



OpenShift Container Platform 4.6

OpenStack へのインストール

OpenShift Container Platform OpenStack クラスターのインストール

OpenShift Container Platform 4.6 OpenStack へのインストール

OpenShift Container Platform OpenStack クラスターのインストール

Enter your first name here. Enter your surname here.

Enter your organisation's name here. Enter your organisational division here.

Enter your email address here.

法律上の通知

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Installing_on_OpenStack.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

本書では、OpenStack Platform に OpenShift Container Platform クラスターをインストールし、アンインストールする方法について説明します。

目次

第1章 OPENSTACK へのインストール	6
1.1. カスタマイズによる OPENSTACK へのクラスターのインストール	6
1.1.1. 前提条件	6
1.1.2. 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	6
1.1.2.1. コントロールプレーンマシン	7
1.1.2.2. コンピュートマシン	7
1.1.2.3. ブートストラップマシン	8
1.1.3. OpenShift Container Platform のインターネットアクセス	8
1.1.4. RHOSP での Swift の有効化	8
1.1.5. 外部ネットワークアクセスの確認	9
1.1.6. インストールプログラムのパラメーターの定義	10
1.1.7. インストールプログラムの取得	12
1.1.8. インストール設定ファイルの作成	13
1.1.8.1. インストール時のクラスター全体のプロキシの設定	14
1.1.9. インストール設定パラメーター	16
1.1.9.1. 必須設定パラメーター	16
1.1.9.2. ネットワーク設定パラメーター	18
1.1.9.3. オプションの設定パラメーター	19
1.1.9.4. 追加の Red Hat OpenStack Platform (RHOSP) 設定パラメーター	23
1.1.9.5. オプションの RHOSP 設定パラメーター	24
1.1.9.6. RHOSP デプロイメントでのカスタムサブネット	27
1.1.9.7. RHOSP のカスタマイズされた install-config.yaml ファイルのサンプル	28
1.1.10. コンピュートマシンのアフィニティーの設定	29
1.1.11. SSH プライベートキーの生成およびエージェントへの追加	31
1.1.12. 環境へのアクセスの有効化	32
1.1.12.1. floating IP アドレスを使ったアクセスの有効化	32
1.1.12.2. Floating IP アドレスなしでのインストールの完了	33
1.1.13. クラスターのデプロイ	34
1.1.14. クラスターステータスの確認	35
1.1.15. CLI の使用によるクラスターへのログイン	36
1.1.16. OpenShift Container Platform の Telemetry アクセス	37
1.1.17. 次のステップ	37
1.2. KURYR を使用する OPENSTACK へのクラスターのインストール	37
1.2.1. 前提条件	37
1.2.2. Kuryr SDN について	38
1.2.3. Kuryr を使用して OpenShift Container Platform を RHOSP にインストールするためのリソースのガイドライン	39
1.2.3.1. クォータの拡大	40
1.2.3.2. Neutron の設定	41
1.2.3.3. Octavia の設定	41
1.2.3.3.1. Octavia OVN ドライバー	44
1.2.3.4. Kuryr を使用したインストールについての既知の制限	45
RHOSP の一般的な制限	45
RHOSP バージョンの制限	45
RHOSP 環境の制限	46
RHOSP のアップグレードの制限	46
1.2.3.5. コントロールプレーンマシン	47
1.2.3.6. コンピュートマシン	47
1.2.3.7. ブートストラップマシン	47
1.2.4. OpenShift Container Platform のインターネットアクセス	48
1.2.5. RHOSP での Swift の有効化	48

1.2.6. 外部ネットワークアクセスの確認	49
1.2.7. インストールプログラムのパラメーターの定義	50
1.2.8. インストールプログラムの取得	52
1.2.9. インストール設定ファイルの作成	52
1.2.9.1. インストール時のクラスター全体のプロキシの設定	54
1.2.10. インストール設定パラメーター	55
1.2.10.1. 必須設定パラメーター	56
1.2.10.2. ネットワーク設定パラメーター	57
1.2.10.3. オプションの設定パラメーター	59
1.2.10.4. 追加の Red Hat OpenStack Platform (RHOSP) 設定パラメーター	63
1.2.10.5. オプションの RHOSP 設定パラメーター	64
1.2.10.6. RHOSP デプロイメントでのカスタムサブネット	67
1.2.10.7. Kuryr を使用した OpenStack のカスタマイズされた install-config.yaml ファイルのサンプル	68
1.2.10.8. Kuryr ポートプール	69
1.2.10.9. インストール時の Kuryr ポートプールの調整	69
1.2.11. コンピュータマシンのアフィニティーの設定	71
1.2.12. SSH プライベートキーの生成およびエージェントへの追加	73
1.2.13. 環境へのアクセスの有効化	75
1.2.13.1. floating IP アドレスを使ったアクセスの有効化	75
1.2.13.2. Floating IP アドレスなしでのインストールの完了	76
1.2.14. クラスターのデプロイ	77
1.2.15. クラスターステータスの確認	78
1.2.16. CLI の使用によるクラスターへのログイン	79
1.2.17. OpenShift Container Platform の Telemetry アクセス	80
1.2.18. 次のステップ	80
1.3. 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	80
1.3.1. 前提条件	81
1.3.2. OpenShift Container Platform のインターネットアクセス	81
1.3.3. 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	81
1.3.3.1. コントロールプレーンマシン	82
1.3.3.2. コンピュータマシン	83
1.3.3.3. ブートストラップマシン	83
1.3.4. Playbook 依存関係のダウンロード	83
1.3.5. インストール Playbook のダウンロード	84
1.3.6. インストールプログラムの取得	85
1.3.7. SSH プライベートキーの生成およびエージェントへの追加	86
1.3.8. Red Hat Enterprise Linux CoreOS (RHCOS) イメージの作成	88
1.3.9. 外部ネットワークアクセスの確認	89
1.3.10. 環境へのアクセスの有効化	89
1.3.10.1. floating IP アドレスを使ったアクセスの有効化	90
1.3.10.2. Floating IP アドレスなしでのインストールの完了	91
1.3.11. インストールプログラムのパラメーターの定義	92
1.3.12. インストール設定ファイルの作成	93
1.3.13. インストール設定パラメーター	95
1.3.13.1. 必須設定パラメーター	95
1.3.13.2. ネットワーク設定パラメーター	97
1.3.13.3. オプションの設定パラメーター	98
1.3.13.4. 追加の Red Hat OpenStack Platform (RHOSP) 設定パラメーター	103
1.3.13.5. オプションの RHOSP 設定パラメーター	104
1.3.13.6. RHOSP デプロイメントでのカスタムサブネット	107
1.3.13.7. RHOSP のカスタマイズされた install-config.yaml ファイルのサンプル	108
1.3.13.8. マシンのカスタムサブネットの設定	109
1.3.13.9. コンピュータマシンプールを空にする	109

1.3.14. Kubernetes マニフェストおよび Ignition 設定ファイルの作成	110
1.3.15. ブートストラップ Ignition ファイルの準備	111
1.3.16. RHOSP でのコントロールプレーンの Ignition 設定ファイルの作成	114
1.3.17. RHOSP でのネットワークリソースの作成	115
1.3.18. RHOSP でのブートストラップマシンの作成	117
1.3.19. RHOSP でのコントロールプレーンの作成	117
1.3.20. CLI の使用によるクラスターへのログイン	118
1.3.21. RHOSP からのブートストラップリソースの削除	118
1.3.22. RHOSP でのコンピュートマシンの作成	119
1.3.23. マシンの証明書署名要求の承認	120
1.3.24. インストールの正常な実行の確認	123
1.3.25. OpenShift Container Platform の Telemetry アクセス	123
1.3.26. 次のステップ	123
1.4. 独自のインフラストラクチャーでの KURYR を使用する OPENSTACK へのクラスターのインストール	123
1.4.1. 前提条件	124
1.4.2. Kuryr SDN について	124
1.4.3. Kuryr を使用して OpenShift Container Platform を RHOSP にインストールするためのリソースのガイド ライン	125
1.4.3.1. クォータの拡大	127
1.4.3.2. Neutron の設定	127
1.4.3.3. Octavia の設定	127
1.4.3.3.1. Octavia OVN ドライバー	131
1.4.3.4. Kuryr を使用したインストールについての既知の制限	131
RHOSP の一般的な制限	131
RHOSP バージョンの制限	132
RHOSP 環境の制限	132
RHOSP のアップグレードの制限	132
1.4.3.5. コントロールプレーンマシン	133
1.4.3.6. コンピュートマシン	133
1.4.3.7. ブートストラップマシン	134
1.4.4. OpenShift Container Platform のインターネットアクセス	134
1.4.5. Playbook 依存関係のダウンロード	134
1.4.6. インストール Playbook のダウンロード	135
1.4.7. インストールプログラムの取得	136
1.4.8. SSH プライベートキーの生成およびエージェントへの追加	137
1.4.9. Red Hat Enterprise Linux CoreOS (RHCOS) イメージの作成	139
1.4.10. 外部ネットワークアクセスの確認	140
1.4.11. 環境へのアクセスの有効化	140
1.4.11.1. floating IP アドレスを使ったアクセスの有効化	141
1.4.11.2. Floating IP アドレスなしでのインストールの完了	142
1.4.12. インストールプログラムのパラメーターの定義	143
1.4.13. インストール設定ファイルの作成	144
1.4.14. インストール設定パラメーター	146
1.4.14.1. 必須設定パラメーター	146
1.4.14.2. ネットワーク設定パラメーター	148
1.4.14.3. オプションの設定パラメーター	149
1.4.14.4. 追加の Red Hat OpenStack Platform (RHOSP) 設定パラメーター	154
1.4.14.5. オプションの RHOSP 設定パラメーター	155
1.4.14.6. RHOSP デプロイメントでのカスタムサブネット	158
1.4.14.7. Kuryr を使用した OpenStack のカスタマイズされた install-config.yaml ファイルのサンプル	159
1.4.14.8. Kuryr ポートプール	160
1.4.14.9. インストール時の Kuryr ポートプールの調整	160
1.4.14.10. マシンのカスタムサブネットの設定	162

1.4.14.11. コンピュートマシンプールを空にする	163
1.4.14.12. ネットワークタイプの変更	163
1.4.15. Kubernetes マニフェストおよび Ignition 設定ファイルの作成	164
1.4.16. ブートストラップ Ignition ファイルの準備	166
1.4.17. RHOSP でのコントロールプレーンの Ignition 設定ファイルの作成	168
1.4.18. RHOSP でのネットワークリソースの作成	169
1.4.19. RHOSP でのブートストラップマシンの作成	171
1.4.20. RHOSP でのコントロールプレーンの作成	171
1.4.21. CLI の使用によるクラスターへのログイン	172
1.4.22. RHOSP からのブートストラップリソースの削除	173
1.4.23. RHOSP でのコンピュートマシンの作成	173
1.4.24. マシンの証明書署名要求の承認	174
1.4.25. インストールの正常な実行の確認	177
1.4.26. OpenShift Container Platform の Telemetry アクセス	177
1.4.27. 次のステップ	177
1.5. 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	178
1.5.1. 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	178
1.5.1.1. その他の制限	179
1.5.2. 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	179
1.5.2.1. コントロールプレーンマシン	180
1.5.2.2. コンピュートマシン	180
1.5.2.3. ブートストラップマシン	180
1.5.3. OpenShift Container Platform のインターネットアクセス	181
1.5.4. RHOSP での Swift の有効化	181
1.5.5. インストールプログラムのパラメーターの定義	182
1.5.6. 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	183
1.5.7. インストール設定ファイルの作成	184
1.5.7.1. インストール時のクラスター全体のプロキシの設定	187
1.5.7.2. インストール設定パラメーター	188
1.5.7.2.1. 必須設定パラメーター	189
1.5.7.2.2. ネットワーク設定パラメーター	190
1.5.7.2.3. オプションの設定パラメーター	192
1.5.7.2.4. 追加の Red Hat OpenStack Platform (RHOSP) 設定パラメーター	197
1.5.7.2.5. オプションの RHOSP 設定パラメーター	198
1.5.7.3. 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	201
1.5.8. コンピュートマシンのアフィニティーの設定	202
1.5.9. SSH プライベートキーの生成およびエージェントへの追加	204
1.5.10. 環境へのアクセスの有効化	206
1.5.10.1. floating IP アドレスを使ったアクセスの有効化	206
1.5.10.2. Floating IP アドレスなしでのインストールの完了	207
1.5.11. クラスターのデプロイ	208
1.5.12. クラスターステータスの確認	209
1.5.13. CLI の使用によるクラスターへのログイン	210
1.5.14. デフォルトの OperatorHub ソースの無効化	211
1.5.15. OpenShift Container Platform の Telemetry アクセス	211
1.5.16. 次のステップ	211
1.6. OPENSTACK でのクラスターのアンインストール	212
1.6.1. インストーラーでプロビジョニングされるインフラストラクチャーを使用するクラスターの削除	212
1.7. 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	213
1.7.1. Playbook 依存関係のダウンロード	213
1.7.2. 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	214

第1章 OPENSTACK へのインストール

1.1. カスタマイズによる OPENSTACK へのクラスタのインストール

OpenShift Container Platform バージョン 4.6 では、Red Hat OpenStack Platform (RHOSP) にカスタマイズされたクラスタ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。

1.1.1. 前提条件

- [OpenShift Container Platform のインストールおよび更新](#) プロセスについての詳細を確認します。
 - OpenShift Container Platform 4.6 が **Available platforms** セクションの RHOSP バージョンと互換性があることを確認します。[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。
- ネットワーク設定がプロバイダーのネットワークに依存しないことを確認します。プロバイダーネットワークはサポートされません。
- 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必要があります。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、[Optimizing storage](#) 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ます。

1.1.2. 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

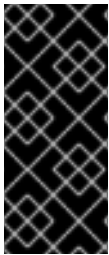
OpenShift Container Platform のインストールをサポートするために、Red Hat OpenStack Platform (RHOSP) クォータは以下の要件を満たす必要があります。

表1.1 RHOSP のデフォルトの OpenShift Container Platform クラスタについての推奨リソース

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	112 GB
vCPU	28
ボリュームストレージ	275 GB

リソース	値
インスタンス	7
セキュリティグループ	3
セキュリティグループルール	60

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、セキュリティグループおよびセキュリティ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、コンピューターマシン、およびブートストラップマシンで設定されます。

1.1.2.1. 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、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.1.2.2. コンピューターマシン

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、2 つの vCPU および 100 GB のストレージ領域があるフレーバー

ヒント

コンピュータマシンは、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。

1.1.2.3. 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、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.1.3. OpenShift Container Platform のインターネットアクセス

OpenShift Container Platform 4.6 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になります。

インターネットへのアクセスは以下を実行するために必要です。

- [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。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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クラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、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。

1.1.4. 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

Swift として知られる [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。

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。

1.1.5. 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一覧に表示されます。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。

1.1.6. 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** についての詳細は、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: shiftstack_user
      password: XXX
      user_domain_name: Default
      project_domain_name: Default
    dev-env:
      region_name: RegionOne
      auth:
        username: 'devuser'
        password: XXX
        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. マシンを認証局の信頼バンドルに追加します。

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

- c. 信頼バンドルを更新します。

```
$ sudo update-ca-trust extract
```

- d. **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。

1.1.7. インストールプログラムの取得

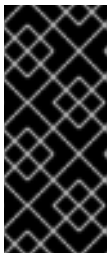
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

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。

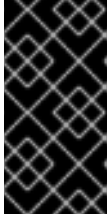
手順

1. OpenShift Cluster Manager サイトの [インフラストラクチャプロバイダー](#) ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. インフラストラクチャプロバイダーを選択します。
3. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、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。ファイル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。



