# OpenShift Container Platform 4.5

## IBM Power へのインストール

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

Last Updated: 2021-08-01

# OpenShift Container Platform 4.5 IBM Power へのインストール

OpenShift Container Platform IBM Power 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



## 概要

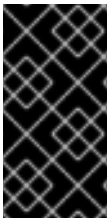
本書では、IBM Power に OpenShift Container Platform クラスターをインストールする方法について説明します。

# 目次

# 第1章 IBM POWER へのインストール

## 1.1. クラスターの IBM POWER へのインストール

OpenShift Container Platform バージョン 4.5 では、プロビジョニングする IBM Power インフラストラクチャーにクラスターをインストールできます。



**重要**

ベアメタルプラットフォーム以外の場合には、追加の考慮点を検討する必要があります。OpenShift Container Platform クラスターをインストールする前に、「guidelines for deploying OpenShift Container Platform on non-tested platforms」にある情報を確認してください。

**前提条件**

- インストールプロセスを開始する前に、既存のインストールファイルを移動するか、または削除する必要があります。これにより、インストールプロセス時に必要なインストールファイルが作成され、更新されます。
- クラスターの NFS を使用した永続ストレージをプロビジョニングします。プライベートイメージレジストリーをデプロイするには、ストレージで **ReadWriteMany** アクセスモードを指定する必要があります。
- 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。
- ファイアウォールを使用する場合、クラスターがアクセスを必要とするサイトを許可するようにファイアウォールを設定する必要があります。

**注記**

プロキシーを設定する場合は、このサイト一覧も確認してください。

### 1.1.1. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

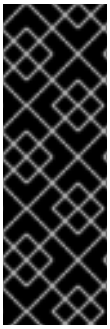
OpenShift Container Platform 4.5 では、クラスターをインストールするためにインターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは Red Hat OpenShift Cluster Manager (OCM) に登録されます。

Red Hat OpenShift Cluster Manager インベントリーが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常である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。

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。

- Red Hat OpenShift Cluster Manager ページにアクセスし、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。

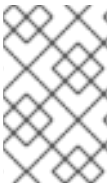
### 1.1.2. ユーザーによってプロビジョニングされるインフラストラクチャーでのクラスターのマシン要件

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含むクラスターの場合、必要なマシンすべてをデプロイする必要があります。

#### 1.1.2.1. 必要なマシン

最小の OpenShift Container Platform クラスターでは以下のホストが必要です。

- 1 つの一時的なブートストラップマシン
- 3 つのコントロールプレーン、またはマスター、マシン
- 少なくとも 2 つのコンピュートマシン (ワーカーマシンとしても知られる)。



**注記**

クラスターでは、ブートストラップマシンが OpenShift Container Platform クラスターを 3 つのコントロールプレーンマシンにデプロイする必要があります。クラスターのインストール後にブートストラップマシンを削除できます。

**重要**

クラスターの高可用性を維持するには、これらのクラスターマシンについて別個の物理ホストを使用します。

ブートストラップおよびコントロールプレーンマシンでは、Red Hat Enterprise Linux CoreOS (RHCOS) をオペレーティングシステムとして使用する必要があります。

RHCOS は Red Hat Enterprise Linux (RHEL) 8 をベースとしており、そのハードウェア認定および要件が継承されることに注意してください。「Red Hat Enterprise Linux テクノロジーの機能と制限 」を参照してください。

#### 1.1.2.2. ネットワーク接続の要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定ファイルをフェッチする必要があります。初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーまたはその静的 IP アドレスが設定されている必要になります。さらに、クラスター内

の各 OpenShift Container Platform ノードは Network Time Protocol (NTP) サーバーにアクセスできる必要があります。DHCP サーバーが NTP サーバー情報を提供する場合、Red Hat Enterprise Linux CoreOS (RHCOS) マシンの chrony タイムサービスは情報を読み取り、NTP サーバーとクロックを同期できます。

#### 1.1.2.3. 最小リソース要件

それぞれのクラスターマシンは、以下の最小要件を満たしている必要があります。

| マシン | オペレーティングシステム | vCPU [1] | 仮想 RAM | ストレージ |
|---|---|---|---|---|
| ブートストラップ | RHCOS | 2 | 16 GB | 120 GB |
| コントロールプレーン | RHCOS | 2 | 16 GB | 120 GB |
| コンピュート | RHCOS | 2 | 8 GB | 120 GB |

1. 1 vCPU は、同時マルチスレッド (SMT) またはハイパースレッディングが有効にされていない場合に 1 つの物理コアと同等です。これが有効にされている場合、以下の数式を使用して対応する比率を計算します: (コアごとのスレッド × コア数) × ソケット数 = vCPU

#### 1.1.2.4. 証明書署名要求の管理

ユーザーがプロビジョニングするインフラストラクチャーを使用する場合、クラスターの自動マシン管理へのアクセスは制限されるため、インストール後にクラスターの証明書署名要求 (CSR) のメカニズムを提供する必要があります。**kube-controller-manager** は kubelet クライアント CSR のみを承認します。**machine-approver** は、kubelet 認証情報を使用して要求される提供証明書の有効性を保証できません。適切なマシンがこの要求を発行したかどうかを確認できないためです。kubelet 提供証明書の要求の有効性を検証し、それらを承認する方法を判別し、実装する必要があります。

### 1.1.3. 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の作成

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する OpenShift Container Platform クラスターをデプロイする前に、基礎となるインフラストラクチャーを作成する必要があります。

**前提条件**

- クラスターでサポートするインフラストラクチャーを作成する前に、「OpenShift Container Platform 4.x のテスト済みインテグレーション」ページを参照してください。

**手順**

1. 各ノードに DHCP を設定するか、または静的 IP アドレスを設定します。
2. 必要なロードバランサーをプロビジョニングします。
3. マシンのポートを設定します。
4. DNS を設定します。

5. ネットワーク接続を確認します。

#### 1.1.3.1. 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のネットワーク要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定をフェッチする必要があります。

初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーまたはその静的 IP アドレスが設定されている必要があります。

クラスターのマシンを長期間管理するために DHCP サーバーを使用することが推奨されています。DHCP サーバーが永続 IP アドレスおよびホスト名をクラスターマシンに提供するように設定されていることを確認します。

クラスターの正常なインストール後に各マスターノードで実行される Kubernetes API サーバーは、クラスターマシンのノード名を解決できる必要があります。API サーバーおよびワーカーノードが異なるゾーンに置かれている場合、デフォルトの DNS 検索ゾーンを、API サーバーでノード名を解決できるように設定することができます。もう 1 つの実行可能な方法として、ノードオブジェクトとすべての DNS 要求の両方において、ホストを完全修飾ドメイン名で常に参照することができます。

マシン間のネットワーク接続を、クラスターのコンポーネントが通信できるように設定する必要があります。すべてのマシンではクラスターの他のすべてのマシンのホスト名を解決できる必要があります。

**表1.1 すべてのマシンに対応するすべてのマシン**

| プロトコル | ポート | 説明 |
|---|---|---|
| ICMP | 該当なし | ネットワーク到達性のテスト |
| TCP | **1936** | メトリクス |
| | **9000**-**9999** | ホストレベルのサービス。 ポート **9100**-**9101** のノードエクスポーター、ポート **9099** の Cluster Version Operator が含まれます。 |
| | **10250**-**10259** | Kubernetes が予約するデフォルトポート |
| | **10256** | openshift-sdn |
| UDP | **4789** | VXLAN および Geneve |
| | **6081** | VXLAN および Geneve |
| | **9000**-**9999** | ポート **9100**-**9101** のノードエクスポーターを含む、ホストレベルのサービス。 |
| TCP/UDP | **30000**-**32767** | Kubernetes ノードポート |

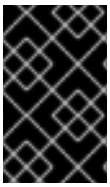
**表1.2 コントロールプレーンへのすべてのマシン**

| プロトコル | ポート | 説明 |
| --- | --- | --- |
| TCP | **6443** | Kubernetes API |

**表1.3 コントロールプレーンマシンへのコントロールプレーンマシン**

| プロトコル | ポート | 説明 |
| --- | --- | --- |
| TCP | **2379-2380** | etcd サーバーおよびピアポート |

**ネットワークトポロジー要件**
クラスター用にプロビジョニングするインフラストラクチャーは、ネットワークトポロジーの以下の要件を満たす必要があります。

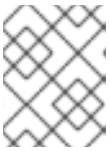


**重要**

OpenShift Container Platform では、すべてのノードが、プラットフォームコンテナーのイメージをプルし、Telemetry データを Red Hat に提供するためにインターネットへの直接のアクセスが必要です。

**ロードバランサー**
OpenShift Container Platform をインストールする前に、以下の要件を満たす 2 つのロードバランサーをプロビジョニングする必要があります。

1. **API ロードバランサー**: プラットフォームと対話およびプラットフォームを設定するためのユーザー向けの共通のエンドポイントを提供します。以下の条件を設定します。
   - Layer 4 の負荷分散のみ。これは、Raw TCP、SSL パススルー、または SSL ブリッジモードと呼ばれます。SSL ブリッジモードを使用する場合は、API ルートの Server Name Indication (SNI) を有効にする必要があります。
   - ステートレス負荷分散アルゴリズム。オプションは、ロードバランサーの実装によって異なります。



**注記**

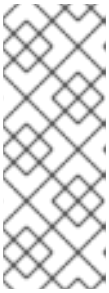
API ロードバランサーが適切に機能するには、セッション永続性は必要ありません。

ロードバランサーのフロントとバックの両方で以下のポートを設定します。

**表1.4 API ロードバランサー**

| ポート | バックエンドマシン (プールメンバー) | 内部 | 外部 | 説明 |
| --- | --- | --- | --- | --- |

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



**注記**

ロードバランサーは、API サーバーが **/readyz** エンドポイントをオフにしてからプールから API サーバーインスタンスを削除するまで最大 30 秒かかるように設定する必要があります。**/readyz** の後の時間枠内でエラーが返されたり、正常になったりする場合は、エンドポイントが削除または追加されているはずです。5 秒または 10 秒ごとにプローブし、2 つの正常な要求が正常な状態になり、3 つの要求が正常な状態になりません。これらは十分にテストされた値です。

2. **Application Ingress ロードバランサー**: クラスター外から送られるアプリケーショントラフィックの Ingress ポイントを提供します。以下の条件を設定します。

   - Layer 4 の負荷分散のみ。これは、Raw TCP、SSL パススルー、または SSL ブリッジモードと呼ばれます。SSL ブリッジモードを使用する場合は、Ingress ルートの Server Name

Indication (SNI) を有効にする必要があります。

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

ロードバランサーのフロントとバックの両方で以下のポートを設定します。

**表1.5 アプリケーション Ingress ロードバランサー**

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

**ヒント**

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

**注記**

Ingress ルーターの作業用の設定が OpenShift Container Platform クラスターに必要です。コントロールプレーンの初期化後に Ingress ルーターを設定する必要があります。

