OpenShift Container Platform 4.3

クラスターの更新

OpenShift Container Platform クラスターの更新

Last Updated: 2020-11-13

OpenShift Container Platform 4.3 クラスターの更新

OpenShift Container Platform クラスターの更新

法律上の通知

概要

本書では、OpenShift Container Platform クラスターを更新し、アップグレードする方法を説明します。クラスターの更新を、クラスターをオフラインにする必要のない単純なプロセスで実行できます。

目次

第1章 クラスターのマイナーバージョン間の更新

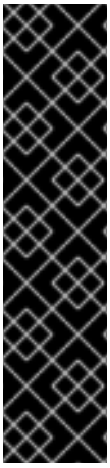
マイナーバージョン間で OpenShift Container Platform クラスターの更新またはアップグレードを実行できます。

注記

oc を使用して更新チャネルを変更するのが容易ではないため、Web コンソールを使用して更新チャネルを変更します。Web コンソール内で更新プロセスを完了することが推奨されます。4.3 チャネルに変更を加えた後に更新を完了するために、CLI を使用してマイナーバージョン内でクラスターを更新する手順を実行できます。

1.1. 前提条件

- **admin** 権限を持つユーザーとしてクラスターにアクセスできること。「Using RBAC to define and apply permissions」を参照してください。
- アップグレードが失敗し、クラスターを直前の状態に復元する必要がある場合に最新の etcd バックアップがあること。

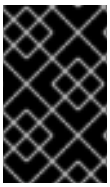


重要

OpenShift Container Platform 4.2 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。

これは、OpenShift Container Platform 4.3.5 の時点でサービス CA が自動的にローテーションされるためです。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。



重要

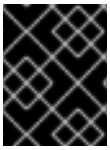
unsupportedConfigOverrides セクションを使用して Operator の設定を変更することはサポートされておらず、クラスターのアップグレードをブロックする可能性があります。クラスターをアップグレードする前に、この設定を削除する必要があります。

1.2. OPENSHIFT CONTAINER PLATFORM の更新サービスについて

OpenShift Container Platform の更新サービスとは、OpenShift Container Platform と Red Hat Enterprise Linux CoreOS (RHCOS) の両方に OTA(over-the-air) 更新を提供するホスト型サービスです。コンポーネント Operator のグラフ、または **頂点** とそれらを結ぶ **辺** を含む図表が提示されます。グラフの辺は、どのバージョンであれば安全に更新できるかを示します。 頂点は、管理されたクラスターコンポーネントの想定された状態を特定する更新のペイロードです。

クラスター内の Cluster Version Operator (CVO) は、OpenShift Container Platform の更新サービスをチェックして、グラフの現在のコンポーネントバージョンとグラフの情報に基づき、有効な更新および更新パスを確認します。ユーザーが更新をリクエストすると、OpenShift Container Platform CVO はその更新のリリースイメージを使ってクラスターをアップグレードします。リリースアーティファクトは、コンテナーイメージとして Quay でホストされます。

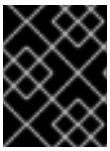
OpenShift Container Platform 更新サービスが互換性のある更新のみを提供できるようにするために、自動化を支援するリリース検証 Pipeline が使用されます。それぞれのリリースアーティファクトについて、他のコンポーネントパッケージだけでなくサポートされているクラウドプラットフォームおよびシステムアーキテクチャーとの互換性の有無が検証されます。Pipeline がリリースの適合性を確認した後に、OpenShift Container Platform 更新サービスは更新が利用可能であることを通知します。



重要

更新サービスが有効な更新をすべて表示するために、更新サービスが表示しないバージョンへの更新を強制することはできません。

連続更新モードでは、2 つのコントローラーが実行されます。1 つのコントローラーはペイロードマニフェストを絶えず更新し、それらをクラスターに適用し、Operator が利用可能か、アップグレード中か、または失敗しているかに応じて Operator の制御されたロールアウトのステータスを出力します。2 つ目のコントローラーは OpenShift Container Platform 更新サービスをポーリングして、更新が利用可能かどうかを判別します。



重要

クラスターを以前のバージョンに戻すこと、つまりロールバックはサポートされていません。サポートされているのは、新規バージョンへのアップグレードのみです。

アップグレードプロセスで、Machine Config Operator (MCO) は新規設定をクラスターマシンに適用します。これは、マシン設定プールの **maxUnavailable** フィールドによって指定されるノードの数を分離し、それらを利用不可としてマークします。デフォルトで、この値は **1** に設定されます。次に、新しい設定を適用して、マシンを再起動します。Red Hat Enterprise Linux (RHEL) マシンをワーカーとして使用する場合、まず OpenShift API をそれらのマシンで更新する必要があるため、MCO はそれらのマシンで kubelet を更新しません。新規バージョンの仕様は古い kubelet に適用されるため、RHEL マシンを **Ready** 状態に戻すことができません。マシンが利用可能になるまで更新を完了することはできません。ただし、利用不可のノードの最大数は、その数のマシンがサービス停止状態のマシンとして分離されても通常のクラスター操作が継続できるようにするために設定されます。