重要

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。

4. 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
```

5. [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。

1.1.8. 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を取得します。

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。



重要

空のディレクトリーを指定します。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の有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。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コピーを行ってください。

- 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。

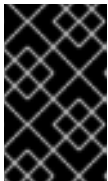
- i. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。



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がある実稼働用の 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およびコンピューターノードに使用する 16 GB 以上の RAM で 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。
 - viii. [Red Hat OpenShift Cluster Manager からプルシークレット](#) を貼り付けます。
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については、[インストール設定パラメーターセクション](#)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

使用可能なパラメーターの詳細については、[インストール設定パラメーター セクション](#) を参照してください。

1.1.8.1. インストール時のクラスター全体のプロキシの設定

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 プロキシを使用することができます。プロキシ設定を **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があるサイトを確認済みで、それらのいずれかがプロキシ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ついてのクラウドプロバイダー API に対する呼び出しを含む) はプロキシされます。プロキシ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)、Google Cloud Platform (GCP)、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> ❶
  httpsProxy: https://<username>:<pswd>@<ip>:<port> ❷
  noProxy: example.com ❸
additionalTrustBundle: | ❹
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

- ❶ 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 URL。URL スキームは **http** である必要があります。
- ❷ 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 URL。
- ❸ プロキシ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 のコンマ区切りの一覧。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をバイパスします。
- ❹ 指定されている場合には、インストールプログラムは、**openshift-config** namespace に **user-ca-bundle** という名前の設定魔府を生成して、追加の CA 証明書を保存します。**additionalTrustBundle** と少なくとも1つのプロキシ設定を指定した場合には、**Proxy** オブジェクトは **trusted CA** フィールドで **user-ca-bundle** 設定マップを参照するように設定されます。その後、Cluster Network Operator は、**trustedCA** パラメーターに指定されたコンテンツを RHCOS トラストバンドルにマージする **trusted-ca-bundle** 設定マップを作成します。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が 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。



注記

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

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を作成します。プロキシ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、追加のプロキシを作成することはできません。

1.1.9. インストール設定パラメーター

OpenShift Container Platform クラスターをデプロイする前に、クラスターをホストするクラウドプラットフォームでアカウントを記述し、クラスターのプラットフォームをオプションでカスタマイズするためにパラメーターの値を指定します。**install-config.yaml** インストール設定ファイルを作成する際に、コマンドラインで必要なパラメーターの値を指定します。クラスターをカスタマイズする場合、**install-config.yaml** ファイルを変更して、プラットフォームについての詳細情報を指定できます。



注記

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。



重要

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.1.9.1. 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1.2 必須パラメーター

パラメーター	説明	値
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。インストーラーは、古い API バージョンをサポートすることもできます。	文字列


パラメーター	説明	値
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、 baseDomain と <metadata.name> 、 <baseDomain> 形式を使用する metadata.name パラメーターの値の組み合わせです。	example.com などの完全修飾ドメイン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。	オブジェクト
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。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。文字列は 14 文字以上でなければなりません。
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 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について参照してください。	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イメージをダウンロードすることを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>


1.1.9.2. 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するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.3 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	オブジェクト  注記 インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN 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。	オブジェクトの配列。以下に例を示します。 <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
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	CIDR (Classless Inter-Domain Routing) 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。
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、 $2^{(32 - 23) - 2}$ Pod IP アドレスを提供します。	サブネット接頭辞。 デフォルト値は 23 です。


パラメーター	説明	値
networking.serviceNetwork	<p>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。</p> <p>OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。</p>	<p>CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>マシンの IP アドレスブロック。</p> <p>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。</p>	<p>オブジェクトの配列。以下に例を示します。</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、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。</p>	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>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。</p> </div> </div>


1.1.9.3. 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1.4 オプションのパラメーター

パラメーター	説明	値
additionalTrustBundle	<p>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。この信頼バンドルは、プロキシが設定される際にも使用できます。</p>	文字列
compute	<p>コンピュータノードを設定するマシンの設定。</p>	<p>machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。</p>

パラメーター	説明	値
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列
compute.hyperthreading	<p>コンピュータ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にされます。</p> <div>  <p>重要</p> <p>同時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ていることを確認します。</p> </div>	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。	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。	aws 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (ワーカーマシンとしても知られる) の数。	2 以上の正の整数。デフォルト値は 3 です。
controlPlane	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。

パラメーター	説明	値
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列
controlPlane.hyperthreading	<p>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にされます。</p> <div>  <div> <p>重要</p> <p>同時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ていることを確認します。</p> </div> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	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。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 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。	サポートされる値は 3 のみです (これはデフォルト値です)。

パラメーター	説明	値
credentialsMode	<p>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。</p> <div>  <p>注記</p> <p>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、Red Hat Operator のクラウド認証情報 Operatorを参照してください。</p> </div>	Mint、Passthrough、Manual 、または空の文字列 ("")。
fips	<p>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 モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>重要</p> <p>FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、x86_64 アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。</p> </div> <div>  <p>注記</p> <p>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。</p> </div>	false または true

パラメーター	説明	値
imageContentSources	release-image コンテンツのソースおよびリポジトリ。	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を指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。
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。	<p>Internal または External。デフォルト値は External です。</p> <p>このパラメーターを Internal に設定することは、クラウド以外のプラットフォームではサポートされません。</p> <div>  <div> <p>重要</p> <p>フィールドの値が Internal に設定されている場合、クラスターは機能しなくなります。詳細は、BZ#1953035 を参照してください。</p> </div> </div>
sshKey	<p>クラスターマシンへのアクセスを認証するための単一または複数の SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の 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。</p> </div> </div>	<p>1つ以上のキー。以下に例を示します。</p> <pre>sshKey: <key1> <key2> <key3></pre>

1.1.9.4. 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1.5 追加の RHOSP パラメーター

パラメーター	説明	値
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。	整数 (例: 30)。
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。	文字列 (例: performance)。
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。	整数 (例: 30)。
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。	文字列 (例: performance)。
platform.openstack.cloud	clouds.yaml ファイルのクラウド一覧にある使用する RHOSP クラウドの名前。	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。	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。	文字列 (例: m1.xlarge)。

1.1.9.5. 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1.6 オプションの RHOSP パラメーター

パラメーター	説明	値
compute.platform.openstack.additionalNetworks	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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。

パラメーター	説明	値
compute.platform.openstack.additionalSecurityGroupIDs	コンピュータマシンに関連付けられた追加のセキュリティグループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。
compute.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: ["zone-1", "zone-2"] 。
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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。
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グループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
controlPlane.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: <code>["zone-1", "zone-2"]</code> 。
platform.openstack.clusterOSImage	<p>インストーラーが RHCOS イメージをダウンロードする場所。</p> <p>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。</p>	<p>HTTP または HTTPS の URL (オプションで SHA-256 形式のチェックサムを使用)。</p> <p>例: <code>http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</code>。この値は、既存の Glance イメージの名前にもなり得ます (例: <code>my-rhcos</code>)。</p>
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。	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>Ingress ポートに関連付ける既存の Floating IP アドレス。このプロパティを使用するには、platform.openstack.externalNetwork プロパティも定義する必要があります。</p>	IP アドレス (例: <code>128.0.0.1</code>)。

パラメーター	説明	値
platform.openstack.lbFloatingIP	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。	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。	文字列としての IP アドレスの一覧。例: ["8.8.8.8", "192.168.1.12"]
platform.openstack.machinesSubnet	<p>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。</p> <p>networking.machineNetwork の最初の項目は machinesSubnet の値に一致する必要があります。</p> <p>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。</p>	文字列としての UUID。例: fa806b2f-ac49-4bceb9db-124bc64209bf 。

1.1.9.6. 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。ノードとポートはこの上に作成されます。

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、以下を確認します。

- ターゲットネットワークおよびサブネットが利用可能である。
- DHCP がターゲットサブネットで有効にされている。
- ターゲットネットワーク上でポートを作成するためのパーミッションがあるインストーラー認証情報を指定できます。

- ネットワーク設定にルーターが必要な場合、これは RHOSP で作成されます。一部の設定は、Floating IP アドレスの変換用のルーターに依存します。
- ネットワーク設定は、プロバイダーのネットワークに依存しません。プロバイダーネットワークはサポートされません。



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.apiVIP** および **platform.openstack.ingressVIP** の値を設定します。

1.1.9.7. RHOSP 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
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: m1.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: OpenShiftSDN
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
```

```
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

1.1.10. コンピュータマシンのアフィニティーの設定

オプションで、インストール時にコンピュータマシンのアフィニティーポリシーを設定できます。インストーラーは、デフォルトでコンピュータマシンのアフィニティーポリシーを選択しません。

インストール後に特定の RHOSP サーバーグループを使用するマシンセットを作成することもできます。



注記

コントロールプレーンマシンは、**soft-anti-affinity** ポリシーで作成されます。

ヒント

[RHOSP インスタンスのスケジューリングおよび配置](#)の詳細は、RHOSP のドキュメントを参照してください。

前提条件

- **install-config.yaml** ファイルを作成し、これに対する変更を完了します。

手順

1. RHOSP コマンドラインインターフェイスを使用して、コンピュータマシンのサーバーグループを作成します。以下に例を示します。

```
$ openstack \
  --os-compute-api-version=2.15 \
  server group create \
  --policy anti-affinity \
  my-openshift-worker-group
```

詳細は、[server group create コマンドのドキュメント](#) を参照してください。

2. 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、マニフェストを作成します。

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

ここでは、以下ようになります。

installation_directory

クラスターの **install-config.yaml** ファイルが含まれるディレクトリーの名前を指定します。

3. **MachineSet** 定義ファイルの **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** を作成します。
4. **spec.template.spec.providerSpec.value** プロパティーの下にある定義に、プロパティー **serverGroupID** を追加します。以下に例を示します。

```

apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
    machine.openshift.io/cluster-api-machine-role: <node_role>
    machine.openshift.io/cluster-api-machine-type: <node_role>
  name: <infrastructure_ID>-<node_role>
  namespace: openshift-machine-api
spec:
  replicas: <number_of_replicas>
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
      machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
        machine.openshift.io/cluster-api-machine-role: <node_role>
        machine.openshift.io/cluster-api-machine-type: <node_role>
        machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
    spec:
      providerSpec:
        value:
          apiVersion: openstackproviderconfig.openshift.io/v1alpha1
          cloudName: openstack
          cloudsSecret:
            name: openstack-cloud-credentials
            namespace: openshift-machine-api
          flavor: <nova_flavor>
          image: <glance_image_name_or_location>
          serverGroupID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeeeee 1
          kind: OpenstackProviderSpec
          networks:
            - filter: {}
              subnets:
                - filter:
                    name: <subnet_name>
                    tags: openshiftClusterID=<infrastructure_ID>
          securityGroups:
            - filter: {}
              name: <infrastructure_ID>-<node_role>
          serverMetadata:
            Name: <infrastructure_ID>-<node_role>
            openshiftClusterID: <infrastructure_ID>
          tags:
            - openshiftClusterID=<infrastructure_ID>
          trunk: true
          userDataSecret:
            name: <node_role>-user-data
          availabilityZone: <optional_openstack_availability_zone>

```

1 サーバグループの UUID をここに追加します。

5. オプション: **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** ファイルをバックアップします。インストールプログラムは、クラスターの作成時に **manifests/** ディレクトリーを削除します。

クラスターのインストール時に、インストーラーは変更した **MachineSet** 定義を使用して RHOSP サーバーグループ内にコンピュータマシンを作成します。

1.1.11. SSH プライベートキーの生成およびエージェントへの追加

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの **~/.ssh/authorized_keys** 一覧に追加されます。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、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 **~/.ssh/id_rsa** などの、新規 SSH キーのパスおよびファイル名を指定します。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。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは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. **ssh-agent** プロセスを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。

- SSH プライベートキーを **ssh-agent** に追加します。

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

出力例

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

- 1 ~/.ssh/id_rsa 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。

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。

1.1.12. 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。

1.1.12.1. floating IP アドレスを使っ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

- 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>
```

- 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>
```

- 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-oc-auth-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.lbFloatingIP**

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があります。

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。

1.1.12.2. Floating IP アドレスなしでのインストールの完了

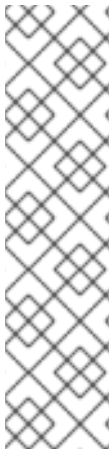
Floating IP アドレスを指定せずに OpenShift Container Platform を Red Hat OpenStack Platform (RHOSP) にインストールすることができます。

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.lbFloatingIP**

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。

1.1.13. 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を取得します。

手順

1. 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、クラスターのデプロイメントを初期化します。

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

- ❶ **<installation_directory>** については、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。
- ❷ 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。



注記

ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

クラスターのデプロイメントが完了すると、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報を含む、クラスターにアクセスするための指示がターミナルに表示されます。

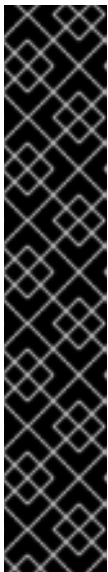
出力例

```
...
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: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



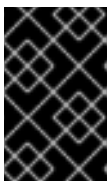
注記

クラスターアクセスおよび認証情報の情報は、インストールが正常に実行される際に **<installation_directory>/openshift_install.log** に出力されます。



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** についての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お勧めします。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。



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。

1.1.14. 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 ❶
```

- ❶ <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ついての情報が含まれます。

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ティング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
```

1.1.15. 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ついての情報が含まれます。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。
- **oc** CLI をインストールしていること。

手順

1. **kubeadmin** 認証情報をエクスポートします。

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

- ❶ <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. 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 へのアクセスおよびその詳細は、[Web コンソールへのアクセス](#) について参照してください。

1.1.16. OpenShift Container Platform の Telemetry アクセス

OpenShift Container Platform 4.6 では、クラスターのヘルス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る Telemetry サービスには、インターネットアクセスが必要です。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、[リモートヘルスマニターリング](#) について参照してください。

1.1.17. 次のステップ

- [クラスターをカスタマイズ](#) します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 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 クラスタトラフィックを設定](#) します。
- RHOSP が 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設定しなかった場合には、[RHOSP のアクセスを Floating IP アドレスで設定](#) します。

1.2. KURYR を使用する OPENSTACK へのクラスターのインストール

OpenShift Container Platform バージョン 4.6 では、Kuryr SDN を使用する Red Hat OpenStack Platform (RHOSP) にカスタマイズされ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。

1.2.1. 前提条件

- [OpenShift Container Platform のインストールおよび更新](#) プロセスについての詳細を確認します。

- OpenShift Container Platform 4.6 が **Available platforms** セクションの RHOSP バージョンと互換性があることを確認します。[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。
- ネットワーク設定がプロバイダーのネットワークに依存しないことを確認します。プロバイダーネットワークはサポートされません。
- 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必要があります。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、[Optimizing storage](#) を参照してください。

1.2.2. Kuryr SDN について

Kuryr は、[Neutron](#) および [Octavia](#) Red Hat OpenStack Platform (RHOSP) サービスを使用して Pod およびサービスのネットワークを提供する Container Network Interface (CNI) プラグインです。

Kuryr と OpenShift Container Platform の統合は主に、RHOSP の仮想マシンで実行する OpenShift Container Platform クラスター用に設計されました。Kuryr は、OpenShift Container Platform Pod を RHOSP SDN にプラグインしてネットワークのパフォーマンスを強化します。さらに、これは Pod と RHOSP 仮想インスタンス間の接続を可能にします。

Kuryr コンポーネントは **openshift-kuryr** namespace を使用して OpenShift Container Platform の Pod としてインストールされます。

- **kuryr-controller: master** ノードにインストールされる単一のサービスインスタンスです。これは、OpenShift Container Platform で **Deployment** としてモデリングされます。
- **kuryr-cni**: 各 OpenShift Container Platform ノードで Kuryr を CNI ドライバーとしてインストールし、設定するコンテナです。これは、OpenShift Container Platform で **DaemonSet** オブジェクトとしてモデリングされます。

Kuryr コントローラーは OpenShift Container Platform API サーバーで Pod、サービスおよび namespace の作成、更新、および削除イベントについて監視します。これは、OpenShift Container Platform API 呼び出しを Neutron および Octavia の対応するオブジェクトにマップします。そのため、Neutron トランクポート機能を実装するすべてのネットワークソリューションを使用して、Kuryr 経由で OpenShift Container Platform をサポートすることができます。これには、Open vSwitch (OVS) および Open Virtual Network (OVN) などのオープンソースソリューションや Neutron と互換性のある市販の SDN が含まれます。

Kuryr は、カプセル化された RHOSP テナントネットワーク上の OpenShift Container Platform デプロイメントに使用することが推奨されています。これは、RHOSP ネットワークでカプセル化された OpenShift Container Platform SDN を実行するなど、二重のカプセル化を防ぐために必要です。

プロバイダーネットワークまたはテナント VLAN を使用する場合は、二重のカプセル化を防ぐために Kuryr を使用する必要はありません。パフォーマンス上の利点はそれほど多くありません。ただし、設定によっては、Kuryr を使用して 2 つのオーバーレイが使用されないようにすることは利点がある場合があります。

Kuryr は、以下のすべての基準が true であるデプロイメントでは推奨されません。

- RHOSP のバージョンが 16 よりも前のバージョンである。
- デプロイメントで UDP サービスが使用されているか、または少数のハイパーバイザーで多数の TCP サービスが使用されている。

または、以下を実行します。

- **ovn-octavia** Octavia ドライバーが無効にされている。
- デプロイメントで、少数のハイパーバイザーで多数の TCP サービスが使用されている。

1.2.3. Kuryr を使用して OpenShift Container Platform を RHOSP にインストールするためのリソースのガイドライン

Kuryr SDN を使用する場合、Pod、サービス、namespace およびネットワークポリシーは RHOSP クォータのリソースを使用します。これにより、最小要件が増加します。また、Kuryr にはデフォルトインストールに必要な要件以外の追加要件があります。

以下のクォータを使用してデフォルトのクラスターの最小要件を満たすようにします。

表1.7 Kuryr を使用する RHOSP のデフォルト OpenShift Container Platform クラスターについての推奨リソース

リソース	値
Floating IP アドレス	3: LoadBalancer タイプに予想されるサービス数
ポート	1500: Pod ごとに1つ必要
ルーター	1
サブネット	250: namespace/プロジェクトごとに1つ必要
ネットワーク	250: namespace/プロジェクトごとに1つ必要
RAM	112 GB
vCPU	28
ボリュームストレージ	275 GB
インスタンス	7
セキュリティグループ	250: サービスおよび NetworkPolicy ごとに1つ必要
セキュリティグループルール	1000
ロードバランサー	100: サービスごとに1つ必要
ロードバランサーリスナー	500: サービスで公開されるポートごとに1つ必要
ロードバランサーノード	500: サービスで公開されるポートごとに1つ必要

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。



重要

OVN Octavia ドライバーではなく Amphora ドライバーで Red Hat OpenStack Platform(RHOSP) バージョン 16 を使用している場合、セキュリティーグループはユーザープロジェクトではなくサービスアカウントに関連付けられます。

リソースを設定する際には、以下の点に注意してください。

- 必要なポート数は Pod 数よりも大きくなる。Kuryr はポートプールを使用して、事前に作成済みのポートを Pod で使用できるようにし、Pod の起動時間を短縮します。
- 各ネットワークポリシーは RHOSP セキュリティーグループにマップされ、**NetworkPolicy** 仕様によっては 1 つ以上のルールがセキュリティーグループに追加される。
- 各サービスは RHOSP ロードバランサーにマップされる。クォータに必要なセキュリティーグループの数を見積もる場合には、この要件を考慮してください。
RHOSP バージョン 15 以前のバージョン、または **ovn-octavia driver** を使用している場合、各ロードバランサーにはユーザープロジェクトと共にセキュリティーグループがあります。
- クォータはロードバランサーのリソース (VM リソースなど) を考慮しませんが、RHOSP デプロイメントのサイズを決定するにはこれらのリソースを考慮する必要があります。デフォルトのインストールには 50 を超えるロードバランサーがあり、クラスターはそれらのロードバランサーに対応する必要があります。
OVN Octavia ドライバーを有効にして RHOSP バージョン 16 を使用している場合は、1 つのロードバランサー仮想マシンのみが生成され、サービスは OVN フロー経由で負荷分散されます。

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томシン、およびブートストラップマシンで設定されます。

Kuryr SDN を有効にするには、使用する環境が以下の要件を満たしている必要があります。

- RHOSP 13+ を実行します。
- オーバークラウドと Octavia を使用します。
- Neutron トランクポートの拡張を使用します。
- ML2/OVS Neutron ドライバーが **ovs-hybrid** の代わりに使用れる場合、**openvswitch** ファイアウォールドライバーを使用します。

1.2.3.1. クォータの拡大

Kuryr SDN を使用する場合、Pod、サービス、namespace、およびネットワークポリシーが使用する Red Hat OpenStack Platform (RHOSP) リソースに対応するためにクォータを引き上げる必要があります。

手順

- 以下のコマンドを実行して、プロジェクトのクォータを増やします。

```
$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets 250 --networks 250 <project>
```

1.2.3.2. Neutron の設定

Kuryr CNI は Neutron トランクの拡張を使用してコンテナを Red Hat OpenStack Platform (RHOSP) SDN にプラグインします。したがって、Kuryr が適切に機能するには **trunks** 拡張を使用する必要があります。

