



OpenShift Container Platform 4.6

IBM Z および LinuxONE へのインストール

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

OpenShift Container Platform 4.6 IBM Z および LinuxONE へのインストール

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

Enter your first name here. Enter your surname here.

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

Enter your email address here.

法律上の通知

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

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

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

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

概要

本書では、IBM Z 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 Z へのインストール	5
1.1. クラスターの IBM Z および LINUXONE へのインストール	5
1.1.1. 前提条件	5
1.1.2. OpenShift Container Platform のインターネットアクセス	5
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でのクラスターのマシン要件	6
1.1.3.1. 必要なマシン	6
1.1.3.2. ネットワーク接続の要件	6
1.1.3.3. IBM Z ネットワーク接続の要件	7
1.1.3.4. 最小リソース要件	7
1.1.3.5. 最小の IBM Z システム環境	7
ハードウェア要件	7
オペレーティングシステム要件	8
IBM Z ネットワーク接続の要件	8
z/VM ゲスト仮想マシンのディスクストレージ	8
ストレージ/メインメモリー	8
1.1.3.6. 推奨される IBM Z システム環境	8
ハードウェア要件	8
オペレーティングシステム要件	9
IBM Z ネットワーク接続の要件	9
z/VM ゲスト仮想マシンのディスクストレージ	9
ストレージ/メインメモリー	9
1.1.3.7. 証明書署名要求の管理	9
1.1.4. 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	10
1.1.4.1. 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ーのネットワーク要件	10
ネットワークポロジー要件	11
ロードバランサー	12
1.1.4.2. 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	13
1.1.5. 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	16
1.1.6. インストールプログラムの取得	17
1.1.7. バイナリーのダウンロードによる OpenShift CLI のインストール	18
1.1.7.1. Linux への OpenShift CLI のインストール	18
1.1.7.2. Windows への OpenShift CLI のインストール	19
1.1.7.3. macOS への OpenShift CLI のインストール	19
1.1.8. インストール設定ファイルの手動作成	20
1.1.8.1. インストール設定パラメーター	21
1.1.8.1.1. 必須設定パラメーター	21
1.1.8.1.2. ネットワーク設定パラメーター	23
1.1.8.1.3. オプションの設定パラメーター	25
1.1.8.2. IBM Z のサンプル install-config.yaml ファイル	28
1.1.9. インストール時のクラスター全体のプロキシの設定	31
1.1.10. Kubernetes マニフェストおよび Ignition 設定ファイルの作成	32
1.1.11. Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成	33
1.1.11.1. 詳細の RHCOS インストールリファレンス	35
RHCOS ブートプロンプトでのルーティングおよびボンディングのオプション	36
1.1.12. クラスターの作成	39
1.1.13. CLI の使用によるクラスターへのログイン	40
1.1.14. マシンの証明書署名要求の承認	40
1.1.15. Operator の初期設定	43
1.1.15.1. イメージレジストリーストレージの設定	44
1.1.15.1.1. IBM Z の場合のレジストリーストレージの設定	44
1.1.15.1.2. 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	46

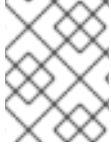
1.1.16. ユーザーによってプロビジョニングされるインフラストラクチャーでのインストールの完了	46
1.1.17. OpenShift Container Platform の Telemetry アクセス	48
1.1.18. デバッグ情報の収集	49
1.1.19. 次のステップ	49
1.2. ネットワークが制限された環境でのクラスターの IBM Z および LINUXONE へのインストール	50
1.2.1. ネットワークが制限された環境でのインストールについて	50
1.2.1.1. その他の制限	51
1.2.2. OpenShift Container Platform のインターネットアクセス	51
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でのクラスターのマシン要件	52
1.2.3.1. 必要なマシン	52
1.2.3.2. ネットワーク接続の要件	52
1.2.3.3. IBM Z ネットワーク接続の要件	52
1.2.3.4. 最小リソース要件	53
1.2.3.5. 最小の IBM Z システム環境	53
ハードウェア要件	53
オペレーティングシステム要件	53
IBM Z ネットワーク接続の要件	54
z/VM ゲスト仮想マシンのディスクストレージ	54
ストレージ/メインメモリー	54
1.2.3.6. 推奨される IBM Z システム環境	54
ハードウェア要件	54
オペレーティングシステム要件	54
IBM Z ネットワーク接続の要件	55
z/VM ゲスト仮想マシンのディスクストレージ	55
ストレージ/メインメモリー	55
1.2.3.7. 証明書署名要求の管理	55
1.2.4. 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	56
1.2.4.1. 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ーのネットワーク要件	56
ネットワークトポロジー要件	57
ロードバランサー	57
1.2.4.2. 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	59
1.2.5. 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	62
1.2.6. インストール設定ファイルの手動作成	63
1.2.6.1. インストール設定パラメーター	64
1.2.6.1.1. 必須設定パラメーター	65
1.2.6.1.2. ネットワーク設定パラメーター	66
1.2.6.1.3. オプションの設定パラメーター	68
1.2.6.2. IBM Z のサンプル install-config.yaml ファイル	72
1.2.6.3. インストール時のクラスター全体のプロキシの設定	75
1.2.7. Kubernetes マニフェストおよび Ignition 設定ファイルの作成	76
1.2.8. Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成	78
1.2.8.1. 詳細の RHCOS インストールリファレンス	80
RHCOS ブートプロンプトでのルーティングおよびボンディングのオプション	80
1.2.9. クラスターの作成	83
1.2.10. CLI の使用によるクラスターへのログイン	84
1.2.11. マシンの証明書署名要求の承認	85
1.2.12. Operator の初期設定	88
1.2.12.1. デフォルトの OperatorHub ソースの無効化	88
1.2.12.2. イメージレジストリーストレージの設定	89
1.2.12.2.1. IBM Z の場合のレジストリーストレージの設定	89
1.2.12.2.2. 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	90
1.2.13. 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	91
1.2.14. OpenShift Container Platform の Telemetry アクセス	93

1.2.15. デバッグ情報の収集	94
1.2.16. 次のステップ	94

第1章 IBM Z へのインストール

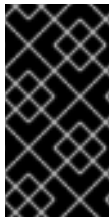
1.1. クラスターの IBM Z および LINUXONE へのインストール

OpenShift Container Platform version 4.6 では、プロビジョニングする IBM Z または LinuxONE 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。



注記

本書は IBM Z のみを参照しますが、これに含まれるすべての情報は LinuxONE にも適用されます。



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。

1.1.1. 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取り除く必要があります。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 を使用して永続ストレージ](#) 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

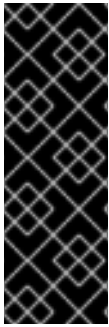
プロキシを設定する場合は、このサイト一覧も確認してください。

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

OpenShift Container Platform 4.6 では、クラスターをインストールするためにインターネットアクセスが必要になります。

インターネットへのアクセスは以下を実行するために必要です。

- [OpenShift Cluster Manager](#) にアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- クラスターのインストールに必要なパッケージを取得するために [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

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

1.1.3. 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。

1.1.3.1. 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改善するには、2つ以上の物理マシンの複数の異なる z/VM インスタンスにコントロールプレーンマシンを分散します。

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。ただし、コンピュートマシンは Red Hat Enterprise Linux CoreOS (RHCOS) または Red Hat Enterprise Linux (RHEL) 7.9 のいずれかを選択できます。

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。

1.1.3.2. 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。マシンは静的 IP アドレスで設定されます。DHCP サーバーは必要ありません。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する必要があります。

1.1.3.3. IBM Z ネットワーク接続の要件

IBM Z の z/VM でインストールするには、レイヤー 2 モードの単一 z/VM 仮想 NIC が必要になります。以下も必要になります。

- 直接接続された OSA または RoCE ネットワークアダプター
- z/VM vSwitch のセットアップ。推奨されるセットアップでは、OSA リンクアグリゲーションを使用します。

1.1.3.4. 最小リソース要件

それぞれのクラスタマシンは、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	IOPS
ブートストラップ	RHCOS	4	16 GB	100 GB	該当なし
コントロールプレーン	RHCOS	4	16 GB	100 GB	該当なし
コンピューター	RHCOS	2	8 GB	100 GB	該当なし

1. 1vCPU 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

1.1.3.5. 最小の IBM Z システム環境

OpenShift Container Platform バージョン 4.6 は、以下の IBM ハードウェアにインストールできます。

- IBM z15 (すべてのモデル)、IBM z14 (すべてのモデル)、IBM z13、および IBM z13s
- LinuxONE(すべてのバージョン)

ハードウェア要件

- 6 つの IFL 相当 (これは、各クラスタで、SMT2 が有効になっている)。
- 最低でもネットワーク接続 1 つ。これで、**LoadBalancer** サービスに接続するだけでなく、クラスタ外のトラフィックに関するデータを提供します。



注記

専用または共有 IFL を使用して、十分なコンピューティングリソースを割り当てることができます。リソース共有は IBM Z の重要な強みの 1 つです。ただし、各ハイパーバイザーレイヤーで容量を正しく調整し、すべての OpenShift Container Platform クラスタに十分なリソースを確保する必要があります。



重要

クラスターの全体的なパフォーマンスに影響を与える可能性があるため、OpenShift Container Platform クラスターの設定に使用される LPAR には十分なコンピューティング能力が必要です。このコンテキストでは、ハイパーバイザーレベルでの LPAR の加重管理、エンタイトルメント、および CPU 共有が重要なロールを果たします。

オペレーティングシステム要件

- z/VM 7.1 以降の 1 インスタンス

z/VM インスタンスで、以下をセットアップします。

- OpenShift Container Platform コントロールプレーンマシンの 3 ゲスト仮想マシン
- OpenShift Container Platform コンピュートマシンの 2 ゲスト仮想マシン
- 一時 OpenShift Container Platform ブートストラップマシンの 1 ゲスト仮想マシン

IBM Z ネットワーク接続の要件

IBM Z の z/VM でインストールするには、レイヤー 2 モードの単一 z/VM 仮想 NIC が必要になります。以下も必要になります。

- 直接接続された OSA または RoCE ネットワークアダプター
- z/VM vSwitch のセットアップ。推奨されるセットアップでは、OSA リンクアグリゲーションを使用します。

z/VM ゲスト仮想マシンのディスクストレージ

- FICON 接続のディスクストレージ (DASD) これらには z/VM ミニディスク、フルパックミニディスク、または専用の DASD を使用でき、これらすべてはデフォルトである CDL としてフォーマットする必要があります。Red Hat Enterprise Linux CoreOS (RHCOS) インストールに必要な最低限の DASD サイズに達するには、拡張アドレスボリューム (EAV) が必要です。利用可能な場合は、HyperPAV を使用して最適なパフォーマンスを確保します。
- FCP 接続のディスクストレージ

ストレージ/メインメモリー

- OpenShift Container Platform コントロールプレーンマシン用に 16 GB
- OpenShift Container Platform コンピュートマシン用に 8 GB
- 一時 OpenShift Container Platform ブートストラップマシン用に 16 GB