追加リソース

- 「管理外の Operator のサポートポリシー」

1.3. OPENSHIFT CONTAINER PLATFORM アップグレードチャネルおよびリリース

OpenShift Container Platform 4.1 で、Red Hat はクラスターのアップグレードの適切なリリースバージョンを推奨するためにチャネルという概念を導入しました。アップグレードのペースを制御することで、これらのアップグレードチャネルからアップグレードストラテジーを選択することができます。アップグレードチャネルは OpenShift Container Platform のマイナーバージョンに関連付けられます。たとえば、OpenShift Container Platform 4.3 アップグレードチャネルには 4.4 リリースへのアップグレードが含まれることはありません。このストラテジーにより、管理者は OpenShift Container Platform の次のマイナーバージョンへのアップグレードに関して明確な決定を行うことができます。アップグレードチャネルはリリースの選択のみを制御し、インストールするクラスターのバージョンには影響を与えません。 OpenShift Container Platform の特定のバージョンの **openshift-install** バイナリーファイルは常に該当バージョンをインストールします。

OpenShift Container Platform 4.3 は以下のアップグレードチャネルを提供します。

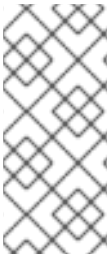
- **candidate-4.3**

- **fast-4.3**
- **stable-4.3**

candidate-4.3 チャネル

candidate-4.3 チャネルには、z-stream (4.3.z) リリースの候補となるビルドが含まれます。リリース候補には、製品のすべての機能が含まれますが、それらがサポートされる訳ではありません。リリース候補を使用して機能の受け入れテストを実行し、OpenShift Container Platform の次のバージョンへの対応を支援します。リリース候補は、名前に **-rc** を含まないものを含め、候補チャネルで利用可能なビルドを指します。候補チャネルでバージョンが利用可能になると、さらに品質のチェックが行われます。品質基準を満たす場合には、これは **fast-4.3** または **stable-4.3** チャネルにプロモートされます。このストラテジーにより、特定のリリースが **candidate-4.3** チャネルと **fast-4.3** または **stable-4.3** チャネルの両方で利用可能な場合、そのリリースは Red Hat でサポートされるバージョンということになります。**candidate-4.3** チャネルには、いずれのチャネルでも推奨されている更新のないリリースバージョンを含めることができます。

candidate-4.3 チャネルを使用して、OpenShift Container Platform の直前のマイナーバージョンからアップグレードできます。



注記

リリース候補は夜間ビルドとは異なります。夜間ビルドは各種機能への早期アクセスのために利用できますが、夜間ビルドへの/からの更新は推奨されておらず、サポートもされていません。夜間ビルドはいずれのアップグレードチャネルでも利用できません。ビルドについての詳細は、OpenShift Container Platform の リリースのステータス を参照できます。

fast-4.3 チャネル

fast-4.3 チャネルは、Red Hat が一般公開リリースとして指定のバージョンを宣言するとすぐに新しいバージョンで更新されます。そのため、これらのリリースは完全にサポートされ、実稼働用の品質があり、これらのリリースのプロモート元の **candidate-4.3** チャネルのリリース候補として利用可能であった間のパフォーマンスにも問題がなかったリリースです。リリースは **fast-4.3** チャネルに表示されてからしばらくすると、**stable-4.3** チャネルに追加されます。リリースは **fast-4.3** チャネルに表示される前に、**stable-4.3** チャネルに表示されることはありません。

fast-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

stable-4.3 チャネル

fast-4.3 チャネルにはエラータの公開後すぐにリリースが組み込まれ、リリースの **stable-4.3** チャネルへの追加は遅延します。この期間中、接続環境のカスタマープログラム(Connected Customer Program) に関わる Red Hat SRE チーム、Red Hat サポートサービス、および実稼働前および実稼働環境からリリースの安定性についてのデータが収集されます。

stable-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

アップグレードバージョンパス

OpenShift Container Platform では、インストールされた OpenShift Container Platform のバージョンと、次のリリースにアクセスするために選択したチャネル内のパスの確認を可能にするアップグレード推奨サービスが提供されます。**fast-4.3** チャネルでは以下を確認できます。

- 4.3.0
- 4.3.1

- 4.3.3
- 4.3.4

このサービスは、テスト済みの重大な問題のないアップグレードのみを推奨します。クラスターが 4.3.1 にあり、OpenShift Container Platform が 4.3.4 を提案している場合、4.3.1 から 4.3.4 に更新しても問題がありません。パッチの連続する番号のみに依存しないようにしてください。たとえば、この例では 4.3.2 はチャネルで利用可能な状態ではなく、これまで利用可能になったことがありません。更新サービスは、既知の脆弱性を含む OpenShift Container Platform のバージョンへの更新を提案しません。

更新の安定性は、チャネルによって異なります。**candidate-4.3** チャネルに更新についての推奨があるからといって、その更新が必ずしもサポートされる訳ではありません。つまり、更新について深刻な問題がまだ検出されていないものの、この更新の安定性についての提案を導くようなトラフィックの安定性はとくに確認されていない可能性があります。**fast-4.3** または **stable-4.3** チャネルに更新の推奨がある場合は、更新がそれぞれのチャネルにある限り完全にサポートされることを示します。リリースがチャネルから削除されることは決してありませんが、深刻な問題を示す更新の推奨はすべてのチャネルから削除されます。更新の推奨が削除された後に開始された更新はサポートされない可能性があります。