さらにデフォルトの ML2/OVS Neutron ドライバーを使用する場合には、セキュリティグループがトランクサブポートで実行され、Kuryr がネットワークポリシーを適切に処理できるように、**ovs_hybrid** ではなく **openvswitch** に設定される必要があります。

1.2.3.3. Octavia の設定

Kuryr SDN は Red Hat OpenStack Platform (RHOSP) の Octavia LBaaS を使用して OpenShift Container Platform サービスを実装します。したがって、Kuryr SDN を使用するように RHOSP に Octavia コンポーネントをインストールし、設定する必要があります。

Octavia を有効にするには、Octavia サービスを RHOSP オーバークラウドのインストール時に組み込むか、またはオーバークラウドがすでに存在する場合は Octavia サービスをアップグレードする必要があります。Octavia を有効にする以下の手順は、オーバークラウドのクリーンインストールまたはオーバークラウドの更新の両方に適用されます。



注記

以下の手順では、Octavia を使用する場合に [RHOSP のデプロイメント](#) 時に必要となる主な手順のみを説明します。また、[レジストリーメソッド](#) が変更されることにも留意してください。

以下の例では、ローカルレジストリーの方法を使用しています。

手順

1. ローカルレジストリーを使用している場合、イメージをレジストリーにアップロードするためのテンプレートを作成します。以下に例を示します。

```
(undercloud) $ openstack overcloud container image prepare \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
--namespace=registry.access.redhat.com/rhosp13 \
--push-destination=<local-ip-from-undercloud.conf>:8787 \
--prefix=openstack- \
--tag-from-label {version}-{release} \
--output-env-file=/home/stack/templates/overcloud_images.yaml \
--output-images-file /home/stack/local_registry_images.yaml
```

2. **local_registry_images.yaml** ファイルに Octavia イメージが含まれることを確認します。以下に例を示します。

```
...
```

```
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
  push_destination: <local-ip-from-undercloud.conf>:8787
```



注記

Octavia コンテナのバージョンは、インストールされている特定の RHOSP リリースによって異なります。

3. コンテナイメージを **registry.redhat.io** からアンダークラウドノードにプルします。

```
(undercloud) $ sudo openstack overcloud container image upload \
  --config-file /home/stack/local_registry_images.yaml \
  --verbose
```

これには、ネットワークおよびアンダークラウドディスクの速度に応じて多少の時間がかかる可能性があります。

4. Octavia ロードバランサーは OpenShift Container Platform API にアクセスするために使用されるため、それらのリスナーの接続のデフォルトタイムアウトを増やす必要があります。デフォルトのタイムアウトは 50 秒です。以下のファイルをオーバークラウドのデプロイコマンドに渡し、タイムアウトを 20 分に増やします。

```
(undercloud) $ cat octavia_timeouts.yaml
parameter_defaults:
  OctaviaTimeoutClientData: 1200000
  OctaviaTimeoutMemberData: 1200000
```



注記

これは RHOSP 13.0.13+ では不要です。

5. Octavia を使用してオーバークラウドをインストールまたは更新します。

```
$ openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
  -e octavia_timeouts.yaml
```



注記

このコマンドには、Octavia に関連付けられたファイルのみが含まれます。これは、RHOSP の特定のインストールによって異なります。詳細は RHOSP のドキュメントを参照してください。Octavia インストールのカスタマイズについての詳細は、[Octavia デプロイメントのプランニング](#) を参照してください。



注記

Kuryr SDN を利用する際には、オーバークラウドのインストールに Neutron の **trunk** 拡張機能が必要です。これは、Director デプロイメントでデフォルトで有効にされます。Neutron バックエンドが ML2/OVS の場合、デフォルトの **ovs-hybrid** の代わりに **openvswitch** ファイアウォールを使用します。バックエンドが ML2/OVN の場合には変更の必要がありません。

6. RHOSP の 13.0.13 よりも前のバージョンでは、プロジェクトの作成後にプロジェクト ID を **octavia.conf** 設定ファイルに追加します。

- トラフィックが Octavia ロードバランサーを通過する場合など、複数のサービス全体でネットワークポリシーを実行するには、Octavia がユーザープロジェクトで Amphora 仮想マシンセキュリティグループを作成するようにする必要があります。この変更により、必要なロードバランサーのセキュリティグループがそのプロジェクトに属し、それらをサービスの分離を実行するように更新できます。



注記

RHOSP の 13.0.13 よりも後のバージョンでは、このタスクは必要ありません。

Octavia は、ロードバランサー VIP へのアクセスを制限する新しい ACL API を実装します。

- a. プロジェクト ID を取得します。

```
$ openstack project show <project>
```

出力例

```
+-----+-----+
| Field | Value |
+-----+-----+
| description |
| domain_id | default |
| enabled | True |
| id | PROJECT_ID |
| is_domain | False |
| name | *<project>* |
| parent_id | default |
| tags | [] |
+-----+-----+
```

- b. プロジェクト ID をコントローラーの **octavia.conf** に追加します。

- i. **stackrc** ファイルを取得します。

```
$ source stackrc # Undercloud credentials
```

- ii. オーバークラウドコントローラーを一覧表示します。

```
$ openstack server list
```

出力例

```

+-----+-----+-----+-----+-----+
| ID              | Name      | Status | Networks |
| Image          | Flavor    |        |          |
+-----+-----+-----+-----+
| 6bef8e73-2ba5-4860-a0b1-3937f8ca7e01 | controller-0 | ACTIVE |
ctlplane=192.168.24.8 | overcloud-full | controller |
| dda3173a-ab26-47f8-a2dc-8473b4a67ab9 | compute-0   | ACTIVE |
ctlplane=192.168.24.6 | overcloud-full | compute   |
+-----+-----+-----+-----+

```

- iii. コントローラーに対して SSH を実行します。

```
$ ssh heat-admin@192.168.24.8
```

- iv. **octavia.conf** ファイルを編集して、プロジェクトを Amphora セキュリティーグループがユーザーのアカウントに設定されているプロジェクトの一覧に追加します。

```

# List of project IDs that are allowed to have Load balancer security groups
# belonging to them.
amp_secgroup_allowed_projects = PROJECT_ID

```

- c. 新しい設定が読み込まれるように Octavia ワーカーを再起動します。

```
controller-0$ sudo docker restart octavia_worker
```



注記

RHOSP 環境によっては、Octavia は UDP リスナーをサポートしない可能性があります。RHOSP の 13.0.13 よりも前のバージョンで Kuryr SDN を使用する場合は、UDP サービスはサポートされません。RHOSP バージョン 16 以降は UDP をサポートします。

1.2.3.3.1. Octavia OVN ドライバー

Octavia は Octavia API を使用して複数のプロバイダードライバーをサポートします。

利用可能なすべての Octavia プロバイダードライバーをコマンドラインで表示するには、以下を入力します。

```
$ openstack loadbalancer provider list
```

出力例

```

+-----+
| name   | description                               |
+-----+
| amphora | The Octavia Amphora driver.               |
| octavia | Deprecated alias of the Octavia Amphora driver. |
| ovn    | Octavia OVN driver.                       |
+-----+

```

RHOSP バージョン 16 以降、Octavia OVN プロバイダードライバー (**ovn**) は RHOSP デプロイメントの OpenShift Container Platform でサポートされます。

ovn は、Octavia および OVN が提供する負荷分散用の統合ドライバーです。これは基本的な負荷分散機能をサポートし、OpenFlow ルールに基づいています。このドライバーは、OVN Neutron ML2 を使用するデプロイメント上の director により Octavia で自動的に有効にされます。

Amphora プロバイダードライバーがデフォルトのドライバーです。ただし、**ovn** が有効にされる場合には、Kuryr がこれを使用します。

Kuryr が Amphora の代わりに **ovn** を使用する場合は、以下の利点があります。

- リソース要件が減少します。Kuryr は、各サービスにロードバランサーの仮想マシンを必要としません。
- ネットワークレイテンシーが短縮されます。
- サービスごとに仮想マシンを使用する代わりに、OpenFlow ルールを使用することで、サービスの作成速度が上がります。
- Amphora 仮想マシンで集中管理されるのではなく、すべてのノードに分散負荷分散アクションが分散されます。

RHOSP クラウドがバージョン 13 から 16 にアップグレードした後に、[クラスターを Octavia OVN ドライバーを使用するように設定](#) できます。

1.2.3.4. Kuryr を使用したインストールについての既知の制限

OpenShift Container Platform を Kuryr SDN で使用する場合は、いくつかの既知の制限があります。

RHOSP の一般的な制限

Kuryr SDN を使用する OpenShift Container Platform は、タイプが **NodePort** の **Service** オブジェクトをサポートしません。

マシンのサブネットがルーターに接続されていない場合や、サブネットが接続されていても、ルーターに外部ゲートウェイが設定されていない場合、Kuryr はタイプが **LoadBalancer** の **Service** オブジェクトの Floating IP を作成できません。

- **Service** オブジェクトで **sessionAffinity=ClientIP** プロパティを設定しても効果はありません。Kuryr はこの設定をサポートしていません。

RHOSP バージョンの制限

OpenShift Container Platform を Kuryr SDN で使用する場合は、RHOSP バージョンに依存するいくつかの制限があります。

- RHOSP の 16 よりも前のバージョンでは、デフォルトの Octavia ロードバランサードライバー (Amphora) を使用します。このドライバーでは、OpenShift Container Platform サービスごとに 1 つの Amphora ロードバランサー仮想マシンをデプロイする必要があります。サービス数が

多すぎると、リソースが不足する可能性があります。

OVN Octavia ドライバーが無効にされている以降のバージョンの RHOSP のデプロイメントでも Amphora ドライバーを使用します。この場合も、RHOSP の以前のバージョンと同じリソースに関する懸念事項を考慮する必要があります。

- バージョン 13.0.13 よりも前の Octavia RHOSP バージョンは UDP リスナーをサポートしません。そのため、OpenShift Container Platform UDP サービスはサポートされません。
- 13.0.13 よりも前の Octavia RHOSP バージョンは、同じポートで複数のプロトコルをリスンできません。TCP や UDP など、同じポートを異なるプロトコルに公開するサービスはサポートされません。
- Kuryr SDN は、サービスによる自動解凍をサポートしていません。

RHOSP 環境の制限

Kuryr SDN を使用する場合に、デプロイメント環境に依存する制限事項があります。

Octavia には UDP プロトコルおよび複数のリスナーのサポートがないため、RHOSP バージョンが 13.0.13 よりも前のバージョンの場合、Kuryr は Pod が DNS 解決に TCP を使用するよう強制します。

Go バージョン 1.12 以前では、CGO サポートが無効にされた状態でコンパイルされたアプリケーションは UDP のみを使用します。この場合、ネイティブの Go リゾルバーは、TCP が DNS 解決に強制的に実行されるかどうかを制御する、**resolv.conf** の **use-vc** オプションを認識しません。その結果、UDP は引き続き DNS 解決に使用されますが、これは失敗します。

TCP の強制を許可するには、環境変数 **CGO_ENABLED** を **1** に設定 (例: **CGO_ENABLED=1**) されている状態でアプリケーションをコンパイルするか、または変数がないことを確認します。

Go バージョン 1.13 以降では、UDP を使用した DNS 解決が失敗する場合に TCP が自動的に使用されます。



注記

Alpine ベースのコンテナを含む musl ベースのコンテナは **use-vc** オプションをサポートしません。

RHOSP のアップグレードの制限

RHOSP のアップグレードプロセスにより、Octavia API が変更され、ロードバランサーに使用される Amphora イメージへのアップグレードが必要になる可能性があります。

API の変更に対応できます。

Amphora イメージがアップグレードされると、RHOSP Operator は既存のロードバランサー仮想マシンを 2 つの方法で処理できます。

- [ロードバランサーのフェイルオーバー](#) をトリガーしてそれぞれの仮想マシンをアップグレードします。
- ユーザーが仮想マシンのアップグレードを行う必要があります。

Operator が最初のオプションを選択する場合、フェイルオーバー時に短い時間のダウンタイムが生じる可能性があります。

Operator が 2 つ目のオプションを選択する場合、既存のロードバランサーは UDP リスナーなどのアップグレードされた Octavia API 機能をサポートしません。この場合、ユーザーはこれらの機能を使用するためにサービスを再作成する必要があります。



重要

OpenShift Container Platform が UDP の負荷分散をサポートする新規の Octavia バージョンを検出する場合、これは DNS サービスを自動的に再作成します。サービスの再作成により、サービスのデフォルトが UDP の負荷分散をサポートするようになります。

再作成により、DNS サービスに約 1 分間のダウンタイムが発生します。

1.2.3.5. 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、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.2.3.6. 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、2 つの vCPU および 100 GB のストレージ領域があるフレーバー

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。

1.2.3.7. 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

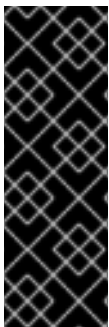
- 少なくとも 16 GB のメモリー、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.2.4. OpenShift Container Platform のインターネットアクセス

OpenShift Container Platform 4.6 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になります。

インターネットへのアクセスは以下を実行するために必要です。

- [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。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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。

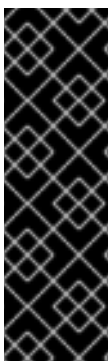


重要

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クラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、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。

1.2.5. 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

Swift として知られる [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。

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。

1.2.6. 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一覧に表示されます。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。

1.2.7. 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** についての詳細は、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: shiftstack_user
      password: XXX
      user_domain_name: Default
      project_domain_name: Default
    dev-env:
      region_name: RegionOne
      auth:
        username: 'devuser'
        password: XXX
        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. マシンを認証局の信頼バンドルに追加します。

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

- c. 信頼バンドルを更新します。

```
$ sudo update-ca-trust extract
```

- d. **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。

1.2.8. インストールプログラムの取得

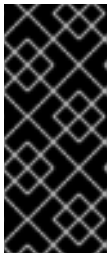
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

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。

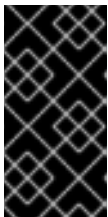
手順

- OpenShift Cluster Manager サイトの [インフラストラクチャプロバイダー](#) ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
- インフラストラクチャプロバイダーを選択します。
- 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、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。ファイル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。



重要

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。

- 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
```

- [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。

1.2.9. 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を取得します。

手順

1. `install-config.yaml` ファイルを作成します。

- 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。



重要

空のディレクトリーを指定します。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の有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。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コピーを行ってください。

- 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。

- 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。

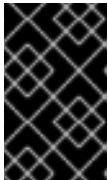


注記

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がある実稼働用の 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。

- 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。
- ii. 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。
- iii. 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。
- iv. コントロールプレーンおよびコンピューターノードに使用する 16 GB 以上の RAM で RHOSP フレーバーを指定します。
- v. 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。
 - viii. [Red Hat OpenShift Cluster Manager からプルシークレット](#) を貼り付けます。
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については、[インストール設定パラメーターセクション](#)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。

1.2.9.1. インストール時のクラスター全体のプロキシの設定

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 プロキシを使用することができます。プロキシ設定を **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ついてのクラウドプロバイダー API に対する呼び出しを含む) はプロキシされます。プロキシ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)、Google Cloud Platform (GCP)、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> ❶
  httpsProxy: https://<username>:<pswd>@<ip>:<port> ❷
  noProxy: example.com ❸
additionalTrustBundle: | ❹
```

```
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
```

...

- 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 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 URL。
- 3 プロキシ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 のコンマ区切りの一覧。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をバイパスします。
- 4 指定されている場合には、インストールプログラムは、**openshift-config** namespace に **user-ca-bundle** という名前の設定魔府を生成して、追加の CA 証明書を保存します。**additionalTrustBundle** と少なくとも1つのプロキシ設定を指定した場合には、**Proxy** オブジェクトは **trusted CA** フィールドで **user-ca-bundle** 設定マップを参照するように設定されます。その後、Cluster Network Operator は、**trustedCA** パラメーターに指定されたコンテンツを RHCOS トラストバンドルにマージする **trusted-ca-bundle** 設定マップを作成します。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が 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。



注記

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

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を作成します。プロキシ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、追加のプロキシを作成することはできません。

1.2.10. インストール設定パラメーター

OpenShift Container Platform クラスターをデプロイする前に、クラスターをホストするクラウドプラットフォームでアカウントを記述し、クラスターのプラットフォームをオプションでカスタマイズするためにパラメーターの値を指定します。**install-config.yaml** インストール設定ファイルを作成する際に、コマンドラインで必要なパラメーターの値を指定します。クラスターをカスタマイズする場合、**install-config.yaml** ファイルを変更して、プラットフォームについての詳細情報を指定できます。



注記

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。

**重要**

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.2.10.1. 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1.8 必須パラメーター

パラメーター	説明	値
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。インストーラーは、古い API バージョンをサポートすることもできます。	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、 baseDomain と <metadata.name> 、 <baseDomain> 形式を使用する metadata.name パラメーターの値の組み合わせです。	example.com などの完全修飾ドメイン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。	オブジェクト
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。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。文字列は 14 文字以上でなければなりません。

パラメーター	説明	値
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 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について参照してください。	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イメージをダウンロードすることを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.2.10.2. ネットワーク設定パラメーター

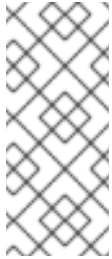
既存のネットワークインフラストラクチャーの要件に基づいて、インストール設定をカスタマイズできます。たとえば、クラスターネットワークの IP アドレスブロックを拡張するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.9 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p> <div>  <div> <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p> </div> </div>

パラメーター	説明	値
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN 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。	オブジェクトの配列。以下に例を示します。 <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
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	CIDR (Classless Inter-Domain Routing) 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。
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、 $2^{(32-23)} = 510$ Pod IP アドレスを提供します。	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。 OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。	CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。 <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
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。	オブジェクトの配列。以下に例を示します。 <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


パラメーター	説明	値
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 以外のすべてのプラットフォームでは、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。	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>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。</p> </div> </div>



1.2.10.3. 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1.10 オプションのパラメーター

パラメーター	説明	値
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。この信頼バンドルは、プロキシが設定される際にも使用できます。	文字列
compute	コンピュータノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列

パラメーター	説明	値
compute.hyperthreading	<p>コンピュータ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にされます。</p> <div>  <div> <p>重要</p> <p>同時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ていることを確認します。</p> </div> </div>	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。	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。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (ワーカーマシンとしても知られる) の数。	2 以上の正の整数。デフォルト値は 3 です。
controlPlane	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列

