1.1. 必須設定パラメーター

必須のインストール設定パラメーターは、以下の表で説明されています。

表12.1 必須パラメーター

パラメーター	説明
apiVersion:	**install-config.yaml** コンテンツの API バージョン。現在のバージョンは **v1** です。インストールプログラムは、古い API バージョンもサポートしている場合があります。 **値:** 文字列
baseDomain:	クラウドプロバイダーのベースドメイン。ベースドメインは、OpenShift Container Platform クラスターコンポーネントへのルートを作成するために使用されます。クラスターの完全な DNS 名は、**<metadata.name>.<baseDomain>** 形式を使用する **baseDomain** と **metadata.name** パラメーターの値を組み合わせたものです。 **値:** **example.com** などの完全修飾ドメイン名またはサブドメイン名。
metadata:	Kubernetes リソース **ObjectMeta**。ここからは **name** パラメーターのみが消費されます。 **値:** オブジェクト

パラメーター	説明
metadata: name:	クラスターの名前。クラスターの DNS レコードはすべて **{{.metadata.name}}.{{.baseDomain}}** のサブドメインです。 **値:** 小文字、ハイフン (-)、およびピリオド (.) からなる文字列 (例: **dev**)。文字列は 14 文字以上でなければなりません。
platform:	インストールを実行する特定のプラットフォームの設定: **aws**、**baremetal**、**azure**、**gcp**、**ibmcloud**、**nutanix**、**openstack**、**powervs**、**vsphere**、または **{}**。**platform.<platform>** パラメーターに関する追加情報は、以下の表で特定のプラットフォームを参照してください。 **値:** オブジェクト
pullSecret:	Red Hat OpenShift Cluster Manager からプルシークレット を取得して、Quay.io などのサービスから OpenShift Container Platform コンポーネントのコンテナーイメージをダウンロードすることを認証します。 **値:** { "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }

12.1.2. ネットワーク設定パラメーター

既存のネットワークインフラストラクチャーの要件に基づいて、インストール設定をカスタマイズできます。たとえば、クラスターネットワークの IP アドレスブロックを拡張したり、デフォルトとは異なる IP アドレスブロックを設定したりすることができます。

クラスターのネットワークパラメーターを設定する前に、次の情報を考慮してください。

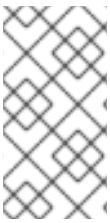
- Red Hat OpenShift Networking OVN-Kubernetes ネットワークプラグインを使用する場合、IPv4 と IPv6 の両方のアドレスファミリーがサポートされます。

- IPv4 アドレスと非リンクローカル IPv6 アドレスの両方をサポートするネットワークを持つ OpenShift Container Platform クラスターにノードをデプロイした場合は、デュアルスタックネットワークを使用するようにクラスターを設定します。
    - デュアルスタックネットワークに設定されたクラスターでは、IPv4 と IPv6 の両方のトラフィックがデフォルトゲートウェイとして同じネットワークインターフェイスを使用する必要があります。これにより、複数のネットワークインターフェイスコントローラー (NIC) 環境で、使用可能なネットワークインターフェイスに基づいて、使用する NIC をクラスターが検出できるようになります。詳細は、**OVN-Kubernetes ネットワークプラグインについて** の「OVN-Kubernetes IPv6 とデュアルスタックの制限」を参照してください。
    - ネットワーク接続の問題を防ぐために、デュアルスタックネットワークをサポートするホストにシングルスタック IPv4 クラスターをインストールしないでください。

両方の IP アドレスファミリーを使用するようにクラスターを設定する場合は、次の要件を確認してください。

- どちらの IP ファミリーも、デフォルトゲートウェイに同じネットワークインターフェイスを使用する必要があります。
- 両方の IP ファミリーにデフォルトゲートウェイが必要です。
- すべてのネットワーク設定パラメーターに対して、IPv4 アドレスと IPv6 アドレスを同じ順序で指定する必要があります。たとえば、次の設定では、IPv4 アドレスが IPv6 アドレスの前に記載されています。

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```



注記

Globalnet は、Red Hat OpenShift Data Foundation ディザスターリカバリーソリューションではサポートされていません。リージョンの障害復旧シナリオでは、クラスターと各クラスター内のサービスネットワークに重複しないプライベート IP アドレスの範囲を使用するようにしてください。

表12.2 ネットワークパラメーター

パラメーター	説明

パラメーター	説明
networking:	クラスターのネットワークの設定。 **値:** オブジェクト  **注記** インストール後に **networking** オブジェクトによって指定されたパラメーターを変更することはできません。
networking: networkType:	インストールする Red Hat OpenShift Networking ネットワークプラグイン。 **値: OVNKubernetes**。**OVNKubernetes** は、Linux ネットワークおよび Linux サーバーと Windows サーバーの両方を含むハイブリッドネットワーク用の Container Network Interface (CNI) プラグインです。デフォルトの値は **OVNKubernetes** です。
networking: clusterNetwork:	Pod の IP アドレスブロック。 デフォルト値は **10.128.0.0/14** で、ホストの接頭辞は **/23** です。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。 **値:** オブジェクトの配列。以下に例を示します。 networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	**networking.clusterNetwork** を使用する場合に必須です。IP アドレスブロック。 IPv4 ネットワーク **値:** Classless Inter-Domain Routing (CIDR) 表記の IP アドレスブロック。IPv4 ブロックの接頭辞長は **0** から **32** の間になります。

パラメーター	説明
networking: clusterNetwork: hostPrefix:	それぞれの個別ノードに割り当てるサブネット接頭辞長。たとえば、**hostPrefix** が **23** に設定される場合、各ノードに指定の **cidr** から **/23** サブネットが割り当てられます。**hostPrefix** 値の **23** は、510 (2^(32 - 23) - 2) Pod IP アドレスを提供します。 **値:** サブネット接頭辞。 デフォルト値は **23** です。
networking: serviceNetwork:	サービスの IP アドレスブロック。デフォルト値は **172.30.0.0/16** です。 OVN-Kubernetes ネットワークプラグインは、サービスネットワークに対して単一の IP アドレスブロックのみをサポートします。 **値:** CIDR 形式の IP アドレスブロックを含む配列。以下に例を示します。 networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	マシンの IP アドレスブロック。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。 **値:** オブジェクトの配列。以下に例を示します。 networking: machineNetwork: - cidr: 10.0.0.0/16

パラメーター	説明
networking: machineNetwork: cidr:	**networking.machineNetwork** を使用する場合に必須です。IP アドレスブロック。libvirt と IBM Power® Virtual Server を除くすべてのプラットフォームのデフォルト値は **10.0.0.0/16** です。libvirt の場合、デフォルト値は **192.168.126.0/24** です。IBM Power® Virtual Server の場合、デフォルト値は **192.168.0.0/24** です。 **値:** CIDR 表記の IP ネットワークブロック。 例: **10.0.0.0/16** **注記** 優先される NIC が置かれている CIDR に一致する **networking.machineNetwork** を設定します。
networking: ovnKubernetesConfig: ipv4: internalJoinSubnet:	**ovn-kubernetes** によって内部的に使用される IPv4 join サブネットを設定します。このサブネットは、ノードネットワークを含め、OpenShift Container Platform が使用している他のサブネットと重複することはできません。サブネットのサイズは、ノード数より大きくする必要があります。インストール後に値を変更することはできません。 **値:** CIDR 表記の IP ネットワークブロック。デフォルト値は **100.64.0.0/16** です。

12.1.3. オプションの設定パラメーター

オプションのインストール設定パラメーターは、以下の表で説明されています。

表12.3 オプションのパラメーター

パラメーター	説明
additionalTrustBundle:	ノードの信頼済み証明書ストアに追加される PEM でエンコードされた X.509 証明書バンドル。このトラストバンドルは、プロキシーが設定されている場合にも使用される可能性があります。 **値:** 文字列