Red Hat は最終的には、**fast-4.3** または **stable-4.3** チャネルのサポートされるリリースから 4.3.z の最新リリースへのサポートされる更新パスを提供します。ただし、問題のあるリリースからの安全なパスが構築され、検証される間に遅延が生じる可能性があります。

高速かつ安定したチャネルの使用およびストラテジー
fast-4.3 および **stable-4.3** チャネルでは、一般公開リリースが利用可能になり次第これを受信するか、または Red Hat がそれらの更新のロールアウトを制御するようにするかを選択することができます。問題がロールアウト時またはロールアウト後に検出される場合、該当バージョンへのアップグレードは **fast-4.3** および **stable-4.3** チャネルの両方でブロックされ、新たに推奨されるアップグレード先の新規バージョンが導入される可能性があります。

fast-4.3 チャネルで実稼働前のシステムを設定し、**stable-4.3** チャネルで実稼働システムを設定してから Red Hat の接続環境のカスタマープログラム(Connected Customer Program) に参加することで、お客様のプロセスを改善することができます。Red Hat はこのプログラムを使用して、ご使用の特定のハードウェアおよびソフトウェア設定に対する更新の影響の有無を確認します。今後のリリースでは、更新が **fast-4.3** から **stable-4.3** チャネルに移行するペースが改善されるか、変更される可能性があります。

ネットワークが制限された環境のクラスター
OpenShift Container Platform クラスターのコンテナーイメージを独自に管理する場合には、製品リリースに関連する Red Hat エラータを確認し、アップグレードへの影響に関するコメントに留意する必要があります。アップグレード時に、インターフェースにこれらのバージョン間の切り替えについての警告が表示される場合があります。そのため、これらの警告を無視するかどうかを決める前に適切なバージョンを選択していることを確認する必要があります。

チャネル間の切り替え
stable-4.3 チャネルから **fast-4.3** チャネルに切り換える場合も、クラスターは引き続きサポートされます。**candidate-4.3** チャネルに常に切り替えることはできますが、そのチャネルの一部のリリースはサポートされないリリース候補である可能性があります。現在のリリースが一般利用公開リリースの場合、**candidate-4.3** チャネルから **fast-4.3** チャネルに切り換えることができます。**fast-4.3** チャネルから **stable-4.3** チャネルに常に切り換えることができますが、現在のリリースが **fast-4.3** に最近プロモートされた場合、リリースが **stable-4.3** にプロモートされるまでに最長 1 日分の遅延が生じる可能性があります。現在のリリースを含まないチャネルに変更すると、アラートが表示され、更新は推奨されません。ただし、元のチャネルにいつでも安全に戻ることができます。

1.4. WEB コンソールを使用したクラスターの更新

更新が利用可能な場合、Web コンソールからクラスターを更新できます。

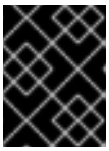
利用可能な OpenShift Container Platform アドバイザリーおよび更新については、カスタマーポータルの エラータのセクションを参照してください。

前提条件

- **admin** 権限を持つユーザーとして Web コンソールにアクセスできること。

手順

1. Web コンソールから、**Administration** > **Cluster Settings** をクリックし、 **Overview** タブの内容を確認します。
2. 実稼働クラスターの場合、**CHANNEL** が **stable-4.3** などの現在のマイナーバージョンの正しいチャネルに設定されていることを確認します。



重要

実稼働クラスターの場合、stable-* または fast-* チャネルにサブスクライブする必要があります。

   - **UPDATE STATUS** が **Updates Available** ではない場合、クラスターをアップグレードすることはできません。
   - **DESIRED VERSION** は、クラスターが実行されているか、または更新されるクラスターのバージョンを示します。

3. **Updates Available**をクリックし、利用可能な最新バージョンを選択し、**Update**をクリックします。**UPDATE STATUS** は **Updating** に切り替わり、**Cluster Operators** タブで Operator のアップグレードの進捗を確認できます。
4. OpenShift Container Platform 4.2 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。OpenShift CLI（**oc**）を必要とする以下のコマンドを使用してこれを実行できます。

```
$ for I in $(oc get ns -o jsonpath='{range .items[*]} {.metadata.name}{"\n"} {end}'); \
    do oc delete pods --all -n $I; \
    sleep 1; \
    done
```

注記

OpenShift Container Platform 4.3.5 の時点で、サービス CA は 自動的にローテーションされるため、すべての Pod を再起動する必要があります。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。

5. 更新が完了し、Cluster Version Operator が利用可能な更新を更新したら、追加の更新が現在のチャネルで利用可能かどうかを確認します。

   - 更新が利用可能な場合は、更新ができなくなるまで、現在のチャネルでの更新を継続します。
   - 利用可能な更新がない場合は、**CHANNEL** を次のマイナーバージョンの stable-* または fast-* チャネルに切り替え、そのチャネルで必要なバージョンに更新します。