パラメーター	説明	値
controlPlane.hyperthreading	<p>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にされます。</p> <div>  <p>重要</p> <p>同時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ていることを確認します。</p> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	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。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 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。	サポートされる値は 3 のみです (これはデフォルト値です)。
credentialsMode	<p>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。</p> <div>  <p>注記</p> <p>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、Red Hat Operator のクラウド認証情報 Operator を参照してください。</p> </div>	Mint 、 Passthrough 、 Manual 、または空の文字列 ("")。

パラメーター	説明	値
fips	<p>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 モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <div> <p>重要</p> <p>FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、x86_64 アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。</p> </div> </div> <div>  <div> <p>注記</p> <p>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。</p> </div> </div>	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。	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を指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。

パラメーター	説明	値
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。	<p>Internal または External。デフォルト値は External です。</p> <p>このパラメーターを Internal に設定することは、クラウド以外のプラットフォームではサポートされません。</p> <div>  <p>重要</p> <p>フィールドの値が Internal に設定されている場合、クラスターは機能しなくなります。詳細は、BZ#1953035 を参照してください。</p> </div>
sshKey	<p>クラスターマシンへのアクセスを認証するための単一または複数の SSH キー。</p> <div>  <p>注記</p> <p>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がある実稼働用の 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。</p> </div>	<p>1つ以上のキー。以下に例を示します。</p> <pre>sshKey: <key1> <key2> <key3></pre>

1.2.10.4. 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1.11 追加の RHOSP パラメーター

パラメーター	説明	値
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。	整数 (例: 30)。

パラメーター	説明	値
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。	文字列 (例: performance)。
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。	整数 (例: 30)。
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。	文字列 (例: performance)。
platform.openstack.cloud	clouds.yaml ファイルのクラウド一覧にある使用する RHOC クラウドの名前。	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。	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。	文字列 (例: m1.xlarge)。

1.2.10.5. 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1.12 オプションの RHOSP パラメーター

パラメーター	説明	値
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。	文字列としての1つ以上の UUID の一覧。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グループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
compute.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: ["zone-1", "zone-2"] 。
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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。
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グループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
controlPlane.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: <code>["zone-1", "zone-2"]</code> 。
platform.openstack.clusterOSImage	<p>インストーラーが RHCOS イメージをダウンロードする場所。</p> <p>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。</p>	<p>HTTP または HTTPS の URL (オプションで SHA-256 形式のチェックサムを使用)。</p> <p>例: <code>http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</code>。この値は、既存の Glance イメージの名前にもなり得ます (例: <code>my-rhcos</code>)。</p>
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。	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	Ingress ポートに関連付ける既存の Floating IP アドレス。このプロパティを使用するには、 platform.openstack.externalNetwork プロパティも定義する必要があります。	IP アドレス (例: <code>128.0.0.1</code>)。

パラメーター	説明	値
platform.openstack.lbFloatingIP	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。	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。	文字列としての IP アドレスの一覧。例: ["8.8.8.8", "192.168.1.12"]
platform.openstack.machinesSubnet	<p>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。</p> <p>networking.machineNetwork の最初の項目は machinesSubnet の値に一致する必要があります。</p> <p>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。</p>	文字列としての UUID。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。

1.2.10.6. RHOSP デプロイメントでのカスタムサブネット

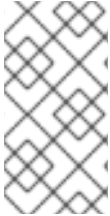
オプションで、選択する Red Hat OpenStack Platform (RHOSP) サブネットにクラスターをデプロイすることができます。サブネットの GUID は、**install-config.yaml** ファイルの **platform.openstack.machinesSubnet** の値として渡されます。

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。ノードとポートはこの上に作成されます。

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、以下を確認します。

- ターゲットネットワークおよびサブネットが利用可能である。
- DHCP がターゲットサブネットで有効にされている。

- ターゲットネットワーク上でポートを作成するためのパーミッションがあるインストーラー認証情報を指定できます。
- ネットワーク設定にルーターが必要な場合、これは RHOSP で作成されます。一部の設定は、Floating IP アドレスの変換用のルーターに依存します。
- ネットワーク設定は、プロバイダーのネットワークに依存しません。プロバイダーネットワークはサポートされません。

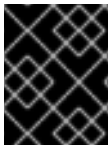


注記

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.apiVIP** および **platform.openstack.ingressVIP** の値を設定します。

1.2.10.7. Kuryr を使用した OpenStack のカスタマイズされた `install-config.yaml` ファイルのサンプル

デフォルトの OpenShift SDN ではなく Kuryr SDN を使用してデプロイするには、**install-config.yaml** ファイルを変更して **Kuryr** を必要な **networking.networkType** として追加してから、デフォルトの OpenShift Container Platform SDN インストール手順に進む必要があります。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
clusterID: os-test
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 ①
networkType: Kuryr
```

```
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
    trunkSupport: true ②
    octaviaSupport: true ③
  pullSecret: '{"auths": ...}'
  sshKey: ssh-ed25519 AAAA...
```

- ① Amphora Octavia ドライバーは、ロードバランサーごとに2つのポートを作成します。そのため、インストーラーが作成するサービスサブネットは、**serviceNetwork** プロパティの値として指定される CIDR のサイズは2倍になります。IP アドレスの競合を防ぐには、範囲をより広くする必要があります。
- ② ③ **trunkSupport** と **octaviaSupport** の両方はインストーラーによって自動的に検出されるため、それらを設定する必要はありません。ただし、ご使用の環境がこれらの両方の要件を満たさないと、Kuryr SDN は適切に機能しません。トランクは Pod を RHOSP ネットワークに接続するために必要であり、Octavia は OpenShift Container Platform サービスを作成するために必要です。

1.2.10.8. Kuryr ポートプール

Kuryr ポートプールでは、Pod 作成のスタンバイ状態の多数のポートを維持します。

ポートをスタンバイ状態に維持すると、Pod の作成時間が必要最小限に抑えることができます。ポートプールを使用しない場合には、Kuryr は Pod が作成または削除されるたびにポートの作成または削除を明示的に要求する必要があります。

Kuryr が使用する Neutron ポートは、namespace に関連付けられるサブネットに作成されます。これらの Pod ポートは、OpenShift Container Platform クラスターノードのプライマリーポートにサブポートとして追加されます。

Kuryr は namespace をそれぞれ、別のサブネットに保存するため、namespace-worker ペアごとに別個のポートプールが維持されます。

クラスターをインストールする前に、**cluster-network-03-config.yml** マニフェストファイルに以下のパラメーターを設定して、ポートプールの動作を設定できます。

- **enablePortPoolsPrepopulation** パラメーターで、プールの事前生成が制御されるので、Kuryr は新規ホストが追加される時や新規 namespace の作成時などの作成時に、Kuryr によりプールにポートが強制的に追加されるようにします。デフォルト値は **false** です。
- **poolMinPorts** パラメーターは、プールに保持する空きポートの最小数です。デフォルト値は **1** です。
- **poolMaxPorts** パラメーターは、プールに保持する空きポートの最大数です。値が **0** の場合は、上限が無効になります。これはデフォルト設定です。
OpenStack ポートのクォータが低い場合や、Pod ネットワークで IP アドレスの数が限定されている場合には、このオプションを設定して、不要なポートが削除されるようにします。
- **poolBatchPorts** パラメーターは、一度に作成可能な Neutron ポートの最大数を定義します。デフォルト値は **3** です。

1.2.10.9. インストール時の Kuryr ポートプールの調整

インストール時に、Pod 作成の速度や効率性を制御するために Kuryr で Red Hat OpenStack Platform (RHOSP) Neutron ポートを管理する方法を設定できます。

前提条件

- **install-config.yaml** ファイルを作成して変更しておく。

手順

1. コマンドラインからマニフェストファイルを作成します。

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

- 1 **<installation_directory>** については、クラスターの **install-config.yaml** ファイルが含まれるディレクトリーの名前を指定します。

2. **cluster-network-03-config.yml** という名前のファイルを **<installation_directory>/manifests/** ディレクトリーに作成します。

```
$ touch <installation_directory>/manifests/cluster-network-03-config.yml 1
```

- 1 **<installation_directory>** については、クラスターの **manifests/** ディレクトリーが含まれるディレクトリー名を指定します。

ファイルの作成後は、以下のようにいくつかのネットワーク設定ファイルが **manifests/** ディレクトリーに置かれます。

```
$ ls <installation_directory>/manifests/cluster-network-*
```

出力例

```
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. エディターで **cluster-network-03-config.yml** ファイルを開き、必要な Cluster Network Operator 設定を記述するカスタムリソース (CR) を入力します。

```
$ oc edit networks.operator.openshift.io cluster
```

4. 要件に合わせて設定を編集します。以下のファイルをサンプルとして紹介しています。

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork:
```

```
- 172.30.0.0/16
defaultNetwork:
  type: Kuryr
  kuryrConfig:
    enablePortPoolsPrepopulation: false ❶
    poolMinPorts: 1 ❷
    poolBatchPorts: 3 ❸
    poolMaxPorts: 5 ❹
    openstackServiceNetwork: 172.30.0.0/15 ❺
```

- ❶ **enablePortPoolsPrepopulation** の値を **true** に設定し、namespace の作成時、または新規ノードがクラスターに追加された後に Kuryr が新規 Neutron ポートを作成するようにします。この設定により、Neutron ポートのクォータが引き上げられますが、Pod の起動に必要なとなる時間を短縮できます。デフォルト値は **false** です。
- ❷ Kuryr は、対象のプール内にある空きポートの数が **poolMinPorts** の値よりも少ない場合には、プールに新規ポートを作成します。デフォルト値は **1** です。
- ❸ **poolBatchPorts** は、空きポートの数が **poolMinPorts** の値よりも少ない場合に作成される新規ポートの数を制御します。デフォルト値は **3** です。
- ❹ プール内の空きポートの数が **poolMaxPorts** の値よりも多い場合に、Kuryr はその値と同じ数になるまでポートを削除します。この値を **0** に設定すると、この上限は無効になり、プールが縮小できないようにします。デフォルト値は **0** です。
- ❺ **openStackServiceNetwork** パラメーターは、RHOSP Octavia の LoadBalancer に割り当てられるネットワークの CIDR 範囲を定義します。

このパラメーターを Amphora ドライバーと併用する場合には、Octavia は、ロードバランサーごとに、このネットワークから IP アドレスを 2 つ (OpenShift 用に 1 つ、VRRP 接続用に 1 つ) 取得します。これらの IP アドレスは OpenShift Container Platform と Neutron でそれぞれ管理されるため、異なるプールから取得する必要があります。したがって、**openStackServiceNetwork serviceNetwork** の値の 2 倍になる必要があり、**serviceNetwork** の値は、**openStackServiceNetwork** で定義された範囲と完全に重複する必要があります。

CNO は、このパラメーターの定義範囲から取得した VRRP IP アドレスが **serviceNetwork** パラメーターの定義範囲と重複しないことを検証します。

このパラメーターが設定されていない場合には、CNO は **serviceNetwork** の拡張値を使用します。この値は、プリフィックスのサイズを 1 つずつ減らして決定します。

5. **cluster-network-03-config.yml** ファイルを保存し、テキストエディターを終了します。
6. オプション: **manifests/cluster-network-03-config.yml** ファイルをバックアップします。インストールプログラムは、クラスターの作成時に **manifests/** ディレクトリーを削除します。

1.2.11. コンピュートマシンのアフィニティーの設定

オプションで、インストール時にコンピュートマシンのアフィニティーポリシーを設定できます。インストーラーは、デフォルトでコンピュートマシンのアフィニティーポリシーを選択しません。

インストール後に特定の RHOSP サーバグループを使用するマシンセットを作成することもできます。



注記

コントロールプレーンマシンは、**soft-anti-affinity** ポリシーで作成されます。

ヒント

[RHOSP インスタンスのスケジューリングおよび配置](#)の詳細は、RHOSP のドキュメントを参照してください。

前提条件

- **install-config.yaml** ファイルを作成し、これに対する変更を完了します。

手順

1. RHOSP コマンドラインインターフェイスを使用して、コンピュータマシンのサーバーグループを作成します。以下に例を示します。

```
$ openstack \
  --os-compute-api-version=2.15 \
  server group create \
  --policy anti-affinity \
  my-openshift-worker-group
```

詳細は、[server group create コマンドのドキュメント](#) を参照してください。

2. 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、マニフェストを作成します。

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

ここでは、以下ようになります。

installation_directory

クラスターの **install-config.yaml** ファイルが含まれるディレクトリーの名前を指定します。

3. **MachineSet** 定義ファイルの **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** を作成します。
4. **spec.template.spec.providerSpec.value** プロパティの下にある定義に、プロパティ **serverGroupID** を追加します。以下に例を示します。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
    machine.openshift.io/cluster-api-machine-role: <node_role>
    machine.openshift.io/cluster-api-machine-type: <node_role>
  name: <infrastructure_ID>-<node_role>
  namespace: openshift-machine-api
spec:
  replicas: <number_of_replicas>
  selector:
    matchLabels:
```

```

machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
template:
  metadata:
    labels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
      machine.openshift.io/cluster-api-machine-role: <node_role>
      machine.openshift.io/cluster-api-machine-type: <node_role>
      machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
  spec:
    providerSpec:
      value:
        apiVersion: openstackproviderconfig.openshift.io/v1alpha1
        cloudName: openstack
        cloudsSecret:
          name: openstack-cloud-credentials
          namespace: openshift-machine-api
        flavor: <nova_flavor>
        image: <glance_image_name_or_location>
        serverGroupID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee 1
        kind: OpenstackProviderSpec
        networks:
          - filter: {}
            subnets:
              - filter:
                  name: <subnet_name>
                  tags: openshiftClusterID=<infrastructure_ID>
        securityGroups:
          - filter: {}
            name: <infrastructure_ID>-<node_role>
        serverMetadata:
          Name: <infrastructure_ID>-<node_role>
          openshiftClusterID: <infrastructure_ID>
        tags:
          - openshiftClusterID=<infrastructure_ID>
        trunk: true
        userDataSecret:
          name: <node_role>-user-data
        availabilityZone: <optional_openstack_availability_zone>

```

1 サーバグループの UUID をここに追加します。

5. オプション: **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** ファイルをバックアップします。インストールプログラムは、クラスターの作成時に **manifests/** ディレクトリを削除します。

クラスターのインストール時に、インストーラーは変更した **MachineSet** 定義を使用して RHOSP サーバグループ内にコンピュートマシンを作成します。

1.2.12. SSH プライベートキーの生成およびエージェントへの追加

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。

**注記**

実稼働環境では、障害復旧およびデバッグが必要です。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの `~/.ssh/authorized_keys` 一覧に追加されます。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、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 `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。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。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

**注記**

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは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. **ssh-agent** プロセスを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。

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

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

出力例

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

- ① ~/.ssh/id_rsa 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。

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。

1.2.13. 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。

1.2.13.1. floating IP アドレスを使っ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.lbFloatingIP**

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があります。

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。

1.2.13.2. 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.lbFloatingIP**

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。

1.2.14. 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を取得します。

手順

1. 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、クラスターのデプロイメントを初期化します。

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

- ❶ **<installation_directory>** については、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。
- ❷ 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。



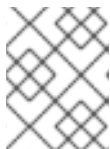
注記

ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

クラスターのデプロイメントが完了すると、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報を含む、クラスターにアクセスするための指示がターミナルに表示されます。

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: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



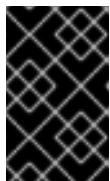
注記

クラスターアクセスおよび認証情報の情報は、インストールが正常に実行される際に **<installation_directory>/openshift_install.log** に出力されます。



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** についての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お勧めします。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。



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。

1.2.15. 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 ❶
```

- ❶ **<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ついての情報が含まれます。

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ター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
```

1.2.16. 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ついての情報が含まれます。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。
- **oc** CLI をインストールしていること。

手順

1. **kubeadmin** 認証情報をエクスポートします。

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

- ❶ **<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. 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 へのアクセスおよびその詳細は、[Web コンソールへのアクセス](#) について参照してください。

1.2.17. OpenShift Container Platform の Telemetry アクセス

OpenShift Container Platform 4.6 では、クラスターのヘルス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る Telemetry サービスには、インターネットアクセスが必要です。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、[リモートヘルスマニターリング](#) について参照してください。

1.2.18. 次のステップ

- [クラスターをカスタマイズ](#) します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 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 クラスタトラフィックを設定](#) します。
- RHOSP が 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設定しなかった場合には、[RHOSP のアクセスを Floating IP アドレスで設定](#) します。

1.3. 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 4.6 では、ユーザーによってプロビジョニングされたインフラストラクチャーを実行するクラスターを Red Hat OpenStack Platform (RHOSP) に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、インストーラーでプロビジョニングされるインストール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ります。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。

1.3.1. 前提条件

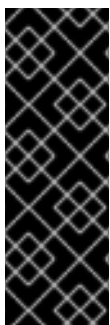
- [OpenShift Container Platform のインストールおよび更新](#) プロセスについての詳細を確認します。
- OpenShift Container Platform 4.6 が **Available platforms** セクションの RHOSP バージョンと互換性があることを確認します。[RHOSP サポートマトリックスの OpenShift Container Platform](#) を参照して、プラットフォームのサポートを異なるバージョン間で比較することもできます。
- ネットワーク設定がプロバイダーのネットワークに依存しないことを確認します。プロバイダーネットワークはサポートされません。
- OpenShift Container Platform をインストールする RHOSP アカウントがあります。
- インストールプログラムを実行するマシンには、以下が含まれます。
 - 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

1.3.2. OpenShift Container Platform のインターネットアクセス

OpenShift Container Platform 4.6 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になります。

インターネットへのアクセスは以下を実行するために必要です。

- [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。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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クラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、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。

1.3.3. 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1.13 RHOSP のデフォルトの OpenShift Container Platform クラスターについての推奨リソース

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	112 GB
vCPU	28
ボリュームストレージ	275 GB
インスタンス	7
セキュリティグループ	3
セキュリティグループルール	60

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、セキュリティグループおよびセキュリティ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設定されます。

1.3.3.1. 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、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.3.3.2. 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、2 つの vCPU および 100 GB のストレージ領域があるフレーバー

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。

1.3.3.3. 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、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.3.4. Playbook 依存関係のダウンロード

ユーザーによってプロビジョニングされたインフラストラクチャーでのインストールプロセスを単純化する Ansible Playbook には、複数の Python モジュールが必要です。インストーラーを実行するマシンで、モジュールのリポジトリを追加し、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. 必要なリポジトリを追加します。

```
$ sudo subscription-manager repos \
--enable=rhel-8-for-x86_64-baseos-rpms \
--enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
--enable=ansible-2.9-for-rhel-8-x86_64-rpms \
--enable=rhel-8-for-x86_64-appstream-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 python3-netaddr
```