**NTP 設定**
OpenShift Container Platform クラスターは、デフォルトでパブリック Network Time Protocol (NTP) サーバーを使用するように設定されます。ローカルのエンタープライズ NTP サーバーを使用する必要があるか、またはクラスターが切断されたネットワークにデプロイされている場合は、特定のタイムサーバーを使用するようにクラスターを設定できます。詳細は、**chrony タイムサービスの設定** のドキュメントを参照してください。

DHCP サーバーが NTP サーバー情報を提供する場合、Red Hat Enterprise Linux CoreOS (RHCOS) マシンの chrony タイムサービスは情報を読み取り、NTP サーバーとクロックを同期できます。

**追加リソース**

- chrony タイムサービスの設定

### 1.1.3.2. 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の DNS 要件

DNS は、名前解決および逆引き名前解決に使用されます。DNS A/AAAA または CNAME レコードは名前解決に使用され、PTR レコードは逆引き名前解決に使用されます。逆引きレコードは、Red Hat Enterprise Linux CoreOS (RHCOS) は逆引きレコードを使用してすべてのノードのホスト名を設定するために重要です。さらに、逆引きレコードは、OpenShift Container Platform が動作するために必要な証明書署名要求 (CSR) を生成するために使用されます。

以下の DNS レコードは、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する OpenShift Container Platform クラスターに必要です。各レコードで、 **<cluster_name>** はクラスター名で、**<base_domain>** は、**install-config.yaml** ファイルに指定するクラスターのベースドメインです。完全な DNS レコードは **<component>.<cluster_name>.<base_domain>.** の形式を取ります。

**表1.6 必要な DNS レコード**

| コンポーネント | レコード | 説明 |
|---|---|---|
| Kubernetes API | **api.<cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコード、および DNS PTR レコードを、コントロールプレーンマシンのロードバランサーを特定するために追加します。これらのレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。 |
| | **api-int.<cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコード、および DNS PTR レコードを、コントロールプレーンマシンのロードバランサーを特定するために追加します。これらのレコードは、クラスター内のすべてのノードで解決できる必要があります。<br><br>  <br>**重要**<br><br>API サーバーは、Kubernetes に記録されるホスト名でワーカーノードを解決できる必要があります。API サーバーがノード名を解決できない場合、プロキシーされる API 呼び出しが失敗し、Pod からログを取得できなくなる可能性があります。 |

| コンポーネント | レコード | 説明 |
| --- | --- | --- |
| ルート | ***.apps.<cluster_name>.<br><base_domain>.** | デフォルトでワーカーノードの Ingress ルーター Pod を実行するマシンをターゲットにするロードバランサーを参照するワイルドカード DNS A/AAAA または CNAME レコードを追加します。これらのレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。 |
| ブートストラップ | **bootstrap.<cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコードおよび DNS PTR レコードを、ブートストラップマシンを特定するために追加します。これらのレコードは、クラスター内のノードで解決できる必要があります。 |
| マスターホスト | **<master><n>.<br><cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコードおよび DNS PTR レコードを、マスターノードの各マシンを特定するために追加します。これらのレコードは、クラスター内のノードで解決できる必要があります。 |
| ワーカーホスト | **<worker><n>.<br><cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコードおよび DNS PTR レコードを、ワーカーノードの各マシンを特定するために追加します。これらのレコードは、クラスター内のノードで解決できる必要があります。 |

### ヒント

**nslookup <hostname>** コマンドを使用して、名前解決を確認することができます。**dig -x <ip_address>** コマンドを使用して、PTR レコードの逆引き名前解決を確認できます。

BIND ゾーンファイルの以下の例は、名前解決の A レコードの例を示しています。この例の目的は、必要なレコードを表示することです。この例では、特定の名前解決サービスを選択するためのアドバイスを提供することを目的としていません。

**例1.1 DNS ゾーンデータベースのサンプル**

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
```

```
    30M  ; retry (30 minutes)
    2W  ; expiry (2 weeks)
    1W )  ; minimum (1 week)
  IN NS ns1.example.com.
  IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

以下の BIND ゾーンファイルの例では、逆引き名前解決の PTR レコードの例を示しています。

**例1.2 逆引きレコードの DNS ゾーンデータベースの例**

```
$TTL 1W
@ IN SOA ns1.example.com. root (
    2019070700 ; serial
    3H  ; refresh (3 hours)
    30M  ; retry (30 minutes)
    2W  ; expiry (2 weeks)
    1W )  ; minimum (1 week)
  IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
```

```
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

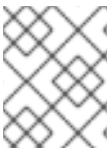
### 1.1.4. SSH プライベートキーの生成およびエージェントへの追加

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
```

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。

1. 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。

### 1.1.5. インストールプログラムの取得

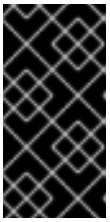
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

**前提条件**

- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、500 MB のローカルディスク領域が必要です。

**手順**

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」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. 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。クラスターインストールの完了後は、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の両方を保持する必要があります。

**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。特定のクラウドプロバイダー用に記載された OpenShift Container Platform のアンインストール手順を完了して、クラスターを完全に削除する必要があります。

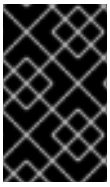
3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar xvf <installation_program>.tar.gz
```

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

### 1.1.6. バイナリーのダウンロードによる CLI のインストール

コマンドラインインターフェースを使用して OpenShift Container Platform と対話するために CLI (**oc**) をインストールすることができます。**oc** は Linux、Windows、または macOS にインストールできます。



**重要**

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.5 のすべてのコマンドを実行することはできません。新規バージョンの **oc** をダウンロードし、インストールします。

#### 1.1.6.1. Linux への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Linux にインストールできます。

**手順**

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページに移動します。
2. インフラストラクチャープロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Linux** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開します。

```
$ tar xvzf <file>
```

5. **oc** バイナリーを、**PATH** にあるディレクトリーに配置します。
**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

#### 1.1.6.2. Windows での CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Windows にインストールできます。

**手順**

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページに移動します。
2. インフラストラクチャープロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Windows** を選択し、**Download command-line tools** をクリックします。
4. ZIP プログラムでアーカイブを解凍します。
5. **oc** バイナリーを、**PATH** にあるディレクトリーに移動します。
**PATH** を確認するには、コマンドプロンプトを開いて以下のコマンドを実行します。

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
C:\> oc <command>
```

#### 1.1.6.3. macOS への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを macOS にインストールできます。

**手順**

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページに移動します。
2. インフラストラクチャープロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **MacOS** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開し、解凍します。
5. **oc** バイナリーをパスにあるディレクトリーに移動します。
**PATH** を確認するには、ターミナルを開き、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

### 1.1.7. インストール設定ファイルの手動作成

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する OpenShift Container Platform のインストールでは、インストール設定ファイルを手動で生成します。

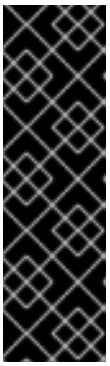
**前提条件**

- OpenShift Container Platform インストーラープログラムおよびクラスターのアクセストークンを取得します。

**手順**

1. 必要なインストールアセットを保存するためのインストールディレクトリーを作成します。

```
$ mkdir <installation_directory>
```



**重要**

ディレクトリーを作成する必要があります。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。

2. 以下の **install-config.yaml** ファイルテンプレートをカスタマイズし、これを **<installation_directory>** に保存します。

**注記**

この設定ファイル **install-config.yaml** に名前を付ける必要があります。

3. **install-config.yaml** ファイルをバックアップし、これを複数のクラスターをインストールするため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の次の手順で使用されます。この時点でこれをバックアップする必要があります。

#### 1.1.7.1. IBM Power のサンプル install-config.yaml ファイル

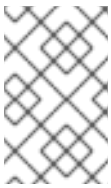
**install-config.yaml** ファイルをカスタマイズして、OpenShift Container Platform クラスターのプラットフォームについての詳細を指定するか、または必要なパラメーターの値を変更することができます。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OpenShiftSDN
  serviceNetwork: (11)
  - 172.30.0.0/16
platform:
  none: {} (12)
fips: false (13)
pullSecret: '{"auths": ...}' (14)
sshKey: 'ssh-ed25519 AAAA...' (15)
```

(1) クラスターのベースドメイン。すべての DNS レコードはこのベースのサブドメインである必要があり、クラスター名が含まれる必要があります。

(2) (5) **controlPlane** セクションは単一マッピングですが、**compute** セクションはマッピングのシーケンスになります。複数の異なるデータ構造の要件を満たすには、 **compute** セクションの最初の行はハイフン **-** で始め、**controlPlane** セクションの最初の行はハイフンで始めることができません。どちらのセクションも、現時点では単一のマシンプールを定義しますが、OpenShift Container Platform の今後のバージョンでは、インストール時の複数のコンピュートプールの定義をサポートする可能性があります。1 つのコントロールプレーンプールのみが使用されます。

(3) (6) 同時マルチスレッド (SMT) または **hyperthreading** を有効/無効にするかどうか。デフォルトでは、SMT 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。パラメーター値を **Disabled** に設定するとこれを無効にすることができます。SMT を無効にする場合、これをすべてのクラスターマシンで無効にする必要があります。これにはコントロールプレーンとコンピュートマシンの両方が含まれます。



**注記**

同時マルチスレッド (SMT) はデフォルトで有効になっています。SMT が BIOS 設定で有効になっていない場合は、**hyperthreading** パラメーターは効果がありません。

**重要**

BIOS または **install-config.yaml** であるかに関係なく **hyperthreading** を無効にする場合、容量計画においてマシンのパフォーマンスの大幅な低下が考慮に入れられていることを確認します。

4 **replicas** パラメーターの値を **0** に設定する必要があります。このパラメーターはクラスターが作成し、管理するワーカーの数を制御します。これは、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する場合にクラスターが実行しない機能です。OpenShift Container Platform のインストールが終了する前に、クラスターが使用するワーカーマシンを手動でデプロイする必要があります。

7 クラスターに追加するコントロールプレーンマシンの数。クラスターをこの値をクラスターの etcd エンドポイント数として使用するため、値はデプロイするコントロールプレーンマシンの数に一致する必要があります。

8 DNS レコードに指定したクラスター名。

9 Pod IP アドレスの割り当てに使用する IP アドレスのブロック。このブロックは既存の物理ネットワークと重複できません。これらの IP アドレスは Pod ネットワークに使用されます。外部ネットワークから Pod にアクセスする必要がある場合、ロードバランサーおよびルーターを、トラフィックを管理するように設定する必要があります。

10 それぞれの個別ノードに割り当てるサブネットプレフィックスの長さ。たとえば、**hostPrefix** が **23** に設定され、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 (510 (2^(32 - 23) - 2) Pod IP アドレスが許可されます)。外部ネットワークからのノードへのアクセスを提供する必要がある場合には、ロードバランサーおよびルーターを、トラフィックを管理するように設定します。

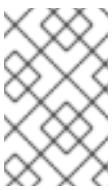
11 サービス IP アドレスに使用する IP アドレスプール。1 つの IP アドレスプールのみを入力できます。外部ネットワークからサービスにアクセスする必要がある場合、ロードバランサーおよびルーターを、トラフィックを管理するように設定します。

12 プラットフォームを **none**に設定する必要があります。IBM Power インフラストラクチャー用に追加のプラットフォーム設定変数を指定することはできません。

13 FIPS モードを有効または無効にするかどうか。デフォルトでは、FIPS モードは有効にされません。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。

14 Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレット。このプルシークレットを使用すると、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する Quay.io など、組み込まれた各種の認証局によって提供されるサービスを使用して認証できます。

15 Red Hat Enterprise Linux CoreOS (RHCOS) の **core** ユーザーのデフォルト SSH キーの公開部分。



**注記**

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

### 1.1.8. 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。

**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。

**前提条件**

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

**手順**

1. クラスターの Kubernetes マニフェストを生成します。

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

**出力例**

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
```