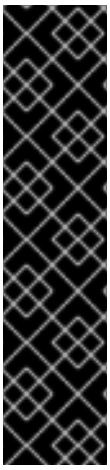
   必要なバージョンに達するまで、いくつかの中間更新を実行する必要がある場合があります。

第2章 WEB コンソールからのマイナーバージョン内でのクラスターの更新

Web コンソールを使用して、OpenShift Container Platform クラスターの更新またはアップグレードを実行できます。

2.1. 前提条件

- **admin** 権限を持つユーザーとしてクラスターにアクセスできること。「Using RBAC to define and apply permissions」を参照してください。
- アップグレードが失敗し、クラスターを直前の状態に復元する必要がある場合に最新の etcd バックアップがあること。



重要

OpenShift Container Platform 4.3.3 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。

これは、OpenShift Container Platform 4.3.5 の時点でサービス CA が自動的にローテーションされるためです。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

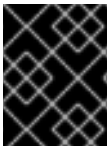
この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。

2.2. OPENSHIFT CONTAINER PLATFORM の更新サービスについて

OpenShift Container Platform の更新サービスとは、OpenShift Container Platform と Red Hat Enterprise Linux CoreOS (RHCOS) の両方に OTA(over-the-air) 更新を提供するホスト型サービスです。コンポーネント Operator のグラフ、または **頂点** とそれらを結ぶ **辺** を含む図表が提示されます。グラフの辺は、どのバージョンであれば安全に更新できるかを示します。 頂点は、管理されたクラスターコンポーネントの想定された状態を特定する更新のペイロードです。

クラスター内の Cluster Version Operator (CVO) は、OpenShift Container Platform の更新サービスをチェックして、グラフの現在のコンポーネントバージョンとグラフの情報に基づき、有効な更新および更新パスを確認します。ユーザーが更新をリクエストすると、OpenShift Container Platform CVO はその更新のリリースイメージを使ってクラスターをアップグレードします。リリースアーティファクトは、コンテナーイメージとして Quay でホストされます。

OpenShift Container Platform 更新サービスが互換性のある更新のみを提供できるようにするために、自動化を支援するリリース検証 Pipeline が使用されます。それぞれのリリースアーティファクトについて、他のコンポーネントパッケージだけでなくサポートされているクラウドプラットフォームおよびシステムアーキテクチャーとの互換性の有無が検証されます。Pipeline がリリースの適合性を確認した後に、OpenShift Container Platform 更新サービスは更新が利用可能であることを通知します。



重要

更新サービスが有効な更新をすべて表示するために、更新サービスが表示しないバージョンへの更新を強制することはできません。

連続更新モードでは、2 つのコントローラーが実行されます。1つのコントローラーはペイロードマニフェストを絶えず更新し、それらをクラスターに適用し、Operator が利用可能か、アップグレード中か、または失敗しているかに応じて Operator の制御されたロールアウトのステータスを出力します。2 つ目のコントローラーは OpenShift Container Platform 更新サービスをポーリングして、更新が利用可能かどうかを判別します。

重要

クラスターを以前のバージョンに戻すこと、つまりロールバックはサポートされていません。サポートされているのは、新規バージョンへのアップグレードのみです。

アップグレードプロセスで、Machine Config Operator (MCO) は新規設定をクラスターマシンに適用します。これは、マシン設定プールの **maxUnavailable** フィールドによって指定されるノードの数を分離し、それらを利用不可としてマークします。デフォルトで、この値は **1** に設定されます。次に、新しい設定を適用して、マシンを再起動します。Red Hat Enterprise Linux (RHEL) マシンをワーカーとして使用する場合、まず OpenShift API をそれらのマシンで更新する必要があるため、MCO はそれらのマシンで kubelet を更新しません。新規バージョンの仕様は古い kubelet に適用されるため、RHEL マシンを **Ready** 状態に戻すことができません。マシンが利用可能になるまで更新を完了することはできません。ただし、利用不可のノードの最大数は、その数のマシンがサービス停止状態のマシンとして分離されても通常のクラスター操作が継続できるようにするために設定されます。

追加リソース

- 「管理外の Operator のサポートポリシー」

2.3. OPENSHIFT CONTAINER PLATFORM アップグレードチャネルおよびリリース

OpenShift Container Platform 4.1 で、Red Hat はクラスターのアップグレードの適切なリリースバージョンを推奨するためにチャネルという概念を導入しました。アップグレードのペースを制御することで、これらのアップグレードチャネルからアップグレードストラテジーを選択することができます。アップグレードチャネルは OpenShift Container Platform のマイナーバージョンに関連付けられます。たとえば、OpenShift Container Platform 4.3 アップグレードチャネルには 4.4 リリースへのアップグレードが含まれることはありません。このストラテジーにより、管理者は OpenShift Container Platform の次のマイナーバージョンへのアップグレードに関して明確な決定を行うことができます。アップグレードチャネルはリリースの選択のみを制御し、インストールするクラスターのバージョンには影響を与えません。 OpenShift Container Platform の特定のバージョンの **openshift-install** バイナリーファイルは常に該当バージョンをインストールします。