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
```

1.3.5. 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-
4.6/upi/openstack/bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/common.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/compute-nodes.yaml
```

```

https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/control-
plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/inventory.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/security-
groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
compute-nodes.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
control-plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
load-balancers.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
containers.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。

1.3.6. 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、インストールファイルをローカルコンピューターに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. OpenShift Cluster Manager サイトの [インフラストラクチャプロバイダー](#) ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. インフラストラクチャプロバイダーを選択します。
3. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、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。ファイル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。



重要

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。

4. 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
```

5. [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。

1.3.7. SSH プライベートキーの生成およびエージェントへの追加

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの `~/.ssh/authorized_keys` 一覧に追加されます。



注記

[AWS キーペア](#) などのプラットフォームに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. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、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> ❶
```

- ❶ ~/.ssh/id_rsa などの、新規 SSH キーのパスおよびファイル名を指定します。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。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは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. **ssh-agent** プロセスを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。

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

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

出力例

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

- ❶ ~/.ssh/id_rsa 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。

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。

1.3.8. Red Hat Enterprise Linux CoreOS (RHCOS) イメージの作成

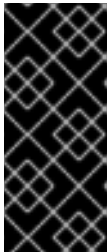
OpenShift Container Platform インストールプログラムでは、Red Hat Enterprise Linux CoreOS (RHCOS) イメージが Red Hat OpenStack Platform (RHOSP) クラスターに存在する必要があります。最新の RHCOS イメージを取得した後、RHOSP CLI を使用してこれをアップロードします。

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 4.6 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中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。



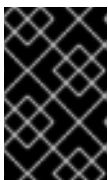
注記

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、または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。

1.3.9. 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一覧に表示されます。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。

1.3.10. 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。

1.3.10.1. floating IP アドレスを使っ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があります。

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。

1.3.10.2. 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。

1.3.11. 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** についての詳細は、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: shiftstack_user
      password: XXX
```

```

    user_domain_name: Default
    project_domain_name: Default
dev-env:
    region_name: RegionOne
auth:
    username: 'devuser'
    password: XXX
    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. マシンを認証局の信頼バンドルに追加します。

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

- c. 信頼バンドルを更新します。

```
$ sudo update-ca-trust extract
```

- d. **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。

1.3.12. 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を取得します。

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。



重要

空のディレクトリーを指定します。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の有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。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コピーを行ってください。

- 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がある実稼働用の 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およびコンピューターノードに使用する 16 GB 以上の RAM で 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。
 - viii. [Red Hat OpenShift Cluster Manager](#) から [プルシークレット](#) を貼り付けます。
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については、[インストール設定パラメーターセクション](#)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。

1.3.13. インストール設定パラメーター

OpenShift Container Platform クラスターをデプロイする前に、クラスターをホストするクラウドプラットフォームでアカウントを記述し、クラスターのプラットフォームをオプションでカスタマイズするためにパラメーターの値を指定します。**install-config.yaml** インストール設定ファイルを作成する際に、コマンドラインで必要なパラメーターの値を指定します。クラスターをカスタマイズする場合、**install-config.yaml** ファイルを変更して、プラットフォームについての詳細情報を指定できます。



注記

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。



重要

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.3.13.1. 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1.14 必須パラメーター

パラメーター	説明	値
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。インストーラーは、古い API バージョンをサポートすることもできます。	文字列

パラメーター	説明	値
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、 baseDomain と <metadata.name> 、 <baseDomain> 形式を使用する metadata.name パラメーターの値の組み合わせです。	example.com などの完全修飾ドメイン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。	オブジェクト
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。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。文字列は 14 文字以上でなければなりません。
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 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について参照してください。	オブジェクト

パラメーター	説明	値
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イメージをダウンロードすることを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.3.13.2. 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するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.15 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p>  <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p>
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。
networking.clusterNetwork	<p>Pod の IP アドレスブロック。</p> <p>デフォルト値は 10.128.0.0/14 で、ホストの接頭辞は /23 です。</p> <p>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。</p>	<p>オブジェクトの配列。以下に例を示します。</p> <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>

パラメーター	説明	値
networking.clusterNetwork.cidr	<p>networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。</p> <p>IPv4 ネットワーク</p>	CIDR (Classless Inter-Domain Routing) 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。
networking.clusterNetwork.hostPrefix	<p>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、$2^{(32 - 23) - 2}$ Pod IP アドレスを提供します。</p>	<p>サブネット接頭辞。</p> <p>デフォルト値は 23 です。</p>
networking.serviceNetwork	<p>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。</p> <p>OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。</p>	<p>CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>マシンの IP アドレスブロック。</p> <p>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。</p>	<p>オブジェクトの配列。以下に例を示します。</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、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。</p>	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>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。</p> </div> </div>

1.3.13.3. 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1.16 オプションのパラメーター

パラメーター	説明	値
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。この信頼バンドルは、プロキシーが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列
compute.hyperthreading	<p>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にされます。</p> <div data-bbox="491 1216 600 1503" data-label="Image"> </div> <p>重要</p> <p>同時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ていることを確認します。</p>	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。	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。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 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。	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	値
controlPlane	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列
controlPlane.hyperthreading	<p>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にされます。</p> <div>  <div> <p>重要</p> <p>同時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ていることを確認します。</p> </div> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	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。	aws 、 azure 、 gcp 、 openstack 、 o virt 、 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。	サポートされる値は 3 のみです (これはデフォルト値です)。

パラメーター	説明	値
credentialsMode	<p>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。</p> <div><p>注記</p><p>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、Red Hat Operatorのクラウド認証情報 Operatorを参照してください。</p></div>	Mint、Passthrough、Manual、 または空の文字列 ("")。

パラメーター	説明	値
fips	<p>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 モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>重要</p> <p>FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、x86_64 アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。</p> </div> <div>  <p>注記</p> <p>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。</p> </div>	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。	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を指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。

パラメーター	説明	値
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。	<p>Internal または External。デフォルト値は External です。</p> <p>このパラメーターを Internal に設定することは、クラウド以外のプラットフォームではサポートされません。</p> <div>  <div> <p>重要</p> <p>フィールドの値が Internal に設定されている場合、クラスターは機能しなくなります。詳細は、BZ#1953035 を参照してください。</p> </div> </div>
sshKey	<p>クラスターマシンへのアクセスを認証するための単一または複数の SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の 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。</p> </div> </div>	<p>1つ以上のキー。以下に例を示します。</p> <pre>sshKey: <key1> <key2> <key3></pre>

1.3.13.4. 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1.17 追加の RHOSP パラメーター

パラメーター	説明	値
compute.platform.openstack.rootVolume.size	コンピュータマシンの場合、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。	整数 (例: 30)。

パラメーター	説明	値
compute.platform.openstack.rootVolume.type	コンピュータマシンの場合、root のボリュームタイプです。	文字列 (例: performance)。
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。	整数 (例: 30)。
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。	文字列 (例: performance)。
platform.openstack.cloud	clouds.yaml ファイルのクラウド一覧にある使用する RHOSP クラウドの名前。	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。	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タマシンに使用する RHOSP フレーバー。	文字列 (例: m1.xlarge)。

1.3.13.5. 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1.18 オプションの RHOSP パラメーター

パラメーター	説明	値
compute.platform.openstack.additionalNetworkIDs	コンピュータ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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。
compute.platform.openstack.additionalSecurityGroupIDs	コンピュータマシンに関連付けられた追加のセキュリティグループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
compute.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: ["zone-1", "zone-2"] 。
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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。
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グループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
controlPlane.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: <code>["zone-1", "zone-2"]</code> 。
platform.openstack.clusterOSImage	<p>インストーラーが RHCOS イメージをダウンロードする場所。</p> <p>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。</p>	<p>HTTP または HTTPS の URL (オプションで SHA-256 形式のチェックサムを使用)。</p> <p>例: <code>http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</code>。この値は、既存の Glance イメージの名前にもなり得ます (例: <code>my-rhcos</code>)。</p>
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。	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>Ingress ポートに関連付ける既存の Floating IP アドレス。このプロパティを使用するには、platform.openstack.externalNetwork プロパティも定義する必要があります。</p>	IP アドレス (例: <code>128.0.0.1</code>)。

パラメーター	説明	値
platform.openstack.lbFloatingIP	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。	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。	文字列としての IP アドレスの一覧。例: ["8.8.8.8", "192.168.1.12"]
platform.openstack.machinesSubnet	<p>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。</p> <p>networking.machineNetwork の最初の項目は machinesSubnet の値に一致する必要があります。</p> <p>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。</p>	文字列としての UUID。例: fa806b2f-ac49-4bceb9db-124bc64209bf 。

1.3.13.6. 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。ノードとポートはこの上に作成されます。

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、以下を確認します。

- ターゲットネットワークおよびサブネットが利用可能である。
- DHCP がターゲットサブネットで有効にされている。
- ターゲットネットワーク上でポートを作成するためのパーミッションがあるインストーラー認証情報を指定できます。

- ネットワーク設定にルーターが必要な場合、これは RHOSP で作成されます。一部の設定は、Floating IP アドレスの変換用のルーターに依存します。
- ネットワーク設定は、プロバイダーのネットワークに依存しません。プロバイダーネットワークはサポートされません。

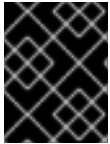


注記

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.apiVIP** および **platform.openstack.ingressVIP** の値を設定します。

1.3.13.7. RHOSP 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
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: m1.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: OpenShiftSDN
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
```

```
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

1.3.13.8. 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。

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、または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["networking"]["machineNetwork"] = [{"cidr": "192.168.0.0/18"}]; ❶
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

- ❶ 必要な Neutron サブネットに一致する値 (例: **192.0.2.0/24**) を挿入します。

- 値を手動で設定するには、ファイルを開き、**networking.machineCIDR** の値を必要な Neutron サブネットに一致する値に設定します。

1.3.13.9. 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、または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。

1.3.14. 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作成するために必要な 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。これはクラスターを作成するために後に使用されます。

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** についての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お勧めします。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. 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> ❶
```

- ❶ **<installation_directory>** については、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およびコンピュート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
```

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。

- 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> ❶
```

- ❶ **<installation_directory>** については、同じインストールディレクトリーを指定します。

以下のファイルはディレクトリーに生成されます。

```
.
├── 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。

1.3.15. 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>", ❶
        "httpHeaders": [{
          "name": "X-Auth-Token", ❷
          "value": "<token_ID>" ❸
        }]
      }]
    }
  }
}
```

```

    }}
  },
  "security": {
    "tls": {
      "certificateAuthorities": [{
        "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" 4
      }]
    }
  },
  "version": "3.1.0"
}

```

- 1 **ignition.config.append.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。

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

独自のインフラストラクチャーを使用して OpenShift Container Platform を 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**。

1.3.17. 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

- 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

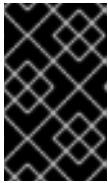
1. オプション: 外部ネットワークの値を **inventory.yaml** Playbook に追加します。

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。

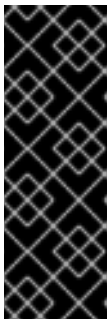
2. オプション: 外部ネットワークおよび Floating IP (FIP) アドレスの値を **inventory.yaml** Playbook に追加します。

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 の値を定義しない場合、インストーラーは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、**network.yaml** Playbook を実行して、ネットワーク、サブネット、およびルーターを作成します。

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

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"
```

1.3.18. 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"
```

1.3.19. 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 Ignition 設定ファイルが作業ディレクトリーにない場合、それらをここにコピーします。
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 v1.14.6+f9b5405 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

1.3.20. 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ついての情報が含まれます。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。
- **oc** CLI をインストールしていること。

手順

1. **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. 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
```

1.3.21. 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。

1.3.22. RHOSP でのコンピュータマシンの作成

コントロールプレーンの起動後、コンピュータ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。

1.3.23. 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ついて 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、または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	v1.19.0
master-1	Ready	master	63m	v1.19.0
master-2	Ready	master	64m	v1.19.0

出力には作成したすべてのマシンが一覧表示されます。



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-bootstrap	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。この一覧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。

- 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 が承認されたら、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、手動の承認が必要です。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ユーザーによってプロビジョニングされるインフラストラクチャーなどのマシン 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、ノードのアイデンティティーを確認します。

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

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

- ❶ **<csr_name>** は、現行の CSR の一覧からの 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 について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 の一覧からの 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  v1.20.0
master-1  Ready   master  73m  v1.20.0
master-2  Ready   master  74m  v1.20.0
worker-0  Ready   worker  11m  v1.20.0
worker-1  Ready   worker  11m  v1.20.0
```



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

- CSR の詳細は、[Certificate Signing Requests](#) を参照してください。

1.3.24. 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。

1.3.25. OpenShift Container Platform の Telemetry アクセス

OpenShift Container Platform 4.6 では、クラスターのヘルス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る Telemetry サービスには、インターネットアクセスが必要です。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、[リモートヘルスマニターリング](#) について参照してください。

1.3.26. 次のステップ

- [クラスターをカスタマイズ](#) します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 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 クラスタトラフィックを設定](#) します。
- RHOSP が 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設定しなかった場合には、[RHOSP のアクセスを Floating IP アドレスで設定](#) します。

1.4. 独自のインフラストラクチャーでの KURYR を使用する OPENSTACK へのクラスターのインストール

OpenShift Container Platform バージョン 4.6 では、ユーザーによってプロビジョニングされたインフラストラクチャーを実行するクラスターを Red Hat OpenStack Platform (RHOSP) に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、インストーラーでプロビジョニングされるインストール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。

1.4.1. 前提条件

- [OpenShift Container Platform のインストールおよび更新](#) プロセスについての詳細を確認します。
 - OpenShift Container Platform 4.6 が **Available platforms** セクションの RHOSP バージョンと互換性があることを確認します。[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。
- ネットワーク設定がプロバイダーのネットワークに依存しないことを確認します。プロバイダーネットワークはサポートされません。
- OpenShift Container Platform をインストールする RHOSP アカウントがあります。
- インストールプログラムを実行するマシンには、以下が含まれます。
 - 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

1.4.2. Kuryr SDN について

[Kuryr](#) は、[Neutron](#) および [Octavia](#) Red Hat OpenStack Platform (RHOSP) サービスを使用して Pod およびサービスのネットワークを提供する Container Network Interface (CNI) プラグインです。

Kuryr と OpenShift Container Platform の統合は主に、RHOSP の仮想マシンで実行する OpenShift Container Platform クラスター用に設計されました。Kuryr は、OpenShift Container Platform Pod を RHOSP SDN にプラグインしてネットワークのパフォーマンスを強化します。さらに、これは Pod と RHOSP 仮想インスタンス間の接続を可能にします。

Kuryr コンポーネントは **openshift-kuryr** namespace を使用して OpenShift Container Platform の Pod としてインストールされます。

- **kuryr-controller: master** ノードにインストールされる単一のサービスインスタンスです。これは、OpenShift Container Platform で **Deployment** としてモデリングされます。
- **kuryr-cni**: 各 OpenShift Container Platform ノードで Kuryr を CNI ドライバーとしてインストールし、設定するコンテナです。これは、OpenShift Container Platform で **DaemonSet** オブジェクトとしてモデリングされます。

Kuryr コントローラーは OpenShift Container Platform API サーバーで Pod、サービスおよび namespace の作成、更新、および削除イベントについて監視します。これは、OpenShift Container Platform API 呼び出しを Neutron および Octavia の対応するオブジェクトにマップします。そのため、Neutron トランクポート機能を実装するすべてのネットワークソリューションを使用して、Kuryr 経由で OpenShift Container Platform をサポートすることができます。これには、Open vSwitch (OVS) および Open Virtual Network (OVN) などのオープンソースソリューションや Neutron と互換性のある市販の SDN が含まれます。

Kuryr は、カプセル化された RHOSP テナントネットワーク上の OpenShift Container Platform デプロイメントに使用することが推奨されています。これは、RHOSP ネットワークでカプセル化された OpenShift Container Platform SDN を実行するなど、二重のカプセル化を防ぐために必要です。

プロバイダーネットワークまたはテナント VLAN を使用する場合は、二重のカプセル化を防ぐために

Kuryr を使用する必要はありません。パフォーマンス上の利点はそれほど多くありません。ただし、設定によっては、Kuryr を使用して 2 つのオーバーレイが使用されないようにすることには利点がある場合があります。

Kuryr は、以下のすべての基準が true であるデプロイメントでは推奨されません。

- RHOSP のバージョンが 16 よりも前のバージョンである。
- デプロイメントで UDP サービスが使用されているか、または少数のハイパーバイザーで多数の TCP サービスが使用されている。

または、以下を実行します。

- **ovn-octavia** Octavia ドライバーが無効にされている。
- デプロイメントで、少数のハイパーバイザーで多数の TCP サービスが使用されている。

1.4.3. Kuryr を使用して OpenShift Container Platform を RHOSP にインストールするためのリソースのガイドライン

Kuryr SDN を使用する場合、Pod、サービス、namespace およびネットワークポリシーは RHOSP クォータのリソースを使用します。これにより、最小要件が増加します。また、Kuryr にはデフォルトインストールに必要な要件以外の追加要件があります。

以下のクォータを使用してデフォルトのクラスターの最小要件を満たすようにします。

表1.19 Kuryr を使用する RHOSP のデフォルト OpenShift Container Platform クラスターについての推奨リソース

リソース	値
Floating IP アドレス	3: LoadBalancer タイプに予想されるサービス数
ポート	1500: Pod ごとに 1 つ必要
ルーター	1
サブネット	250: namespace/プロジェクトごとに 1 つ必要
ネットワーク	250: namespace/プロジェクトごとに 1 つ必要
RAM	112 GB
vCPU	28
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	250: サービスおよび NetworkPolicy ごとに 1 つ必要

リソース	値
セキュリティグループルール	1000
ロードバランサー	100: サービスごとに1つ必要
ロードバランサーリスナー	500: サービスで公開されるポートごとに1つ必要
ロードバランサーノード	500: サービスで公開されるポートごとに1つ必要

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。



重要

OVN Octavia ドライバーではなく Amphora ドライバーで Red Hat OpenStack Platform(RHOSP) バージョン 16 を使用している場合、セキュリティグループはユーザープロジェクトではなくサービスアカウントに関連付けられます。