1 **<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。

インストールプロセスの後の部分で独自のコンピュートマシンを作成するため、この警告を無視しても問題がありません。

2. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes マニフェストファイルを変更し、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いようにします。
   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、その値を **False** に設定します。
   c. ファイルを保存し、終了します。
3. Ignition 設定ファイルを取得します。

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

```
-
```

1. **<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
```

### 1.1.9. Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされる IBM Power インフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。ISO イメージまたはネットワーク PXE ブートを使用する手順を実行してマシンを作成することができます。

#### 1.1.9.1. ISO イメージを使用した Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

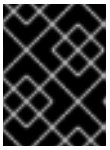
ユーザーによってプロビジョニングされる IBM Power インフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。ISO イメージを使用してマシンを作成することができます。

**前提条件**

- クラスターの Ignition 設定ファイルを取得していること。
- お使いのコンピューターからアクセスでき、作成するマシンがアクセスできる HTTP サーバーへのアクセスがあること。

**手順**

1. インストールプログラムが作成したコントロールプレーン、コンピュート、およびブートストラップ Ignition 設定を HTTP サーバーにアップロードします。これらのファイルの URL をメモします。



**重要**

インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

2. 「RHCOS イメージミラー」ページからオペレーティングシステムのインスタンスをインストールするために優先される方法で必要な RHCOS イメージを取得します。

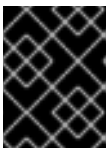
**重要**

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。この手順には ISO イメージのみを使用します。RHCOS qcow2 イメージは、ベアメタルのインストールではサポートされません。

ISO ファイルおよび RAW ディスクファイルをダウンロードする必要があります。これらのファイルの名前は以下の例のようになります。

- ISO: **rhcos-<version>-installer.<architecture>.iso**
- 圧縮された metal RAW: **rhcos-<version>-metal.<architecture>.raw.gz**

3. RAW RHCOS イメージファイルのいずれかを HTTP サーバーにアップロードし、その URL をメモします。



**重要**

インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

4. ISO を使用し、RHCOS インストールを開始します。以下のインストールオプションのいずれかを使用します。
   - ディスクに ISO イメージを書き込み、これを直接起動します。
   - LOM インターフェースで ISO リダイレクトを使用します。
5. インスタンスの起動後に、**TAB** または **E** キーを押してカーネルコマンドラインを編集します。
6. パラメーターをカーネルコマンドラインに追加します。

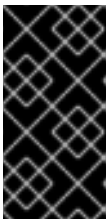
```
coreos.inst=yes
coreos.inst.install_dev=sda 1
coreos.inst.image_url=<image_URL> 2
coreos.inst.ignition_url=http://example.com/config.ign 3
ip=<dhcp or static IP address> 4 5
bond=<bonded_interface> 6
```

1. インストール先のシステムのブロックデバイスを指定します。
2. サーバーにアップロードした RAW イメージの URL を指定します。
3. このマシンタイプの Ignition 設定ファイルの URL を指定します。
4. **ip=dhcp** を設定するか、各ノードに個別の静的 IP アドレス（**ip=**）および DNS サーバー（**nameserver=**）を設定します。詳細は、「**高度なネットワークの設定**」を参照してください。
5. 複数のネットワークインターフェースまたは DNS サーバーを使用する場合は、「**高度なネットワークの設定**」を参照してください。

6 オプションで、「**高度なネットワークの設定**」で説明されているように、**bond=** オプションを使用して、複数のネットワークインターフェースを単一のインターフェースにボン

7. Enter を押してインストールを完了します。RHCOS のインストール後に、システムは再起動します。システムの再起動後、指定した Ignition 設定ファイルを適用します。

8. 継続してクラスターのマシンを作成します。



**重要**

この時点でブートストラップおよびコントロールプレーンマシンを作成する必要があります。コントロールプレーンマシンがデフォルトのスケジュール対象にされていない場合、クラスターのインストール前に少なくとも 2 つのコンピュートマシンを作成します。

#### 1.1.9.2. PXE ブートによる Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされる IBM Power インフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。PXE ブートを使用してマシンを作成することができます。

**前提条件**

- クラスターの Ignition 設定ファイルを取得していること。
- 使用しているコンピューターからアクセス可能な HTTP サーバーおよび TFTP サーバーへのアクセスがあること。

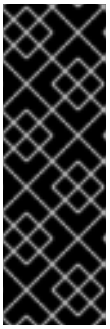
**手順**

1. インストールプログラムが作成したマスター、ワーカーおよびブートストラップの Ignition 設定を HTTP サーバーにアップロードします。これらのファイルの URL をメモします。

**重要**

インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

2. Red Hat カスタマーポータルの「製品のダウンロード」ページまたは 「RHCOS イメージミラー」ページから圧縮された metal RAW **イメージ、kernel** および **initramfs** ファイルを取得します。



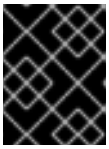
**重要**

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。この手順には RAW イメージのみを使用します。RHCOS qcow2 イメージは、ベアメタルのインストールではサポートされません。

ファイル名には、OpenShift Container Platform のバージョン名が含まれます。以下の例のようになります。

- 圧縮されたメタル RAW イメージ: **rhcos-<version>-<architecture>-metal.<architecture>.raw.gz**
- **kernel**: **rhcos-<version>-<architecture>-installer-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-<architecture>-installer-initramfs.<architecture>.img**

3. RAW イメージを HTTP サーバーにアップロードします。
4. 使用する起動方法に必要な追加ファイルをアップロードします。
   - 従来の PXE の場合、**kernel** および **initramfs** ファイルを TFTP サーバーにアップロードします。
   - iPXE の場合、**kernel** および **initramfs** ファイルを HTTP サーバーにアップロードします。



**重要**

インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

5. RHCOS のインストール後にマシンがローカルディスクから起動されるようにネットワークブートインフラストラクチャーを設定します。
6. RHCOS イメージに PXE インストールを設定します。
ご使用の環境についての以下の例で示されるメニューエントリーのいずれかを変更し、イメージおよび Ignition ファイルが適切にアクセスできることを確認します。
   - PXE の場合:

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL rhcos-<version>-<architecture>-installer-kernel-<architecture> (1)
    APPEND ip=dhcp rd.neednet=1 initrd=rhcos-<version>-<architecture>-installer-initramfs.<architecture>.img coreos.inst=yes coreos.inst.install_dev=sda coreos.inst.image_url=http://<HTTP_server>/rhcos-<version>-<architecture>-metal.<architecture>.raw.gz coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) TFTP サーバーで利用可能な **kernel** ファイルの場所を指定します。

(2) 複数の NIC を使用する場合、**ip** オプションに単一インターフェースを指定します。たとえば、**eno1** という名前の NIC で DHCP を使用するには、 **ip=eno1:dhcp** を設定します。

(3) HTTP または TFTP サーバーにアップロードした RHCOS ファイルの場所を指定します。**initrd** パラメーター値は、TFTP サーバーの **initramfs** ファイルの場所です。**coreos.inst.image_url** パラメーター値は、HTTP サーバーの圧縮されたメタル RAW イメージの場所であり、 **coreos.inst.ignition_url** パラメーター値は HTTP サーバーのブートストラップ Ignition 設定ファイルの場所になります。

**注記**

この設定では、グラフィカルコンソールを使用するマシンでシリアルコンソールアクセスを有効にしません。別のコンソールを設定するには、**APPEND** 行に 1 つ以上の **console=** 引数を追加します。たとえば、**console=tty0 console=ttyS0** を追加して、最初の PC シリアルポートをプライマリーコンソールとして、グラフィカルコンソールをセカンダリーコンソールとして設定します。詳細は、「How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?」を参照してください。

7. UEFI を使用する場合は、以下の操作を実行します。

   a. システムの起動に必要な EFI バイナリーおよび **grub.cfg** ファイルを提供します。**shim.efi** バイナリーと **grubx64.efi** バイナリーが必要です。

      - RHCOS ISO をホストにマウントし、**images/efiboot.img** ファイルをホストにマウントして、必要な EFI バイナリーを展開します。**efiboot.img** マウントポイントから、**EFI/redhat/shimx64.efi** および **EFI/redhat/grubx64.efi** ファイルを TFTP サーバーにコピーします。

        ```
        # mkdir -p /mnt/{iso,efiboot}
        # mount -o loop rhcos-installer.x86_64.iso /mnt/iso
        # mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
        # cp /mnt/efiboot/EFI/redhat/{shimx64.efi,grubx64.efi} .
        # umount /mnt/{efiboot,iso}
        ```

   b. RHCOS ISO に含まれている **EFI/redhat/grub.cfg** ファイルを TFTP サーバーにコピーします。

   c. **grub.cfg** ファイルを編集し、以下の引数を追加します。

      ```
      menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
       linux rhcos-<version>-<architecture>-installer-kernel-<architecture> nomodeset rd.neednet=1 coreos.inst=yes coreos.inst.install_dev=sda coreos.inst.image_url=http://<HTTP_server>/rhcos-<version>-<architecture>-metal.<architecture>.raw.gz coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1
       initrd rhcos-<version>-<architecture>-installer-initramfs.<architecture>.img 2
      }
      ```

      1 **linux** 行の項目の最初の引数は、TFTP サーバーにアップロードした **kernel** ファイルの場所です。**coreos.inst.image_url** パラメーター値には、HTTP サーバーにアップロードした圧縮されたメタル RAW イメージの場所を指定します。**coreos.inst.ignition_url** パラメーターには、HTTP サーバーにアップロードしたブートストラップ Ingition 設定ファイルの場所を指定します。

      2 TFTP サーバーにアップロードした **initramfs** ファイルの場所を指定します。

8. 継続してクラスターのマシンを作成します。

**重要**

この時点でブートストラップおよびコントロールプレーンマシンを作成する必要があります。コントロールプレーンマシンがデフォルトのスケジュール対象にされていない場合、クラスターのインストール前に少なくとも 2 つのコンピュートマシンを作成します。

### 1.1.10. クラスターの作成

OpenShift Container Platform クラスターを作成するには、ブートストラッププロセスが、インストールプログラムで生成した Ignition 設定ファイルを使用してプロビジョニングしたマシンで完了するのを待機します。

**前提条件**

- クラスターに必要なインフラストラクチャーを作成する。
- インストールプログラムを取得し、クラスターの Ignition 設定ファイルを生成している。
- Ignition 設定ファイルを使用して、クラスターの RHCOS マシンを作成済している。
- お使いのマシンでインターネットに直接アクセスできるか、または HTTP または HTTPS プロキシーが利用できる。

**手順**

1. ブートストラッププロセスをモニターします。