OpenShift Container Platform 4.3 は以下のアップグレードチャネルを提供します。

- **candidate-4.3**
- **fast-4.3**
- **stable-4.3**

candidate-4.3 チャネル

candidate-4.3 チャネルには、z-stream (4.3.z) リリースの候補となるビルドが含まれます。リリース候補には、製品のすべての機能が含まれますが、それらがサポートされる訳ではありません。リリース候補を使用して機能の受け入れテストを実行し、OpenShift Container Platform の次のバージョンへの対応を支援します。リリース候補は、名前に **-rc** を含まないものを含め、候補チャネルで利用可能なビルドを指します。候補チャネルでバージョンが利用可能になると、さらに品質のチェックが行われます。品質基準を満たす場合には、これは **fast-4.3** または **stable-4.3** チャネルにプロモートされます。このス

トラテジーにより、特定のリリースが **candidate-4.3** チャネルと **fast-4.3** または **stable-4.3** チャネルの両方で利用可能な場合、そのリリースは Red Hat でサポートされるバージョンということになります。**candidate-4.3** チャネルには、いずれのチャネルでも推奨されている更新のないリリースバージョンを含めることができます。

candidate-4.3 チャネルを使用して、OpenShift Container Platform の直前のマイナーバージョンからアップグレードできます。

注記

リリース候補は夜間ビルドとは異なります。夜間ビルドは各種機能への早期アクセスのために利用できますが、夜間ビルドへの/からの更新は推奨されておらず、サポートもされていません。夜間ビルドはいずれのアップグレードチャネルでも利用できません。ビルドについての詳細は、OpenShift Container Platform の リリースのステータス を参照できます。

fast-4.3 チャネル

fast-4.3 チャネルは、Red Hat が一般公開リリースとして指定のバージョンを宣言するとすぐに新しいバージョンで更新されます。そのため、これらのリリースは完全にサポートされ、実稼働用の品質があり、これらのリリースのプロモート元の **candidate-4.3** チャネルのリリース候補として利用可能であった間のパフォーマンスにも問題がなかったリリースです。リリースは **fast-4.3** チャネルに表示されてからしばらくすると、**stable-4.3** チャネルに追加されます。リリースは **fast-4.3** チャネルに表示される前に、**stable-4.3** チャネルに表示されることはありません。

fast-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

stable-4.3 チャネル

fast-4.3 チャネルにはエラータの公開後すぐにリリースが組み込まれ、リリースの **stable-4.3** チャネルへの追加は遅延します。この期間中、接続環境のカスタマープログラム(Connected Customer Program) に関わる Red Hat SRE チーム、Red Hat サポートサービス、および実稼働前および実稼働環境からリリースの安定性についてのデータが収集されます。

stable-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

アップグレードバージョンパス

OpenShift Container Platform では、インストールされた OpenShift Container Platform のバージョンと、次のリリースにアクセスするために選択したチャネル内のパスの確認を可能にするアップグレード推奨サービスが提供されます。**fast-4.3** チャネルでは以下を確認できます。

- 4.3.0
- 4.3.1
- 4.3.3
- 4.3.4

このサービスは、テスト済みの重大な問題のないアップグレードのみを推奨します。クラスターが 4.3.1 にあり、OpenShift Container Platform が 4.3.4 を提案している場合、4.3.1 から 4.3.4 に更新しても問題がありません。パッチの連続する番号のみに依存しないようにしてください。たとえば、この例では 4.3.2 はチャネルで利用可能な状態ではなく、これまで利用可能になったことがありません。更新サービスは、既知の脆弱性を含む OpenShift Container Platform のバージョンへの更新を提案しません。

更新の安定性は、チャネルによって異なります。**candidate-4.3** チャネルに更新についての推奨がある

からといって、その更新が必ずしもサポートされる訳ではありません。つまり、更新について深刻な問題がまだ検出されていないものの、この更新の安定性についての提案を導くようなトラフィックの安定性はとくに確認されていない可能性があります。**fast-4.3** または **stable-4.3** チャネルに更新の推奨がある場合は、更新がそれぞれのチャネルにある限り完全にサポートされることを示します。リリースがチャネルから削除されることは決してありませんが、深刻な問題を示す更新の推奨はすべてのチャネルから削除されます。更新の推奨が削除された後に開始された更新はサポートされない可能性があります。

Red Hat は最終的には、**fast-4.3** または **stable-4.3** チャネルのサポートされるリリースから 4.3.z の最新リリースへのサポートされる更新パスを提供します。ただし、問題のあるリリースからの安全なパスが構築され、検証される間に遅延が生じる可能性があります。

高速かつ安定したチャネルの使用およびストラテジー
fast-4.3 および **stable-4.3** チャネルでは、一般公開リリースが利用可能になり次第これを受信するか、または Red Hat がそれらの更新のロールアウトを制御するようにするかを選択することができます。問題がロールアウト時またはロールアウト後に検出される場合、該当バージョンへのアップグレードは **fast-4.3** および **stable-4.3** チャネルの両方でブロックされ、新たに推奨されるアップグレード先の新規バージョンが導入される可能性があります。

fast-4.3 チャネルで実稼働前のシステムを設定し、**stable-4.3** チャネルで実稼働システムを設定してから Red Hat の接続環境のカスタマープログラム(Connected Customer Program) に参加することで、お客様のプロセスを改善することができます。Red Hat はこのプログラムを使用して、ご使用の特定のハードウェアおよびソフトウェア設定に対する更新の影響の有無を確認します。今後のリリースでは、更新が **fast-4.3** から **stable-4.3** チャネルに移行するペースが改善されるか、変更される可能性があります。