パラメーター	説明
capabilities:	オプションのコアクラスターコンポーネントのインストールを制御します。オプションのコンポーネントを無効にすることで、OpenShift Container Platform クラスターのフットプリントを削減できます。詳細は、**インストール** の「クラスター機能ページ」を参照してください。 **値:** 文字列の配列
capabilities: baselineCapabilitySet:	有効にするオプション機能の初期セットを選択します。有効な値は **None**、**v4.11**、**v4.12**、**vCurrent** です。デフォルト値は **vCurrent** です。 **値:** 文字列
capabilities: additionalEnabledCapabilities:	オプションの機能のセットを、**baselineCapabilitySet** で指定したものを超えて拡張します。このパラメーターでは複数の機能を指定できます。 **値:** 文字列の配列
cpuPartitioningMode:	ワークロードパーティション設定を使用して、OpenShift Container Platform サービス、クラスター管理ワークロード、およびインフラストラクチャー Pod を分離し、予約された CPU セットで実行できます。ワークロードパーティショニングを有効にできるのはインストール時のみです。インストール後は無効にできません。このフィールドはワークロードのパーティショニングを有効にしますが、特定の CPU を使用するようにワークロードを設定するわけではありません。詳細は、**スケーラビリティとパフォーマンス** セクションの **ワークロードパーティショニング** ページを参照してください。 **値:** **None** または **AllNodes**。デフォルト値は **None** です。
compute:	コンピュートノードを構成するマシンの設定。 **値:** **MachinePool** オブジェクトの配列。

パラメーター	説明
compute: architecture:	プール内のマシンの命令セットアーキテクチャーを決定します。現在、さまざまなアーキテクチャーのクラスターはサポートされていません。すべてのプールは同じアーキテクチャーを指定する必要があります。有効な値は、**amd64** と **arm64** です。 すべてのインストールオプションが 64 ビット ARM アーキテクチャーをサポートしているわけではありません。使用するインストールオプションがプラットフォームでサポートされているか確認するには、**クラスターインストール方法の選択およびそのユーザー向けの準備** の **各種プラットフォームでサポートされているインストール方法** を参照してください。 **値:** 文字列
compute: hyperthreading:	コンピュートマシンで同時マルチスレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時マルチスレッドはマシンのコアのパフォーマンスを上げるために有効化されます。  **重要** 同時マルチスレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。 **値:** **Enabled** または **Disabled**
compute: name:	**compute** を使用する場合に必須です。マシンプールの名前。 **値:** **worker**
compute: platform:	**compute** を使用する場合に必須です。このパラメーターを使用して、ワーカーマシンをホストするクラウドプロバイダーを指定します。このパラメーターの値は **controlPlane.platform** パラメーターの値に一致する必要があります。 **値:** **aws**、**azure**、**gcp**、**ibmcloud**、**nutanix**、**openstack**、**powervs**、**vsphere**、または **{}**
compute: replicas:	プロビジョニングするコンピュートマシン (ワーカーマシンとしても知られる) の数。 **値:** **2** 以上の正の整数。デフォルト値は **3** です。

パラメーター	説明
featureSet:	機能セットのクラスターを有効にします。機能セットは、デフォルトで有効にされない OpenShift Container Platform 機能のコレクションです。インストール中に機能セットを有効にする方法の詳細は、「機能ゲートの使用による各種機能の有効化」を参照してください。 **値**: 文字列。**TechPreviewNoUpgrade** など、有効にする機能セットの名前。
controlPlane:	コントロールプレーンを構成するマシンの設定。 **値**: **MachinePool** オブジェクトの配列。
controlPlane: architecture:	プール内のマシンの命令セットアーキテクチャーを決定します。現在、さまざまなアーキテクチャーのクラスターはサポートされていません。すべてのプールは同じアーキテクチャーを指定する必要があります。有効な値は、**amd64** と **arm64** です。 すべてのインストールオプションが 64 ビット ARM アーキテクチャーをサポートしているわけではありません。使用するインストールオプションがプラットフォームでサポートされているか確認するには、**クラスターインストール方法の選択およびそのユーザー向けの準備** の **各種プラットフォームでサポートされているインストール方法** を参照してください。 **値**: 文字列
controlPlane: hyperthreading:	コントロールプレーンマシンで同時マルチスレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時マルチスレッドはマシンのコアのパフォーマンスを上げるために有効化されます。  **重要** 同時マルチスレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。 **値**: **Enabled** または **Disabled**

パラメーター	説明
controlPlane: name:	**controlPlane** を使用する場合に必須です。マシンプールの名前。 **値: master**
controlPlane: platform:	**controlPlane** を使用する場合に必須です。このパラメーターを使用して、コントロールプレーンマシンをホストするクラウドプロバイダーを指定します。このパラメーターの値は **compute.platform** パラメーターの値に一致する必要があります。 **値:** **aws**、**azure**、**gcp**、**ibmcloud**、**nutanix**、**openstack**、**powervs**、**vsphere**、または **{}**
controlPlane: replicas:	プロビジョニングするコントロールプレーンマシンの数。 **値:** サポートされる値は **3** です。シングルノード OpenShift をデプロイする場合は **1** です。
credentialsMode:	Cloud Credential Operator (CCO) モード。モードを指定しないと、CCO は指定された認証情報の機能を動的に判別しようとします。この場合、複数のモードがサポートされるプラットフォームで Mint モードが優先されます。  **注記** すべてのクラウドプロバイダーですべての CCO モードがサポートされているわけではありません。CCO モードの詳細は、**認証と認可** コンテンツの「クラウドプロバイダーの認証情報の管理」を参照してください。 **値: Mint**、**Passthrough**、**Manual**、または空の文字列 (**""**)。

パラメーター	説明
fips:	FIPS モードを有効または無効にします。デフォルトは **false** (無効) です。FIPS モードを有効にすると、OpenShift Container Platform が稼働している Red Hat Enterprise Linux CoreOS (RHCOS) マシンで、デフォルトの Kubernetes 暗号化スイートが無視され、代わりに RHCOS が提供する暗号化モジュールが使用されます。  **重要** クラスターで FIPS モードを有効にするには、FIPS モードで動作するように設定された Red Hat Enterprise Linux (RHEL) コンピューターからインストールプログラムを実行する必要があります。RHEL で FIPS モードを設定する方法の詳細は、RHEL から FIPS モードへの切り替え を参照してください。 FIPS モードでブートされた Red Hat Enterprise Linux (RHEL) または Red Hat Enterprise Linux CoreOS (RHCOS) を実行する場合、OpenShift Container Platform コアコンポーネントは、x86_64、ppc64le、および s390x アーキテクチャーのみで、FIPS 140-2/140-3 検証のために NIST に提出された RHEL 暗号化ライブラリーを使用します。  **重要** Azure File ストレージを使用している場合、FIPS モードを有効にすることはできません。 **値:** **false** または **true**
imageContentSources:	release-image コンテンツのソースおよびリポジトリー。 **値:** オブジェクトの配列。この表の以下の行で説明されているように、**source** およびオプションで **mirrors** が含まれます。
imageContentSources: source:	**imageContentSources** を使用する場合に必須です。ユーザーが参照するリポジトリーを指定します (例: イメージプル仕様)。 **値:** 文字列