```
$ ./openshift-install --dir=<installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 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。

**出力例**

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.18.3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API サーバーでこれがコントロールプレーンマシンにブートストラップされていることを示すシグナルが出されるとコマンドは成功します。

2. ブートストラッププロセスが完了したら、ブートストラップマシンをロードバランサーから削除します。

**重要**

この時点で、ブートストラップマシンをロードバランサーから削除する必要があります。さらに、マシン自体を削除し、再フォーマットすることができます。

### 1.1.11. 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ます。
- **oc** CLI をインストールします。

**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
```

### 1.1.12. 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  v1.18.3
master-1  Ready     master  63m  v1.18.3
master-2  Ready     master  64m  v1.18.3
worker-0  NotReady  worker  76s  v1.18.3
worker-1  NotReady  worker  70s  v1.18.3
```

出力には作成したすべてのマシンが一覧表示されます。

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。

- 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 --no-run-if-empty oc adm certificate approve
```

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。

**追加情報**

- CSR の詳細は、「Certificate Signing Requests」を参照してください。

### 1.1.13. Operator の初期設定

コントロールプレーンの初期化後に、一部の Operator を利用可能にするためにそれらをすぐに設定する必要があります。

**前提条件**

- コントロールプレーンが初期化されています。

**手順**

1. クラスターコンポーネントがオンラインになることを確認します。

```
$ watch -n5 oc get clusteroperators
```

**出力例**

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.5.4     True        False         False      69s
cloud-credential                           4.5.4     True        False         False      12m
cluster-autoscaler                         4.5.4     True        False         False      11m
console                                    4.5.4     True        False         False      46s
dns                                        4.5.4     True        False         False      11m
image-registry                             4.5.4     True        False         False      5m26s
ingress                                    4.5.4     True        False         False      5m36s
kube-apiserver                             4.5.4     True        False         False      8m53s
kube-controller-manager                    4.5.4     True        False         False      7m24s
kube-scheduler                             4.5.4     True        False         False      12m
machine-api                                4.5.4     True        False         False      12m
machine-config                             4.5.4     True        False         False      7m36s
marketplace                                4.5.4     True        False         False      7m54m
monitoring                                 4.5.4     True        False         False      7h54s
network                                    4.5.4     True        False         False      5m9s
node-tuning                                4.5.4     True        False         False      11m
openshift-apiserver                        4.5.4     True        False         False      11m
openshift-controller-manager               4.5.4     True        False         False      5m943s
openshift-samples                          4.5.4     True        False         False      3m55s
operator-lifecycle-manager                 4.5.4     True        False         False      11m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      11m
service-ca                                 4.5.4     True        False         False      11m
service-catalog-apiserver                  4.5.4     True        False         False      5m26s
service-catalog-controller-manager         4.5.4     True        False         False      5m25s
storage                                    4.5.4     True        False         False      5m30s
```

2. 利用不可の Operator を設定します。

#### 1.1.13.1. イメージレジストリーストレージの設定

イメージレジストリー Operator は、デフォルトストレージを提供しないプラットフォームでは最初は利用できません。インストール後に、レジストリー Operator を使用できるようにレジストリーをストレージを使用するように設定する必要があります。

実稼働クラスターに必要な永続ボリュームの設定についての手順が示されます。該当する場合、空のディレクトリーをストレージの場所として設定する方法が表示されます。これは、実稼働以外のクラスターでのみ利用できます。

アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

#### 1.1.13.1.1. ベアメタルの場合のレジストリーストレージの設定

クラスター管理者は、インストール後にレジストリーをストレージを使用できるように設定する必要があります。

**前提条件**

- クラスター管理者のパーミッション。
- ベアメタル上のクラスター。
- Red Hat OpenShift Container Storage などのクラスターのプロビジョニングされた永続ストレージ。

**重要**

OpenShift Container Platform は、1 つのレプリカのみが存在する場合にイメージレジストリーストレージの **ReadWriteOnce** アクセスをサポートします。2 つ以上のレプリカで高可用性をサポートするイメージレジストリーをデプロイするには、**ReadWriteMany** アクセスが必要です。

- 100Gi の容量が必要です。

**手順**

1. レジストリーをストレージを使用できるように設定するには、**configs.imageregistry/cluster** リソースの **spec.storage.pvc** を変更します。

**注記**

共有ストレージを使用する場合は、外部からアクセスを防ぐためにセキュリティー設定を確認します。

2. レジストリー Pod がないことを確認します。

```
$ oc get pod -n openshift-image-registry
```

**注記**

ストレージタイプが **emptyDIR** の場合、レプリカ数が **1** を超えることはありません。

3. レジストリー設定を確認します。

```
$ oc edit configs.imageregistry.operator.openshift.io
```

**出力例**

```
storage:
  pvc:
    claim:
```

**claim** フィールドを空のままにし、**image-registry-storage** PVC の自動作成を可能にします。

4. **clusteroperator** ステータスを確認します。

```
$ oc get clusteroperator image-registry
```

#### 1.1.13.1.2. 実稼働以外のクラスターでのイメージレジストリーのストレージの設定

イメージレジストリー Operator のストレージを設定する必要があります。実稼働用以外のクラスターの場合、イメージレジストリーは空のディレクトリーに設定することができます。これを実行する場合、レジストリーを再起動するとすべてのイメージが失われます。

**手順**

1. イメージレジストリーストレージを空のディレクトリーに設定するには、以下を実行します。

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

**警告**

実稼働用以外のクラスターにのみこのオプションを設定します。

イメージレジストリー Operator がそのコンポーネントを初期化する前にこのコマンドを実行する場合、**oc patch** コマンドは以下のエラーを出して失敗します。

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

数分待機した後に、このコマンドを再び実行します。

2. イメージのビルドおよびプッシュを有効にするためにレジストリーが managed に設定されていることを確認します。

- 以下を実行します。

```
$ oc edit configs.imageregistry/cluster
```

次に、行を変更します。

```
managementState: Removed
```

次のように変更してください。

```
managementState: Managed
```

### 1.1.14. 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でのインストールの完了

Operator 設定の完了後に、提供するインフラストラクチャーでのクラスターのインストールを終了できます。

**前提条件**

- コントロールプレーンが初期化されています。
- Operator の初期設定を完了済みです。

**手順**

1. 以下のコマンドを使用して、すべてのクラスターコンポーネントがオンラインであることを確認します。

```
$ watch -n5 oc get clusteroperators
```

**出力例**

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.5.4     True        False         False      7m56s
cloud-credential                           4.5.4     True        False         False      31m
cluster-autoscaler                         4.5.4     True        False         False      16m
console                                    4.5.4     True        False         False      10m
csi-snapshot-controller                    4.5.4     True        False         False      16m
dns                                        4.5.4     True        False         False      22m
etcd                                       4.5.4     False       False         False      25s
image-registry                             4.5.4     True        False         False      16m
ingress                                    4.5.4     True        False         False      16m
insights                                   4.5.4     True        False         False      17m
kube-apiserver                             4.5.4     True        False         False      19m
kube-controller-manager                    4.5.4     True        False         False      20m
kube-scheduler                             4.5.4     True        False         False      20m
kube-storage-version-migrator              4.5.4     True        False         False      16m
machine-api                                4.5.4     True        False         False      22m
machine-config                             4.5.4     True        False         False      22m
marketplace                                4.5.4     True        False         False      16m
monitoring                                 4.5.4     True        False         False      10m
network                                    4.5.4     True        False         False      23m
node-tuning                                4.5.4     True        False         False      23m
openshift-apiserver                        4.5.4     True        False         False      17m
openshift-controller-manager               4.5.4     True        False         False      15m
openshift-samples                          4.5.4     True        False         False      16m
operator-lifecycle-manager                 4.5.4     True        False         False      22m
```

```
operator-lifecycle-manager-catalog        4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver  4.5.4     True        False         False      18m
service-ca                                4.5.4     True        False         False      23m
service-catalog-apiserver                 4.5.4     True        False         False      23m
service-catalog-controller-manager        4.5.4     True        False         False      23m
storage                                   4.5.4     True        False         False      17m
```

あるいは、以下のコマンドを使用すると、すべてのクラスターが利用可能な場合に通知されます。また、このコマンドは認証情報を取得して表示します。

```
$ ./openshift-install --dir=<installation_directory> wait-for install-complete 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。

**出力例**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator が Kubernetes API サーバーから OpenShift Container Platform クラスターのデプロイを終了するとコマンドは成功します。

**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。

2. Kubernetes API サーバーが Pod と通信していることを確認します。
   a. すべての Pod の一覧を表示するには、以下のコマンドを使用します。

```
$ oc get pods --all-namespaces
```

**出力例**

```
NAMESPACE                        NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator     openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running    1          9m
openshift-apiserver              apiserver-67b9g                               1/1     Running     0
3m
openshift-apiserver              apiserver-ljcmx                               1/1     Running     0
1m
openshift-apiserver              apiserver-z25h4                               1/1     Running     0
2m
```

```
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running     0          5m
...
```

b. 以下のコマンドを使用して、直前のコマンドの出力に一覧表示される Pod のログを表示します。

```
$ oc logs <pod_name> -n <namespace> 1
```

1 直前のコマンドの出力にあるように、Pod 名および namespace を指定します。

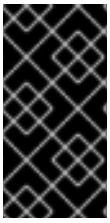
Pod のログが表示される場合、Kubernetes API サーバーはクラスターマシンと通信できます。

**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。

## 1.2. ネットワークが制限された環境での IBM POWER へのクラスターのインストール

OpenShift Container Platform バージョン 4.5 では、クラスターをネットワークが制限された環境でプロビジョニングする IBM Power インフラストラクチャーにインストールできます。



**重要**

ベアメタルプラットフォーム以外の場合には、追加の考慮点を検討する必要があります。OpenShift Container Platform クラスターをインストールする前に、「guidelines for deploying OpenShift Container Platform on non-tested platforms」にある情報を確認してください。