1.1.3.6. 推奨される IBM Z システム環境

ハードウェア要件

- 6 つの IFL 相当がそれぞれ割り当てられた LPARS 3 つ (これは、各クラスターで、SMT2 が有効になっている)。
- ネットワーク接続 2 つ。これで、**LoadBalancer** サービスに接続するだけでなく、クラスター外のトラフィックに関するデータを提供します。
- HiperSockets。ノードに直接割り当てられるか、または z/VM ゲストに対して透過性を持たせ

るために z/VM VSWITCH でブリッジしてノードに割り当てられます。HyperSockets をノードに直接接続するには、RHEL 8 ゲスト経由で外部ネットワークにゲートウェイを設定し、Hipersockets ネットワークにブリッジする必要があります。

オペレーティングシステム要件

- 高可用性を確保する場合は z/VM 7.1 以降の 2 または 3 インスタンス

z/VM インスタンスで、以下を設定します。

- OpenShift Container Platform コントロールプレーンマシン用に 3 ゲスト仮想マシン (z/VM インスタンスごとに 1 つ)
- OpenShift Container Platform コンピュートマシン用に 6 以上のゲスト仮想マシン (z/VM インスタンス全体に分散)
- 一時 OpenShift Container Platform ブートストラップマシンの 1 ゲスト仮想マシン
- オーバーコミット環境で必須コンポーネントの可用性を確保するには、CP コマンドの **SET SHARE** を使用してコントロールプレーンの優先度を引き上げます。インフラストラクチャーノードが存在する場合は、同じ操作を行います。IBM ドキュメントの [SET SHARE](#) を参照してください。

IBM Z ネットワーク接続の要件

IBM Z の z/VM でインストールするには、レイヤー 2 モードの単一 z/VM 仮想 NIC が必要になります。以下も必要になります。

- 直接接続された OSA または RoCE ネットワークアダプター
- z/VM vSwitch のセットアップ。推奨されるセットアップでは、OSA リンクアグリゲーションを使用します。

z/VM ゲスト仮想マシンのディスクストレージ

- FICON 接続のディスクストレージ (DASD) これらには z/VM ミニディスク、フルパックミニディスク、または専用の DASD を使用でき、これらすべてはデフォルトである CDL としてフォーマットする必要があります。Red Hat Enterprise Linux CoreOS (RHCOS) インストールに必要な最低限の DASD サイズに達するには、拡張アドレスボリューム (EAV) が必要です。利用可能な場合は、HyperPAV および High Performance FICON (zHPF) を使用して最適なパフォーマンスを確保します。
- FCP 接続のディスクストレージ

ストレージ/メインメモリー

- OpenShift Container Platform コントロールプレーンマシン用に 16 GB
- OpenShift Container Platform コンピュートマシン用に 8 GB
- 一時 OpenShift Container Platform ブートストラップマシン用に 16 GB

1.1.3.7. 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。

関連情報

- IBM ドキュメントの [Bridging a HiperSockets LAN with a z/VM Virtual Switch](#) を参照してください。
- パフォーマンスの最適化については、[Scaling HyperPAV alias devices on Linux guests on z/VM](#) を参照してください。
- LPAR の加重管理とエンタイトルメントについては、[LPAR パフォーマンスのトピック](#) を参照してください。

1.1.4. 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. 静的 IP アドレスをセットアップします。
2. FTP サーバーをセットアップします。
3. 必要なロードバランサーをプロビジョニングします。
4. マシンのポートを設定します。
5. DNS を設定します。
6. ネットワーク接続を確認します。

1.1.4.1. 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ー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 FTP サーバーが必要になります。

マシンに永続 IP アドレスおよびホスト名があることを確認します。

Kubernetes API サーバーはクラスターマシンのノード名を解決する必要があります。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する必要があります。

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ト、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トを含む、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。



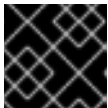
重要

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。



重要

API ロードバランサーのセッションの永続性は設定しないでください。

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
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。

関連情報

- [chrony タイムサービスの設定](#)

1.1.4.2. 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>.<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>.<base_domain>	<p>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。</p> <div>  <div> <p>重要</p> <p>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する必要があります。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。</p> </div> </div>
ルート	*.apps.<cluster_name>.<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>.<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>.<cluster_name>.<base_domain>	コントロールプレーンノード (別名マスターノード) の各マシンを識別するための DNS A/AAAA または CNAME レコードと DNS PTR レコード。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>.<cluster_name>.<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
```

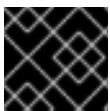
1.1.5. SSH プライベートキーの生成およびエージェントへの追加

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。



重要

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

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

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t ed25519 -N "" \
-f <path>/<file_name> ①
```

- 1 `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が `~/.ssh` ディレクトリーにあることを確認します。

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



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

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

出力例

```
Agent pid 31874
```



注記

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

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

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

出力例

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

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

次のステップ

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

1.1.6. インストールプログラムの取得

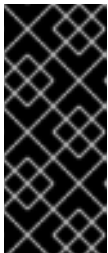
OpenShift Container Platform をインストールする前に、インストールファイルをプロビジョニングマシンにダウンロードします。

前提条件

- Linux を実行するマシン (例: 500 MB のローカルディスク領域のある Red Hat Enterprise Linux 8) が必要です。

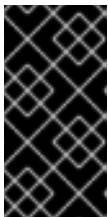
手順

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



重要

インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターのインストール完了後は、インストールプログラムおよびインストールプログラムが作成するファイルを保持する必要があります。ファイルはいずれもクラスターを削除するために必要になります。



重要

インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。クラスターを削除するには、特定のクラウドプロバイダー用の OpenShift Container Platform のアンインストール手順を実行します。

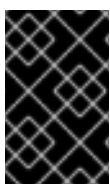
4. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

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

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

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

コマンドラインインターフェイスを使用して 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 4.6 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。

1.1.7.1. Linux への OpenShift 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 カスタマーポータル[の OpenShift Container Platform ダウンロードページ](#) に移動します。
2. **Version** ドロップダウンメニューで適切なバージョンを選択します。
3. **OpenShift v4.6 Linux Client** エントリーの横にある **Download Now** をクリックして、ファイルを保存します。
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
```

OpenShift CLI のインストール後に、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.1.7.2. Windows への OpenShift 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 カスタマーポータル[の OpenShift Container Platform ダウンロードページ](#) に移動します。
2. **Version** ドロップダウンメニューで適切なバージョンを選択します。
3. **OpenShift v4.6 Windows Client** エントリーの横にある **Download Now** をクリックして、ファイルを保存します。
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
```

OpenShift CLI のインストール後に、**oc** コマンドを使用して利用できます。

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

1.1.7.3. macOS への OpenShift 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 カスタマーポータルでの [OpenShift Container Platform ダウンロードページ](#) に移動します。
2. **Version** ドロップダウンメニューで適切なバージョンを選択します。
3. **OpenShift v4.6 MacOSX Client** エントリーの横にある **Download Now** をクリックして、ファイルを保存します。
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
```

OpenShift CLI のインストール後に、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.1.8. 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

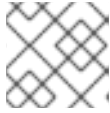
1. 必要なインストールアセットを保存するためのインストールディレクトリーを作成します。

```
$ 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、複数のクラスターをインストールするのに使用できるようにします。

**重要**

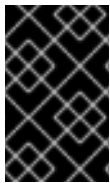
install-config.yaml ファイルは、インストールプロセスの次の手順で使用されます。この時点でこれをバックアップする必要があります。

1.1.8.1. インストール設定パラメーター

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

**注記**

インストール後は、これらのパラメーターを **install-config.yaml** ファイルで変更することはできません。

**重要**

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

1.1.8.1.1. 必須設定パラメーター

必須のインストール設定パラメーターは、以下の表で説明されています。

表1.7 必須パラメーター

パラメーター	説明	値
apiVersion	install-config.yaml コンテンツの API バージョン。現在のバージョンは v1 です。インストーラーは、古い API バージョンをサポートすることもできます。	文字列

パラメーター	説明	値
baseDomain	クラウドプロバイダーのベースドメイン。ベースドメインは、OpenShift Container Platform クラスターコンポーネントへのルートを作成するために使用されます。クラスターの完全な DNS 名は、 baseDomain と <metadata.name> 、 <baseDomain> 形式を使用する metadata.name パラメーターの値の組み合わせです。	example.com などの完全修飾ドメインまたはサブドメイン名。
metadata	Kubernetes リソース ObjectMeta 。ここからは name パラメーターのみが消費されます。	オブジェクト
metadata.name	クラスターの名前。クラスターの DNS レコードはすべて {{.metadata.name}} 、 {{.baseDomain}} のサブドメインです。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 platform.<platform> パラメーターに関する追加情報は、以下の表で特定のプラットフォームについて参照してください。	オブジェクト

パラメーター	説明	値
pullSecret	Red Hat OpenShift Cluster Manager からプルシークレットを取得して、Quay.io などのサービスから OpenShift Container Platform コンポーネントのコンテナイメージをダウンロードすることを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.1.8.1.2. ネットワーク設定パラメーター

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

IPv4 アドレスのみがサポートされます。

表1.8 ネットワークパラメーター

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

パラメーター	説明	値
networking.clusterNetwork.cidr	<p>networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。</p> <p>IPv4 ネットワーク</p>	CIDR (Classless Inter-Domain Routing) 表記の IP アドレスブロック。IPv4 ブロックの接頭辞長は 0 から 32 の間になります。
networking.clusterNetwork.hostPrefix	<p>それぞれの個別ノードに割り当てるサブネット接頭辞長。たとえば、hostPrefix が 23 に設定される場合、各ノードに指定の cidr から /23 サブネットが割り当てられます。hostPrefix 値の 23 は、$2^{(32 - 23) - 2}$ Pod IP アドレスを提供します。</p>	<p>サブネット接頭辞。</p> <p>デフォルト値は 23 です。</p>
networking.serviceNetwork	<p>サービスの IP アドレスブロック。デフォルト値は 172.30.0.0/16 です。</p> <p>OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。</p>	<p>CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。</p> <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	<p>マシンの IP アドレスブロック。</p> <p>複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。</p> <p>複数の IP カーネル引数を指定する場合、machineNetwork.cidr の値はプライマリーネットワークの CIDR である必要があります。</p>	<p>オブジェクトの配列。以下に例を示します。</p> <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>
networking.machineNetwork.cidr	<p>networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、デフォルト値は 10.0.0.0/16 です。libvirt の場合、デフォルト値は 192.168.126.0/24 です。</p>	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>優先される NIC が置かれている CIDR に一致する networking.machineNetwork を設定します。</p> </div> </div>