ネットワークが制限された環境のクラスター
OpenShift Container Platform クラスターのコンテナーイメージを独自に管理する場合には、製品リリースに関連する Red Hat エラータを確認し、アップグレードへの影響に関するコメントに留意する必要があります。アップグレード時に、インターフェースにこれらのバージョン間の切り替えについての警告が表示される場合があります。そのため、これらの警告を無視するかどうかを決める前に適切なバージョンを選択していることを確認する必要があります。

チャネル間の切り替え
stable-4.3 チャネルから **fast-4.3** チャネルに切り換える場合も、クラスターは引き続きサポートされます。**candidate-4.3** チャネルに常に切り替えることはできますが、そのチャネルの一部のリリースはサポートされないリリース候補である可能性があります。現在のリリースが一般利用公開リリースの場合、**candidate-4.3** チャネルから **fast-4.3** チャネルに切り換えることができます。**fast-4.3** チャネルから **stable-4.3** チャネルに常に切り換えることができますが、現在のリリースが **fast-4.3** に最近プロモートされた場合、リリースが **stable-4.3** にプロモートされるまでに最長 1 日分の遅延が生じる可能性があります。現在のリリースを含まないチャネルに変更すると、アラートが表示され、更新は推奨されません。ただし、元のチャネルにいつでも安全に戻ることができます。

2.4. WEB コンソールを使用したクラスターの更新

更新が利用可能な場合、Web コンソールからクラスターを更新できます。

利用可能な OpenShift Container Platform アドバイザリーおよび更新については、カスタマーポータルの エラータのセクションを参照してください。

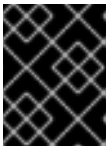
前提条件

- **admin** 権限を持つユーザーとして Web コンソールにアクセスできること。

手順

1. Web コンソールから、**Administration** > **Cluster Settings** をクリックし、 **Overview** タブの内容を確認します。

2. 実稼働クラスターの場合、**CHANNEL** が **stable-4.3** などの現在のマイナーバージョンの、更新する必要のあるバージョンの正しいチャネルに設定されていることを確認します。



重要

実稼働クラスターの場合、stable-* または fast-* チャネルにサブスクライブする必要があります。

- **UPDATE STATUS** が **Updates Available** ではない場合、クラスターをアップグレードすることはできません。

- **DESIRED VERSION** は、クラスターが実行されているか、または更新されるクラスターのバージョンを示します。

3. **Updates Available**をクリックし、更新するバージョンで、利用可能な最新バージョンを選択し、**Update** をクリックします。**UPDATE STATUS** は **Updating** に切り替わり、**Cluster Operators** タブで Operator のアップグレードの進捗を確認できます。

4. OpenShift Container Platform 4.2 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。OpenShift CLI（**oc**）を必要とする以下のコマンドを使用してこれを実行できます。

```
$ for I in $(oc get ns -o jsonpath='{range .items[*]} {.metadata.name}{"\n"} {end}'); \
    do oc delete pods --all -n $I; \
    sleep 1; \
    done
```

注記

OpenShift Container Platform 4.3.5 の時点で、サービス CA は 自動的にローテーションされるため、すべての Pod を再起動する必要があります。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。

5. 更新が完了し、Cluster Version Operator が利用可能な更新を更新したら、追加の更新が現在のチャネルで利用可能かどうかを確認します。

- 更新が利用可能な場合は、更新ができなくなるまで、現在のチャネルでの更新を継続します。

- 利用可能な更新がない場合は、**CHANNEL** を次のマイナーバージョンの stable-* または fast-* チャネルに切り替え、そのチャネルで必要なバージョンに更新します。

必要なバージョンに達するまで、いくつかの中間更新を実行する必要がある場合があります。

第3章 CLI の使用によるマイナーバージョン内でのクラスターの更新

OpenShift CLI (**oc**) を使用して OpenShift Container Platform クラスターをマイナーバージョン内で更新するか、またはアップグレードすることができます。

3.1. 前提条件

- **admin** 権限を持つユーザーとしてクラスターにアクセスできること。「Using RBAC to define and apply permissions」を参照してください。
- アップグレードが失敗し、クラスターを直前の状態に復元する必要がある場合に最新の etcd バックアップがあること。

重要

OpenShift Container Platform 4.3.3 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。

これは、OpenShift Container Platform 4.3.5 の時点でサービス CA が自動的にローテーションされるためです。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。