**前提条件**

- ネットワークが制限された環境でインストールのミラーレジストリーを作成し、お使いの OpenShift Container Platform **のバージョンの imageContentSources** データを取得します。**docker.io/ibmcom/registry-ppc64le:2.6.2.5** イメージを使用します。
- インストールプロセスを開始する前に、既存のインストールファイルを移動するか、または削除する必要があります。これにより、インストールプロセス時に必要なインストールファイルが作成され、更新されます。

**重要**

インストールメディアにアクセスできるマシンからインストール手順が実行されるようにします。

- クラスターの永続ストレージをプロビジョニングします。プライベートイメージレジストリーをデプロイするには、ストレージで **ReadWriteMany** アクセスモードを指定する必要があります。

- 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。
- ファイアウォールを使用し、Telemetry を使用する予定がある場合は、クラスターがアクセスする必要のあるサイトを許可するようにファイアウォールを設定する必要があります。

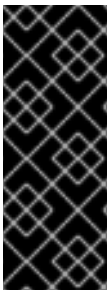
**注記**

プロキシーを設定する場合は、このサイト一覧も確認してください。

### 1.2.1. 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 4.5 では、ソフトウェアコンポーネントを取得するためにインターネットへのアクティブな接続を必要としないインストールを実行できます。ネットワークが制限された環境のインストールは、クラスターのインストール先となるクラウドプラットフォームに応じて、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。

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。



**重要**

ユーザーによってプロビジョニングされるインストールの設定は複雑であるため、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ネットワークが制限されたインストールを試行する前に、標準的な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実行することを検討してください。このテストが完了すると、ネットワークが制限されたインストール時に発生する可能性のある問題の切り分けやトラブルシューティングがより容易になります。

#### 1.2.1.1. 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。

### 1.2.2. 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するクラスターのマシン要件

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含むクラスターの場合、必要なマシンすべてをデプロイする必要があります。

#### 1.2.2.1. 必要なマシン

最小の OpenShift Container Platform クラスターでは以下のホストが必要です。

- 1 つの一時的なブートストラップマシン
- 3 つのコントロールプレーン、またはマスター、マシン

- 少なくとも 2 つのコンピュートマシン (ワーカーマシンとしても知られる)。

**注記**

クラスターでは、ブートストラップマシンが OpenShift Container Platform クラスターを 3 つのコントロールプレーンマシンにデプロイする必要があります。クラスターのインストール後にブートストラップマシンを削除できます。

**重要**

クラスターの高可用性を維持するには、これらのクラスターマシンについて別個の物理ホストを使用します。

ブートストラップおよびコントロールプレーンマシンでは、Red Hat Enterprise Linux CoreOS (RHCOS) をオペレーティングシステムとして使用する必要があります。

RHCOS は Red Hat Enterprise Linux (RHEL) 8 をベースとしており、そのハードウェア認定および要件が継承されることに注意してください。「Red Hat Enterprise Linux テクノロジーの機能と制限 」を参照してください。

#### 1.2.2.2. ネットワーク接続の要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定ファイルをフェッチする必要があります。初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーまたはその静的 IP アドレスが設定されている必要になります。さらに、クラスター内の各 OpenShift Container Platform ノードは Network Time Protocol (NTP) サーバーにアクセスできる必要があります。DHCP サーバーが NTP サーバー情報を提供する場合、Red Hat Enterprise Linux CoreOS (RHCOS) マシンの chrony タイムサービスは情報を読み取り、NTP サーバーとクロックを同期できます。

#### 1.2.2.3. 最小リソース要件

それぞれのクラスターマシンは、以下の最小要件を満たしている必要があります。

| マシン | オペレーティングシステム | vCPU [1] | 仮想 RAM | ストレージ |
|---|---|---|---|---|
| ブートストラップ | RHCOS | 2 | 16 GB | 120 GB |
| コントロールプレーン | RHCOS | 2 | 16 GB | 120 GB |
| コンピュート | RHCOS | 2 | 8 GB | 120 GB |

1. 1 vCPU は、同時マルチスレッド (SMT) またはハイパースレッディングが有効にされていない場合に 1 つの物理コアと同等です。これが有効にされている場合、以下の数式を使用して対応する比率を計算します: (コアごとのスレッド × コア数) × ソケット数 = vCPU

#### 1.2.2.4. 証明書署名要求の管理

ユーザーがプロビジョニングするインフラストラクチャーを使用する場合、クラスターの自動マシン管理へのアクセスは制限されるため、インストール後にクラスターの証明書署名要求 (CSR) のメカニズムを提供する必要があります。**kube-controller-manager** は kubelet クライアント CSR のみを承認します。**machine-approver** は、kubelet 認証情報を使用して要求される提供証明書の有効性を保証できません。適切なマシンがこの要求を発行したかどうかを確認できないためです。kubelet 提供証明書の要求の有効性を検証し、それらを承認する方法を判別し、実装する必要があります。

### 1.2.3. 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の作成

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する OpenShift Container Platform クラスターをデプロイする前に、基礎となるインフラストラクチャーを作成する必要があります。

**前提条件**

- クラスターでサポートするインフラストラクチャーを作成する前に、「OpenShift Container Platform 4.x のテスト済みインテグレーション」ページを参照してください。

**手順**

1. 各ノードに DHCP を設定するか、または静的 IP アドレスを設定します。
2. 必要なロードバランサーをプロビジョニングします。
3. マシンのポートを設定します。
4. DNS を設定します。
5. ネットワーク接続を確認します。

#### 1.2.3.1. 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のネットワーク要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定をフェッチする必要があります。

初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーまたはその静的 IP アドレスが設定されている必要があります。

クラスターのマシンを長期間管理するために DHCP サーバーを使用することが推奨されています。DHCP サーバーが永続 IP アドレスおよびホスト名をクラスターマシンに提供するように設定されていることを確認します。

クラスターの正常なインストール後に各マスターノードで実行される Kubernetes API サーバーは、クラスターマシンのノード名を解決できる必要があります。API サーバーおよびワーカーノードが異なるゾーンに置かれている場合、デフォルトの DNS 検索ゾーンを、API サーバーでノード名を解決できるように設定することができます。もう 1 つの実行可能な方法として、ノードオブジェクトとすべての DNS 要求の両方において、ホストを完全修飾ドメイン名で常に参照することができます。

マシン間のネットワーク接続を、クラスターのコンポーネントが通信できるように設定する必要があります。すべてのマシンではクラスターの他のすべてのマシンのホスト名を解決できる必要があります。

**表1.7 すべてのマシンに対応するすべてのマシン**

| プロトコル | ポート | 説明 |
| --- | --- | --- |
| ICMP | 該当なし | ネットワーク到達性のテスト |
| TCP | **1936** | メトリクス |
| | **9000**-**9999** | ホストレベルのサービス。 ポート **9100**-**9101** のノードエクスポーター、ポート **9099** の Cluster Version Operator が含まれます。 |
| | **10250**-**10259** | Kubernetes が予約するデフォルトポート |
| | **10256** | openshift-sdn |
| UDP | **4789** | VXLAN および Geneve |
| | **6081** | VXLAN および Geneve |
| | **9000**-**9999** | ポート **9100**-**9101** のノードエクスポーターを含む、ホストレベルのサービス。 |
| TCP/UDP | **30000**-**32767** | Kubernetes ノードポート |

**表1.8 コントロールプレーンへのすべてのマシン**

| プロトコル | ポート | 説明 |
| --- | --- | --- |
| TCP | **6443** | Kubernetes API |

**表1.9 コントロールプレーンマシンへのコントロールプレーンマシン**

| プロトコル | ポート | 説明 |
| --- | --- | --- |
| TCP | **2379**-**2380** | etcd サーバーおよびピアポート |

**ネットワークトポロジー要件**
クラスター用にプロビジョニングするインフラストラクチャーは、ネットワークトポロジーの以下の要件を満たす必要があります。

**ロードバランサー**
OpenShift Container Platform をインストールする前に、以下の要件を満たす 2 つのロードバランサーをプロビジョニングする必要があります。

1. **API ロードバランサー**: プラットフォームと対話およびプラットフォームを設定するためのユーザー向けの共通のエンドポイントを提供します。以下の条件を設定します。

- Layer 4 の負荷分散のみ。これは、Raw TCP、SSL パススルー、または SSL ブリッジモードと呼ばれます。SSL ブリッジモードを使用する場合は、API ルートの Server Name Indication (SNI) を有効にする必要があります。
- ステートレス負荷分散アルゴリズム。オプションは、ロードバランサーの実装によって異なります。

**注記**

API ロードバランサーが適切に機能するには、セッション永続性は必要ありません。

ロードバランサーのフロントとバックの両方で以下のポートを設定します。

**表1.10 API ロードバランサー**

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

**注記**

ロードバランサーは、API サーバーが **/readyz** エンドポイントをオフにしてからプールから API サーバーインスタンスを削除するまで最大 30 秒かかるように設定する必要があります。**/readyz** の後の時間枠内でエラーが返されたり、正常になったりする場合は、エンドポイントが削除または追加されているはずです。5 秒または 10 秒ごとにプローブし、2 つの正常な要求が正常な状態になり、3 つの要求が正常な状態になりません。これらは十分にテストされた値です。

2. **Application Ingress ロードバランサー**: クラスター外から送られるアプリケーショントラフィックの Ingress ポイントを提供します。以下の条件を設定します。

   - Layer 4 の負荷分散のみ。これは、Raw TCP、SSL パススルー、または SSL ブリッジモードと呼ばれます。SSL ブリッジモードを使用する場合は、Ingress ルートの Server Name Indication (SNI) を有効にする必要があります。
   - 選択可能なオプションやプラットフォーム上でホストされるアプリケーションの種類に基づいて、接続ベースの永続化またはセッションベースの永続化が推奨されます。

   ロードバランサーのフロントとバックの両方で以下のポートを設定します。

**表1.11 アプリケーション Ingress ロードバランサー**

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

**ヒント**

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

**注記**

Ingress ルーターの作業用の設定が OpenShift Container Platform クラスターに必要です。コントロールプレーンの初期化後に Ingress ルーターを設定する必要があります。

**NTP 設定**

OpenShift Container Platform クラスターは、デフォルトでパブリック Network Time Protocol (NTP) サーバーを使用するように設定されます。ローカルのエンタープライズ NTP サーバーを使用する必要があるか、またはクラスターが切断されたネットワークにデプロイされている場合は、特定のタイムサーバーを使用するようにクラスターを設定できます。詳細は、**chrony タイムサービスの設定** のドキュメントを参照してください。

DHCP サーバーが NTP サーバー情報を提供する場合、Red Hat Enterprise Linux CoreOS (RHCOS) マシンの chrony タイムサービスは情報を読み取り、NTP サーバーとクロックを同期できます。