1.1.8.1.3. オプションの設定パラメーター



オプションのインストール設定パラメーターは、以下の表で説明されています。

表1.9 オプションのパラメーター

パラメーター	説明	値
additionalTrustBundle	ノードの信頼済み証明書ストアに追加される PEM でエンコードされた X.509 証明書バンドル。この信頼バンドルは、プロキシーが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。 詳細は、以下の Machine-pool の表を参照してください。
compute.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列
compute.hyperthreading	<p>コンピュートマシンで同時マルチスレッドまたは hyperthreading を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。</p> <div data-bbox="491 1346 600 1632" data-label="Image"> </div> <p>重要</p> <p>同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れていることを確認します。</p>	Enabled または Disabled
compute.name	compute を使用する場合に必須です。マシンプールの名前。	worker
compute.platform	compute を使用する場合に必須です。このパラメーターを使用して、ワーカーマシンをホストするクラウドプロバイダーを指定します。このパラメーターの値は controlPlane.platform パラメーターの値に一致する必要があります。	aws 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}

パラメーター	説明	値
compute.replicas	プロビジョニングするコンピュートマシン (ワーカーマシンとしても知られる) の数。	2 以上の正の整数。デフォルト値は 3 です。
controlPlane	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
controlPlane.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列
controlPlane.hyperthreading	<p>コントロールプレーンマシンで同時マルチスレッドまたは hyperthreading を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。</p> <div data-bbox="491 1084 600 1370" data-label="Image"> </div> <p>重要</p> <p>同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れていることを確認します。</p>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	master
controlPlane.platform	controlPlane を使用する場合に必須です。このパラメーターを使用して、コントロールプレーンマシンをホストするクラウドプロバイダーを指定します。このパラメーターの値は compute.platform パラメーターの値に一致する必要があります。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 vsphere 、または {}
controlPlane.replicas	プロビジョニングするコントロールプレーンマシンの数。	サポートされる値は 3 のみです (これはデフォルト値です)。

パラメーター	説明	値
credentialsMode	<p>Cloud Credential Operator (CCO) モード。モードを指定しないと、CCO は指定された認証情報の機能を動的に判別しようとします。この場合、複数のモードがサポートされるプラットフォームで Mint モードが優先されます。</p> <div>  <p>注記</p> <p>すべてのクラウドプロバイダーですべての CCO モードがサポートされているわけではありません。CCO モードの詳細は、Red Hat Operator のクラウド認証情報 Operatorを参照してください。</p> </div>	Mint、Passthrough、Manual 、または空の文字列 ("")。
fips	<p>FIPS モードを有効または無効にします。デフォルトは false (無効) です。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>重要</p> <p>FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、x86_64 アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。</p> </div> <div>  <p>注記</p> <p>Azure File ストレージを使用している場合、FIPS モードを有効にすることはできません。</p> </div>	false または true

パラメーター	説明	値
imageContentSources	release-image コンテンツのソースおよびリポジトリ。	オブジェクトの配列。この表の以下の行で説明されているように、 source およびオプションで mirrors が含まれます。
imageContentSources.source	imageContentSources を使用する場合に必須です。ユーザーが参照するリポジトリを指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。
publish	Kubernetes API、OpenShift ルートなどのクラスターのユーザーに表示されるエンドポイントをパブリッシュまたは公開する方法。	<p>Internal または External。デフォルト値は External です。</p> <p>このパラメーターを Internal に設定することは、クラウド以外のプラットフォームではサポートされません。</p> <div>  <div> <p>重要</p> <p>フィールドの値が Internal に設定されている場合、クラスターは機能しなくなります。詳細は、BZ#1953035 を参照してください。</p> </div> </div>
sshKey	<p>クラスターマシンへのアクセスを認証するための単一または複数の SSH キー。</p> <div>  <div> <p>注記</p> <p>インストールのデバッグまたは障害復旧を実行する必要がある実稼働用の OpenShift Container Platform クラスターでは、ssh-agent プロセスが使用する SSH キーを指定します。</p> </div> </div>	<p>1つ以上のキー。以下に例を示します。</p> <pre>sshKey: <key1> <key2> <key3></pre>

1.1.8.2. IBM Z 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 ❶
compute: ❷
- hyperthreading: Enabled ❸
  name: worker
  replicas: 0 ❹
  architecture : s390x
controlPlane: ❺
  hyperthreading: Enabled ❻
  name: master
  replicas: 3 ❼
  architecture : s390x
metadata:
  name: test ❽
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 ❾
    hostPrefix: 23 ❿
  networkType: OpenShiftSDN
  serviceNetwork: ❶❶
  - 172.30.0.0/16
platform:
  none: {} ❶❷
fips: false ❶❸
pullSecret: '{"auths": ...}' ❶❹
sshKey: 'ssh-ed25519 AAAA...' ❶❺

```

❶ 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ます。クラスター名が含まれる必要があります。

❷ ❺ **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。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。

❸ ❻ 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
- 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。



注記

クラス E の CIDR 範囲は、将来の使用のために予約されています。クラス E CIDR 範囲を使用するには、ネットワーク環境がクラス E CIDR 範囲内の IP アドレスを受け入れるように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接頭辞長。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。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。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 Z 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。



重要

FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、**x86_64** アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。

- 14 [Red Hat OpenShift Cluster Manager](#) からの **プルシークレット**。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナイメージを提供する Quay.io など、組み込まれた各種の認証局によって提供されるサービスで認証できます。
- 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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

1.1.9. インストール時のクラスター全体のプロキシの設定

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

前提条件

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



注記

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

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

手順

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> ❶
  httpsProxy: https://<username>:<pswd>@<ip>:<port> ❷
  noProxy: example.com ❸
  additionalTrustBundle: | ❹
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
  ...
```

- ❶ クラスター外の HTTP 接続を作成するために使用するプロキシ URL。URL スキームは **http** である必要があります。
- ❷ クラスター外で HTTPS 接続を作成するために使用するプロキシ URL。
- ❸ プロキシから除外するための宛先ドメイン名、IP アドレス、または他のネットワーク CIDR のコンマ区切りの一覧。サブドメインのみと一致するように、ドメインの前に . を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。* を使用し、すべての宛先のプロキシをバイパスします。
- ❹ 指定されている場合には、インストールプログラムは、**openshift-config** namespace に **user-ca-bundle** という名前の設定魔府を生成して、追加の CA 証明書を保存します。この **additionalTrustBundle** フィールドは、1 つのプロキシ設定を指定した場合に

9. **additionalTrustBundle** と少なくとも一つのプロキシー設定を指定した場合に、**Proxy** オブジェクトは **trustedCA** フィールドで **user-ca-bundle** 設定マップを参照するように設定されます。その後、Cluster Network Operator は、**trustedCA** パラメーターに指定されたコンテンツを RHCOS トラストバンドルにマージする **trusted-ca-bundle** 設定マップを作成します。**additionalTrustBundle** フィールドは、プロキシーのアイデンティティ証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。



注記

インストールプログラムは、プロキシーの **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。

1.1.10. Kubernetes マニフェストおよび Ignition 設定ファイルの作成

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

インストール設定ファイルは Kubernetes マニフェストに変換されます。マニフェストは Ignition 設定ファイルにラップされます。これはクラスターを作成するために後に使用されます。



重要

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

前提条件

- OpenShift Container Platform インストールプログラムを取得していること。

- **install-config.yaml** インストール設定ファイルを作成していること。

手順

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

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

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

2. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes マニフェストファイルの **mastersSchedulable** パラメーターが **false** に設定されていることを確認します。この設定により、Pod がコントロールプレーンマシンにスケジュールされなくなります。
 - a. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** ファイルを開きます。
 - b. **mastersSchedulable** パラメーターを見つけ、これが **false** に設定されていることを確認します。
 - c. ファイルを保存し、終了します。
3. Ignition 設定ファイルを作成するには、インストールプログラムが含まれるディレクトリーから以下のコマンドを実行します。

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

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

以下のファイルはディレクトリーに生成されます。

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

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

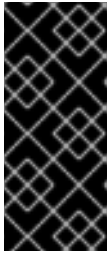
プロビジョニングする IBM Z インフラストラクチャーにクラスターをインストールする前に、クラスターが使用する RHCOS を z/VM ゲスト仮想マシンにインストールする必要があります。マシンを作成するには、以下の手順を実行します。

前提条件

- 作成するマシンがアクセスできるプロビジョニングマシンで稼働している FTP サーバー。

手順

1. プロビジョニングマシンで Linux にログインします。
2. [RHCOS イメージミラー](#) から Red Hat Enterprise Linux CoreOS (RHCOS) カーネル、initramfs および rootfs 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中で最も新しいバージョンのイメージをダウンロードする必要があります。この手順で説明されている適切な kernel、initramfs、および rootfs アーティファクトのみを使用します。

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

- kernel: **rhcos-<version>-live-kernel-<architecture>**
- initramfs: **rhcos-<version>-live-initramfs.<architecture>.img**
- rootfs: **rhcos-<version>-live-rootfs.<architecture>.img**



注記

rootfs イメージは FCP および DASD の場合と同じです。

3. パラメーターファイルを作成します。以下のパラメーターは特定の仮想マシンに固有のもので
す。
 - **coreos.inst.install_dev=** の場合、DASD インストールに **dasda** を指定するか、または FCP に **sda** を指定します。FCP には **zfcf.allow_lun_scan=0** が必要なことに注意してください。
 - **rd.dasd=** の場合、RHCOS がインストールされる DASD を指定します。
 - **rd.zfcf=<adapter>,<wwpn>,<lun>** は、RHCOS をインストールする FCP ディスクを指定します。
 - **ip=** には、以下の 7 つのエントリーを指定します。
 - i. マシンの IP アドレス。
 - ii. 空の文字列。
 - iii. ゲートウェイ。
 - iv. ネットマスク。
 - v. **hostname.domainname** 形式のマシンホストおよびドメイン名。この値を省略して、RHCOS に決定させるようにします。
 - vi. ネットワークインターフェイス名。この値を省略して、RHCOS に決定させるようにします。

vii. 静的 IP アドレスを使用する場合、空の文字列になります。

- **coreos.inst.ignition_url=** の場合、マシンロールの Ignition ファイルを指定します。**bootstrap.ign**、**master.ign**、または **worker.ign** を使用します。HTTP プロトコルおよび HTTPS プロトコルのみがサポートされます。
- **coreos.live.rootfs_url=** の場合、起動しているカーネルおよび initramfs の一致する rootfs アーティファクトを指定します。HTTP プロトコルおよび HTTPS プロトコルのみがサポートされます。
- その他のパラメーターはそのまま利用できます。
ブートストラップマシンのパラメーターファイルのサンプル **bootstrap-0.parm**:

```
rd.neednet=1 \
console=ttySCLP0 \
coreos.inst.install_dev=dasda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-
rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

パラメーターファイルのすべてのオプションを1行で記述し、改行文字がないことを確認します。

4. FTP などを使用し、initramfs、kernel、パラメーターファイル、および RHCOS イメージを z/VM に転送します。FTP でファイルを転送し、仮想リーダーから起動する方法については、[Z/VM 環境へのインストール](#) を参照してください。
5. ブートストラップノードになる z/VM ゲスト仮想マシンの仮想リーダーに対してファイルの punch を実行します。
IBM ドキュメントの [PUNCH](#) を参照してください。

ヒント

CP PUNCH コマンドを使用するか、Linux を使用している場合は、**vmur** コマンドを使用して2つの z/VM ゲスト仮想マシン間でファイルを転送できます。

6. ブートストラップマシンで CMS にログインします。
7. リーダーからブートストラップマシンに対して IPL を実行します。