パラメーター	説明
imageContentSources: mirrors:	同じイメージが含まれている可能性があるリポジトリーを 1 つ以上指定します。 **値**: 文字列の配列
platform: aws: lbType:	AWS で NLB ロードバランサータイプを設定するために必要です。有効な値は **Classic** または **NLB** です。値が指定されていない場合、インストールプログラムはデフォルトで **Classic** になります。インストールプログラムは、ここで指定された値をイングレスクラスター設定オブジェクトに設定します。他の Ingress コントローラーのロードバランサータイプを指定しない場合、それらはこのパラメーターに設定されたタイプを使用します。 **値**: **Classic** または **NLB**。デフォルト値は **Classic** です。
publish:	Kubernetes API、OpenShift ルートなどのクラスターのユーザーに表示されるエンドポイントをパブリッシュまたは公開する方法。 **値**: **Internal** または **External**。インターネットからアクセスできないプライベートクラスターをデプロイするには、**publish** パラメーターを **Internal** に設定します。デフォルト値は **External** です。
sshKey:	クラスターマシンへのアクセスを認証するための SSH キー。 **注記** インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。 **値**: たとえば、**sshKey: ssh-ed25519 AAAA..** です。

注記

AWS アカウントでサービスコントロールポリシー (SCP) が有効になっている場合は、**credentialsMode** パラメーターを **Mint**、**Passthrough** または **Manual** に設定する必要があります。Google Cloud で共有 Virtual Private Cloud (VPC) にインストールする場合は、**credentialsMode** を **Passthrough** または **Manual** に設定する必要があります。

重要

このパラメーターを **Manual** に設定すると、管理者レベルのシークレットを **kube-system** プロジェクトに保存する代替手段が有効になりますが、追加の設定手順が必要になります。詳細は、「管理者レベルのシークレットを kube-system プロジェクトに保存する代替方法」を参照してください。

12.1.4. オプションの AWS 設定パラメーター

オプションの AWS 設定パラメーターは、以下の表で説明されています。

表12.4 オプションの AWS パラメーター

パラメーター	説明
compute: platform: aws: amiID:	クラスターのコンピュートマシンの起動に使用される AWS AMI。これは、カスタム RHCOS AMI を必要とするリージョンに必要です。 **値:** 設定した AWS リージョンに属する公開済みまたはカスタムの RHCOS AMI。利用可能な AMI ID は、**AWS インフラストラクチャーの RHCOS AMI** を参照してください。
compute: platform: aws: iamProfile:	マシンに使用する IAM インスタンスプロファイルの名前。インストールプログラムで IAM インスタンスプロファイルを作成する場合は、**iamProfile** パラメーターを使用しないでください。**iamProfile** パラメーターまたは **iamRole** パラメーターのいずれかを指定できますが、両方を指定することはできません。 **値:** 文字列
compute: platform: aws: iamRole:	マシンに使用する IAM インスタンスロールの名前。IAM ロールを指定すると、インストールプログラムによってインスタンスプロファイルが作成されます。インストールプログラムで IAM インスタンスロールを作成する場合は、**iamRole** パラメーターを選択しないでください。**iamRole** または **iamProfile** パラメーターのいずれかを指定できますが、両方を指定することはできません。 **値:** 文字列
compute: platform: aws: rootVolume: iops:	ルートボリュームに予約される 1 秒あたりの入出力操作 (IOPS)。 **値:** 整数 (例: **4000**)。

パラメーター	説明
`compute:` `  platform:` `  aws:` `  rootVolume:` `  size:`	ルートボリュームのサイズ (GiB)。 **値:** 整数 (例:**500**)。
`compute:` `  platform:` `  aws:` `  rootVolume:` `  type:`	root ボリュームのタイプです。 **値:** 有効な AWS EBS ボリュームタイプ (例:**io1**)。
`compute:` `  platform:` `  aws:` `  rootVolume:` `  kmsKeyARN:`	KMS キーの Amazon リソース名 (キー ARN)。これは、ワーカーノードのオペレーティングシステムボリュームを特定の KMS キーで暗号化するために必要です。 **値:** 有効なキー ID またはキー ARN。
`compute:` `  platform:` `  aws:` `  type:`	コンピュートマシンの EC2 インスタンスタイプ。 **値:** 有効な AWS インスタンスタイプ (例:**m4.2xlarge**)。「カスタマイズによる AWS へのクラスターのインストール」ページの「AWS のテスト済みインスタンスタイプ」の表を参照してください。
`compute:` `  platform:` `  aws:` `  zones:`	インストールプログラムがコンピュートマシンプールのマシンを作成するアベイラビリティーゾーン。独自の VPC を指定する場合は、そのアベイラビリティーゾーンにサブネットを指定する必要があります。 **値:** YAML シーケンス 内の有効な AWS アベイラビリティーゾーン (**us-east-1c** など) のリスト。
`controlPlane:` `  platform:` `  aws:` `  amiID:`	クラスターのコントロールプレーンマシンを起動するために使用される AWS AMI。これは、カスタム RHCOS AMI を必要とするリージョンに必要です。 **値:** 設定した AWS リージョンに属する公開済みまたはカスタムの RHCOS AMI。利用可能な AMI ID は、**AWS インフラストラクチャーの RHCOS AMI** を参照してください。

パラメーター	説明
controlPlane: platform: aws: iamProfile:	マシンに使用する IAM インスタンスプロファイルの名前。インストールプログラムで IAM インスタンスプロファイルを作成する場合は、**iamProfile** パラメーターを使用しないでください。**iamProfile** パラメーターまたは **iamRole** パラメーターのいずれかを指定できますが、両方を指定することはできません。 **値:** 文字列
controlPlane: platform: aws: iamRole:	マシンに使用する IAM インスタンスロールの名前。IAM ロールを指定すると、インストールプログラムによってインスタンスプロファイルが作成されます。インストールプログラムで IAM インスタンスロールを作成する場合は、**iamRole** パラメーターを使用しないでください。**iamRole** または **iamProfile** パラメーターのいずれかを指定できますが、両方を指定することはできません。 **値:** 文字列
controlPlane: platform: aws: rootVolume: iops:	コントロールプレーンマシン上のルートボリューム用に予約されている 1 秒あたりの入出力操作数 (IOPS)。 **値:** 整数 (例: **4000**)。
controlPlane: platform: aws: rootVolume: size:	コントロールプレーンマシンのルートボリュームのサイズ (GiB)。 **値:** 整数 (例: **500**)。
controlPlane: platform: aws: rootVolume: type:	コントロールプレーンマシンのルートボリュームのタイプ。 **値:** 有効な AWS EBS ボリュームタイプ (例: **io1**)。
controlPlane: platform: aws: rootVolume: kmsKeyARN:	KMS キーの Amazon リソース名 (キー ARN)。これは、特定の KMS キーでコントロールプレーンノードのオペレーティングシステムボリュームを暗号化するために必要です。 **値:** 有効なキー ID とキー ARN。

パラメーター	説明
controlPlane: platform: aws: type:	コントロールプレーンマシンの EC2 インスタンスタイプ。 **値:** 有効な AWS インスタンスタイプ (例: **m6i.xlarge**)。「カスタマイズによる AWS へのクラスターのインストール」ページの「AWS のテスト済みインスタンスタイプ」の表を参照してください。
controlPlane: platform: aws: zones:	インストールプログラムがコントロールプレーンマシンプールのマシンを作成するアベイラビリティーゾーン。 **値:** YAML シーケンス 内の有効な AWS アベイラビリティーゾーン (**us-east-1c** など) のリスト。
platform: aws: amiID:	クラスターのすべてのマシンを起動するために使用される AWS AMI。これが設定されている場合、AMI はクラスターと同じリージョンに属する必要があります。これは、カスタム RHCOS AMI を必要とするリージョンに必要です。 **値:** 設定した AWS リージョンに属する公開済みまたはカスタムの RHCOS AMI。利用可能な AMI ID は、**AWS インフラストラクチャーの RHCOS AMI** を参照してください。
platform: aws: hostedZone:	クラスターの既存の Route 53 プライベートホストゾーン。独自の VPC を指定する場合も、既存のホストゾーンのみを使用できます。ホストゾーンは、インストール前にユーザーによって提供される VPC に関連付けられている必要があります。また、ホストゾーンのドメインはクラスタードメインまたはクラスタードメインの親である必要があります。定義されていない場合は、インストールプログラムは新規のホストゾーンを作成します。 **値:** 文字列 (例:**Z3URY6TWQ91KVV**)。
platform: aws: hostedZoneRole:	指定されたホストゾーンを含むアカウントの既存の IAM ロールの Amazon Resource Name (ARN)。インストールプログラムとクラスター Operator は、ホストゾーンで操作を実行するときにこのロールを引き受けます。このパラメーターは、クラスターを共有 VPC にインストールする場合にのみ使用してください。 **値:** 文字列。例: (**arn:aws:iam::1234567890:role/shared-vpc-role**)。