**追加リソース**

- chrony タイムサービスの設定

### 1.2.3.2. 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の DNS 要件

DNS は、名前解決および逆引き名前解決に使用されます。DNS A/AAAA または CNAME レコードは名前解決に使用され、PTR レコードは逆引き名前解決に使用されます。逆引きレコードは、Red Hat Enterprise Linux CoreOS (RHCOS) は逆引きレコードを使用してすべてのノードのホスト名を設定するために重要です。さらに、逆引きレコードは、OpenShift Container Platform が動作するために必要な証明書署名要求 (CSR) を生成するために使用されます。

以下の DNS レコードは、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する OpenShift Container Platform クラスターに必要です。各レコードで、 **<cluster_name>** はクラスター名で、**<base_domain>** は、**install-config.yaml** ファイルに指定するクラスターのベースドメインです。完全な DNS レコードは **<component>.<cluster_name>.<base_domain>.** の形式を取ります。

**表1.12 必要な DNS レコード**

| コンポーネント | レコード | 説明 |
|---|---|---|
| Kubernetes API | **api.<cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコード、および DNS PTR レコードを、コントロールプレーンマシンのロードバランサーを特定するために追加します。これらのレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。 |

| コンポーネント | レコード | 説明 |
| --- | --- | --- |
| | **api-int.<cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコード、および DNS PTR レコードを、コントロールプレーンマシンのロードバランサーを特定するために追加します。これらのレコードは、クラスター内のすべてのノードで解決できる必要があります。<br><br>  <br>**重要**<br>API サーバーは、Kubernetes に記録されるホスト名でワーカーノードを解決できる必要があります。API サーバーがノード名を解決できない場合、プロキシーされる API 呼び出しが失敗し、Pod からログを取得できなくなる可能性があります。 |
| ルート | ***.apps.<cluster_name>.<br><base_domain>.** | デフォルトでワーカーノードの Ingress ルーター Pod を実行するマシンをターゲットにするロードバランサーを参照するワイルドカード DNS A/AAAA または CNAME レコードを追加します。これらのレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。 |
| ブートストラップ | **bootstrap.<cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコードおよび DNS PTR レコードを、ブートストラップマシンを特定するために追加します。これらのレコードは、クラスター内のノードで解決できる必要があります。 |

| コンポーネント | レコード | 説明 |
| --- | --- | --- |
| マスターホスト | **<master><n>.<br><cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコードおよび DNS PTR レコードを、マスターノードの各マシンを特定するために追加します。これらのレコードは、クラスター内のノードで解決できる必要があります。 |
| ワーカーホスト | **<worker><n>.<br><cluster_name>.<br><base_domain>.** | DNS A/AAAA または CNAME レコードおよび DNS PTR レコードを、ワーカーノードの各マシンを特定するために追加します。これらのレコードは、クラスター内のノードで解決できる必要があります。 |

### ヒント

**nslookup <hostname>** コマンドを使用して、名前解決を確認することができます。**dig -x <ip_address>** コマンドを使用して、PTR レコードの逆引き名前解決を確認できます。

BIND ゾーンファイルの以下の例は、名前解決の A レコードの例を示しています。この例の目的は、必要なレコードを表示することです。この例では、特定の名前解決サービスを選択するためのアドバイスを提供することを目的としていません。

**例1.3 DNS ゾーンデータベースのサンプル**

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
```

```
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

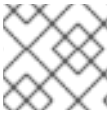
以下の BIND ゾーンファイルの例では、逆引き名前解決の PTR レコードの例を示しています。

**例1.4 逆引きレコードの DNS ゾーンデータベースの例**

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

### 1.2.4. 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

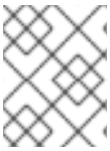
クラスターでインストールのデバッグまたは障害復旧を実行する必要がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



**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
```

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。

1. 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。

### 1.2.5. インストール設定ファイルの手動作成

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する OpenShift Container Platform のインストールでは、インストール設定ファイルを手動で生成します。

**前提条件**

- OpenShift Container Platform インストーラープログラムおよびクラスターのアクセストークンを取得します。

**手順**

1. 必要なインストールアセットを保存するためのインストールディレクトリーを作成します。

```
$ mkdir <installation_directory>
```

**重要**

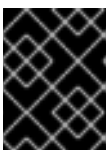
ディレクトリーを作成する必要があります。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

2. 以下の **install-config.yaml** ファイルテンプレートをカスタマイズし、これを **<installation_directory>** に保存します。

**注記**

この設定ファイル **install-config.yaml** に名前を付ける必要があります。

3. **install-config.yaml** ファイルをバックアップし、これを複数のクラスターをインストールするため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の次の手順で使用されます。この時点でこれをバックアップする必要があります。

#### 1.2.5.1. IBM Power のサンプル install-config.yaml ファイル

**install-config.yaml** ファイルをカスタマイズして、OpenShift Container Platform クラスターのプラットフォームについての詳細を指定するか、または必要なパラメーターの値を変更することができます。

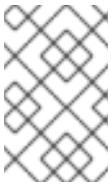
```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OpenShiftSDN
  serviceNetwork: (11)
  - 172.30.0.0/16
platform:
  none: {} (12)
fips: false (13)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' (14)
sshKey: 'ssh-ed25519 AAAA...' (15)
additionalTrustBundle: | (16)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (17)
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: registry.svc.ci.openshift.org/ocp/release
```

(1) クラスターのベースドメイン。すべての DNS レコードはこのベースのサブドメインである必要があり、クラスター名が含まれる必要があります。

(2)(5) **controlPlane** セクションは単一マッピングですが、**compute** セクションはマッピングのシーケンスになります。複数の異なるデータ構造の要件を満たすには、 **compute** セクションの最初の行はハイフン **-** で始め、**controlPlane** セクションの最初の行はハイフンで始めることができません。どちらのセクションも、現時点では単一のマシンプールを定義しますが、OpenShift Container Platform の今後のバージョンでは、インストール時の複数のコンピュートプールの定義をサポートする可能性があります。1 つのコントロールプレーンプールのみが使用されます。

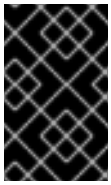
(3)(6) 同時マルチスレッド (SMT) または **hyperthreading** を有効/無効にするかどうか。デフォルトでは、SMT はマシンのコアのパフォーマンスを上げるために有効にされます。パラメーター値を **Disabled** に設定するとこれを無効にすることができます。SMT を無効にする場合、これをすべて

のクラスターマシンで無効にする必要があります。これにはコントロールプレーンとコンピュートマシンの両方が含まれます。



**注記**

同時マルチスレッド (SMT) はデフォルトで有効になっています。SMT が BIOS 設定で有効になっていない場合は、**hyperthreading** パラメーターは効果がありません。



**重要**

BIOS または **install-config.yaml** であるかに関係なく **hyperthreading** を無効にする場合、容量計画においてマシンのパフォーマンスの大幅な低下が考慮に入れられていることを確認します。

4 **replicas** パラメーターの値を **0** に設定する必要があります。このパラメーターはクラスターが作成し、管理するワーカーの数を制御します。これは、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する場合にクラスターが実行しない機能です。OpenShift Container Platform のインストールが終了する前に、クラスターが使用するワーカーマシンを手動でデプロイする必要があります。

7 クラスターに追加するコントロールプレーンマシンの数。クラスターをこの値をクラスターの etcd エンドポイント数として使用するため、値はデプロイするコントロールプレーンマシンの数に一致する必要があります。

8 DNS レコードに指定したクラスター名。

9 Pod IP アドレスの割り当てに使用する IP アドレスのブロック。このブロックは既存の物理ネットワークと重複できません。これらの IP アドレスは Pod ネットワークに使用されます。外部ネットワークから Pod にアクセスする必要がある場合、ロードバランサーおよびルーターを、トラフィックを管理するように設定する必要があります。

10 それぞれの個別ノードに割り当てるサブネットプレフィックスの長さ。たとえば、**hostPrefix** が **23** に設定され、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 (510 (2^(32 - 23) - 2) Pod IP アドレスが許可されます)。外部ネットワークからのノードへのアクセスを提供する必要がある場合には、ロードバランサーおよびルーターを、トラフィックを管理するように設定します。

11 サービス IP アドレスに使用する IP アドレスプール。1 つの IP アドレスプールのみを入力できます。外部ネットワークからサービスにアクセスする必要がある場合、ロードバランサーおよびルーターを、トラフィックを管理するように設定します。

12 プラットフォームを **none**に設定する必要があります。IBM Power インフラストラクチャー用に追加のプラットフォーム設定変数を指定することはできません。

13 FIPS モードを有効または無効にするかどうか。デフォルトでは、FIPS モードは有効にされません。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。

14 **<local_registry>** については、レジストリードメイン名と、ミラーレジストリーがコンテンツを提供するために使用するポートをオプションで指定します。例: **registry.example.com** または **registry.example.com:5000<credentials>** について、ミラーレジストリーの base64 でエンコードされたユーザー名およびパスワードを指定します。

15 Red Hat Enterprise Linux CoreOS (RHCOS) の **core** ユーザーのデフォルト SSH キーの公開部分。

**注記**

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

16 ミラーレジストリーに使用した証明書ファイルの内容を指定します。

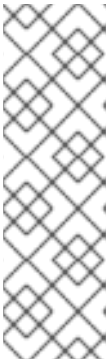
17 リポジトリーのミラーリングに使用するコマンドの出力の **imageContentSources** セクションを指定します。

### 1.2.5.2. インストール時のクラスター全体のプロキシーの設定

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

**前提条件**

- 既存の **install-config.yaml** ファイルが必要です。
- クラスターがアクセスする必要のあるサイトを確認し、プロキシーをバイパスする必要があるかどうかを判別します。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) はプロキシーされます。**Proxy** オブジェクトの **spec.noProxy** フィールドにサイトを追加し、必要に応じてプロキシーをバイパスします。



**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、プロキシー設定を追加します。以下は例になります。

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: http://<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ー URL。URL スキームは **http** である必要があります。追加のプロキシー設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシーネットワークを使用する場合には、**httpProxy** 値を指定することはできません。

2. クラスター外で HTTPS 接続を作成するために使用するプロキシー URL。このフィールドが指定されていない場合、HTTP および HTTPS 接続の両方に **httpProxy** が使用されます。追加のプロキシー設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシーネットワークを使用する場合には、**httpsProxy** 値を指定することはできません。

3. プロキシーを除外するための宛先ドメイン名、ドメイン、IP アドレス、または他のネットワーク CIDR のカンマ区切りの一覧。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ーをバイパスします。

4. 指定されている場合、インストールプログラムは HTTPS 接続のプロキシーに必要な 1 つ以上の追加の CA 証明書が含まれる **user-ca-bundle** という名前の設定マップを **openshift-config** namespace に生成します。次に Cluster Network Operator は、これらのコンテンツを Red Hat Enterprise Linux CoreOS (RHCOS) 信頼バンドルにマージする **trusted-ca-bundle** 設定マップを作成し、この設定マップは **Proxy** オブジェクトの **trustedCA** フィールドで参照されます。**additionalTrustBundle** フィールドは、プロキシーのアイデンティティー証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。追加のプロキシー設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシーネットワークを使用する場合には、MITM CA 証明書を指定する必要があります。

**注記**

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

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

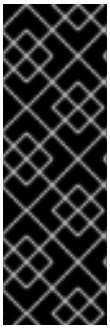
インストールプログラムは、指定の **install-config.yaml** ファイルのプロキシー設定を使用する **cluster** という名前のクラスター全体のプロキシーを作成します。プロキシー設定が指定されていない場合、**cluster Proxy** オブジェクトが依然として作成されますが、これには **spec** がありません。

**注記**

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

### 1.2.6. 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。



**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。

**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、これらのファイルがミラーホスト上に置かれます。
- **install-config.yaml** インストール設定ファイルを作成します。

**手順**

1. クラスターの Kubernetes マニフェストを生成します。

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

**出力例**

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
```