```
$ ipl c
```

IBM ドキュメントの [IPL](#) を参照してください。

8. クラスタ内の他のマシンについてこの手順を繰り返します。

1.1.11.1. 詳細の RHCOS インストールリファレンス

このセクションでは、Red Hat Enterprise Linux CoreOS (RHCOS) の手動インストールプロセスを変更できるようにするネットワーク設定および他の高度なオプションについて説明します。以下の表では、RHCOS ライブインストーラーおよび **coreos-installer** コマンドで利用できるカーネル引数およびコマ

ンドラインのオプションを説明します。

RHCOS ブートプロンプトでのルーティングおよびボンディングのオプション

ISO イメージから RHCOS をインストールする場合、そのイメージを起動してノードのネットワークを設定する際に手動でカーネル引数を追加できます。ネットワークの引数が使用される場合、インストールはデフォルトで DHCP を使用するよう設定されます。



重要

ネットワーク引数を追加する場合、**rd.neednet=1** カーネル引数も追加する必要があります。

以下の表では、ライブ ISO インストールに **ip=**、**nameserver=**、および **bond=** カーネル引数を使用する方法を説明します。



注記

順序は、カーネル引数の **ip=**、**nameserver=**、および **bond=** を追加する場合に重要です。

ISO のルーティングおよびボンディングのオプション

以下の表は、Red Hat Enterprise Linux CoreOS (RHCOS) ノードのネットワーク設定の例を示しています。これらは、システムの起動時に **dracut** ツールに渡されるネットワークオプションです。**dracut** でサポートされるネットワークオプションの詳細は、man ページの **dracut.cmdline** を参照してください。

詳細	例
IP アドレスを設定するには、DHCP (ip=dhcp) を使用するか、または個別の静的 IP アドレス (ip=<host_ip>) を設定します。次に、各ノードの DNS サーバーの IP アドレス (nameserver=<dns_ip>) を特定します。この例では、以下を設定します。 <ul style="list-style-type: none">ノードの IP アドレス: 10.10.10.2ゲートウェイアドレス: 10.10.10.254ネットワーク: 255.255.255.0ホスト名: core0.example.comDNS サーバーアドレス: 4.4.4.41	<pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41</pre>
複数の ip= エントリを指定して、複数のネットワークインターフェイスを指定します。	<pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</pre>

詳細	例
<p>オプション: rd.route= value を設定して、追加のネットワークへのルートを設定できます。</p> <p>追加のネットワークゲートウェイがプライマリーネットワークゲートウェイと異なる場合、デフォルトゲートウェイはプライマリーネットワークゲートウェイである必要があります。</p>	<p>デフォルトゲートウェイを設定するには、以下の手順に従います。</p> <pre>ip=::10.10.10.254:::</pre> <p>追加ネットワークのルートを設定するには、以下を実行します。</p> <pre>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</pre>
<p>2 つ以上のネットワークインターフェイスがあり、1 つのインターフェイスのみが使用される場合などに、1 つのインターフェイスで DHCP を無効にします。</p>	<pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=:::core0.example.com:enp2s0:none</pre>
<p>複数のネットワークインターフェイスを持つシステムで、DHCP および静的 IP 設定を組み合わせることができます。</p>	<pre>ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</pre>
<p>オプション: vlan= パラメーターを使用して、個別のインターフェイスに VLAN を設定できます。</p>	<p>ネットワークインターフェイスに VLAN を設定し、静的 IP アドレスを使用するには、以下を実行します。</p> <pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0</pre> <p>ネットワークインターフェイス上に VLAN を設定し、DHCP を使用するには、以下を行います。</p> <pre>ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0</pre>
<p>各サーバーに nameserver= エントリーを追加して、複数の DNS サーバーを指定できます。</p>	<pre>nameserver=1.1.1.1 nameserver=8.8.8.8</pre>

詳細	例
<p>オプション: 複数のネットワークインターフェイスを単一のインターフェイスにボンディングすることは、bond= オプションを使用によってサポートされます。次の2つの例では、以下のようになります。</p> <ul style="list-style-type: none"> ● ボンディングされたインターフェイスを設定する構文は bond=name[:network_interfaces][:options] です。 ● name は、ボンディングデバイス名 (bond0) で、network_interfaces は物理 (イーサネット) インターフェイス (em1,em2) のコンマ区切り一覧を表します。options はボンディングオプションのコンマ区切りの一覧です。modinfo bonding を入力して、利用可能なオプションを表示します。 ● Bond= を使用してボンディングされたインターフェイスを作成する場合は、IP アドレスの割り当て方法とボンディングされたインターフェイスのその他の情報を指定する必要があります。 	<p>DHCP を使用するようにボンディングされたインターフェイスを設定するには、ボンドの IP アドレスを dhcp に設定します。以下に例を示します。</p> <pre>bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp</pre> <p>静的 IP アドレスを使用するようにボンディングされたインターフェイスを設定するには、必要な特定の IP アドレスと関連情報を入力します。以下に例を示します。</p> <pre>bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none</pre>
<p>オプション: vlan= パラメーターを使用して、ボンディングされたインターフェイスに VLAN を設定できます。</p>	<p>ボンディングされたインターフェイスを VLAN で設定し、DHCP を使用するには、以下を行います。</p> <pre>ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0</pre> <p>ボンディングされたインターフェイスを VLAN で設定し、静的 IP アドレスを使用するには、以下を行います。</p> <pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0</pre>

詳細	例
<p>オプション: team= パラメーターを使用することで、ボンディングの代わりにネットワークチーミングを使用できます。この例では、以下のように設定されています。</p> <ul style="list-style-type: none"> チームインターフェイス設定の構文は team= name [:network_interfaces] です。 name はチームデバイス名 (team0)、network_interfaces は物理 (イーサネット) インターフェイス (em1、em2) のコンマ区切りリストを表します。 <div data-bbox="161 651 272 909" data-label="Image"></div> <p>注記</p> <p>RHCOS が次のバージョンの RHEL に切り替わると、チーミングは非推奨になる予定です。詳細は、Red Hat ナレッジベースア티クル libvirt-lxc を使用した Linux コンテナ (廃止) を参照してください。</p>	<p>ネットワークチームを設定する方法:</p> <pre>team=team0:em1,em2 ip=team0:dhcp</pre>

1.1.12. 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 プロキシが利用できる。

手順

- 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 \ ❶
--log-level=info ❷
```

- ❶ **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。
- ❷ 異なるインストールの詳細情報を表示するには、**info** ではなく、**warn**、**debug**、または **error** を指定します。

出力例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 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。

1.1.13. CLI の使用によるクラスターへのログイン

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

前提条件

- OpenShift Container Platform クラスターをデプロイしていること。
- **oc** CLI をインストールしていること。

手順

1. **kubeadmin** 認証情報をエクスポートします。

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

1.1.14. 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

- マシンのクラスターに追加されています。

手順

1. クラスターがマシンを認識していることを確認します。

```
$ oc get nodes
```

出力例

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	63m	v1.19.0
master-1	Ready	master	63m	v1.19.0
master-2	Ready	master	64m	v1.19.0

出力には作成したすべてのマシンが一覧表示されます。



注記

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

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

```
$ oc get csr
```

出力例

NAME	AGE	REQUESTOR	CONDITION
csr-mddf5	20m	system:node:master-01.example.com	Approved,Issued
csr-z5rln	16m	system:node:worker-21.example.com	Approved,Issued

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



注記

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



注記

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

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

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

- ❶ **<csr_name>** は、現行の CSR の一覧からの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

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



注記

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

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

```
$ oc get csr
```

出力例

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

5. 残りの CSR が承認されず、それらが **Pending** ステータスにある場合、クラスターマシンの CSR を承認します。

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

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

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
```

- すべてのクライアントおよびサーバーの CSR が承認された後に、マシンのステータスが **Ready** になります。以下のコマンドを実行して、これを確認します。

```
$ oc get nodes
```

出力例

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   73m   v1.20.0
master-1  Ready    master   73m   v1.20.0
master-2  Ready    master   74m   v1.20.0
worker-0  Ready    worker   11m   v1.20.0
worker-1  Ready    worker   11m   v1.20.0
```



注記

サーバー CSR の承認後にマシンが **Ready** ステータスに移行するまでに数分の時間がかかる場合があります。

関連情報

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

1.1.15. 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

- 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 SINCE	VERSION	AVAILABLE	PROGRESSING	DEGRADED
authentication	4.6.0	True	False	False 3h56m
cloud-credential	4.6.0	True	False	False 29h
cluster-autoscaler	4.6.0	True	False	False 29h

config-operator	4.6.0	True	False	False	6h39m
console	4.6.0	True	False	False	3h59m
csi-snapshot-controller	4.6.0	True	False	False	4h12m
dns	4.6.0	True	False	False	4h15m
etcd	4.6.0	True	False	False	29h
image-registry	4.6.0	True	False	False	3h59m
ingress	4.6.0	True	False	False	4h30m
insights	4.6.0	True	False	False	29h
kube-apiserver	4.6.0	True	False	False	29h
kube-controller-manager	4.6.0	True	False	False	29h
kube-scheduler	4.6.0	True	False	False	29h
kube-storage-version-migrator	4.6.0	True	False	False	4h2m
machine-api	4.6.0	True	False	False	29h
machine-approver	4.6.0	True	False	False	6h34m
machine-config	4.6.0	True	False	False	3h56m
marketplace	4.6.0	True	False	False	4h2m
monitoring	4.6.0	True	False	False	6h31m
network	4.6.0	True	False	False	29h
node-tuning	4.6.0	True	False	False	4h30m
openshift-apiserver	4.6.0	True	False	False	3h56m
openshift-controller-manager	4.6.0	True	False	False	4h36m
openshift-samples	4.6.0	True	False	False	4h30m
operator-lifecycle-manager	4.6.0	True	False	False	29h
operator-lifecycle-manager-catalog	4.6.0	True	False	False	29h
operator-lifecycle-manager-packageserver	4.6.0	True	False	False	3h59m
service-ca	4.6.0	True	False	False	29h
storage	4.6.0	True	False	False	4h30m

2. 利用不可の Operator を設定します。

1.1.15.1. 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。

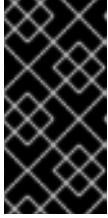
アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

1.1.15.1.1. IBM Z 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。
- IBM Z にクラスターがある。
- クラスター用にプロビジョニングされる永続ストレージ。



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
```

5. 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
```