パラメーター	説明
`platform:` `  aws:` `  region:`	インストールプログラムがすべてのクラスターリソースを作成する AWS リージョン。 **値:** 有効な AWS リージョン (例: **us-east-1**)。次のコマンドを実行すると、AWS CLI を使用して、選択したインスタンスタイプに基づいて利用可能なリージョンにアクセスできます。 `$ aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge`  **重要** ARM ベースの AWS インスタンスで実行する場合は、AWS Graviton プロセッサーが利用可能なリージョンを入力するようにしてください。AWS ドキュメントの グローバルアベイラビリティー マップを参照してください。現在、AWS Graviton3 プロセッサーは一部のリージョンでのみ利用できます。
`platform:` `  aws:` `  serviceEndpoints:` `  - name:` `  url:`	AWS サービスのエンドポイント名と URL。カスタムエンドポイントは、FIPS など、代替 AWS エンドポイントを使用しなければならない場合にのみ必要です。カスタム API エンドポイントは、EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53、および STS AWS サービスに指定できます。 **値:** 有効な AWS サービスエンドポイント 名と有効な AWS サービスエンドポイント URL。

パラメーター	説明
platform: aws: userTags:	インストールプログラムが、作成するすべてのリソースに対するタグとして追加するキーと値のマップ。 **値:** **<key>: <value>** 形式のキー値ペアなど、有効な YAML マップ。AWS タグの詳細は、AWS ドキュメントの Tagging Your Amazon EC2 Resources を参照してください。  **注記** インストール時に最大 25 個のユーザー定義タグを追加できます。残りの 25 個のタグは、OpenShift Container Platform 用に予約されています。
platform: aws: propagateUserTags:	クラスター内 Operator に対し、Operator が作成する AWS リソースのタグに指定されたユーザータグを組み込むフラグ。 **値:** ブール値 (例: **true** または **false**)。
platform: aws: publicIpv4Pool:	**publish** が **External** に設定されている場合に、Elastic IP (EIP) を割り当てるために使用されるパブリック IPv4 プール ID。プールは、クラスターと同じ AWS アカウントとリージョンでプロビジョニングおよびアドバタイズする必要があります。プール内に 2n + 1 個の IPv4 アドレスがあることを確認する必要があります。**n** は、API、NAT ゲートウェイ、およびブートストラップノードの Network Load Balancer (NLB) をデプロイするために使用する AWS ゾーンの合計数です。AWS での独自の持ち込み IP アドレス (BYOIP) の詳細は、Onboard your BYOIP を参照してください。 **値:** 有効な パブリック IPv4 プール ID  **注記** BYOIP は、ネットワーク制限のないカスタマイズされたインストールの場合にのみ有効にできます。
platform: aws: preserveBootstrapIgnition:	ブートストラップの完了後に S3 バケットが削除されないようにします。 **値:** **true** または **false**。デフォルト値は **false** で、S3 バケットが削除されます。

パラメーター	説明
platform: aws: vpc: subnets:	自動的に作成されたサブネットの代わりに使用される既存の VPC 内のサブネットのリスト。サブネットを指定するには、サブネット ID と、そのサブネットに適用されるロールのオプションのリストを指定します。サブネット ID を指定しても、サブネットのロールを指定しない場合は、サブネットのロールが自動的に決定されます。ロールを指定しない場合は、VPC 内の他のサブネットに **kubernetes.io/cluster/.: .** または **kubernetes.io/cluster/unmanaged: true** タグが付いていることを確認する必要があります。 サブネットは、指定する同じ **machineNetwork[].cidr** 範囲の一部である必要があります。 パブリッククラスターの場合は、各アベイラビリティーゾーンにパブリックサブネットとプライベートサブネットを指定します。 プライベートクラスターには、各アベイラビリティーゾーンのプライベートサブネットを指定します。 AWS Local Zone を使用するクラスターの場合、エッジマシンプールが確実に作成されるように、AWS Local Zone のサブネットをこのリストに追加する必要があります。 **値: id** パラメーターと **roles** パラメーターのペアのリスト。
platform: aws: vpc: subnets: - id:	インストールプログラムによって作成されたサブネットの代わりに使用される既存のサブネットの ID。 **値:** 文字列。サブネット ID は、"subnet-" で始まる英数字のみを含む一意の ID である必要があります。ID はちょうど 24 文字の長さである必要があります。

パラメーター	説明
platform: aws: vpc: subnets: - id: roles: - type:	**platform.aws.vpc.subnets.id** で指定されたサブネットに適用される 1 つ以上のロール。サブネットにロールを指定する場合、各サブネットには少なくとも 1 つのロールが割り当てられている必要があり、**ClusterNode**、**IngressControllerLB**、**ControlPlaneExternalLB**、**BootstrapNode**、および **ControlPlaneInternalLB** のロールが少なくとも 1 つのサブネットに割り当てられている必要があります。ただし、クラスタースコープが内部の場合、**ControlPlaneExternalLB** ロールは必要ありません。 **EdgeNode** ロールは、AWS Local Zones 内のサブネットにのみ割り当てることができます。 **値:** 1 つ以上のロールタイプのリスト。有効な値には、**ClusterNode**、**EdgeNode**、**BootstrapNode**、**IngressControllerLB**、**ControlPlaneExternalLB**、**ControlPlaneInternalLB** などがあります。

12.1.5. 追加の Red Hat OpenStack Platform (RHOSP) 設定パラメーター

追加の RHOSP 設定パラメーターは以下の表で説明されています。

表12.5 追加の RHOSP パラメーター

パラメーター	説明
compute: platform: openstack: rootVolume: size:	コンピュートマシンの場合、root ボリュームのギガバイトのサイズになります。この値を設定しない場合、マシンは一時ストレージを使用します。 **値:** 整数 (例: **30**)。
compute: platform: openstack: rootVolume: types:	コンピュートマシンの場合は、root のボリュームタイプです。 **値:** 文字列のリスト (例: {**performance-host1**, **performance-host2**, **performance-host3**})。[1]