1 **<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。

インストールプロセスの後の部分で独自のコンピュートマシンを作成するため、この警告を無視しても問題がありません。

2. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes マニフェストファイルを変更し、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いようにします。
   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、その値を **False** に設定します。
   c. ファイルを保存し、終了します。
3. Ignition 設定ファイルを取得します。

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

1 **<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
```

### 1.2.7. Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされる IBM Power インフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。ISO イメージまたはネットワーク PXE ブートを使用する手順を実行してマシンを作成することができます。

#### 1.2.7.1. ISO イメージを使用した Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされる IBM Power インフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。ISO イメージを使用してマシンを作成することができます。

**前提条件**

- クラスターの Ignition 設定ファイルを取得していること。
- お使いのコンピューターからアクセスでき、作成するマシンがアクセスできる HTTP サーバーへのアクセスがあること。

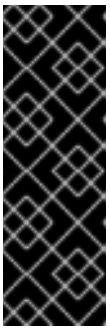
**手順**

1. インストールプログラムが作成したコントロールプレーン、コンピュート、およびブートストラップ Ignition 設定を HTTP サーバーにアップロードします。これらのファイルの URL をメモします。

**重要**

インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

2. 「RHCOS イメージミラー」ページからオペレーティングシステムのインスタンスをインストールするために優先される方法で必要な RHCOS イメージを取得します。



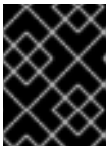
**重要**

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。この手順には ISO イメージのみを使用します。RHCOS qcow2 イメージは、ベアメタルのインストールではサポートされません。

ISO ファイルおよび RAW ディスクファイルをダウンロードする必要があります。これらのファイルの名前は以下の例のようになります。

- ISO: **rhcos-<version>-installer.<architecture>.iso**
- 圧縮された metal RAW: **rhcos-<version>-metal.<architecture>.raw.gz**

3. RAW RHCOS イメージファイルのいずれかを HTTP サーバーにアップロードし、その URL をメモします。



**重要**

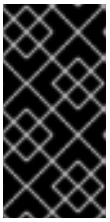
インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

4. ISO を使用し、RHCOS インストールを開始します。以下のインストールオプションのいずれかを使用します。
   - ディスクに ISO イメージを書き込み、これを直接起動します。
   - LOM インターフェースで ISO リダイレクトを使用します。
5. インスタンスの起動後に、**TAB** または **E** キーを押してカーネルコマンドラインを編集します。
6. パラメーターをカーネルコマンドラインに追加します。

```
coreos.inst=yes
coreos.inst.install_dev=sda 1
coreos.inst.image_url=<image_URL> 2
coreos.inst.ignition_url=http://example.com/config.ign 3
ip=<dhcp or static IP address> 4 5
bond=<bonded_interface> 6
```

1. インストール先のシステムのブロックデバイスを指定します。
2. サーバーにアップロードした RAW イメージの URL を指定します。
3. このマシンタイプの Ignition 設定ファイルの URL を指定します。
4. **ip=dhcp** を設定するか、各ノードに個別の静的 IP アドレス（**ip=**）および DNS サーバー（**nameserver=**）を設定します。詳細は、「**高度なネットワークの設定**」を参照してください。
5. 複数のネットワークインターフェースまたは DNS サーバーを使用する場合は、「**高度なネットワークの設定**」を参照してください。
6. オプションで、「**高度なネットワークの設定**」で説明されているように、**bond=** オプションを使用して、複数のネットワークインターフェースを単一のインターフェースにボンディングできます。

7. Enter を押してインストールを完了します。RHCOS のインストール後に、システムは再起動します。システムの再起動後、指定した Ignition 設定ファイルを適用します。
8. 継続してクラスターのマシンを作成します。



**重要**

この時点でブートストラップおよびコントロールプレーンマシンを作成する必要があります。コントロールプレーンマシンがデフォルトのスケジュール対象にされていない場合、クラスターのインストール前に少なくとも 2 つのコンピュートマシンを作成します。

#### 1.2.7.2. PXE ブートによる Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされる IBM Power インフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。PXE ブートを使用してマシンを作成することができます。

**前提条件**

- クラスターの Ignition 設定ファイルを取得していること。
- 使用しているコンピューターからアクセス可能な HTTP サーバーおよび TFTP サーバーへのアクセスがあること。

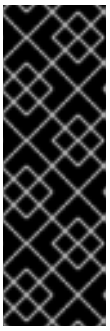
**手順**

1. インストールプログラムが作成したマスター、ワーカーおよびブートストラップの Ignition 設定を HTTP サーバーにアップロードします。これらのファイルの URL をメモします。

**重要**

インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

2. Red Hat カスタマーポータルの「製品のダウンロード」ページまたは 「RHCOS イメージミラー」ページから圧縮された metal RAW **イメージ、kernel** および **initramfs** ファイルを取得します。



**重要**

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。この手順には RAW イメージのみを使用します。RHCOS qcow2 イメージは、ベアメタルのインストールではサポートされません。

ファイル名には、OpenShift Container Platform のバージョン名が含まれます。以下の例のようになります。

- 圧縮されたメタル RAW イメージ: **rhcos-<version>-<architecture>-metal.<architecture>.raw.gz**
- **kernel**: **rhcos-<version>-<architecture>-installer-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-<architecture>-installer-initramfs.<architecture>.img**

3. RAW イメージを HTTP サーバーにアップロードします。

4. 使用する起動方法に必要な追加ファイルをアップロードします。

   - 従来の PXE の場合、**kernel** および **initramfs** ファイルを TFTP サーバーにアップロードします。

   - iPXE の場合、**kernel** および **initramfs** ファイルを HTTP サーバーにアップロードします。

**重要**

インストールの完了後にコンピュートマシンをさらにクラスターに追加する予定の場合には、これらのファイルを削除しないでください。

5. RHCOS のインストール後にマシンがローカルディスクから起動されるようにネットワークブートインフラストラクチャーを設定します。

6. RHCOS イメージに PXE インストールを設定します。
ご使用の環境についての以下の例で示されるメニューエントリーのいずれかを変更し、イメージおよび Ignition ファイルが適切にアクセスできることを確認します。

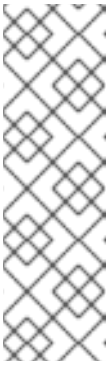
   - PXE の場合:

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL rhcos-<version>-<architecture>-installer-kernel-<architecture> (1)
    APPEND ip=dhcp rd.neednet=1 initrd=rhcos-<version>-<architecture>-installer-
initramfs.<architecture>.img coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://<HTTP_server>/rhcos-<version>-<architecture>-metal.
<architecture>.raw.gz coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2)
(3)
```

(1) TFTP サーバーで利用可能な **kernel** ファイルの場所を指定します。

(2) 複数の NIC を使用する場合、**ip** オプションに単一インターフェースを指定します。たとえば、**eno1** という名前の NIC で DHCP を使用するには、 **ip=eno1:dhcp** を設定します。

(3) HTTP または TFTP サーバーにアップロードした RHCOS ファイルの場所を指定します。**initrd** パラメーター値は、TFTP サーバーの **initramfs** ファイルの場所です。**coreos.inst.image_url** パラメーター値は、HTTP サーバーの圧縮されたメタル RAW イメージの場所であり、 **coreos.inst.ignition_url** パラメーター値は HTTP サーバーのブートストラップ Ignition 設定ファイルの場所になります。



**注記**

この設定では、グラフィカルコンソールを使用するマシンでシリアルコンソールアクセスを有効にしません。別のコンソールを設定するには、**APPEND** 行に 1 つ以上の **console=** 引数を追加します。たとえば、**console=tty0 console=ttyS0** を追加して、最初の PC シリアルポートをプライマリーコンソールとして、グラフィカルコンソールをセカンダリーコンソールとして設定します。詳細は、「How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?」を参照してください。

7. UEFI を使用する場合は、以下の操作を実行します。

   a. システムの起動に必要な EFI バイナリーおよび **grub.cfg** ファイルを提供します。**shim.efi** バイナリーと **grubx64.efi** バイナリーが必要です。

      - RHCOS ISO をホストにマウントし、**images/efiboot.img** ファイルをホストにマウントして、必要な EFI バイナリーを展開します。**efiboot.img** マウントポイントから、**EFI/redhat/shimx64.efi** および **EFI/redhat/grubx64.efi** ファイルを TFTP サーバーにコピーします。

        ```
        # mkdir -p /mnt/{iso,efiboot}
        # mount -o loop rhcos-installer.x86_64.iso /mnt/iso
        # mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
        # cp /mnt/efiboot/EFI/redhat/{shimx64.efi,grubx64.efi} .
        # umount /mnt/{efiboot,iso}
        ```

   b. RHCOS ISO に含まれている **EFI/redhat/grub.cfg** ファイルを TFTP サーバーにコピーします。

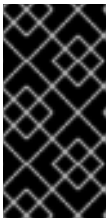
   c. **grub.cfg** ファイルを編集し、以下の引数を追加します。

      ```
      menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --
      class gnu --class os {
       linux rhcos-<version>-<architecture>-installer-kernel-<architecture> nomodeset
      rd.neednet=1 coreos.inst=yes coreos.inst.install_dev=sda
      coreos.inst.image_url=http://<HTTP_server>/rhcos-<version>-<architecture>-metal.
      <architecture>.raw.gz coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1
       initrd rhcos-<version>-<architecture>-installer-initramfs.<architecture>.img 2
      }
      ```

      1. **linux** 行の項目の最初の引数は、TFTP サーバーにアップロードした **kernel** ファイルの場所です。**coreos.inst.image_url** パラメーター値には、HTTP サーバーにアップロードした圧縮されたメタル RAW イメージの場所を指定します。**coreos.inst.ignition_url** パラメーターには、HTTP サーバーにアップロードしたブートストラップ Ingition 設定ファイルの場所を指定します。