リソースを設定する際には、以下の点に注意してください。

- 必要なポート数は Pod 数よりも大きくなる。Kuryr はポートプールを使用して、事前に作成済みのポートを Pod で使用できるようにし、Pod の起動時間を短縮します。
- 各ネットワークポリシーは RHOSP セキュリティグループにマップされ、**NetworkPolicy** 仕様によっては1つ以上のルールがセキュリティグループに追加される。
- 各サービスは RHOSP ロードバランサーにマップされる。クォータに必要なセキュリティグループの数を見積もる場合には、この要件を考慮してください。
RHOSP バージョン 15 以前のバージョン、または **ovn-octavia driver** を使用している場合、各ロードバランサーにはユーザープロジェクトと共にセキュリティグループがあります。
- クォータはロードバランサーのリソース (VM リソースなど) を考慮しませんが、RHOSP デプロイメントのサイズを決定するにはこれらのリソースを考慮する必要があります。デフォルトのインストールには 50 を超えるロードバランサーがあり、クラスターはそれらのロードバランサーに対応する必要があります。
OVN Octavia ドライバーを有効にして RHOSP バージョン 16 を使用している場合は、1つのロードバランサー仮想マシンのみが生成され、サービスは OVN フロー経由で負荷分散されます。

OpenShift Container Platform デプロイメントは、コントロールプレーンマシン、コンピューティングマシン、およびブートストラップマシンで設定されます。

Kuryr SDN を有効にするには、使用する環境が以下の要件を満たしている必要があります。

- RHOSP 13+ を実行します。
- オーバークラウドと Octavia を使用します。
- Neutron トランクポートの拡張を使用します。
- ML2/OVS Neutron ドライバーが **ovs-hybrid** の代わりに使用れる場合、**openvswitch** ファイアウォールドライバを使用します。

1.4.3.1. クォータの拡大

Kuryr SDN を使用する場合、Pod、サービス、namespace、およびネットワークポリシーが使用する Red Hat OpenStack Platform (RHOSP) リソースに対応するためにクォータを引き上げる必要があります。

手順

- 以下のコマンドを実行して、プロジェクトのクォータを増やします。

```
$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets 250 --networks 250 <project>
```

1.4.3.2. Neutron の設定

Kuryr CNI は Neutron トランクの拡張を使用してコンテナを Red Hat OpenStack Platform (RHOSP) SDN にプラグインします。したがって、Kuryr が適切に機能するには **trunks** 拡張を使用する必要があります。

さらにデフォルトの ML2/OVS Neutron ドライバーを使用する場合には、セキュリティグループがトランクサブポートで実行され、Kuryr がネットワークポリシーを適切に処理できるように、**ovs_hybrid** ではなく **openvswitch** に設定される必要があります。

1.4.3.3. Octavia の設定

Kuryr SDN は Red Hat OpenStack Platform (RHOSP) の Octavia LBaaS を使用して OpenShift Container Platform サービスを実装します。したがって、Kuryr SDN を使用するように RHOSP に Octavia コンポーネントをインストールし、設定する必要があります。

Octavia を有効にするには、Octavia サービスを RHOSP オーバークラウドのインストール時に組み込むか、またはオーバークラウドがすでに存在する場合は Octavia サービスをアップグレードする必要があります。Octavia を有効にする以下の手順は、オーバークラウドのクリーンインストールまたはオーバークラウドの更新の両方に適用されます。



注記

以下の手順では、Octavia を使用する場合に [RHOSP のデプロイメント](#) 時に必要となる主な手順のみを説明します。また、[レジストリーメソッド](#) が変更されることにも留意してください。

以下の例では、ローカルレジストリーの方法を使用しています。

手順

- ローカルレジストリーを使用している場合、イメージをレジストリーにアップロードするためのテンプレートを作成します。以下に例を示します。

```
(undercloud) $ openstack overcloud container image prepare \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
--namespace=registry.access.redhat.com/rhosp13 \
--push-destination=<local-ip-from-undercloud.conf>:8787 \
--prefix=openstack- \
--tag-from-label {version}-{release} \
--output-env-file=/home/stack/templates/overcloud_images.yaml \
--output-images-file /home/stack/local_registry_images.yaml
```

- local_registry_images.yaml** ファイルに Octavia イメージが含まれることを確認します。以下に例を示します。

```
...
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
  push_destination: <local-ip-from-undercloud.conf>:8787
```



注記

Octavia コンテナのバージョンは、インストールされている特定の RHOSP リリースによって異なります。

- コンテナイメージを **registry.redhat.io** からアンダークラウドノードにプルします。

```
(undercloud) $ sudo openstack overcloud container image upload \
--config-file /home/stack/local_registry_images.yaml \
--verbose
```

これには、ネットワークおよびアンダークラウドディスクの速度に応じて多少の時間がかかる可能性があります。

- Octavia ロードバランサーは OpenShift Container Platform API にアクセスするために使用されるため、それらのリスナーの接続のデフォルトタイムアウトを増やす必要があります。デフォルトのタイムアウトは 50 秒です。以下のファイルをオーバークラウドのデプロイコマンドに渡し、タイムアウトを 20 分に増やします。

```
(undercloud) $ cat octavia_timeouts.yaml
parameter_defaults:
  OctaviaTimeoutClientData: 1200000
  OctaviaTimeoutMemberData: 1200000
```



注記

これは RHOSP 13.0.13+ では不要です。

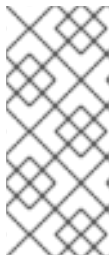
5. Octavia を使用してオーバークラウドをインストールまたは更新します。

```
$ openstack overcloud deploy --templates \
-e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
-e octavia_timeouts.yaml
```



注記

このコマンドには、Octavia に関連付けられたファイルのみが含まれます。これは、RHOSP の特定のインストールによって異なります。詳細は RHOSP のドキュメントを参照してください。Octavia インストールのカスタマイズについての詳細は、[Octavia デプロイメントのプランニング](#) を参照してください。



注記

Kuryr SDN を利用する際には、オーバークラウドのインストールに Neutron の **trunk** 拡張機能が必要です。これは、Director デプロイメントでデフォルトで有効にされます。Neutron バックエンドが ML2/OVS の場合、デフォルトの **ovs-hybrid** の代わりに **openvswitch** ファイアウォールを使用します。バックエンドが ML2/OVN の場合には変更の必要がありません。

6. RHOSP の 13.0.13 よりも前のバージョンでは、プロジェクトの作成後にプロジェクト ID を **octavia.conf** 設定ファイルに追加します。

- トラフィックが Octavia ロードバランサーを通過する場合など、複数のサービス全体でネットワークポリシーを実行するには、Octavia がユーザープロジェクトで Amphora 仮想マシンセキュリティグループを作成するようする必要があります。この変更により、必要なロードバランサーのセキュリティグループがそのプロジェクトに属し、それらをサービスの分離を実行するように更新できます。



注記

RHOSP の 13.0.13 よりも後のバージョンでは、このタスクは必要ありません。

Octavia は、ロードバランサー VIP へのアクセスを制限する新しい ACL API を実装します。

a. プロジェクト ID を取得します。

```
$ openstack project show <project>
```

出力例

```
+-----+-----+
| Field  | Value                                |
+-----+-----+
| description |                                          |
| domain_id | default                              |
| enabled    | True                                |
| id         | PROJECT_ID                          |
| is_domain  | False                               |
| name       | *<project>*                         |
```

```
| parent_id | default |
| tags      | []      |
+-----+
```

- b. プロジェクト ID をコントローラーの **octavia.conf** に追加します。

- i. **stackrc** ファイルを取得します。

```
$ source stackrc # Undercloud credentials
```

- ii. オーバークラウドコントローラーを一覧表示します。

```
$ openstack server list
```

出力例

```
+-----+-----+-----+-----+-----+
| ID              | Name          | Status | Networks |
| Image          | Flavor        |        |          |
+-----+-----+-----+-----+
| 6bef8e73-2ba5-4860-a0b1-3937f8ca7e01 | controller-0 | ACTIVE | ctlplane=192.168.24.8 | overcloud-full | controller |
| dda3173a-ab26-47f8-a2dc-8473b4a67ab9 | compute-0   | ACTIVE | ctlplane=192.168.24.6 | overcloud-full | compute    |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

- iii. コントローラーに対して SSH を実行します。

```
$ ssh heat-admin@192.168.24.8
```

- iv. **octavia.conf** ファイルを編集して、プロジェクトを Amphora セキュリティーグループがユーザーのアカウントに設定されているプロジェクトの一覧に追加します。

```
# List of project IDs that are allowed to have Load balancer security groups
# belonging to them.
amp_secgroup_allowed_projects = PROJECT_ID
```

- c. 新しい設定が読み込まれるように Octavia ワーカーを再起動します。

```
controller-0$ sudo docker restart octavia_worker
```



注記

RHOSP 環境によっては、Octavia は UDP リスナーをサポートしない可能性があります。RHOSP の 13.0.13 よりも前のバージョンで Kuryr SDN を使用する場合、UDP サービスはサポートされません。RHOSP バージョン 16 以降は UDP をサポートします。

1.4.3.3.1. Octavia OVN ドライバー

Octavia は Octavia API を使用して複数のプロバイダードライバーをサポートします。

利用可能なすべての Octavia プロバイダードライバーをコマンドラインで表示するには、以下を入力します。

```
$ openstack loadbalancer provider list
```

出力例

```
+-----+-----+
| name   | description                               |
+-----+-----+
| amphora | The Octavia Amphora driver.              |
| octavia | Deprecated alias of the Octavia Amphora driver. |
| ovn     | Octavia OVN driver.                      |
+-----+-----+
```

RHOSP バージョン 16 以降、Octavia OVN プロバイダードライバー (**ovn**) は RHOSP デプロイメントの OpenShift Container Platform でサポートされます。

ovn は、Octavia および OVN が提供する負荷分散用の統合ドライバーです。これは基本的な負荷分散機能をサポートし、OpenFlow ルールに基づいています。このドライバーは、OVN Neutron ML2 を使用するデプロイメント上の director により Octavia で自動的に有効にされます。

Amphora プロバイダードライバーがデフォルトのドライバーです。ただし、**ovn** が有効にされる場合には、Kuryr がこれを使用します。

Kuryr が Amphora の代わりに **ovn** を使用する場合は、以下の利点があります。

- リソース要件が減少します。Kuryr は、各サービスにロードバランサーの仮想マシンを必要としません。
- ネットワークレイテンシーが短縮されます。
- サービスごとに仮想マシンを使用する代わりに、OpenFlow ルールを使用することで、サービスの作成速度が上がります。
- Amphora 仮想マシンで集中管理されるのではなく、すべてのノードに分散負荷分散アクションが分散されます。

1.4.3.4. Kuryr を使用したインストールについての既知の制限

OpenShift Container Platform を Kuryr SDN で使用する場合、いくつかの既知の制限があります。

RHOSP の一般的な制限

Kuryr SDN を使用する OpenShift Container Platform は、タイプが **NodePort** の **Service** オブジェクトをサポートしません。

マシンのサブネットがルーターに接続されていない場合や、サブネットが接続されていても、ルーターに外部ゲートウェイが設定されていない場合、Kuryr はタイプが **LoadBalancer** の **Service** オブジェクトの Floating IP を作成できません。

- **Service** オブジェクトで **sessionAffinity=ClientIP** プロパティを設定しても効果はありません。Kuryr はこの設定をサポートしていません。

RHOSP バージョンの制限

OpenShift Container Platform を Kuryr SDN で使用する場合は、RHOSP バージョンに依存するいくつかの制限があります。

- RHOSP の 16 よりも前のバージョンでは、デフォルトの Octavia ロードバランサードライバー (Amphora) を使用します。このドライバーでは、OpenShift Container Platform サービスごとに 1 つの Amphora ロードバランサー仮想マシンをデプロイする必要があります。サービス数が多すぎると、リソースが不足する可能性があります。
OVN Octavia ドライバーが無効にされている以降のバージョンの RHOSP のデプロイメントでも Amphora ドライバーを使用します。この場合も、RHOSP の以前のバージョンと同じリソースに関する懸念事項を考慮する必要があります。
- バージョン 13.0.13 よりも前の Octavia RHOSP バージョンは UDP リスナーをサポートしません。そのため、OpenShift Container Platform UDP サービスはサポートされません。
- 13.0.13 よりも前の Octavia RHOSP バージョンは、同じポートで複数のプロトコルをリッスンできません。TCP や UDP など、同じポートを異なるプロトコルに公開するサービスはサポートされません。
- Kuryr SDN は、サービスによる自動解凍をサポートしていません。

RHOSP 環境の制限

Kuryr SDN を使用する場合に、デプロイメント環境に依存する制限事項があります。

Octavia には UDP プロトコルおよび複数のリスナーのサポートがないため、RHOSP バージョンが 13.0.13 よりも前のバージョンの場合、Kuryr は Pod が DNS 解決に TCP を使用するよう強制します。

Go バージョン 1.12 以前では、CGO サポートが無効にされた状態でコンパイルされたアプリケーションは UDP のみを使用します。この場合、ネイティブの Go リゾルバーは、TCP が DNS 解決に強制的に実行されるかどうかを制御する、**resolv.conf** の **use-vc** オプションを認識しません。その結果、UDP は引き続き DNS 解決に使用されますが、これは失敗します。

TCP の強制を許可するには、環境変数 **CGO_ENABLED** を **1** に設定 (例: **CGO_ENABLED=1**) されている状態でアプリケーションをコンパイルするか、または変数がないことを確認します。

Go バージョン 1.13 以降では、UDP を使用した DNS 解決が失敗する場合に TCP が自動的に使用されます。



注記

Alpine ベースのコンテナを含む musl ベースのコンテナは **use-vc** オプションをサポートしません。

RHOSP のアップグレードの制限

RHOSP のアップグレードプロセスにより、Octavia API が変更され、ロードバランサーに使用される Amphora イメージへのアップグレードが必要になる可能性があります。

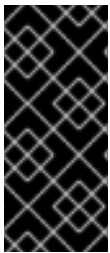
API の変更に対応できます。

Amphora イメージがアップグレードされると、RHOSP Operator は既存のロードバランサー仮想マシンを2つの方法で処理できます。

- [ロードバランサーのフェイルオーバー](#) をトリガーしてそれぞれの仮想マシンをアップグレードします。
- ユーザーが仮想マシンのアップグレードを行う必要があります。

Operator が最初のオプションを選択する場合、フェイルオーバー時に短い時間のダウンタイムが生じる可能性があります。

Operator が2つ目のオプションを選択する場合、既存のロードバランサーはUDP リスナーなどのアップグレードされた Octavia API 機能をサポートしません。この場合、ユーザーはこれらの機能を使用するためにサービスを再作成する必要があります。



重要

OpenShift Container Platform が UDP の負荷分散をサポートする新規の Octavia バージョンを検出する場合、これは DNS サービスを自動的に再作成します。サービスの再作成により、サービスのデフォルトが UDP の負荷分散をサポートようになります。

再作成により、DNS サービスに約1分間のダウンタイムが発生します。

1.4.3.5. 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、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.4.3.6. 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、2 つの vCPU および 100 GB のストレージ領域があるフレーバー

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。

1.4.3.7. 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、4 つの vCPU および 100 GB のストレージ領域があるフレイバー

1.4.4. OpenShift Container Platform のインターネットアクセス

OpenShift Container Platform 4.6 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になります。

インターネットへのアクセスは以下を実行するために必要です。

- [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。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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クラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、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。

1.4.5. Playbook 依存関係のダウンロード

ユーザーによってプロビジョニングされたインフラストラクチャーでのインストールプロセスを単純化する Ansible Playbook には、複数の Python モジュールが必要です。インストーラーを実行するマシンで、モジュールのリポジトリを追加し、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. 必要なリポジトリを追加します。

```
$ sudo subscription-manager repos \
--enable=rhel-8-for-x86_64-baseos-rpms \
--enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
--enable=ansible-2.9-for-rhel-8-x86_64-rpms \
--enable=rhel-8-for-x86_64-appstream-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 python3-netaddr
```

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
```

1.4.6. 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。

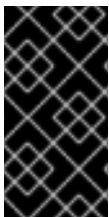
```
$ xargs -n 1 curl -O <<< '
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/common.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/compute-nodes.yaml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

```

https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/control-
plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/inventory.yaml
https://raw.githubusercontent.com/openshift/installer/release-
4.6/upi/openstack/network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/security-
groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
compute-nodes.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
control-plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
load-balancers.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-
containers.yaml'

```

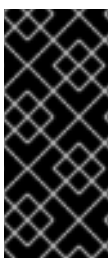
Playbook はマシンにダウンロードされます。



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。

1.4.7. インストールプログラムの取得

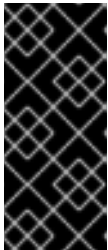
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

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。

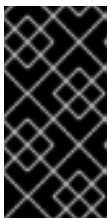
手順

1. OpenShift Cluster Manager サイトの [インフラストラクチャプロバイダー](#) ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. インフラストラクチャプロバイダーを選択します。
3. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、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。ファイル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。



重要

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。

4. 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
```

5. [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。

1.4.8. SSH プライベートキーの生成およびエージェントへの追加

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの `~/.ssh/authorized_keys` 一覧に追加されます。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、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> ❶
```

- ❶ ~/.ssh/id_rsa などの、新規 SSH キーのパスおよびファイル名を指定します。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。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは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. **ssh-agent** プロセスを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。

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

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

出力例

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

- ❶ ~/.ssh/id_rsa 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。

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。

1.4.9. Red Hat Enterprise Linux CoreOS (RHCOS) イメージの作成

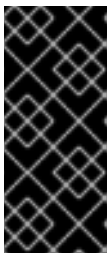
OpenShift Container Platform インストールプログラムでは、Red Hat Enterprise Linux CoreOS (RHCOS) イメージが Red Hat OpenStack Platform (RHOSP) クラスターに存在する必要があります。最新の RHCOS イメージを取得した後、RHOSP CLI を使用してこれをアップロードします。

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 4.6 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中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、または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。

1.4.10. 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一覧に表示されます。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。

1.4.11. 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。

1.4.11.1. floating IP アドレスを使っ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があります。

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。

1.4.11.2. 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。

`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ラーが失敗するのを防ぐために、プロキシ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。

1.4.12. 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** についての詳細は、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: shiftstack_user
      password: XXX
```

```

    user_domain_name: Default
    project_domain_name: Default
dev-env:
    region_name: RegionOne
auth:
    username: 'devuser'
    password: XXX
    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. マシンを認証局の信頼バンドルに追加します。

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

- c. 信頼バンドルを更新します。