パラメーター	説明
compute: platform: openstack: rootVolume: type:	コンピュートマシンの場合、root のボリュームタイプです。このプロパティーは非推奨となり、**compute.platform.openstack.rootVolume.types** に置き換えられます。 **値:** 文字列 (例:**performance**)。[2]
compute: platform: openstack: rootVolume: zones:	コンピュートマシンの場合、ルートボリュームをインストールする Cinder 可用性ゾーン。このパラメーターに値を設定しない場合、インストールプログラムはデフォルトのアベイラビリティーゾーンを選択します。このパラメーターは、**compute.platform.openstack.zones** が定義されている場合には必須です。 **値:** 文字列のリスト (例;**["zone-1", "zone-2"]**)。
controlPlane: platform: openstack: rootVolume: size:	コントロールプレーンマシンの場合、root ボリュームのギガバイトのサイズになります。この値を設定しない場合、マシンは一時ストレージを使用します。 **値:** 整数 (例:**30**)。
controlPlane: platform: openstack: rootVolume: types:	コントロールプレーンマシンの場合は、root ボリュームのタイプです。 **値:** 文字列のリスト (例: {**performance-host1**, **performance-host2**, **performance-host3**})。[1]
controlPlane: platform: openstack: rootVolume: type:	コントロールプレーンマシンの場合、root ボリュームのタイプです。このプロパティーは非推奨となり、**compute.platform.openstack.rootVolume.types** に置き換えられます。 **値:** 文字列 (例:**performance**)。[2]
controlPlane: platform: openstack: rootVolume: zones:	コントロールプレーンマシンの root ボリュームをインストールする Cinder アベイラビリティーゾーン。この値を設定しない場合、インストールプログラムはデフォルトのアベイラビリティーゾーンを選択します。**controlPlane.platform.openstack.zones** が定義されている場合、このパラメーターは必須です。 **値:** 文字列のリスト (例;**["zone-1", "zone-2"]**)。

パラメーター	説明
platform: openstack: cloud:	**clouds.yaml** ファイルのクラウドリストにある使用する RHOSP クラウドの名前。 可能であれば、**clouds.yaml** ファイルのクラウド設定では、ユーザー名とパスワードの組み合わせではなく、アプリケーションの認証情報を使用します。アプリケーション認証情報を使用すると、ユーザー名とパスワードのローテーションに伴うシークレットの伝達による中断を回避できます。 **値:** 文字列 (例:**MyCloud**)。
platform: openstack: externalNetwork:	インストールに使用される RHOSP の外部ネットワーク名。 **値:** 文字列 (例:**external**)。
platform: openstack: computeFlavor:	コントロールプレーンおよびコンピュートマシンに使用する RHOSP フレーバー。 このプロパティーは非推奨にされています。すべてのマシンプールのデフォルトとしてフレーバーを使用するには、これを **platform.openstack.defaultMachinePlatform** プロパティーで **type** キーの値として追加します。それぞれのマシンプールのフレーバー値を個別に設定することもできます。 **値:** 文字列 (例:**m1.xlarge**)。

1. マシンプールが **zones** を定義している場合、タイプの数は 1 つの項目であるか、**zones** 内の項目の数と一致できます。たとえば、**zones** に項目が 3 つある場合は、タイプの数を 2 にすることはできません。
2. このプロパティーへの既存の参照がある場合、インストールプログラムは、それに対応する値を **controlPlane.platform.openstack.rootVolume.types** フィールドに入力します。

12.1.6. オプションの RHOSP 設定パラメーター

オプションの RHOSP 設定パラメーターは、以下の表で説明されています。

表12.6 オプションの RHOSP パラメーター

パラメーター	説明
compute: platform: openstack: additionalNetworkIDs:	コンピュートマシンに関連付けられた追加のネットワーク。追加ネットワーク用に許可されるアドレスのペアは作成されません。 **値:** 1つ以上の文字列形式の UUID を含むリスト。例: **fa806b2f-ac49-4bce-b9db-124bc64209bf**。
compute: platform: openstack: additionalSecurityGroupIDs:	コンピュートマシンに関連付けられた追加のセキュリティーグループ。 **値:** 1つ以上の文字列形式の UUID を含むリスト。例: **7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**.
compute: platform: openstack: zones:	マシンをインストールする RHOSP Compute (Nova) アベイラビリティーゾーン (AZs)。このパラメーターが設定されていない場合、インストールプログラムは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。 **値:** 文字列のリスト。例: **["zone-1", "zone-2"]**。
compute: platform: openstack: serverGroupPolicy:	プール内のコンピュートマシンを含むグループに適用するサーバーグループポリシー。作成後にサーバーグループのポリシーまたは所属を変更することはできません。サポートされているオプションには、**anti-affinity**、**soft-affinity**、および **soft-anti-affinity** が含まれます。デフォルト値は **soft-anti-affinity** です。 **affinity** ポリシーは移行を防止するため、RHOSP のアップグレードに影響します。**affinity** ポリシーはサポートされていません。 厳密な **anti-affinity** ポリシーを使用する場合は、インスタンスの移行中に追加の RHOSP ホストが必要です。 **値:** マシンプールに適用するサーバーグループポリシー。たとえば、**soft-affinity**。
controlPlane: platform: openstack: additionalNetworkIDs:	コントロールプレーンマシンに関連付けられた追加のネットワーク。追加ネットワーク用に許可されるアドレスのペアは作成されません。 コントロールプレーンマシンに接続されている追加のネットワークも、ブートストラップノードに接続されています。 **値:** 1つ以上の文字列形式の UUID を含むリスト。例: **fa806b2f-ac49-4bce-b9db-124bc64209bf**。

パラメーター	説明
controlPlane: platform: openstack: additionalSecurityGroupIDs:	コントロールプレーンマシンに関連付けられた追加のセキュリティーグループ。 **値:** 1 つ以上の文字列形式の UUID を含むリスト。例: **7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**.
controlPlane: platform: openstack: zones:	マシンをインストールする RHOSP Compute (Nova) アベイラビリティーゾーン (AZs)。このパラメーターが設定されていない場合、インストールプログラムは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。 **値:** 文字列のリスト。例: **["zone-1", "zone-2"]**。
controlPlane: platform: openstack: serverGroupPolicy:	プール内のコントロールプレーンマシンを含むグループに適用するサーバーグループポリシー。作成後にサーバーグループのポリシーまたは所属を変更することはできません。サポートされているオプションには、**anti-affinity**、**soft-affinity**、および **soft-anti-affinity** が含まれます。デフォルト値は **soft-anti-affinity** です。 **affinity** ポリシーは移行を防止するため、RHOSP のアップグレードに影響します。**affinity** ポリシーはサポートされていません。 厳密な **anti-affinity** ポリシーを使用する場合は、インスタンスの移行中に追加の RHOSP ホストが必要です。 **値:** マシンプールに適用するサーバーグループポリシー。たとえば、**soft-affinity**。
platform: openstack: clusterOSImage:	インストールプログラムが RHCOS イメージをダウンロードする場所。 ネットワークが制限された環境でインストールを実行するには、このパラメーターを設定する必要があります。 **値:** HTTP または HTTPS URL。必要に応じて SHA-256 チェックサムを付けることもできます。 例: **http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**。この値は、既存の Glance イメージの名前にもなり得ます (例: **my-rhcos**)。

パラメーター	説明
platform: openstack: clusterOSImageProperties:	インストールプログラムによって Glance にアップロードされる ClusterOSImage に追加するプロパティー。このプロパティーは、**platform.openstack.clusterOSImage** が既存の Glance イメージに設定されている場合は無視されます。 このプロパティーを使用し、ノードあたり 26 PV の RHOSP のデフォルト永続ボリューム (PV) の制限を超過することができます。制限を超えるには、**hw_scsi_model** プロパティーの値を **virtio-scsi** に設定し、**hw_disk_bus** の値を **scsi** に設定します。 このプロパティーを使用し、**hw_qemu_guest_agent** プロパティーを **yes** の値で追加して QEMU ゲストエージェントを有効にすることもできます。 **値:** 文字列プロパティーのセット。以下に例を示します。 clusterOSImageProperties: hw_scsi_model: "virtio-scsi" hw_disk_bus: "scsi" hw_qemu_guest_agent: "yes"
platform: openstack: controlPlanePort: fixedIPs:	マシンが使用するサブネット。 **値:** クラスターのインストールで使用するサブネット名または UUID のリスト。
platform: openstack: controlPlanePort: network:	マシンが使用するネットワーク。 **値:** クラスターのインストールで使用する RHOSP ネットワークの UUID または名前。