1.1.15.1.2. 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

- 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。

1.1.16. 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

- 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 SINCE	VERSION AVAILABLE	PROGRESSING	DEGRADED
authentication	4.6.0 True False	False	3h56m
cloud-credential	4.6.0 True False	False	29h
cluster-autoscaler	4.6.0 True False	False	29h
config-operator	4.6.0 True False	False	6h39m
console	4.6.0 True False	False	3h59m
csi-snapshot-controller	4.6.0 True False	False	4h12m
dns	4.6.0 True False	False	4h15m
etcd	4.6.0 True False	False	29h
image-registry	4.6.0 True False	False	3h59m
ingress	4.6.0 True False	False	4h30m
insights	4.6.0 True False	False	29h
kube-apiserver	4.6.0 True False	False	29h
kube-controller-manager	4.6.0 True False	False	29h
kube-scheduler	4.6.0 True False	False	29h
kube-storage-version-migrator	4.6.0 True False	False	4h2m
machine-api	4.6.0 True False	False	29h
machine-approver	4.6.0 True False	False	6h34m
machine-config	4.6.0 True False	False	3h56m
marketplace	4.6.0 True False	False	4h2m
monitoring	4.6.0 True False	False	6h31m
network	4.6.0 True False	False	29h
node-tuning	4.6.0 True False	False	4h30m
openshift-apiserver	4.6.0 True False	False	3h56m
openshift-controller-manager	4.6.0 True False	False	4h36m
openshift-samples	4.6.0 True False	False	4h30m
operator-lifecycle-manager	4.6.0 True False	False	29h
operator-lifecycle-manager-catalog	4.6.0 True False	False	29h
operator-lifecycle-manager-packageserver	4.6.0 True False	False	3h59m
service-ca	4.6.0 True False	False	29h
storage	4.6.0 True False	False	4h30m

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 ❶
```

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

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

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> ❶
```

- ❶ 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。

1.1.17. OpenShift Container Platform の Telemetry アクセス

OpenShift Container Platform 4.6 では、クラスターのヘルスと更新の成功に関するメトリクスを提供するためにデフォルトで実行される Telemetry サービスには、インターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは

[OpenShift Cluster Manager](#) に登録されます。

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

関連情報

- Telemetry サービスの詳細は、[リモートヘルスマニターリング](#) について参照してください。

1.1.18. デバッグ情報の収集

IBM Z での OpenShift Container Platform インストールに関する特定の問題のトラブルシューティングおよびデバッグに役立つ可能性のあるデバッグ情報を収集できます。

前提条件

- **oc** CLI ツールをインストールしていること。

手順

1. クラスターにログインします。

```
$ oc login
```

2. ハードウェア情報を収集するノードで、デバッグコンテナを起動します。

```
$ oc debug node/<nodename>
```

3. **/host** ファイルシステムに切り替え、**toolbox** を起動します。

```
$ chroot /host  
$ toolbox
```

4. **dbginfo** データを収集します。

```
$ dbginfo.sh
```

5. その後に、**scp** を使用するなどしてデータを取得できます。

関連情報

- [How to generate SOSREPORT within OpenShift4 nodes without SSH](#)

1.1.19. 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クラスターの IBM Z および LINUXONE へのインストール

OpenShift Container Platform バージョン 4.6 では、クラスターを制限されたネットワークでプロビジョニングする IBM Z および LinuxONE 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。



注記

本書は IBM Z のみを参照しますが、これに含まれるすべての情報は LinuxONE にも適用されます。



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。
- 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 NFS を使用して [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 4.6 では、ソフトウェアコンポーネントを取得するためにインターネットへのアクティブな接続を必要としないインストールを実行できます。ネットワークが制限された環境のインストールは、クラスタ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。

クラウドプラットフォーム上でネットワークが制限されたインストールの実行を選択した場合でも、そのクラウド API へのアクセスが必要になります。Amazon Web Service の Route 53 DNS や IAM サービスなどの一部のクラウド機能には、インターネットアクセスが必要です。ネットワークによっては、ベアメタルハードウェアまたは VMware vSphere へのインストールには、インターネットアクセスが必要になる場合があります。

ネットワークが制限されたインストールを完了するには、OpenShift Container Platform レジストリーのコンテンツをミラーリングし、インストールメディアを含むレジストリーを作成する必要があります。このミラーは、インターネットと制限されたネットワークの両方にアクセスできるミラーホストで、または制限に対応する他の方法を使用して作成できます。



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

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

インターネットへのアクセスは以下を実行するために必要です。

- [OpenShift Cluster Manager](#) にアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- クラスターのインストールに必要なパッケージを取得するために [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

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

1.2.3. 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。

1.2.3.1. 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つのコンピュータマシン (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改善するには、2つ以上の物理マシンの複数の異なる z/VM インスタンスにコントロールプレーンマシンを分散します。

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。ただし、コンピュータマシンは Red Hat Enterprise Linux CoreOS (RHCOS) または Red Hat Enterprise Linux (RHEL) 7.9 のいずれかを選択できます。

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。

1.2.3.2. 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。マシンは静的 IP アドレスで設定されます。DHCP サーバーは必要ありません。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する必要があります。

1.2.3.3. IBM Z ネットワーク接続の要件

IBM Z の z/VM でインストールするには、レイヤー 2 モードの単一 z/VM 仮想 NIC が必要になります。以下も必要になります。

- 直接接続された OSA または RoCE ネットワークアダプター
- z/VM vSwitch のセットアップ。推奨されるセットアップでは、OSA リンクアグリゲーションを使用します。

1.2.3.4. 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	IOPS
ブートストラップ	RHCOS	4	16 GB	100 GB	該当なし
コントロールプレーン	RHCOS	4	16 GB	100 GB	該当なし
コンピューター	RHCOS	2	8 GB	100 GB	該当なし

- 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

1.2.3.5. 最小の IBM Z システム環境

OpenShift Container Platform バージョン 4.6 は、以下の IBM ハードウェアにインストールできます。

- IBM z15 (すべてのモデル)、IBM z14 (すべてのモデル)、IBM z13、および IBM z13s
- LinuxONE(すべてのバージョン)

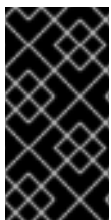
ハードウェア要件

- 6 つの IFL 相当 (これは、各クラスターで、SMT2 が有効になっている)。
- 最低でもネットワーク接続1つ。これで、**LoadBalancer** サービスに接続するだけでなく、クラスター外のトラフィックに関するデータを提供します。



注記

専用または共有 IFL を使用して、十分なコンピューティングリソースを割り当てることができます。リソース共有は IBM Z の重要な強みの1つです。ただし、各ハイパーバイザーレイヤーで容量を正しく調整し、すべての OpenShift Container Platform クラスターに十分なリソースを確保する必要があります。



重要

クラスターの全体的なパフォーマンスに影響を与える可能性があるため、OpenShift Container Platform クラスターの設定に使用される LPAR には十分なコンピューティング能力が必要です。このコンテキストでは、ハイパーバイザーレベルでの LPAR の加重管理、エンタイトルメント、および CPU 共有が重要なロールを果たします。

オペレーティングシステム要件

- z/VM 7.1 以降の1インスタンス

z/VM インスタンスで、以下をセットアップします。

- OpenShift Container Platform コントロールプレーンマシンの 3 ゲスト仮想マシン
- OpenShift Container Platform コンピュートマシンの 2 ゲスト仮想マシン
- 一時 OpenShift Container Platform ブートストラップマシンの 1 ゲスト仮想マシン

IBM Z ネットワーク接続の要件

IBM Z の z/VM でインストールするには、レイヤー 2 モードの単一 z/VM 仮想 NIC が必要になります。以下も必要になります。

- 直接接続された OSA または RoCE ネットワークアダプター
- z/VM vSwitch のセットアップ。推奨されるセットアップでは、OSA リンクアグリゲーションを使用します。

z/VM ゲスト仮想マシンのディスクストレージ

- FICON 接続のディスクストレージ (DASD) これらには z/VM ミニディスク、フルパックミニディスク、または専用の DASD を使用でき、これらすべてはデフォルトである CDL としてフォーマットする必要があります。Red Hat Enterprise Linux CoreOS (RHCOS) インストールに必要な最低限の DASD サイズに達するには、拡張アドレスボリューム (EAV) が必要です。利用可能な場合は、HyperPAV を使用して最適なパフォーマンスを確保します。
- FCP 接続のディスクストレージ

ストレージ/メインメモリー

- OpenShift Container Platform コントロールプレーンマシン用に 16 GB
- OpenShift Container Platform コンピュートマシン用に 8 GB
- 一時 OpenShift Container Platform ブートストラップマシン用に 16 GB

1.2.3.6. 推奨される IBM Z システム環境

ハードウェア要件

- 6 つの IFL 相当がそれぞれ割り当てられた LPARS 3 つ (これは、各クラスターで、SMT2 が有効になっている)。
- ネットワーク接続 2 つ。これで、**LoadBalancer** サービスに接続するだけでなく、クラスター外のトラフィックに関するデータを提供します。
- HiperSockets。ノードに直接割り当てられるか、または z/VM ゲストに対して透過性を持たせるために z/VM VSWITCH でブリッジしてノードに割り当てられます。HiperSockets をノードに直接接続するには、RHEL 8 ゲスト経由で外部ネットワークにゲートウェイを設定し、Hipersockets ネットワークにブリッジする必要があります。

オペレーティングシステム要件

- 高可用性を確保する場合は z/VM 7.1 以降の 2 または 3 インスタンス

z/VM インスタンスで、以下を設定します。

- OpenShift Container Platform コントロールプレーンマシン用に 3 ゲスト仮想マシン (z/VM インスタンスごとに 1 つ)

- OpenShift Container Platform コンピュータマシン用に 6 以上のゲスト仮想マシン (z/VM インスタンス全体に分散)
- 一時 OpenShift Container Platform ブートストラップマシンの 1 ゲスト仮想マシン
- オーバーコミット環境で必須コンポーネントの可用性を確保するには、CP コマンドの **SET SHARE** を使用してコントロールプレーンの優先度を引き上げます。インフラストラクチャードが存在する場合は、同じ操作を行います。IBM ドキュメントの [SET SHARE](#) を参照してください。

IBM Z ネットワーク接続の要件

IBM Z の z/VM でインストールするには、レイヤー 2 モードの単一 z/VM 仮想 NIC が必要になります。以下も必要になります。

- 直接接続された OSA または RoCE ネットワークアダプター
- z/VM vSwitch のセットアップ。推奨されるセットアップでは、OSA リンクアグリゲーションを使用します。

z/VM ゲスト仮想マシンのディスクストレージ

- FICON 接続のディスクストレージ (DASD) これらには z/VM ミニディスク、フルパックミニディスク、または専用の DASD を使用でき、これらすべてはデフォルトである CDL としてフォーマットする必要があります。Red Hat Enterprise Linux CoreOS (RHCOS) インストールに必要な最低限の DASD サイズに達するには、拡張アドレスボリューム (EAV) が必要です。利用可能な場合は、HyperPAV および High Performance FICON (zHPF) を使用して最適なパフォーマンスを確保します。
- FCP 接続のディスクストレージ