```
$ sudo update-ca-trust extract
```

- d. **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。

1.4.13. 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を取得します。

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。



重要

空のディレクトリーを指定します。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の有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。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コピーを行ってください。

- 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がある実稼働用の 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およびコンピューターノードに使用する 16 GB 以上の RAM で 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。
 - viii. [Red Hat OpenShift Cluster Manager からプルシークレット](#) を貼り付けます。
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については、[インストール設定パラメーターセクション](#)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。

1.4.14. インストール設定パラメーター

OpenShift Container Platform クラスターをデプロイする前に、クラスターをホストするクラウドプラットフォームでアカウントを記述し、クラスターのプラットフォームをオプションでカスタマイズするためにパラメーターの値を指定します。**install-config.yaml** インストール設定ファイルを作成する際に、コマンドラインで必要なパラメーターの値を指定します。クラスターをカスタマイズする場合、**install-config.yaml** ファイルを変更して、プラットフォームについての詳細情報を指定できます。



注記

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。



重要

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.4.14.1. 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1.20 必須パラメーター

パラメーター	説明	値
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。インストーラーは、古い API バージョンをサポートすることもできます。	文字列

パラメーター	説明	値
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、 baseDomain と <metadata.name> 、 <baseDomain> 形式を使用する metadata.name パラメーターの値の組み合わせです。	example.com などの完全修飾ドメイン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。	オブジェクト
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。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。文字列は 14 文字以上でなければなりません。
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 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について参照してください。	オブジェクト

パラメーター	説明	値
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イメージをダウンロードすることを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.4.14.2. 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するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.21 ネットワークパラメーター


パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p>  <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p>
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。
networking.clusterNetwork	<p>Pod の IP アドレスブロック。</p> <p>デフォルト値は 10.128.0.0/14 で、ホストの接頭辞は /23 です。</p> <p>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。</p>	<p>オブジェクトの配列。以下に例を示します。</p> <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>

パラメーター	説明	値
networking.clusterNetwork.cidr	<p>networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。</p> <p>IPv4 ネットワーク</p>	CIDR (Classless Inter-Domain Routing) 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。
networking.clusterNetwork.hostPrefix	<p>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、$2^{(32 - 23) - 2}$ Pod IP アドレスを提供します。</p>	<p>サブネット接頭辞。</p> <p>デフォルト値は 23 です。</p>
networking.serviceNetwork	<p>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。</p> <p>OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。</p>	<p>CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>マシンの IP アドレスブロック。</p> <p>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。</p>	<p>オブジェクトの配列。以下に例を示します。</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、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。</p>	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>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。</p> </div> </div>

1.4.14.3. 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1.22 オプションのパラメーター

パラメーター	説明	値
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。この信頼バンドルは、プロキシーが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列
compute.hyperthreading	<p>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にされます。</p> <div>  <p>重要</p> <p>同時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ていることを確認します。</p> </div>	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。	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。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 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。	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	値
controlPlane	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。 詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列
controlPlane.hyperthreading	<p>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にされます。</p> <div>  <div> <p>重要</p> <p>同時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ていることを確認します。</p> </div> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	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。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 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。	サポートされる値は 3 のみです (これはデフォルト値です)。

パラメーター	説明	値
credentialsMode	<p>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。</p> <div><p>注記</p><p>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、Red Hat Operator のクラウド認証情報 Operatorを参照してください。</p></div>	Mint、Passthrough、Manual 、または空の文字列 ("")。

パラメーター	説明	値
fips	<p>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 モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>重要</p> <p>FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、x86_64 アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。</p> </div> <div>  <p>注記</p> <p>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。</p> </div>	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。	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を指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。

パラメーター	説明	値
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。	<p>Internal または External。デフォルト値は External です。</p> <p>このパラメーターを Internal に設定することは、クラウド以外のプラットフォームではサポートされません。</p> <div>  <div> <p>重要</p> <p>フィールドの値が Internal に設定されている場合、クラスターは機能しなくなります。詳細は、BZ#1953035 を参照してください。</p> </div> </div>
sshKey	<p>クラスターマシンへのアクセスを認証するための単一または複数の SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の 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。</p> </div> </div>	<p>1つ以上のキー。以下に例を示します。</p> <pre>sshKey: <key1> <key2> <key3></pre>

1.4.14.4. 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1.23 追加の RHOSP パラメーター

パラメーター	説明	値
compute.platform.openstack.rootVolume.size	コンピュータマシンの場合、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。	整数 (例: 30)。

パラメーター	説明	値
compute.platform.openstack.rootVolume.type	コンピュータマシンの場合、root のボリュームタイプです。	文字列 (例: performance)。
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。	整数 (例: 30)。
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。	文字列 (例: performance)。
platform.openstack.cloud	clouds.yaml ファイルのクラウド一覧にある使用する RHOSP クラウドの名前。	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。	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タマシンに使用する RHOSP フレーバー。	文字列 (例: m1.xlarge)。

1.4.14.5. 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1.24 オプションの RHOSP パラメーター

パラメーター	説明	値
compute.platform.openstack.additionalNetworkIDs	コンピュータ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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。
compute.platform.openstack.additionalSecurityGroupIDs	コンピュータマシンに関連付けられた追加のセキュリティグループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
compute.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: ["zone-1", "zone-2"] 。
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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。
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グループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
controlPlane.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: <code>["zone-1", "zone-2"]</code> 。
platform.openstack.clusterOSImage	<p>インストーラーが RHCOS イメージをダウンロードする場所。</p> <p>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。</p>	<p>HTTP または HTTPS の URL (オプションで SHA-256 形式のチェックサムを使用)。</p> <p>例: <code>http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</code>。この値は、既存の Glance イメージの名前にもなり得ます (例: <code>my-rhcos</code>)。</p>
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。	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	<p>Ingress ポートに関連付ける既存の Floating IP アドレス。このプロパティを使用するには、platform.openstack.externalNetwork プロパティも定義する必要があります。</p>	IP アドレス (例: <code>128.0.0.1</code>)。

パラメーター	説明	値
platform.openstack.lbFloatingIP	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。	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。	文字列としての IP アドレスの一覧。例: ["8.8.8.8", "192.168.1.12"]
platform.openstack.machinesSubnet	<p>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。</p> <p>networking.machineNetwork の最初の項目は machinesSubnet の値に一致する必要があります。</p> <p>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。</p>	文字列としての UUID。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。

1.4.14.6. RHOSP デプロイメントでのカスタムサブネット

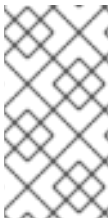
オプションで、選択する Red Hat OpenStack Platform (RHOSP) サブネットにクラスターをデプロイすることができます。サブネットの GUID は、**install-config.yaml** ファイルの **platform.openstack.machinesSubnet** の値として渡されます。

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。ノードとポートはこの上に作成されます。

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、以下を確認します。

- ターゲットネットワークおよびサブネットが利用可能である。
- DHCP がターゲットサブネットで有効にされている。
- ターゲットネットワーク上でポートを作成するためのパーミッションがあるインストーラー認証情報を指定できます。

- ネットワーク設定にルーターが必要な場合、これは RHOSP で作成されます。一部の設定は、Floating IP アドレスの変換用のルーターに依存します。
- ネットワーク設定は、プロバイダーのネットワークに依存しません。プロバイダーネットワークはサポートされません。



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.apiVIP** および **platform.openstack.ingressVIP** の値を設定します。

1.4.14.7. Kuryr を使用した OpenStack のカスタマイズされた `install-config.yaml` ファイルのサンプル

デフォルトの OpenShift SDN ではなく Kuryr SDN を使用してデプロイするには、**install-config.yaml** ファイルを変更して **Kuryr** を必要な **networking.networkType** として追加してから、デフォルトの OpenShift Container Platform SDN インストール手順に進む必要があります。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
clusterID: os-test
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 ①
  networkType: Kuryr
platform:
  openstack:
    cloud: mycloud
```

```
externalNetwork: external
computeFlavor: m1.xlarge
lbFloatingIP: 128.0.0.1
trunkSupport: true ②
octaviaSupport: true ③
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

- ① Amphora Octavia ドライバーは、ロードバランサーごとに 2 つのポートを作成します。そのため、インストーラーが作成するサービスサブネットは、**serviceNetwork** プロパティの値として指定される CIDR のサイズは 2 倍になります。IP アドレスの競合を防ぐには、範囲をより広くする必要があります。
- ② ③ **trunkSupport** と **octaviaSupport** の両方はインストーラーによって自動的に検出されるため、それらを設定する必要はありません。ただし、ご使用の環境がこれらの両方の要件を満たさないと、Kuryr SDN は適切に機能しません。トランクは Pod を RHOSP ネットワークに接続するために必要であり、Octavia は OpenShift Container Platform サービスを作成するために必要です。

1.4.14.8. Kuryr ポートプール

Kuryr ポートプールでは、Pod 作成のスタンバイ状態の多数のポートを維持します。

ポートをスタンバイ状態に維持すると、Pod の作成時間が必要最小限に抑えることができます。ポートプールを使用しない場合には、Kuryr は Pod が作成または削除されるたびにポートの作成または削除を明示的に要求する必要があります。

Kuryr が使用する Neutron ポートは、namespace に関連付けられるサブネットに作成されます。これらの Pod ポートは、OpenShift Container Platform クラスターノードのプライマリーポートにサブポートとして追加されます。

Kuryr は namespace をそれぞれ、別のサブネットに保存するため、namespace-worker ペアごとに別個のポートプールが維持されます。

クラスターをインストールする前に、**cluster-network-03-config.yml** マニフェストファイルに以下のパラメーターを設定して、ポートプールの動作を設定できます。

- **enablePortPoolsPrepopulation** パラメーターで、プールの事前生成が制御されるので、Kuryr は新規ホストが追加される時や新規 namespace の作成時などの作成時に、Kuryr によりプールにポートが強制的に追加されるようにします。デフォルト値は **false** です。
- **poolMinPorts** パラメーターは、プールに保持する空きポートの最小数です。デフォルト値は **1** です。
- **poolMaxPorts** パラメーターは、プールに保持する空きポートの最大数です。値が **0** の場合は、上限が無効になります。これはデフォルト設定です。
OpenStack ポートのクォータが低い場合や、Pod ネットワークで IP アドレスの数が限定されている場合には、このオプションを設定して、不要なポートが削除されるようにします。
- **poolBatchPorts** パラメーターは、一度に作成可能な Neutron ポートの最大数を定義します。デフォルト値は **3** です。

1.4.14.9. インストール時の Kuryr ポートプールの調整

インストール時に、Pod 作成の速度や効率性を制御するために Kuryr で Red Hat OpenStack Platform (RHOSP) Neutron ポートを管理する方法を設定できます。

前提条件

- **install-config.yaml** ファイルを作成して変更しておく。

手順

1. コマンドラインからマニフェストファイルを作成します。

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

- 1 **<installation_directory>** については、クラスターの **install-config.yaml** ファイルが含まれるディレクトリーの名前を指定します。

2. **cluster-network-03-config.yml** という名前のファイルを **<installation_directory>/manifests/** ディレクトリーに作成します。

```
$ touch <installation_directory>/manifests/cluster-network-03-config.yml 1
```

- 1 **<installation_directory>** については、クラスターの **manifests/** ディレクトリーが含まれるディレクトリー名を指定します。

ファイルの作成後は、以下のようにいくつかのネットワーク設定ファイルが **manifests/** ディレクトリーに置かれます。

```
$ ls <installation_directory>/manifests/cluster-network-*
```

出力例

```
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. エディターで **cluster-network-03-config.yml** ファイルを開き、必要な Cluster Network Operator 設定を記述するカスタムリソース (CR) を入力します。

```
$ oc edit networks.operator.openshift.io cluster
```

4. 要件に合わせて設定を編集します。以下のファイルをサンプルとして紹介しています。

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  serviceNetwork:
    - 172.30.0.0/16
  defaultNetwork:
    type: Kuryr
    kuryrConfig:
```

```
enablePortPoolsPrepopulation: false ❶
poolMinPorts: 1 ❷
poolBatchPorts: 3 ❸
poolMaxPorts: 5 ❹
openstackServiceNetwork: 172.30.0.0/15 ❺
```

- ❶ **enablePortPoolsPrepopulation** の値を **true** に設定し、namespace の作成時、または新規ノードがクラスターに追加された後に Kuryr が新規 Neutron ポートを作成するようにします。この設定により、Neutron ポートのクォータが引き上げられますが、Pod の起動に必要なとなる時間を短縮できます。デフォルト値は **false** です。
- ❷ Kuryr は、対象のプール内にある空きポートの数が **poolMinPorts** の値よりも少ない場合には、プールに新規ポートを作成します。デフォルト値は **1** です。
- ❸ **poolBatchPorts** は、空きポートの数が **poolMinPorts** の値よりも少ない場合に作成される新規ポートの数を制御します。デフォルト値は **3** です。
- ❹ プール内の空きポートの数が **poolMaxPorts** の値よりも多い場合に、Kuryr はその値と同じ数になるまでポートを削除します。この値を **0** に設定すると、この上限は無効になり、プールが縮小できないようにします。デフォルト値は **0** です。
- ❺ **openStackServiceNetwork** パラメーターは、RHOSP Octavia の LoadBalancer に割り当てられるネットワークの CIDR 範囲を定義します。

このパラメーターを Amphora ドライバーと併用する場合には、Octavia は、ロードバランサーごとに、このネットワークから IP アドレスを 2 つ (OpenShift 用に 1 つ、VRRP 接続用に 1 つ) 取得します。これらの IP アドレスは OpenShift Container Platform と Neutron でそれぞれ管理されるため、異なるプールから取得する必要があります。したがって、**openStackServiceNetwork serviceNetwork** の値の 2 倍になる必要があり、**serviceNetwork** の値は、**openStackServiceNetwork** で定義された範囲と完全に重複する必要があります。

CNO は、このパラメーターの定義範囲から取得した VRRP IP アドレスが **serviceNetwork** パラメーターの定義範囲と重複しないことを検証します。

このパラメーターが設定されていない場合には、CNO は **serviceNetwork** の拡張値を使用します。この値は、プリフィックスのサイズを 1 つずつ減らして決定します。

5. **cluster-network-03-config.yml** ファイルを保存し、テキストエディターを終了します。
6. オプション: **manifests/cluster-network-03-config.yml** ファイルをバックアップします。インストールプログラムは、クラスターの作成時に **manifests/** ディレクトリーを削除します。

1.4.14.10. 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。

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、または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["networking"]["machineNetwork"] = [{"cidr": "192.168.0.0/18"}]; ❶
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

❶ 必要な Neutron サブネットに一致する値 (例: **192.0.2.0/24**) を挿入します。

- 値を手動で設定するには、ファイルを開き、**networking.machineCIDR** の値を必要な Neutron サブネットに一致する値に設定します。

1.4.14.11. 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、または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。

1.4.14.12. ネットワークタイプの変更

デフォルトで、インストールプログラムは **OpenShiftSDN** ネットワークタイプを選択します。代わりに Kuryr を使用するには、プログラムが生成したインストール設定ファイルの値を変更します。

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. コマンドプロンプトで、**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、または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["networking"]["networkType"] = "Kuryr";
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、**networking.networkType** を **"Kuryr"** に設定します。

1.4.15. 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作成するために必要な 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。これはクラスターを作成するために後に使用されます。

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** についての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お勧めします。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. 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> ❶
```

- ❶ **<installation_directory>** については、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およびコンピューティング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
```

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。

- 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ティング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> ❶
```

- ❶ **<installation_directory>** については、同じインストールディレクトリーを指定します。

以下のファイルはディレクトリーに生成されます。

```
.
├── 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。

1.4.16. 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)

```

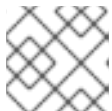
2. RHOSP CLI を使用して、ブートストラップ Ignition ファイルを使用するイメージを作成します。

```
$ 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>", ❶
        "httpHeaders": [{
          "name": "X-Auth-Token", ❷
          "value": "<token_ID>" ❸
        }]
      }]
    },
    "security": {
      "tls": {
        "certificateAuthorities": [{
          "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>" ❹
        }]
      }
    },
    "version": "3.1.0"
  }
}
```

- ❶ **ignition.config.append.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。
- ❷ **httpHeaders** の **name** を **"X-Auth-Token"** に設定します。
- ❸ **httpHeaders** の **value** をトークンの ID に設定します。
- ❹ 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。

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

独自のインフラストラクチャーを使用して OpenShift Container Platform を 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**。

1.4.18. 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

- 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。

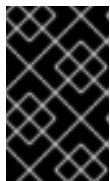
手順

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 の値を定義しない場合、インストーラーは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、**network.yaml** Playbook を実行して、ネットワーク、サブネット、およびルーターを作成します。

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

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"
```

1.4.19. 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"
```

1.4.20. 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 Ignition 設定ファイルが作業ディレクトリーにない場合、それらをここにコピーします。
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 v1.14.6+f9b5405 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

1.4.21. 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ついての情報が含まれます。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。
- **oc** CLI をインストールしていること。

手順

1. **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. 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
```

1.4.22. 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。

1.4.23. RHOSP でのコンピュータマシンの作成

コントロールプレーンの起動後、コンピュータ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。

1.4.24. 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ついて 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、または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	v1.19.0
master-1	Ready	master	63m	v1.19.0
master-2	Ready	master	64m	v1.19.0

出力には作成したすべてのマシンが一覧表示されます。



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。この一覧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 が承認されたら、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、手動の承認が必要です。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ユーザーによってプロビジョニングされるインフラストラクチャーなどのマシン 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、ノードのアイデンティティーを確認します。

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

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

❶ **<csr_name>** は、現行の CSR の一覧からの 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 について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 の一覧からの 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   v1.20.0
master-1  Ready    master   73m   v1.20.0
master-2  Ready    master   74m   v1.20.0
worker-0  Ready    worker   11m   v1.20.0
worker-1  Ready    worker   11m   v1.20.0
```



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

- CSR の詳細は、[Certificate Signing Requests](#) を参照してください。

1.4.25. 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。

1.4.26. OpenShift Container Platform の Telemetry アクセス

OpenShift Container Platform 4.6 では、クラスターのヘルス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る Telemetry サービスには、インターネットアクセスが必要です。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、[リモートヘルスマニターリング](#) について参照してください。

1.4.27. 次のステップ

- [クラスターをカスタマイズ](#) します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 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 クラスタートラフィックを設定](#) します。
- RHOSP が 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設定しなかった場合には、[RHOSP のアクセスを Floating IP アドレスで設定](#) します。