      2. TFTP サーバーにアップロードした **initramfs** ファイルの場所を指定します。

8. 継続してクラスターのマシンを作成します。



**重要**

この時点でブートストラップおよびコントロールプレーンマシンを作成する必要があります。コントロールプレーンマシンがデフォルトのスケジュール対象にされていない場合、クラスターのインストール前に少なくとも 2 つのコンピュートマシンを作成します。

### 1.2.8. クラスターの作成

OpenShift Container Platform クラスターを作成するには、ブートストラッププロセスが、インストールプログラムで生成した Ignition 設定ファイルを使用してプロビジョニングしたマシンで完了するのを待機します。

**前提条件**

- クラスターに必要なインフラストラクチャーを作成する。
- インストールプログラムを取得し、クラスターの Ignition 設定ファイルを生成している。
- Ignition 設定ファイルを使用して、クラスターの RHCOS マシンを作成済している。

**手順**

1. ブートストラッププロセスをモニターします。

```
$ ./openshift-install --dir=<installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 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。

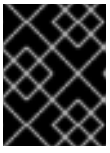
2 異なるインストールの詳細情報を表示するには、**info** ではなく、**warn**、**debug**、または **error** を指定します。

**出力例**

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.18.3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API サーバーでこれがコントロールプレーンマシンにブートストラップされていることを示すシグナルが出されるとコマンドは成功します。

2. ブートストラッププロセスが完了したら、ブートストラップマシンをロードバランサーから削除します。



**重要**

この時点で、ブートストラップマシンをロードバランサーから削除する必要があります。さらに、マシン自体を削除し、再フォーマットすることができます。

### 1.2.9. 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ます。
- **oc** CLI をインストールします。

**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
```

### 1.2.10. 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  v1.18.3
master-1  Ready     master  63m  v1.18.3
```

```
master-2  Ready    master  64m  v1.18.3
worker-0  NotReady  worker  76s  v1.18.3
worker-1  NotReady  worker  70s  v1.18.3
```

出力には作成したすべてのマシンが一覧表示されます。

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。

- 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 --no-run-if-empty oc adm certificate approve
```

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。

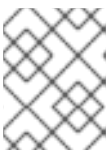
     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
     {{end}}{{end}}' | xargs oc adm certificate approve
     ```

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。

**追加情報**

- CSR の詳細は、「Certificate Signing Requests」を参照してください。

### 1.2.11. Operator の初期設定

コントロールプレーンの初期化後に、一部の Operator を利用可能にするためにそれらをすぐに設定する必要があります。

**前提条件**

- コントロールプレーンが初期化されています。

**手順**

1. クラスターコンポーネントがオンラインになることを確認します。

```
$ watch -n5 oc get clusteroperators
```

**出力例**

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.5.4     True        False         False      69s
cloud-credential                           4.5.4     True        False         False      12m
cluster-autoscaler                         4.5.4     True        False         False      11m
console                                    4.5.4     True        False         False      46s
dns                                        4.5.4     True        False         False      11m
image-registry                             4.5.4     True        False         False      5m26s
ingress                                    4.5.4     True        False         False      5m36s
kube-apiserver                             4.5.4     True        False         False      8m53s
kube-controller-manager                    4.5.4     True        False         False      7m24s
kube-scheduler                             4.5.4     True        False         False      12m
machine-api                                4.5.4     True        False         False      12m
machine-config                             4.5.4     True        False         False      7m36s
marketplace                                4.5.4     True        False         False      7m54m
monitoring                                 4.5.4     True        False         False      7h54s
network                                    4.5.4     True        False         False      5m9s
node-tuning                                4.5.4     True        False         False      11m
openshift-apiserver                        4.5.4     True        False         False      11m
openshift-controller-manager               4.5.4     True        False         False      5m943s
openshift-samples                          4.5.4     True        False         False      3m55s
operator-lifecycle-manager                 4.5.4     True        False         False      11m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      11m
service-ca                                 4.5.4     True        False         False      11m
service-catalog-apiserver                  4.5.4     True        False         False      5m26s
service-catalog-controller-manager         4.5.4     True        False         False      5m25s
storage                                    4.5.4     True        False         False      5m30s
```

2. 利用不可の Operator を設定します。

#### 1.2.11.1. イメージレジストリーストレージの設定

イメージレジストリー Operator は、デフォルトストレージを提供しないプラットフォームでは最初は利用できません。インストール後に、レジストリー Operator を使用できるようにレジストリーをストレージを使用するように設定する必要があります。

実稼働クラスターに必要な永続ボリュームの設定についての手順が示されます。該当する場合、空のディレクトリーをストレージの場所として設定する方法が表示されます。これは、実稼働以外のクラスターでのみ利用できます。

アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

#### 1.2.11.1.1. ベアメタルの場合のレジストリーストレージの設定

クラスター管理者は、インストール後にレジストリーをストレージを使用できるように設定する必要があります。

**前提条件**

- クラスター管理者のパーミッション。
- ベアメタル上のクラスター。
- Red Hat OpenShift Container Storage などのクラスターのプロビジョニングされた永続ストレージ。

**重要**

OpenShift Container Platform は、1 つのレプリカのみが存在する場合にイメージレジストリーストレージの **ReadWriteOnce** アクセスをサポートします。2 つ以上のレプリカで高可用性をサポートするイメージレジストリーをデプロイするには、**ReadWriteMany** アクセスが必要です。

- 100Gi の容量が必要です。

**手順**

1. レジストリーをストレージを使用できるように設定するには、**configs.imageregistry/cluster** リソースの **spec.storage.pvc** を変更します。

**注記**

共有ストレージを使用する場合は、外部からアクセスを防ぐためにセキュリティー設定を確認します。

2. レジストリー Pod がないことを確認します。

```
$ oc get pod -n openshift-image-registry
```

**注記**

ストレージタイプが **emptyDIR** の場合、レプリカ数が **1** を超えることはありません。

3. レジストリー設定を確認します。

```
$ oc edit configs.imageregistry.operator.openshift.io
```

**出力例**

```
storage:
  pvc:
    claim:
```

**claim** フィールドを空のままにし、**image-registry-storage** PVC の自動作成を可能にします。

4. **clusteroperator** ステータスを確認します。

```
$ oc get clusteroperator image-registry
```

#### 1.2.11.1.2. 実稼働以外のクラスターでのイメージレジストリーのストレージの設定

イメージレジストリー Operator のストレージを設定する必要があります。実稼働用以外のクラスターの場合、イメージレジストリーは空のディレクトリーに設定することができます。これを実行する場合、レジストリーを再起動するとすべてのイメージが失われます。

**手順**

1. イメージレジストリーストレージを空のディレクトリーに設定するには、以下を実行します。

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

**警告**

実稼働用以外のクラスターにのみこのオプションを設定します。

イメージレジストリー Operator がそのコンポーネントを初期化する前にこのコマンドを実行する場合、**oc patch** コマンドは以下のエラーを出して失敗します。

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

数分待機した後に、このコマンドを再び実行します。

2. イメージのビルドおよびプッシュを有効にするためにレジストリーが managed に設定されていることを確認します。

- 以下を実行します。

```
$ oc edit configs.imageregistry/cluster
```

次に、行を変更します。

```
managementState: Removed
```

次のように変更してください。

```
managementState: Managed
```

-

### 1.2.12. 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でのインストールの完了

Operator 設定の完了後に、提供するインフラストラクチャーでのクラスターのインストールを終了できます。

**前提条件**

- コントロールプレーンが初期化されています。
- Operator の初期設定を完了済みです。

**手順**

1. 以下のコマンドを使用して、すべてのクラスターコンポーネントがオンラインであることを確認します。

```
$ watch -n5 oc get clusteroperators
```

**出力例**

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.5.4     True        False         False      7m56s
cloud-credential                           4.5.4     True        False         False      31m
cluster-autoscaler                         4.5.4     True        False         False      16m
console                                    4.5.4     True        False         False      10m
csi-snapshot-controller                    4.5.4     True        False         False      16m
dns                                        4.5.4     True        False         False      22m
etcd                                       4.5.4     False       False         False      25s
image-registry                             4.5.4     True        False         False      16m
ingress                                    4.5.4     True        False         False      16m
insights                                   4.5.4     True        False         False      17m
kube-apiserver                             4.5.4     True        False         False      19m
kube-controller-manager                    4.5.4     True        False         False      20m
kube-scheduler                             4.5.4     True        False         False      20m
kube-storage-version-migrator              4.5.4     True        False         False      16m
machine-api                                4.5.4     True        False         False      22m
machine-config                             4.5.4     True        False         False      22m
marketplace                                4.5.4     True        False         False      16m
monitoring                                 4.5.4     True        False         False      10m
network                                    4.5.4     True        False         False      23m
node-tuning                                4.5.4     True        False         False      23m
openshift-apiserver                        4.5.4     True        False         False      17m
openshift-controller-manager               4.5.4     True        False         False      15m
openshift-samples                          4.5.4     True        False         False      16m
operator-lifecycle-manager                 4.5.4     True        False         False      22m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver   4.5.4     True        False         False      18m
service-ca                                 4.5.4     True        False         False      23m
service-catalog-apiserver                  4.5.4     True        False         False      23m
service-catalog-controller-manager         4.5.4     True        False         False      23m
storage                                    4.5.4     True        False         False      17m
```

あるいは、以下のコマンドを使用すると、すべてのクラスターが利用可能な場合に通知されます。また、このコマンドは認証情報を取得して表示します。

```
$ ./openshift-install --dir=<installation_directory> wait-for install-complete 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。

**出力例**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator が Kubernetes API サーバーから OpenShift Container Platform クラスターのデプロイを終了するとコマンドは成功します。

**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。

2. Kubernetes API サーバーが Pod と通信していることを確認します。

   a. すべての Pod の一覧を表示するには、以下のコマンドを使用します。

   ```
   $ oc get pods --all-namespaces
   ```

   **出力例**

   ```
   NAMESPACE                    NAME                                        READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator      openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running     1          9m
   openshift-apiserver               apiserver-67b9g                              1/1     Running     0
   3m
   openshift-apiserver               apiserver-ljcmx                              1/1     Running     0
   1m
   openshift-apiserver               apiserver-z25h4                              1/1     Running     0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8       1/1
   Running     0          5m
   ...
   ```

   b. 以下のコマンドを使用して、直前のコマンドの出力に一覧表示される Pod のログを表示します。

   ```
   $ oc logs <pod_name> -n <namespace> 1
   ```

1 直前のコマンドの出力にあるように、Pod 名および namespace を指定します。

Pod のログが表示される場合、Kubernetes API サーバーはクラスターマシンと通信できます。

**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。