ストレージ/メインメモリー

- OpenShift Container Platform コントロールプレーンマシン用に 16 GB
- OpenShift Container Platform コンピュータマシン用に 8 GB
- 一時 OpenShift Container Platform ブートストラップマシン用に 16 GB

1.2.3.7. 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。

関連情報

- IBM ドキュメントの [Bridging a HiperSockets LAN with a z/VM Virtual Switch](#) を参照してください。
- パフォーマンスの最適化については、[Scaling HyperPAV alias devices on Linux guests on z/VM](#) を参照してください。
- LPAR の加重管理とエンタイトルメントについては、[LPAR パフォーマンスのトピック](#) を参照してください。

1.2.4. 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。

1.2.4.1. 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ー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。

Kubernetes API サーバーはクラスターマシンのノード名を解決する必要があります。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する必要があります。

表1.10 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

プロトコル	ポート	説明
	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1.11 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1.12 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

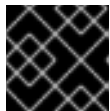
ネットワークポロジー要件

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。



重要

API ロードバランサーのセッションの永続性は設定しないでください。

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1.13 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1.14 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。

```
!restricted:
```

関連情報

- [chrony タイムサービスの設定](#)

1.2.4.2. 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1.15 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>.<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>.<base_domain>.	<p>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。</p> <div>  <div> <p>重要</p> <p>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する必要があります。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。</p> </div> </div>
ルート	*.apps.<cluster_name>.<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>.<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>.<cluster_name>.<base_domain>.	コントロールプレーンノード (別名マスターノード) の各マシンを識別するための DNS A/AAAA または CNAME レコードと DNS PTR レコード。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>.<cluster_name>.<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

```

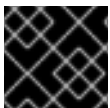
1.2.5. SSH プライベートキーの生成およびエージェントへの追加

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。



重要

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

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

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t ed25519 -N "" \
-f <path>/<file_name>
```

1

~/.ssh/id_rsa などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が ~/.ssh ディレクトリーにあることを確認します。

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



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

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

出力例

```
Agent pid 31874
```



注記

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

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

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

出力例

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

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

次のステップ

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

1.2.6. 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。
- リポジトリのミラーリングに使用するコマンドの出力で **imageContentSources** セクションを取得します。
- ミラーレジストリーの証明書の内容を取得します。

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を作成します。

```
$ mkdir <installation_directory>
```



重要

ディレクトリ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。

- **docker.io** などの、RHCOS がデフォルトで信頼するレジストリーを使用しない限り、**additionalTrustBundle** セクションにミラーリポジトリーの証明書の内容を指定する必要があります。ほとんどの場合、ミラーの証明書を指定する必要があります。
 - 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** セクションを組み込む必要があります。
3. **install-config.yaml** ファイルをバックアップし、複数のクラスターをインストールするのに使用できるようにします。



重要

install-config.yaml ファイルは、インストールプロセス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。

1.2.6.1. インストール設定パラメーター

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



注記

インストール後は、これらのパラメーターを **install-config.yaml** ファイルで変更することはできません。



重要

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

1.2.6.1.1. 必須設定パラメーター

必須のインストール設定パラメーターは、以下の表で説明されています。

表1.16 必須パラメーター

パラメーター	説明	値
apiVersion	install-config.yaml コンテンツの API バージョン。現在のバージョンは v1 です。インストーラーは、古い API バージョンをサポートすることもできます。	文字列
baseDomain	クラウドプロバイダーのベースドメイン。ベースドメインは、OpenShift Container Platform クラスターコンポーネントへのルートを作成するために使用されます。クラスターの完全な DNS 名は、 baseDomain と <metadata.name> 、 <baseDomain> 形式を使用する metadata.name パラメーターの値の組み合わせです。	example.com などの完全修飾ドメインまたはサブドメイン名。
metadata	Kubernetes リソース ObjectMeta 。ここからは name パラメーターのみが消費されます。	オブジェクト
metadata.name	クラスターの名前。クラスターの DNS レコードはすべて {{.metadata.name}} 、 {{.baseDomain}} のサブドメインです。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。

パラメーター	説明	値
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 platform.<platform> パラメーターに関する追加情報は、以下の表で特定のプラットフォームについて参照してください。	オブジェクト
pullSecret	Red Hat OpenShift Cluster Manager からプルシークレットを取得して、Quay.io などのサービスから OpenShift Container Platform コンポーネントのコンテナイメージをダウンロードすることを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.2.6.1.2. ネットワーク設定パラメーター


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

IPv4 アドレスのみがサポートされます。

表1.17 ネットワークパラメーター

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

パラメーター	説明	値
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。
networking.clusterNetwork	Pod の IP アドレスブロック。 デフォルト値は 10.128.0.0/14 で、ホストの接頭辞は /23 です。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。 IPv4 ネットワーク	CIDR (Classless Inter-Domain Routing) 表記の IP アドレスブロック。IPv4 ブロックの接頭辞長は 0 から 32 の間になります。
networking.clusterNetwork.hostPrefix	それぞれの個別ノードに割り当てるサブネット接頭辞長。たとえば、 hostPrefix が 23 に設定される場合、各ノードに指定の cidr から /23 サブネットが割り当てられます。 hostPrefix 値の 23 は、510 ($2^{(32-23)} - 2$) Pod IP アドレスを提供します。	サブネット接頭辞。 デフォルト値は 23 です。
networking.serviceNetwork	サービスの IP アドレスブロック。デフォルト値は 172.30.0.0/16 です。 OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。	CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。 networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	マシンの IP アドレスブロック。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。 複数の IP カーネル引数を指定する場合、 machineNetwork.cidr の値はプライマリーネットワークの CIDR である必要があります。	オブジェクトの配列。以下に例を示します。 networking: machineNetwork: - cidr: 10.0.0.0/16

パラメーター	説明	値
networking.machineNetwork.cidr	networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、デフォルト値は 10.0.0.0/16 です。libvirt の場合、デフォルト値は 192.168.126.0/24 です。	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>優先される NIC が置かれている CIDR に一致する networking.machineNetwork を設定します。</p> </div> </div>


1.2.6.1.3. オプションの設定パラメーター



オプションのインストール設定パラメーターは、以下の表で説明されています。



表1.18 オプションのパラメーター

パラメーター	説明	値
additionalTrustBundle	ノードの信頼済み証明書ストアに追加される PEM でエンコードされた X.509 証明書バンドル。この信頼バンドルは、プロキシが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
compute.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
compute.hyperthreading	<p>コンピュータマシンで同時マルチスレッドまたは hyperthreading を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。</p> <div>  <div> <p>重要</p> <p>同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れていることを確認します。</p> </div> </div>	Enabled または Disabled
compute.name	compute を使用する場合に必須です。マシンプールの名前。	worker
compute.platform	compute を使用する場合に必須です。このパラメーターを使用して、ワーカーマシンをホストするクラウドプロバイダーを指定します。このパラメーターの値は controlPlane.platform パラメーターの値に一致する必要があります。	aws 、 azure 、 gcp 、 openstack 、 ovirt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (ワーカーマシンとしても知られる) の数。	2 以上の正の整数。デフォルト値は 3 です。
controlPlane	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
controlPlane.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
controlPlane.hyperthreading	<p>コントロールプレーンマシンで同時マルチスレッドまたは hyperthreading を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。</p> <div>  <p>重要</p> <p>同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れていることを確認します。</p> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	master
controlPlane.platform	controlPlane を使用する場合に必須です。このパラメーターを使用して、コントロールプレーンマシンをホストするクラウドプロバイダーを指定します。このパラメーターの値は compute.platform パラメーターの値に一致する必要があります。	aws 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}
controlPlane.replicas	プロビジョニングするコントロールプレーンマシンの数。	サポートされる値は 3 のみです (これはデフォルト値です)。
credentialsMode	<p>Cloud Credential Operator (CCO) モード。モードを指定しないと、CCO は指定された認証情報の機能を動的に判別しようとします。この場合、複数のモードがサポートされるプラットフォームで Mint モードが優先されます。</p> <div>  <p>注記</p> <p>すべてのクラウドプロバイダーですべての CCO モードがサポートされているわけではありません。CCO モードの詳細は、Red Hat Operator のクラウド認証情報 Operator を参照してください。</p> </div>	Mint 、 Passthrough 、 Manual 、または空の文字列 ("")。

パラメーター	説明	値
fips	<p>FIPS モードを有効または無効にします。デフォルトは false (無効) です。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>重要</p> <p>FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、x86_64 アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。</p> </div> <div>  <p>注記</p> <p>Azure File ストレージを使用している場合、FIPS モードを有効にすることはできません。</p> </div>	false または true
imageContentSources	release-image コンテンツのソースおよびリポジトリ。	オブジェクトの配列。この表の以下の行で説明されているように、 source およびオプションで mirrors が含まれます。
imageContentSources.source	imageContentSources を使用する場合に必須です。ユーザーが参照するリポジトリを指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。

パラメーター	説明	値
publish	Kubernetes API、OpenShift ルートなどのクラスターのユーザーに表示されるエンドポイントをパブリッシュまたは公開する方法。	<p>Internal または External。デフォルト値は External です。</p> <p>このパラメーターを Internal に設定することは、クラウド以外のプラットフォームではサポートされません。</p> <div>  <div> <p>重要</p> <p>フィールドの値が Internal に設定されている場合、クラスターは機能しなくなります。詳細は、BZ#1953035 を参照してください。</p> </div> </div>
sshKey	<p>クラスターマシンへのアクセスを認証するための単一または複数の SSH キー。</p> <div>  <div> <p>注記</p> <p>インストールのデバッグまたは障害復旧を実行する必要がある実稼働用の OpenShift Container Platform クラスターでは、ssh-agent プロセスが使用する SSH キーを指定します。</p> </div> </div>	<p>1つ以上のキー。以下に例を示します。</p> <pre>sshKey: <key1> <key2> <key3></pre>

1.2.6.2. IBM Z 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
```

[illegible]

- 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。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ター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。



注記

クラス E の CIDR 範囲は、将来の使用のために予約されています。クラス E CIDR 範囲を使用するには、ネットワーク環境がクラス E CIDR 範囲内の IP アドレスを受け入れるように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接頭辞長。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。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。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 Z 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。

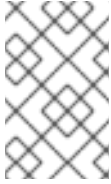


重要

FIPS 検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーの使用は、**x86_64** アーキテクチャーの OpenShift Container Platform デプロイメントでのみサポートされています。

- 14 **<local_registry>** については、レジストリドメイン名と、ミラーレジストリーがコンテンツを提

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

- 16 **additionalTrustBundle** パラメーターおよび値を追加します。この値は、ミラーレジストリーに使用した証明書ファイルの内容である必要があります。これはミラーレジストリー用に生成した既存の、信頼される認証局または自己署名証明書である可能性があります。
- 17 リポジトリ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。

1.2.6.3. インストール時のクラスター全体のプロキシの設定

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

前提条件

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



注記

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

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

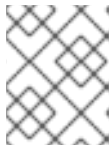
手順

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
```