パラメーター	説明
platform: openstack: defaultMachinePlatform:	デフォルトのマシンプールプラットフォームの設定。 **値:** { "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }
platform: openstack: ingressFloatingIP:	Ingress ポートに関連付ける既存の Floating IP アドレス。このプロパティーを使用するには、**platform.openstack.externalNetwork** プロパティーも定義する必要があります。 **値:** IP アドレス (例:**128.0.0.1**)。
platform: openstack: apiFloatingIP:	API ロードバランサーに関連付ける既存の Floating IP アドレス。このプロパティーを使用するには、**platform.openstack.externalNetwork** プロパティーも定義する必要があります。 **値:** IP アドレス (例:**128.0.0.1**)。
platform: openstack: externalDNS:	クラスターインスタンスが DNS 解決に使用する外部 DNS サーバーの IP アドレス。 **値:** 文字列形式の IP アドレスのリスト。例: **["8.8.8.8", "192.168.1.12"]**
platform: openstack: loadbalancer:	デフォルトの内部ロードバランサーを使用するかどうか。値が **UserManaged** に設定されている場合、このデフォルトのロードバランサーは無効になり、外部のユーザー管理のロードバランサーを使用するクラスターをデプロイできるようになります。パラメーターが設定されていない場合、または値が **OpenShiftManagedDefault** の場合、クラスターはデフォルトのロードバランサーを使用します。 **値:** **UserManaged** または **OpenShiftManagedDefault**。

パラメーター	説明
platform: openstack: machinesSubnet:	クラスターのノードが使用する RHOSP サブネットの UUID。ノードおよび仮想 IP (VIP) ポートがこのサブネットに作成されます。 **networking.machineNetwork** の最初の項目は **machinesSubnet** の値に一致する必要があります。 カスタムサブネットにデプロイする場合、OpenShift Container Platform インストーラーに外部 DNS サーバーを指定することはできません。代わりに、DNS を RHOSP のサブネットに追加 します。 **値:** 文字列形式の UUID。例:**fa806b2f-ac49-4bce-b9db-124bc64209bf**。

12.1.7. 追加の Google Cloud 設定パラメーター

追加の Google Cloud 設定パラメーターについて、次の表で説明します。

表12.7 追加の Google Cloud パラメーター

パラメーター	説明
controlPlane: platform: gcp: osImage: project:	オプション: デフォルトで、インストールプログラムは、コントロールプレーンマシンの起動に使用する Red Hat Enterprise Linux CoreOS (RHCOS) イメージをダウンロードしてインストールします。インストールプログラムがコントロールプレーンマシンのみに使用するカスタム RHCOS イメージの場所を指定することで、デフォルトの動作をオーバーライドできます。デフォルトのイメージを使用する場合、コントロールプレーンマシンはライセンスコストに影響しません。一方、Google Cloud Marketplace イメージをコントロールプレーンマシンに適用する場合は、使用コストが発生します。 **値:** 文字列。イメージが置かれている Google Cloud プロジェクトの名前。
controlPlane: platform: gcp: osImage: name:	インストールプログラムがコントロールプレーンマシンを起動するために使用するカスタム RHCOS イメージの名前。**controlPlane.platform.gcp.osImage.project** を使用する場合、このフィールドは必須です。 **値:** 文字列。RHCOS イメージの名前。

パラメーター	説明
compute: platform: gcp: osImage: project:	オプション: デフォルトで、インストールプログラムはコンピュートマシンの起動に使用する RHCOS イメージをダウンロードしてインストールします。インストールプログラムがコンピュートマシンのみに使用するカスタム RHCOS イメージの場所を指定することで、デフォルトの動作をオーバーライドできます。 **値:** 文字列。イメージが置かれている Google Cloud プロジェクトの名前。
compute: platform: gcp: osImage: name:	インストールプログラムがコンピュートマシンの起動に使用するカスタム RHCOS イメージの名前。**compute.platform.gcp.osImage.project** を使用する場合、このフィールドは必須です。 **値:** 文字列。RHCOS イメージの名前。
compute: platform: gcp: serviceAccount:	インストール中に使用する Google Cloud サービスアカウントのメールアドレスを指定します。このサービスアカウントは、コンピュートマシンのプロビジョニングに使用されます。 **値:** 文字列。サービスアカウントのメールアドレス。
platform: gcp: network:	クラスターをデプロイする既存 Virtual Private Cloud (VPC) の名前。クラスターを共有 VPC にデプロイする場合は、共有 VPC を含む Google Cloud プロジェクトの名前で **platform.gcp.networkProjectID** を設定する必要があります。 **値:** 文字列。
platform: gcp: networkProjectID:	オプション: クラスターをデプロイする共有 VPC を含む Google Cloud プロジェクトの名前。 **値:** 文字列。
platform: gcp: projectID:	インストールプログラムがクラスターをインストールする Google Cloud プロジェクトの名前。 **値:** 文字列。

パラメーター	説明
platform: gcp: userProvisionedDNS:	デフォルトの cluster-provisioned DNS ソリューションの代わりに、user-provisioned DNS を有効にします。この機能を使用する場合は、**api.<cluster_name>.<base_domain>.** および ***.apps.<cluster_name>.<base_domain>.** のレコードを含む独自の DNS ソリューションを提供する必要があります。 **値:** **Enabled** または **Disabled**。デフォルト値は、**Disabled** です。**userProvisionedDNS** はテクノロジープレビュー機能です。
platform: gcp: region:	クラスターをホストする Google Cloud リージョンの名前。 **値:** **us-central1** などの有効なリージョン名。
platform: gcp: controlPlaneSubnet:	コントロールプレーンマシンをデプロイする既存サブネットの名前。 **値:** サブネット名。
platform: gcp: computeSubnet:	コンピュートマシンをデプロイする既存サブネットの名前。 **値:** サブネット名。
platform: gcp: defaultMachinePlatform: zones:	インストールプログラムがマシンを作成するアベイラビリティーゾーン。 **値:** YAML シーケンス 内の有効な Google Cloud アベイラビリティーゾーン (**us-central1-a** など) のリスト。  **重要** Google Cloud 64 ビット ARM インフラストラクチャーでクラスターを実行する場合は、Ampere Altra Arm CPU が利用可能なゾーンを使用するようにしてください。「Google Cloud アベイラビリティーゾーン」のリンクから、64 ビット ARM プロセッサーと互換性のあるゾーンを確認できます。

パラメーター	説明
platform: gcp: defaultMachinePlatform: osDisk: diskSizeGB:	ディスクのサイズ (GB 単位)。 **値:** 16 GB から 65536 GB までの任意のサイズ。
platform: gcp: defaultMachinePlatform: osDisk: diskType:	Google Cloud ディスクタイプ。 **値:** すべてのマシンのデフォルトディスクタイプ。有効な値は、**pd-balanced**、**pd-ssd**、**pd-standard**、または **hyperdisk-balanced** です。デフォルト値は **pd-ssd** です。コントロールプレーンマシンは **pd-standard** ディスクタイプを使用できないため、デフォルトのマシンプラットフォームディスクタイプとして **pd-standard** を指定する場合は、**controlPlane.platform.gcp.osDisk.diskType** パラメーターを使用して別のディスクタイプを指定する必要があります。
platform: gcp: defaultMachinePlatform: osImage: project:	オプション: デフォルトで、インストールプログラムは、コントロールプレーンおよびコンピュートマシンの起動に使用される RHCOS イメージをダウンロードしてインストールします。インストールプログラムが両方のタイプのマシンに使用するカスタム RHCOS イメージの場所を指定することで、デフォルトの動作をオーバーライドできます。 **値:** 文字列。イメージが置かれている Google Cloud プロジェクトの名前。
platform: gcp: defaultMachinePlatform: osImage: name:	インストールプログラムがコントロールプレーンとコンピュートマシンの起動に使用するカスタム RHCOS イメージの名前。**platform.gcp.defaultMachinePlatform.osImage.project** を使用する場合、このフィールドは必須です。 **値:** 文字列。RHCOS イメージの名前。
platform: gcp: defaultMachinePlatform: tags:	オプション: コントロールプレーンおよびコンピュートマシンに追加する別のネットワークタグ。 **値:** 1 つ以上の文字列 (例:**network-tag1**)。