3.2. OPENSHIFT CONTAINER PLATFORM の更新サービスについて

OpenShift Container Platform の更新サービスとは、OpenShift Container Platform と Red Hat Enterprise Linux CoreOS (RHCOS) の両方に OTA(over-the-air) 更新を提供するホスト型サービスです。コンポーネント Operator のグラフ、または **頂点** とそれらを結ぶ **辺** を含む図表が提示されます。グラフの辺は、どのバージョンであれば安全に更新できるかを示します。 頂点は、管理されたクラスターコンポーネントの想定された状態を特定する更新のペイロードです。

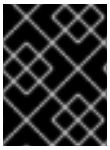
クラスター内の Cluster Version Operator (CVO) は、OpenShift Container Platform の更新サービスをチェックして、グラフの現在のコンポーネントバージョンとグラフの情報に基づき、有効な更新および更新パスを確認します。ユーザーが更新をリクエストすると、OpenShift Container Platform CVO はその更新のリリースイメージを使ってクラスターをアップグレードします。リリースアーティファクトは、コンテナーイメージとして Quay でホストされます。

OpenShift Container Platform 更新サービスが互換性のある更新のみを提供できるようにするために、自動化を支援するリリース検証 Pipeline が使用されます。それぞれのリリースアーティファクトについて、他のコンポーネントパッケージだけでなくサポートされているクラウドプラットフォームおよびシステムアーキテクチャーとの互換性の有無が検証されます。Pipeline がリリースの適合性を確認した後に、OpenShift Container Platform 更新サービスは更新が利用可能であることを通知します。

重要

更新サービスが有効な更新をすべて表示するために、更新サービスが表示しないバージョンへの更新を強制することはできません。

連続更新モードでは、2 つのコントローラーが実行されます。1つのコントローラーはペイロードマニフェストを絶えず更新し、それらをクラスターに適用し、Operator が利用可能か、アップグレード中か、または失敗しているかに応じて Operator の制御されたロールアウトのステータスを出力します。2 つ目のコントローラーは OpenShift Container Platform 更新サービスをポーリングして、更新が利用可能かどうかを判別します。



重要

クラスターを以前のバージョンに戻すこと、つまりロールバックはサポートされていません。サポートされているのは、新規バージョンへのアップグレードのみです。

アップグレードプロセスで、Machine Config Operator (MCO) は新規設定をクラスターマシンに適用します。これは、マシン設定プールの **maxUnavailable** フィールドによって指定されるノードの数を分離し、それらを利用不可としてマークします。デフォルトで、この値は **1** に設定されます。次に、新しい設定を適用して、マシンを再起動します。Red Hat Enterprise Linux (RHEL) マシンをワーカーとして使用する場合、まず OpenShift API をそれらのマシンで更新する必要があるため、MCO はそれらのマシンで kubelet を更新しません。新規バージョンの仕様は古い kubelet に適用されるため、RHEL マシンを **Ready** 状態に戻すことができません。マシンが利用可能になるまで更新を完了することはできません。ただし、利用不可のノードの最大数は、その数のマシンがサービス停止状態のマシンとして分離されても通常のクラスター操作が継続できるようにするために設定されます。

追加リソース

- 「管理外の Operator のサポートポリシー」

3.3. OPENSHIFT CONTAINER PLATFORM アップグレードチャネルおよびリリース

OpenShift Container Platform 4.1 で、Red Hat はクラスターのアップグレードの適切なリリースバージョンを推奨するためにチャネルという概念を導入しました。アップグレードのペースを制御することで、これらのアップグレードチャネルからアップグレードストラテジーを選択することができます。アップグレードチャネルは OpenShift Container Platform のマイナーバージョンに関連付けられます。たとえば、OpenShift Container Platform 4.3 アップグレードチャネルには 4.4 リリースへのアップグレードが含まれることはありません。このストラテジーにより、管理者は OpenShift Container Platform の次のマイナーバージョンへのアップグレードに関して明確な決定を行うことができます。アップグレードチャネルはリリースの選択のみを制御し、インストールするクラスターのバージョンには影響を与えません。 OpenShift Container Platform の特定のバージョンの **openshift-install** バイナリーファイルは常に該当バージョンをインストールします。

OpenShift Container Platform 4.3 は以下のアップグレードチャネルを提供します。

- **candidate-4.3**
- **fast-4.3**
- **stable-4.3**

candidate-4.3 チャネル

candidate-4.3 チャネルには、z-stream (4.3.z) リリースの候補となるビルドが含まれます。リリース候補には、製品のすべての機能が含まれますが、それらがサポートされる訳ではありません。リリース候補を使用して機能の受け入れテストを実行し、OpenShift Container Platform の次のバージョンへの対応を支援します。リリース候補は、名前に **-rc** を含まないものを含め、候補チャネルで利用可能なビルドを指します。候補チャネルでバージョンが利用可能になると、さらに品質のチェックが行われます。品質基準を満たす場合には、これは **fast-4.3** または **stable-4.3** チャネルにプロモートされます。このス