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

- 1 クラスター外の HTTP 接続を作成するために使用するプロキシ URL。URL スキームは **http** である必要があります。
- 2 クラスター外で HTTPS 接続を作成するために使用するプロキシ URL。
- 3 プロキシから除外するための宛先ドメイン名、IP アドレス、または他のネットワーク CIDR のコンマ区切りの一覧。サブドメインのみと一致するように、ドメインの前に **.** を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。***** を使用し、すべての宛先のプロキシをバイパスします。
- 4 指定されている場合には、インストールプログラムは、**openshift-config** namespace に **user-ca-bundle** という名前の設定魔府を生成して、追加の CA 証明書を保存します。**additionalTrustBundle** と少なくとも 1 つのプロキシ設定を指定した場合には、**Proxy** オブジェクトは **trusted CA** フィールドで **user-ca-bundle** 設定マップを参照するように設定されます。その後、Cluster Network Operator は、**trustedCA** パラメーターに指定されたコンテンツを RHCOS トラストバンドルにマージする **trusted-ca-bundle** 設定マップを作成します。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。



注記

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

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

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



注記

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

1.2.7. Kubernetes マニフェストおよび Ignition 設定ファイルの作成

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

インストール設定ファイルは Kubernetes マニフェストに変換されます。マニフェストは Ignition 設定ファイルにラップされます。これはクラスターを作成するために後に使用されます。

重要

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

前提条件

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

手順

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

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

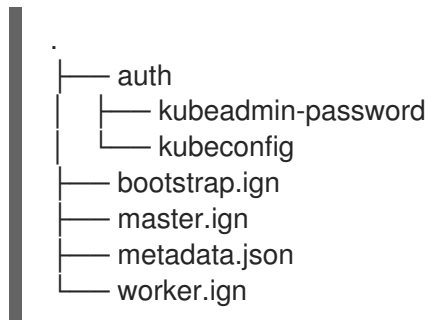
- ❶ **<installation_directory>** については、作成した **install-config.yaml** ファイルが含まれるインストールディレクトリーを指定します。

2. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes マニフェストファイルの **mastersSchedulable** パラメーターが **false** に設定されていることを確認します。この設定により、Pod がコントロールプレーンマシンにスケジュールされなくなります。
 - a. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** ファイルを開きます。
 - b. **mastersSchedulable** パラメーターを見つけ、これが **false** に設定されていることを確認します。
 - c. ファイルを保存し、終了します。
3. Ignition 設定ファイルを作成するには、インストールプログラムが含まれるディレクトリーから以下のコマンドを実行します。

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

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

以下のファイルはディレクトリーに生成されます。



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

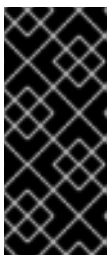
プロビジョニングする IBM Z インフラストラクチャーにクラスターをインストールする前に、クラスターが使用する RHCOS を z/VM ゲスト仮想マシンにインストールする必要があります。マシンを作成するには、以下の手順を実行します。

前提条件

- 作成するマシンがアクセスできるプロビジョニングマシンで稼働している FTP サーバー。

手順

1. プロビジョニングマシンで Linux にログインします。
2. [RHCOS イメージミラー](#) から Red Hat Enterprise Linux CoreOS (RHCOS) カーネル、initramfs および rootfs 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中で最も新しいバージョンのイメージをダウンロードする必要があります。この手順で説明されている適切な kernel、initramfs、および rootfs アーティファクトのみを使用します。

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

- kernel: **rhcos-<version>-live-kernel-<architecture>**
- initramfs: **rhcos-<version>-live-initramfs.<architecture>.img**
- rootfs: **rhcos-<version>-live-rootfs.<architecture>.img**



注記

rootfs イメージは FCP および DASD の場合と同じです。

3. パラメーターファイルを作成します。以下のパラメーターは特定の仮想マシンに固有のものです。
 - **coreos.inst.install_dev=** の場合、DASD インストールに **dasda** を指定するか、または FCP に **sda** を指定します。FCP には **zfcpl.allow_lun_scan=0** が必要なことに注意してください。

- **rd.dasd=** の場合、RHCOS がインストールされる DASD を指定します。
- **rd.zfcp=<adapter>,<wwpn>,<lun>** は、RHCOS をインストールする FCP ディスクを指定します。
- **ip=** には、以下の 7 つのエントリーを指定します。
 - i. マシンの IP アドレス。
 - ii. 空の文字列。
 - iii. ゲートウェイ。
 - iv. ネットマスク。
 - v. **hostname.domainname** 形式のマシンホストおよびドメイン名。この値を省略して、RHCOS に決定させるようにします。
 - vi. ネットワークインターフェイス名。この値を省略して、RHCOS に決定させるようにします。
 - vii. 静的 IP アドレスを使用する場合、空の文字列になります。
- **coreos.inst.ignition_url=** の場合、マシンロールの Ignition ファイルを指定します。**bootstrap.ign**、**master.ign**、または **worker.ign** を使用します。HTTP プロトコルおよび HTTPS プロトコルのみがサポートされます。
- **coreos.live.rootfs_url=** の場合、起動しているカーネルおよび initramfs の一致する rootfs アーティファクトを指定します。HTTP プロトコルおよび HTTPS プロトコルのみがサポートされます。
- その他のパラメーターはそのまま利用できます。
ブートストラップマシンのパラメーターファイルのサンプル **bootstrap-0.parm:**

```
rd.neednet=1 \
console=ttySclp0 \
coreos.inst.install_dev=dasda \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-
rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

パラメーターファイルのすべてのオプションを 1 行で記述し、改行文字がないことを確認します。

4. FTP などを使用し、initramfs、kernel、パラメーターファイル、および RHCOS イメージを z/VM に転送します。FTP でファイルを転送し、仮想リーダーから起動する方法については、[Z/VM 環境へのインストール](#) を参照してください。
5. ブートストラップノードになる z/VM ゲスト仮想マシンの仮想リーダーに対してファイルの punch を実行します。
IBM ドキュメントの [PUNCH](#) を参照してください。

ヒント

CP PUNCH コマンドを使用するか、Linux を使用している場合は、**vmur** コマンドを使用して 2 つの z/VM ゲスト仮想マシン間でファイルを転送できます。

6. ブートストラップマシンで CMS にログインします。
7. リーダーからブートストラップマシンに対して IPL を実行します。

```
$ ipl c
```

IBM ドキュメントの [IPL](#) を参照してください。

8. クラスタ内の他のマシンについてこの手順を繰り返します。

1.2.8.1. 詳細の RHCOS インストールリファレンス

このセクションでは、Red Hat Enterprise Linux CoreOS (RHCOS) の手動インストールプロセスを変更できるようにするネットワーク設定および他の高度なオプションについて説明します。以下の表では、RHCOS ライブインストーラーおよび **coreos-installer** コマンドで使用できるカーネル引数およびコマンドラインのオプションを説明します。

RHCOS ブートプロンプトでのルーティングおよびボンディングのオプション

ISO イメージから RHCOS をインストールする場合、そのイメージを起動してノードのネットワークを設定する際に手動でカーネル引数を追加できます。ネットワークの引数が使用される場合、インストールはデフォルトで DHCP を使用するように設定されます。



重要

ネットワーク引数を追加する場合、**rd.neednet=1** カーネル引数も追加する必要があります。

以下の表では、ライブ ISO インストールに **ip=**、**nameserver=**、および **bond=** カーネル引数を使用する方法を説明します。



注記

順序は、カーネル引数の **ip=**、**nameserver=**、および **bond=** を追加する場合に重要です。

ISO のルーティングおよびボンディングのオプション

以下の表は、Red Hat Enterprise Linux CoreOS (RHCOS) ノードのネットワーク設定の例を示しています。これらは、システムの起動時に **dracut** ツールに渡されるネットワークオプションです。**dracut** でサポートされるネットワークオプションの詳細は、man ページの **dracut.cmdline** を参照してください。

詳細	例
<p>IP アドレスを設定するには、DHCP (ip=dhcp) を使用するか、または個別の静的 IP アドレス (ip=<host_ip>) を設定します。次に、各ノードの DNS サーバーの IP アドレス (nameserver=<dns_ip>) を特定します。この例では、以下を設定します。</p> <ul style="list-style-type: none"> ● ノードの IP アドレス: 10.10.10.2 ● ゲートウェイアドレス: 10.10.10.254 ● ネットワーク: 255.255.255.0 ● ホスト名: core0.example.com ● DNS サーバーアドレス: 4.4.4.41 	<pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41</pre>
<p>複数の ip= エントリーを指定して、複数のネットワークインターフェイスを指定します。</p>	<pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</pre>
<p>オプション: rd.route= value を設定して、追加のネットワークへのルートを設定できます。</p> <p>追加のネットワークゲートウェイがプライマリーネットワークゲートウェイと異なる場合、デフォルトゲートウェイはプライマリーネットワークゲートウェイである必要があります。</p>	<p>デフォルトゲートウェイを設定するには、以下の手順に従います。</p> <pre>ip>::10.10.10.254:::</pre> <p>追加ネットワークのルートを設定するには、以下を実行します。</p> <pre>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</pre>
<p>2 つ以上のネットワークインターフェイスがあり、1 つのインターフェイスのみが使用される場合などに、1 つのインターフェイスで DHCP を無効にします。</p>	<pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=:::core0.example.com:enp2s0:none</pre>
<p>複数のネットワークインターフェイスを持つシステムで、DHCP および静的 IP 設定を組み合わせることができます。</p>	<pre>ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</pre>

詳細	例
<p>オプション: vlan= パラメーターを使用して、個別のインターフェイスに VLAN を設定できます。</p>	<p>ネットワークインターフェイスに VLAN を設定し、静的 IP アドレスを使用するには、以下を実行します。</p> <pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0</pre> <p>ネットワークインターフェイス上に VLAN を設定し、DHCP を使用するには、以下を行います。</p> <pre>ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0</pre>
<p>各サーバーに nameserver= エントリーを追加して、複数の DNS サーバーを指定できます。</p>	<pre>nameserver=1.1.1.1 nameserver=8.8.8.8</pre>
<p>オプション: 複数のネットワークインターフェイスを単一のインターフェイスにボンディングすることは、bond= オプションを使用によってサポートされます。次の 2 つの例では、以下のようになります。</p> <ul style="list-style-type: none"> ● ボンディングされたインターフェイスを設定する構文は bond=name[:network_interfaces][:options] です。 ● name は、ボンディングデバイス名 (bond0) で、network_interfaces は物理 (イーサネット) インターフェイス (em1,em2) のコンマ区切り一覧を表します。options はボンディングオプションのコンマ区切りの一覧です。modinfo bonding を入力して、利用可能なオプションを表示します。 ● Bond= を使用してボンディングされたインターフェイスを作成する場合は、IP アドレスの割り当て方法とボンディングされたインターフェイスのその他の情報を指定する必要があります。 	<p>DHCP を使用するようにボンディングされたインターフェイスを設定するには、ボンドの IP アドレスを dhcp に設定します。以下に例を示します。</p> <pre>bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp</pre> <p>静的 IP アドレスを使用するようにボンディングされたインターフェイスを設定するには、必要な特定の IP アドレスと関連情報を入力します。以下に例を示します。</p> <pre>bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none</pre>

詳細	例
<p>オプション: vlan= パラメーターを使用して、ボンディングされたインターフェイスに VLAN を設定できます。</p>	<p>ボンディングされたインターフェイスを VLAN で設定し、DHCP を使用するには、以下を行います。</p> <pre>ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0</pre> <p>ボンディングされたインターフェイスを VLAN で設定し、静的 IP アドレスを使用するには、以下を行います。</p> <pre>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0</pre>
<p>オプション: team= パラメーターを使用することで、ボンディングの代わりにネットワークチームングを使用できます。この例では、以下のように設定されています。</p> <ul style="list-style-type: none"> チームインターフェイス設定の構文は team= name [:network_interfaces] です。 name はチームデバイス名 (team0)、network_interfaces は物理 (イーサネット) インターフェイス (em1、em2) のコンマ区切りリストを表します。 <div data-bbox="162 1272 271 1527" data-label="Image"> </div> <p>注記</p> <p>RHCOS が次のバージョンの RHEL に切り替わると、チームングは非推奨になる予定です。詳細は、Red Hat ナレッジベースアークル libvirt-lxc を使用した Linux コンテナ (廃止) を参照してください。</p>	<p>ネットワークチームを設定する方法:</p> <pre>team=team0:em1,em2 ip=team0:dhcp</pre>

1.2.9. 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 \ ❶
--log-level=info ❷
```