パラメーター	説明
`platform:` `  gcp:` `  defaultMachinePlatform:` `  type:`	コントロールプレーンおよびコンピュートマシンの Google Cloud マシンタイプ。 **値:** Google Cloud マシンタイプ (例:**n1-standard-4**)。
`platform:` `  gcp:` `  defaultMachinePlatform:` `  osDisk:` `  encryptionKey:` `  kmsKey:` `  name:`	マシンのディスク暗号化に使用されるお客様が管理する暗号化キーの名前。 **値:** 暗号鍵の名前。
`platform:` `  gcp:` `  defaultMachinePlatform:` `  osDisk:` `  encryptionKey:` `  kmsKey:` `  keyRing:`	KMS キーが属する Key Management Service (KMS) キーリングの名前。 **値:** KMS キーリング名。
`platform:` `  gcp:` `  defaultMachinePlatform:` `  osDisk:` `  encryptionKey:` `  kmsKey:` `  location:`	KMS キーリングが存在する Google Cloud の場所。 **値:** Google Cloud の場所。
`platform:` `  gcp:` `  defaultMachinePlatform:` `  osDisk:` `  encryptionKey:` `  kmsKey:` `  projectID:`	KMS キーリングが存在するプロジェクトの ID。この値は、設定されていない場合、デフォルトで **platform.gcp.projectID** パラメーターの値になります。 **値:** Google Cloud プロジェクト ID。

パラメーター	説明
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKeyServiceAccount:	コントロールプレーンおよびコンピュートマシンの暗号化要求に使用される Google Cloud サービスアカウント。存在しない場合には、Compute Engine のデフォルトのサービスアカウントが使用されます。Google Cloud サービスアカウントの詳細は、Google のドキュメントの service accounts を参照してください。 **値:** Google Cloud サービスアカウントのメールアドレス (例: **<service_account_name>@<project_id>.iam.gserviceaccount.com**)。
platform: gcp: defaultMachinePlatform: secureBoot:	クラスター内のすべてのマシンに Shielded VM セキュアブートを有効にするかどうか。Shielded VM には、セキュアブート、ファームウェアと整合性の監視、ルートキット保護などの追加のセキュリティープロトコルがあります。Shielded VM の詳細は、Shielded VM に関する Google のドキュメントを参照してください。 **値**: **Enabled** または **Disabled**。デフォルト値は、**Disabled** です。

パラメーター	説明
platform: gcp: defaultMachinePlatform: confidentialCompute:	クラスター内のすべてのマシンに Confidential 仮想マシンを使用するかどうか。Confidential 仮想マシンは処理中のデータを暗号化します。Confidential Computing の詳細は、Confidential Computing に関する Google のドキュメントを参照してください。 対応している値は次のとおりです。 ● **Enabled**: Confidential Computing プラットフォームが自動的に選択されます。 ● **Disabled**: Confidential Computing を無効にします。 ● **AMDEncryptedVirtualization** は、AMD Secure Encrypted Virtualization (AMD SEV) による Confidential Computing を可能にします。 ● **AMDEncryptedVirtualizationNestedPaging** は、AMD Secure Encrypted Virtualization Secure Nested Paging (AMD SEV-SNP) による Confidential Computing を可能にします。 ● **IntelTrustedDomainExtensions** は、Intel Trusted Domain Extensions (Intel TDX) を使用した Confidential Computing を可能にします。 **Disabled** 以外の値を指定する場合は、**platform.gcp.defaultMachinePlatform.onHostMaintenance** を **Terminate** に設定し、Confidential Computing をサポートするリージョンとマシンタイプを指定する必要があります。詳細は、Supported configurations に関する Google のドキュメントを参照してください。 **値:** 文字列。
platform: gcp: defaultMachinePlatform: onHostMaintenance:	ソフトウェアやハードウェアの更新など、ホストメンテナンスイベント中のすべての VM の動作を指定します。Confidential 仮想マシンの場合は、このパラメーターを **Terminate** に設定する必要があります。Confidential 仮想マシンはライブ VM 移行をサポートしていません。 **値:** **Terminate** または **Migrate**。デフォルト値は、**Migrate** です。

パラメーター	説明
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: name:	コントロールプレーンマシンのディスク暗号化に使用されるお客様が管理する暗号化キーの名前。 **値:** 暗号鍵の名前。
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: keyRing:	コントロールプレーンマシンの場合、KMS キーが属する KMS キーリングの名前。 **値:** KMS キーリング名。
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: location:	コントロールプレーンマシン用のキーリングが存在する Google Cloud の場所。KMS の場所の詳細は、Google のドキュメント Cloud KMS locations を参照してください。 **値:** キーリングの Google Cloud の場所。
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: projectID:	コントロールプレーンマシンの場合、KMS キーリングが存在するプロジェクトの ID。設定されていない場合、この値は VM プロジェクト ID にデフォルト設定されます。 **値:** Google Cloud プロジェクト ID。
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKeyServiceAccount:	コントロールプレーンマシンの暗号化要求に使用される Google Cloud サービスアカウント。存在しない場合には、Compute Engine のデフォルトのサービスアカウントが使用されます。Google Cloud サービスアカウントの詳細は、Google のドキュメントの service accounts を参照してください。 **値:** Google Cloud サービスアカウントのメールアドレス (例: **<service_account_name>@<project_id>.iam.gserviceaccount.com**)。

パラメーター	説明
controlPlane: platform: gcp: osDisk: diskSizeGB:	ディスクのサイズ (GB 単位)。この値はコントロールプレーンマシンに適用されます。 **値:** 16 から 65536 までの任意の整数。
controlPlane: platform: gcp: osDisk: diskType:	コントロールプレーンマシンの Google Cloud ディスクタイプ。 **値:** 有効な値は、**pd-balanced**、**pd-ssd**、または **hyperdisk-balanced** です。デフォルト値は **pd-ssd** です。
controlPlane: platform: gcp: tags:	オプション: コントロールプレーンマシンに追加する別のネットワークタグ。このパラメーターを設定すると、コントロールプレーンマシンの **platform.gcp.defaultMachinePlatform.tags** パラメーターが上書きされます。 **値:** 1 つ以上の文字列 (例:**control-plane-tag1**)。
controlPlane: platform: gcp: type:	コントロールプレーンマシンの Google Cloud マシンタイプ。設定されている場合、このパラメーターは **platform.gcp.defaultMachinePlatform.type** パラメーターを上書きします。 **値:** Google Cloud マシンタイプ (例:**n1-standard-4**)。
controlPlane: platform: gcp: zones:	インストールプログラムがコントロールプレーンマシンを作成するアベイラビリティーゾーン。 **値:** YAML シーケンス 内の有効な Google Cloud アベイラビリティーゾーン (**us-central1-a** など) のリスト。  **重要** Google Cloud 64 ビット ARM インフラストラクチャーでクラスターを実行する場合は、Ampere Altra Arm CPU が利用可能なゾーンを使用するようにしてください。「Google Cloud アベイラビリティーゾーン」のリンクから、64 ビット ARM プロセッサーと互換性のあるゾーンを確認できます。