1.5. ネットワークが制限された環境での OPENSTACK へのクラスタのインストール

OpenShift Container Platform 4.6 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、クラスタをネットワークが制限された環境で Red Hat OpenStack Platform (RHOSP) にインストールできます。

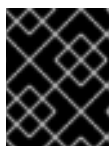


注記

ネットワークが制限された環境でのインストールは、インストーラーでプロビジョニングされるインストールでのみサポートされます。

前提条件

- [ミラーホストでレジストリーを作成](#) し、OpenShift Container Platform の使用しているバージョン用の **imageContentSources** データを取得します。



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します。

- [OpenShift Container Platform のインストールおよび更新](#) プロセスについての詳細を確認します。
 - アーキテクチャドキュメントの [利用可能なプラットフォームの一覧](#) を参照して、OpenShift Container Platform 4.6 が RHOSP バージョンと互換性があることを確認します。 [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。
- ネットワーク設定がプロバイダーのネットワークに依存しないことを確認します。プロバイダーネットワークはサポートされません。
- RHOSP でメタデータサービスが有効化されています。

1.5.1. 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 4.6 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、インストーラーでプロビジョニングされるインフラストラクチャーまたはユーザーによってプロビジョニングされるインフラストラクチャー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または VMware vSphere へのインストールには、インターネットアクセスが必要になる場合があります。

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 Container Platform 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。

1.5.1.1. 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、開発者カタログのコンテンツは、必要とされるイメージストリームタグにアクセスできないために使用できません。

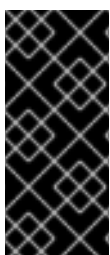
1.5.2. OpenShift Container Platform を RHOSP にインストールするリソースのガイドライン

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1.25 RHOSP のデフォルトの OpenShift Container Platform クラスターについての推奨リソース

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	112 GB
vCPU	28
ボリュームストレージ	275 GB
インスタンス	7
セキュリティグループ	3
セキュリティグループルール	60

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、コンピューターマシン、およびブートストラップマシンで設定されます。

1.5.2.1. 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、4 つの vCPU および 100 GB のストレージ領域があるフレーバー

1.5.2.2. コンピューターマシン

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、2 つの vCPU および 100 GB のストレージ領域があるフレーバー

ヒント

コンピューターマシンは、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。

1.5.2.3. 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

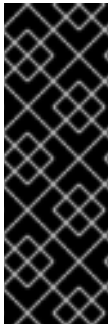
- 少なくとも 16 GB のメモリー、4 つの vCPU および 100 GB のストレージ領域があるフレイバー

1.5.3. OpenShift Container Platform のインターネットアクセス

OpenShift Container Platform 4.6 では、クラスターをインストールするために必要なイメージを取得するために、インターネットアクセスが必要になります。

インターネットへのアクセスは以下を実行するために必要です。

- [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。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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。

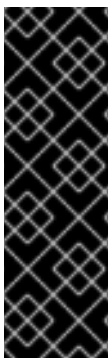


重要

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クラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、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。

1.5.4. 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

Swift として知られる [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。

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。

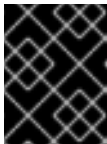
1.5.5. インストールプログラムのパラメーターの定義

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** についての詳細は、RHOSP ドキュメントの [Config files](#) を参照してください。

```
clouds:
  swiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: swiftstack
      username: swiftstack_user
      password: XXX
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: 'devuser'
      password: XXX
      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) を使用する場合は、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. マシンを認証局の信頼バンドルに追加します。

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

- c. 信頼バンドルを更新します。

```
$ sudo update-ca-trust extract
```

- d. **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。

1.5.6. 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

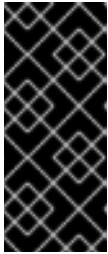
Red Hat Enterprise Linux CoreOS (RHCOS) イメージをダウンロードし、OpenShift Container Platform をネットワークが制限された Red Hat OpenStack Platform (RHOSP) 環境にインストールします。

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 4.6 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中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、または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。

1.5.7. 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、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** 値を使用します。
- ミラーレジストリーの証明書の内容を取得します。
- Red Hat Enterprise Linux CoreOS (RHCOS) イメージを取得し、これをアクセス可能な場所にアップロードします。

手順

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

- 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。



重要

空のディレクトリーを指定します。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の有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。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コピーを行ってください。

- 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。
 - オプション: クラスタ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。

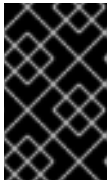


注記

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がある実稼働用の OpenShift Container Platform クラスタでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

- 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。
- 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。
- 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。

4. 必要な **install-config.yaml** ファイルに他の変更を加えます。利用可能なパラメーターの詳細については、**インストール設定パラメーターセクション**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。

1.5.7.1. インストール時のクラスター全体のプロキシの設定

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 プロキシを使用することができます。プロキシ設定を **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ついてのクラウドプロバイダー API に対する呼び出しを含む) はプロキシされます。プロキシ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)、Google Cloud Platform (GCP)、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-----
...
```

- 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 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 URL。
- 3 プロキシ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 のコンマ区切りの一覧。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をバイパスします。
- 4 指定されている場合には、インストールプログラムは、**openshift-config** namespace に **user-ca-bundle** という名前の設定魔府を生成して、追加の CA 証明書を保存します。**additionalTrustBundle** と少なくとも 1 つのプロキシ設定を指定した場合には、**Proxy** オブジェクトは **trusted CA** フィールドで **user-ca-bundle** 設定マップを参照するように設定されます。その後、Cluster Network Operator は、**trustedCA** パラメーターに指定されたコンテンツを RHCOS トラストバンドルにマージする **trusted-ca-bundle** 設定マップを作成します。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が 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。

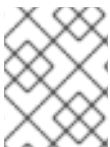


注記

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

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を作成します。プロキシ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、追加のプロキシを作成することはできません。

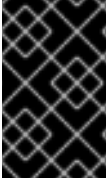
1.5.7.2. インストール設定パラメーター

OpenShift Container Platform クラスターをデプロイする前に、クラスターをホストするクラウドプラットフォームでアカウントを記述し、クラスターのプラットフォームをオプションでカスタマイズするためにパラメーターの値を指定します。**install-config.yaml** インストール設定ファイルを作成する際に、コマンドラインで必要なパラメーターの値を指定します。クラスターをカスタマイズする場合、**install-config.yaml** ファイルを変更して、プラットフォームについての詳細情報を指定できます。



注記

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。



重要

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.5.7.2.1. 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1.26 必須パラメーター

パラメーター	説明	値
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。インストーラーは、古い API バージョンをサポートすることもできます。	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、 baseDomain と <metadata.name> 。 <baseDomain> 形式を使用する metadata.name パラメーターの値の組み合わせです。	example.com などの完全修飾ドメイン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。	オブジェクト
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。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。文字列は 14 文字以上でなければなりません。


パラメーター	説明	値
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 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について参照してください。	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イメージをダウンロードすることを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.5.7.2.2. 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するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.27 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p>  <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p>

パラメーター	説明	値
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN 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。	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	CIDR (Classless Inter-Domain Routing) 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。
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、 $2^{(32 - 23)} - 2$ Pod IP アドレスを提供します。	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。 OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。	CIDR 形式の IP アドレスブロックを持つ配列。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。	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	値
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 以外のすべてのプラットフォームでは、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。	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>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。</p> </div> </div>

1.5.7.2.3. 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1.28 オプションのパラメーター


パラメーター	説明	値
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。この信頼バンドルは、プロキシが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列

パラメーター	説明	値
compute.hyperthreading	<p>コンピュータ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にされます。</p> <div>  <p>重要</p> <p>同時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ていることを確認します。</p> </div>	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。	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。	aws 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (ワーカーマシンとしても知られる) の数。	2 以上の正の整数。デフォルト値は 3 です。
controlPlane	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
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。現時点で異種クラスターはサポートされていないため、すべてのプールが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 (デフォルト) です。	文字列

パラメーター	説明	値
controlPlane.hyperthreading	<p>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にされます。</p> <div>  <div> <p>重要</p> <p>同時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ていることを確認します。</p> </div> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	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。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 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。	サポートされる値は 3 のみです (これはデフォルト値です)。

パラメーター	説明	値
credentialsMode	<p>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。</p> <div><p>注記</p><p>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、Red Hat Operatorのクラウド認証情報 Operatorを参照してください。</p></div>	Mint、Passthrough、Manual 、または空の文字列 ("")。

パラメーター	説明	値
fips	<p>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 モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>重要</p> <p>FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、x86_64 アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。</p> </div> <div>  <p>注記</p> <p>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。</p> </div>	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。	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を指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。

パラメーター	説明	値
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。	<p>Internal または External。デフォルト値は External です。</p> <p>このパラメーターを Internal に設定することは、クラウド以外のプラットフォームではサポートされません。</p> <div>  <div> <p>重要</p> <p>フィールドの値が Internal に設定されている場合、クラスターは機能しなくなります。詳細は、BZ#1953035 を参照してください。</p> </div> </div>
sshKey	<p>クラスターマシンへのアクセスを認証するための単一または複数の SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の 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。</p> </div> </div>	<p>1つ以上のキー。以下に例を示します。</p> <pre>sshKey: <key1> <key2> <key3></pre>

1.5.7.2.4. 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1.29 追加の RHOSP パラメーター

パラメーター	説明	値
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。	整数 (例: 30)。
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。	文字列 (例: performance)。

パラメーター	説明	値
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。	整数 (例: 30)。
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。	文字列 (例: performance)。
platform.openstack.cloud	clouds.yaml ファイルのクラウド一覧にある使用する RHOSP クラウドの名前。	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。	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。	文字列 (例: m1.xlarge)。

1.5.7.2.5. 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1.30 オプションの RHOSP パラメーター

パラメーター	説明	値
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。	文字列としての1つ以上の UUID の一覧。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グループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
compute.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: ["zone-1", "zone-2"] 。
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。	文字列としての1つ以上の UUID の一覧。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。
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グループ。	文字列としての1つ以上の UUID の一覧。例: 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7 。

パラメーター	説明	値
controlPlane.platform.openstack.zones	<p>マシンをインストールする RHOSP Compute (Nova) アベイラビリティゾーン (AZs)。このパラメーターが設定されていない場合、インストーラーは RHOSP 管理者が設定した Nova のデフォルト設定に依存します。</p> <p>Kuryr を使用するクラスターでは、RHOSP Octavia はアベイラビリティゾーンをサポートしません。ロードバランサーおよび Amphora プロバイダードライバーを使用している場合、Amphora 仮想マシンに依存する OpenShift Container Platform サービスは、このプロパティの値に基づいて作成されません。</p>	文字列の一覧例: <code>["zone-1", "zone-2"]</code> 。
platform.openstack.clusterOSImage	<p>インストーラーが RHCOS イメージをダウンロードする場所。</p> <p>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。</p>	<p>HTTP または HTTPS の URL (オプションで SHA-256 形式のチェックサムを使用)。</p> <p>例: <code>http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</code>。この値は、既存の Glance イメージの名前にもなり得ます (例: <code>my-rhcos</code>)。</p>
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。	<pre>{ "type": "ml.large", "rootVolume": { "size": 30, "type": "performance" } }</pre>
platform.openstack.ingressFloatingIP	Ingress ポートに関連付ける既存の Floating IP アドレス。このプロパティを使用するには、 platform.openstack.externalNetwork プロパティも定義する必要があります。	IP アドレス (例: <code>128.0.0.1</code>)。

パラメーター	説明	値
platform.openstack.lbFloatingIP	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。	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。	文字列としての IP アドレスの一覧。例: ["8.8.8.8", "192.168.1.12"]
platform.openstack.machinesSubnet	<p>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。</p> <p>networking.machineNetwork の最初の項目は machinesSubnet の値に一致する必要があります。</p> <p>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。</p>	文字列としての UUID。例: fa806b2f-ac49-4bce-b9db-124bc64209bf 。

1.5.7.3. 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
clusterID: os-test
controlPlane:
  name: master
```


ヒント

[RHOSP インスタンスのスケジューリングおよび配置](#)の詳細は、RHOSP のドキュメントを参照してください。

前提条件

- **install-config.yaml** ファイルを作成し、これに対する変更を完了します。

手順

1. RHOSP コマンドラインインターフェイスを使用して、コンピュータマシンのサーバーグループを作成します。以下に例を示します。

```
$ openstack \
  --os-compute-api-version=2.15 \
  server group create \
  --policy anti-affinity \
  my-openshift-worker-group
```

詳細は、[server group create コマンドのドキュメント](#) を参照してください。

2. 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、マニフェストを作成します。

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

ここでは、以下ようになります。

installation_directory

クラスターの **install-config.yaml** ファイルが含まれるディレクトリーの名前を指定します。

3. **MachineSet** 定義ファイルの **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** を作成します。
4. **spec.template.spec.providerSpec.value** プロパティーの下にある定義に、プロパティー **serverGroupID** を追加します。以下に例を示します。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
    machine.openshift.io/cluster-api-machine-role: <node_role>
    machine.openshift.io/cluster-api-machine-type: <node_role>
  name: <infrastructure_ID>-<node_role>
  namespace: openshift-machine-api
spec:
  replicas: <number_of_replicas>
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
      machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
  template:
    metadata:
```

```

labels:
  machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
  machine.openshift.io/cluster-api-machine-role: <node_role>
  machine.openshift.io/cluster-api-machine-type: <node_role>
  machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
spec:
  providerSpec:
    value:
      apiVersion: openstackproviderconfig.openshift.io/v1alpha1
      cloudName: openstack
      cloudsSecret:
        name: openstack-cloud-credentials
        namespace: openshift-machine-api
      flavor: <nova_flavor>
      image: <glance_image_name_or_location>
      serverGroupID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee 1
      kind: OpenstackProviderSpec
      networks:
        - filter: {}
          subnets:
            - filter:
                name: <subnet_name>
                tags: openshiftClusterID=<infrastructure_ID>
      securityGroups:
        - filter: {}
          name: <infrastructure_ID>-<node_role>
      serverMetadata:
        Name: <infrastructure_ID>-<node_role>
        openshiftClusterID: <infrastructure_ID>
      tags:
        - openshiftClusterID=<infrastructure_ID>
      trunk: true
      userDataSecret:
        name: <node_role>-user-data
      availabilityZone: <optional_openstack_availability_zone>

```

1 サーバーグループの UUID をここに追加します。

5. オプション: **manifests/99_openshift-cluster-api_worker-machineset-0.yaml** ファイルをバックアップします。インストールプログラムは、クラスターの作成時に **manifests/** ディレクトリを削除します。

クラスターのインストール時に、インストーラーは変更した **MachineSet** 定義を使用して RHOSP サーバーグループ内にコンピュータマシンを作成します。

1.5.9. SSH プライベートキーの生成およびエージェントへの追加

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。

**注記**

実稼働環境では、障害復旧およびデバッグが必要です。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの `~/.ssh/authorized_keys` 一覧に追加されます。

**注記**

[AWS キーペア](#) などのプラットフォームに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. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、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 `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。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。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

**注記**

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは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. **ssh-agent** プロセスを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。

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

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

出力例

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

- ❶ ~/.ssh/id_rsa 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。

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。

1.5.10. 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。

1.5.10.1. floating IP アドレスを使っ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.lbFloatingIP**

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があります。

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。

1.5.10.2. 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。

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.lbFloatingIP**

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。

1.5.11. 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を取得します。

手順

1. 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、クラスターのデプロイメントを初期化します。

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

- ❶ **<installation_directory>** については、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。
- ❷ 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。



注記

ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

クラスターのデプロイメントが完了すると、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報を含む、クラスターにアクセスするための指示がターミナルに表示されます。

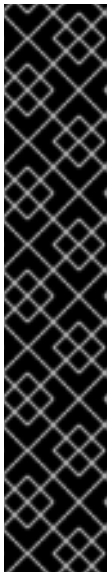
出力例

```
...
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: "4vYBz-Ee6gm-ymBZj-
Wt5AL"
INFO Time elapsed: 36m22s
```



注記

クラスターアクセスおよび認証情報の情報は、インストールが正常に実行される際に **<installation_directory>/.openshift_install.log** に出力されます。



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** についての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お勧めします。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。



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。

1.5.12. 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 ❶
```

- ❶ **<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ついての情報が含まれます。

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ティング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
```

1.5.13. 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ついての情報が含まれます。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。
- **oc** CLI をインストールしていること。

手順

1. **kubeadmin** 認証情報をエクスポートします。

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

- ❶ **<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. 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 へのアクセスおよびその詳細は、[Web コンソールへのアクセス](#) について参照してください。

1.5.14. デフォルトの OperatorHub 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** → **Global 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し、削除し、無効にし、有効にすることができます。

1.5.15. OpenShift Container Platform の Telemetry アクセス

OpenShift Container Platform 4.6 では、クラスターのヘルス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る Telemetry サービスには、インターネットアクセスが必要です。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、[リモートヘルスマニターリング](#) について参照してください。

1.5.16. 次のステップ

- [クラスターをカスタマイズ](#) します。
- 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る CA がある場合、[信頼ストアを設定](#) してこれをクラスターに追加します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 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。
- [ネットワークが制限された環境での Operator Lifecycle Manager \(OLM\) の使用](#) 方法について参照します。
- RHOSP が 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設定しなかった場合には、[RHOSP のアクセスを Floating IP アドレスで設定](#) します。

1.6. 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。

1.6.1. インストーラーでプロビジョニングされるインフラストラクチャーを使用するクラスターの削除

インストーラーでプロビジョニングされるインフラストラクチャーを使用するクラスターは、クラウドから削除できます。



注記

アンインストール後に、とくにユーザーによってプロビジョニングされるインフラストラクチャー (UPI) クラスターで適切に削除されていないリソースがあるかどうかについて、クラウドプロバイダーを確認します。インストーラーが作成されなかったり、インストーラーがアクセスできない場合には、リソースがある可能性があります。

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ります。
- 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。

手順

1. クラスターをインストールするため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 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にある **metadata.json** ファイルが必要になります。

2. オプション: **<installation_directory>** ディレクトリおよび OpenShift Container Platform インストールプログラムを削除します。

1.7. 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

ユーザーによってプロビジョニングされたインフラストラクチャーの 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。

1.7.1. Playbook 依存関係のダウンロード

ユーザーによってプロビジョニングされたインフラストラクチャー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. 必要なリポジトリを追加します。

```
$ sudo subscription-manager repos \
--enable=rhel-8-for-x86_64-baseos-rpms \
--enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
--enable=ansible-2.9-for-rhel-8-x86_64-rpms \
--enable=rhel-8-for-x86_64-appstream-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
```

1.7.2. 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。