- ❶ **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。
- ❷ 異なるインストールの詳細情報を表示するには、**info** ではなく、**warn**、**debug**、または **error** を指定します。

出力例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 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。

1.2.10. CLI の使用によるクラスターへのログイン

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

前提条件

- OpenShift Container Platform クラスターをデプロイしていること。
- **oc** CLI をインストールしていること。

手順

1. **kubeadmin** 認証情報をエクスポートします。

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

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

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

```
$ oc whoami
```

出力例

```
system:admin
```

1.2.11. マシンの証明書署名要求の承認

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

前提条件

- マシンがクラスターに追加されています。

手順

1. クラスターがマシンを認識していることを確認します。

```
$ oc get nodes
```

出力例

NAME	STATUS	ROLES	AGE	VERSION
master-0	Ready	master	63m	v1.19.0
master-1	Ready	master	63m	v1.19.0
master-2	Ready	master	64m	v1.19.0

出力には作成したすべてのマシンが一覧表示されます。



注記

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

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

```
$ oc get csr
```

出力例

NAME	AGE	REQUESTOR	CONDITION
csr-8b2br	15m	system:serviceaccount:openshift-machine-config-operator:node-bootstrap	Pending

```
csr-8vnps 15m system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

この例では、2つのマシンがクラスターに参加しています。この一覧にはさらに多くの承認された CSR が表示される可能性があります。

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

注記

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

注記

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

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

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

- ❶ **<csr_name>** は、現行の CSR の一覧からの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

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

注記

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

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

```
$ oc get csr
```

出力例

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

5. 残りの CSR が承認されず、それらが **Pending** ステータスにある場合、クラスターマシンの CSR を承認します。

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

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

1 **<csr_name>** は、現行の CSR の一覧からの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

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

6. すべてのクライアントおよびサーバーの CSR が承認された後に、マシンのステータスが **Ready** になります。以下のコマンドを実行して、これを確認します。

```
$ oc get nodes
```

出力例

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   73m   v1.20.0
master-1  Ready    master   73m   v1.20.0
master-2  Ready    master   74m   v1.20.0
worker-0  Ready    worker   11m   v1.20.0
worker-1  Ready    worker   11m   v1.20.0
```



注記

サーバー CSR の承認後にマシンが **Ready** ステータスに移行するまでに数分の時間がかかる場合があります。

関連情報

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

1.2.12. 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 SINCE	VERSION	AVAILABLE	PROGRESSING	DEGRADED
authentication	4.6.0	True	False	False 3h56m
cloud-credential	4.6.0	True	False	False 29h
cluster-autoscaler	4.6.0	True	False	False 29h
config-operator	4.6.0	True	False	False 6h39m
console	4.6.0	True	False	False 3h59m
csi-snapshot-controller	4.6.0	True	False	False 4h12m
dns	4.6.0	True	False	False 4h15m
etcd	4.6.0	True	False	False 29h
image-registry	4.6.0	True	False	False 3h59m
ingress	4.6.0	True	False	False 4h30m
insights	4.6.0	True	False	False 29h
kube-apiserver	4.6.0	True	False	False 29h
kube-controller-manager	4.6.0	True	False	False 29h
kube-scheduler	4.6.0	True	False	False 29h
kube-storage-version-migrator	4.6.0	True	False	False 4h2m
machine-api	4.6.0	True	False	False 29h
machine-approver	4.6.0	True	False	False 6h34m
machine-config	4.6.0	True	False	False 3h56m
marketplace	4.6.0	True	False	False 4h2m
monitoring	4.6.0	True	False	False 6h31m
network	4.6.0	True	False	False 29h
node-tuning	4.6.0	True	False	False 4h30m
openshift-apiserver	4.6.0	True	False	False 3h56m
openshift-controller-manager	4.6.0	True	False	False 4h36m
openshift-samples	4.6.0	True	False	False 4h30m
operator-lifecycle-manager	4.6.0	True	False	False 29h
operator-lifecycle-manager-catalog	4.6.0	True	False	False 29h
operator-lifecycle-manager-packageserver	4.6.0	True	False	False 3h59m
service-ca	4.6.0	True	False	False 29h
storage	4.6.0	True	False	False 4h30m

2. 利用不可の Operator を設定します。

1.2.12.1. デフォルトの OperatorHub ソースの無効化

Red Hat によって提供されるコンテンツを調達する Operator カタログおよびコミュニティプロジェクトは、OpenShift Container Platform のインストール時にデフォルトで OperatorHub に設定されます。ネットワークが制限された環境では、クラスター管理者としてデフォルトのカタログを無効にする必要があります。

手順

- **disableAllDefaultSources: true** を **OperatorHub** オブジェクトに追加して、デフォルトカタログのソースを無効にします。

```
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

ヒント

または、Web コンソールを使用してカタログソースを管理できます。**Administration → Cluster Settings → Global Configuration → OperatorHub** ページから、**Sources** タブをクリックして、個別のソースを作成し、削除し、無効にし、有効にすることができます。

1.2.12.2. 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是利用できません。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。

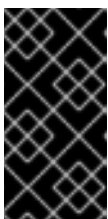
アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

1.2.12.2.1. IBM Z 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。
- IBM Z にクラスターがある。
- クラスター用にプロビジョニングされる永続ストレージ。



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
```

5. 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
```

1.2.12.2.2. 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

- 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。

1.2.13. 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

- 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 SINCE	VERSION	AVAILABLE	PROGRESSING	DEGRADED
authentication	4.6.0	True	False	False 3h56m
cloud-credential	4.6.0	True	False	False 29h
cluster-autoscaler	4.6.0	True	False	False 29h

config-operator	4.6.0	True	False	False	6h39m
console	4.6.0	True	False	False	3h59m
csi-snapshot-controller	4.6.0	True	False	False	4h12m
dns	4.6.0	True	False	False	4h15m
etcd	4.6.0	True	False	False	29h
image-registry	4.6.0	True	False	False	3h59m
ingress	4.6.0	True	False	False	4h30m
insights	4.6.0	True	False	False	29h
kube-apiserver	4.6.0	True	False	False	29h
kube-controller-manager	4.6.0	True	False	False	29h
kube-scheduler	4.6.0	True	False	False	29h
kube-storage-version-migrator	4.6.0	True	False	False	4h2m
machine-api	4.6.0	True	False	False	29h
machine-approver	4.6.0	True	False	False	6h34m
machine-config	4.6.0	True	False	False	3h56m
marketplace	4.6.0	True	False	False	4h2m
monitoring	4.6.0	True	False	False	6h31m
network	4.6.0	True	False	False	29h
node-tuning	4.6.0	True	False	False	4h30m
openshift-apiserver	4.6.0	True	False	False	3h56m
openshift-controller-manager	4.6.0	True	False	False	4h36m
openshift-samples	4.6.0	True	False	False	4h30m
operator-lifecycle-manager	4.6.0	True	False	False	29h
operator-lifecycle-manager-catalog	4.6.0	True	False	False	29h
operator-lifecycle-manager-packageserver	4.6.0	True	False	False	3h59m
service-ca	4.6.0	True	False	False	29h
storage	4.6.0	True	False	False	4h30m

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

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
3m			
openshift-apiserver	apiserver-ljcmx	1/1	Running
1m			
openshift-apiserver	apiserver-z25h4	1/1	Running
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> ❶
```

❶ 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せん。

3. [Cluster registration](#) ページでクラスターを登録します。

1.2.14. OpenShift Container Platform の Telemetry アクセス

OpenShift Container Platform 4.6 では、クラスターのヘルスと更新の成功に関するメトリクスを提供

するためにデフォルトで実行される Telemetry サービスには、インターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [OpenShift Cluster Manager](#) に登録されます。

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

関連情報

- Telemetry サービスの詳細は、[リモートヘルスマニターリング](#) について参照してください。

1.2.15. デバッグ情報の収集

IBM Z での OpenShift Container Platform インストールに関する特定の問題のトラブルシューティングおよびデバッグに役立つ可能性のあるデバッグ情報を収集できます。

前提条件

- oc** CLI ツールをインストールしていること。

手順

1. クラスターにログインします。

```
$ oc login
```

2. ハードウェア情報を収集するノードで、デバッグコンテナを起動します。

```
$ oc debug node/<nodename>
```

3. `/host` ファイルシステムに切り替え、**toolbox** を起動します。

```
$ chroot /host  
$ toolbox
```

4. **dbginfo** データを収集します。

```
$ dbginfo.sh
```

5. その後に、**scp** を使用するなどしてデータを取得できます。

関連情報

- [How to generate SOSREPORT within OpenShift Container Platform version 4 nodes without SSH](#)

1.2.16. 次のステップ

- [クラスターをカスタマイズ](#) します。

- クラスターのインストールに使用したミラーレジストリーに信頼される CA がある場合、[信頼ストアを設定](#)してこれをクラスターに追加します。