パラメーター	説明
controlPlane: platform: gcp: secureBoot:	コントロールプレーンマシンの Shielded VM セキュアブートを有効にするかどうか。Shielded VM には、セキュアブート、ファームウェアと整合性の監視、ルートキット保護などの追加のセキュリティープロトコルがあります。Shielded VM の詳細は、Shielded VM に関する Google のドキュメントを参照してください。 **値:** **Enabled** または **Disabled**。デフォルト値は、**Disabled** です。
controlPlane: platform: gcp: confidentialCompute:	コントロールプレーンマシンに Confidential 仮想マシンを使用するかどうか。Confidential 仮想マシンは処理中のデータを暗号化します。Confidential Computing の詳細は、Confidential Computing に関する Google のドキュメントを参照してください。 対応している値は次のとおりです。 ● **Enabled**: Confidential Computing プラットフォームが自動的に選択されます。 ● **Disabled**: Confidential Computing を無効にします。 ● **AMDEncryptedVirtualization** は、AMD Secure Encrypted Virtualization (AMD SEV) による Confidential Computing を可能にします。 ● **AMDEncryptedVirtualizationNestedPaging** は、AMD Secure Encrypted Virtualization Secure Nested Paging (AMD SEV-SNP) による Confidential Computing を可能にします。 ● **IntelTrustedDomainExtensions** は、Intel Trusted Domain Extensions (Intel TDX) を使用した Confidential Computing を可能にします。 **Disabled** 以外の値を指定する場合は、**controlPlane.platform.gcp.defaultMachinePlatform.onHostMaintenance** を **Terminate** に設定する必要があります。 **値:** 文字列。

パラメーター	説明
controlPlane: platform: gcp: onHostMaintenance:	ソフトウェアやハードウェアの更新など、ホストメンテナンスイベント中のコントロールプレーン VM の動作を指定します。Confidential 仮想マシンの場合は、このパラメーターを **Terminate** に設定する必要があります。Confidential 仮想マシンはライブ VM 移行をサポートしていません。 **値:** **Terminate** または **Migrate**。デフォルト値は、**Migrate** です。
controlPlane: platform: gcp: serviceAccount:	インストール中に使用する Google Cloud サービスアカウントのメールアドレスを指定します。このサービスアカウントは、コントロールプレーンマシンのプロビジョニングに使用されます。 **重要** 共有 VPC にインストールする際に、サービスアカウントが指定されていない場合、インストールプログラムのサービスアカウントに、ホストプロジェクトの **resourcemanager.projects.getIamPolicy** 権限と **resourcemanager.projects.setIamPolicy** 権限が付与されている必要があります。 **値:** 文字列。サービスアカウントのメールアドレス。
compute: platform: gcp: osDisk: encryptionKey: kmsKey: name:	コントロールマシンのディスク暗号化に使用されるお客様が管理する暗号化キーの名前。 **値:** 暗号鍵の名前。
compute: platform: gcp: osDisk: encryptionKey: kmsKey: keyRing:	コンピュートマシンの場合、KMS キーが属する KMS キーリングの名前。 **値:** KMS キーリング名。

パラメーター	説明
compute: platform: gcp: osDisk: encryptionKey: kmsKey: location:	コンピュートマシン用のキーリングが存在する Google Cloud の場所。KMS の場所の詳細は、Google のドキュメント Cloud KMS locations を参照してください。 **値:** キーリングの Google Cloud の場所。
compute: platform: gcp: osDisk: encryptionKey: kmsKey: projectID:	コンピュートマシンの場合、KMS キーリングが存在するプロジェクトの ID。設定されていない場合、この値は VM プロジェクト ID にデフォルト設定されます。 **値:** Google Cloud プロジェクト ID。
compute: platform: gcp: osDisk: encryptionKey: kmsKeyServiceAccount:	コンピュートマシンの暗号化要求に使用される Google Cloud サービスアカウント。この値が設定されていない場合には、Compute Engine のデフォルトのサービスアカウントが使用されます。Google Cloud サービスアカウントの詳細は、Google のドキュメントの service accounts を参照してください。 **値:** Google Cloud サービスアカウントのメールアドレス (例: **<service_account_name>@<project_id>.iam.gserviceaccount.com**)。
compute: platform: gcp: osDisk: diskSizeGB:	ディスクのサイズ (GB 単位)。この値はコンピュートマシンに適用されます。 **値:** 16 から 65536 までの任意の整数。
compute: platform: gcp: osDisk: diskType:	コンピュートマシンの Google Cloud ディスクタイプ。 **値:** 有効な値は、**pd-balanced**、**pd-ssd**、**pd-standard**、または **hyperdisk-balanced** です。デフォルト値は **pd-ssd** です。

パラメーター	説明
compute: platform: gcp: tags:	オプション: コンピュートマシンに追加する別のネットワークタグ。このパラメーターを設定すると、コンピュートマシンの **platform.gcp.defaultMachinePlatform.tags** パラメーターが上書きされます。 **値:** 1 つ以上の文字列 (例:**compute-network-tag1**)。
compute: platform: gcp: type:	コンピュートマシンの Google Cloud マシンタイプ。設定されている場合、このパラメーターは **platform.gcp.defaultMachinePlatform.type** パラメーターを上書きします。 **値:** Google Cloud マシンタイプ (例:**n1-standard-4**)。
compute: platform: gcp: zones:	インストールプログラムがコンピュートマシンを作成するアベイラビリティーゾーン。 **値:** YAML シーケンス 内の有効な Google Cloud アベイラビリティーゾーン (**us-central1-a** など) のリスト。  **重要** Google Cloud 64 ビット ARM インフラストラクチャーでクラスターを実行する場合は、Ampere Altra Arm CPU が利用可能なゾーンを使用するようにしてください。「Google Cloud アベイラビリティーゾーン」のリンクから、64 ビット ARM プロセッサーと互換性のあるゾーンを確認できます。
compute: platform: gcp: secureBoot:	コンピュートマシン用に Shielded VM のセキュアブートを有効にするかどうか。Shielded VM には、セキュアブート、ファームウェアと整合性の監視、ルートキット保護などの追加のセキュリティープロトコルがあります。Shielded VM の詳細は、Shielded VM に関する Google のドキュメントを参照してください。 **値:** **Enabled** または **Disabled**。デフォルト値は、**Disabled** です。

パラメーター	説明
compute: platform: gcp: confidentialCompute:	コンピュートマシンに Confidential 仮想マシンを使用するかどうか。Confidential 仮想マシンは処理中のデータを暗号化します。Confidential Computing の詳細は、Confidential Computing に関する Google のドキュメントを参照してください。 対応している値は次のとおりです。 ● **Enabled**: Confidential Computing プラットフォームが自動的に選択されます。 ● **Disabled**: Confidential Computing を無効にします。 ● **AMDEncryptedVirtualization** は、AMD Secure Encrypted Virtualization (AMD SEV) による Confidential Computing を可能にします。 ● **AMDEncryptedVirtualizationNestedPaging** は、AMD Secure Encrypted Virtualization Secure Nested Paging (AMD SEV-SNP) による Confidential Computing を可能にします。 ● **IntelTrustedDomainExtensions** は、Intel Trusted Domain Extensions (Intel TDX) を使用した Confidential Computing を可能にします。 **Disabled** 以外の値を指定する場合は、**compute.platform.gcp.onHostMaintenance** を **Terminate** に設定する必要があります。 **値:** 文字列。
compute: platform: gcp: onHostMaintenance:	ソフトウェアやハードウェアの更新など、ホストメンテナンスイベント中のコンピューティング VM の動作を指定します。Confidential 仮想マシンの場合は、このパラメーターを **Terminate** に設定する必要があります。Confidential 仮想マシンはライブ VM 移行をサポートしていません。 **値:** **Terminate** または **Migrate**。デフォルト値は、**Migrate** です。