トラテジーにより、特定のリリースが **candidate-4.3** チャネルと **fast-4.3** または **stable-4.3** チャネルの両方で利用可能な場合、そのリリースは Red Hat でサポートされるバージョンということになります。**candidate-4.3** チャネルには、いずれのチャネルでも推奨されている更新のないリリースバージョンを含めることができます。

candidate-4.3 チャネルを使用して、OpenShift Container Platform の直前のマイナーバージョンからアップグレードできます。

注記

リリース候補は夜間ビルドとは異なります。夜間ビルドは各種機能への早期アクセスのために利用できますが、夜間ビルドへの/からの更新は推奨されておらず、サポートもされていません。夜間ビルドはいずれのアップグレードチャネルでも利用できません。ビルドについての詳細は、OpenShift Container Platform の リリースのステータス を参照できます。

fast-4.3 チャネル

fast-4.3 チャネルは、Red Hat が一般公開リリースとして指定のバージョンを宣言するとすぐに新しいバージョンで更新されます。そのため、これらのリリースは完全にサポートされ、実稼働用の品質があり、これらのリリースのプロモート元の **candidate-4.3** チャネルのリリース候補として利用可能であった間のパフォーマンスにも問題がなかったリリースです。リリースは **fast-4.3** チャネルに表示されてからしばらくすると、**stable-4.3** チャネルに追加されます。リリースは **fast-4.3** チャネルに表示される前に、**stable-4.3** チャネルに表示されることはありません。

fast-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

stable-4.3 チャネル

fast-4.3 チャネルにはエラータの公開後すぐにリリースが組み込まれ、リリースの **stable-4.3** チャネルへの追加は遅延します。この期間中、接続環境のカスタマープログラム(Connected Customer Program) に関わる Red Hat SRE チーム、Red Hat サポートサービス、および実稼働前および実稼働環境からリリースの安定性についてのデータが収集されます。

stable-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

アップグレードバージョンパス

OpenShift Container Platform では、インストールされた OpenShift Container Platform のバージョンと、次のリリースにアクセスするために選択したチャネル内のパスの確認を可能にするアップグレード推奨サービスが提供されます。**fast-4.3** チャネルでは以下を確認できます。

- 4.3.0
- 4.3.1
- 4.3.3
- 4.3.4

このサービスは、テスト済みの重大な問題のないアップグレードのみを推奨します。クラスターが 4.3.1 にあり、OpenShift Container Platform が 4.3.4 を提案している場合、4.3.1 から 4.3.4 に更新しても問題がありません。パッチの連続する番号のみに依存しないようにしてください。たとえば、この例では 4.3.2 はチャネルで利用可能な状態ではなく、これまで利用可能になったことがありません。更新サービスは、既知の脆弱性を含む OpenShift Container Platform のバージョンへの更新を提案しません。

更新の安定性は、チャネルによって異なります。**candidate-4.3** チャネルに更新についての推奨がある

からといって、その更新が必ずしもサポートされる訳ではありません。つまり、更新について深刻な問題がまだ検出されていないものの、この更新の安定性についての提案を導くようなトラフィックの安定性はとくに確認されていない可能性があります。**fast-4.3** または **stable-4.3** チャネルに更新の推奨がある場合は、更新がそれぞれのチャネルにある限り完全にサポートされることを示します。リリースがチャネルから削除されることは決してありませんが、深刻な問題を示す更新の推奨はすべてのチャネルから削除されます。更新の推奨が削除された後に開始された更新はサポートされない可能性があります。

Red Hat は最終的には、**fast-4.3** または **stable-4.3** チャネルのサポートされるリリースから 4.3.z の最新リリースへのサポートされる更新パスを提供します。ただし、問題のあるリリースからの安全なパスが構築され、検証される間に遅延が生じる可能性があります。

高速かつ安定したチャネルの使用およびストラテジー
fast-4.3 および **stable-4.3** チャネルでは、一般公開リリースが利用可能になり次第これを受信するか、または Red Hat がそれらの更新のロールアウトを制御するようにするかを選択することができます。問題がロールアウト時またはロールアウト後に検出される場合、該当バージョンへのアップグレードは **fast-4.3** および **stable-4.3** チャネルの両方でブロックされ、新たに推奨されるアップグレード先の新規バージョンが導入される可能性があります。

fast-4.3 チャネルで実稼働前のシステムを設定し、**stable-4.3** チャネルで実稼働システムを設定してから Red Hat の接続環境のカスタマープログラム(Connected Customer Program) に参加することで、お客様のプロセスを改善することができます。Red Hat はこのプログラムを使用して、ご使用の特定のハードウェアおよびソフトウェア設定に対する更新の影響の有無を確認します。今後のリリースでは、更新が **fast-4.3** から **stable-4.3** チャネルに移行するペースが改善されるか、変更される可能性があります。

ネットワークが制限された環境のクラスター
OpenShift Container Platform クラスターのコンテナーイメージを独自に管理する場合には、製品リリースに関連する Red Hat エラータを確認し、アップグレードへの影響に関するコメントに留意する必要があります。アップグレード時に、インターフェースにこれらのバージョン間の切り替えについての警告が表示される場合があります。そのため、これらの警告を無視するかどうかを決める前に適切なバージョンを選択していることを確認する必要があります。

チャネル間の切り替え
stable-4.3 チャネルから **fast-4.3** チャネルに切り換える場合も、クラスターは引き続きサポートされます。**candidate-4.3** チャネルに常に切り替えることはできますが、そのチャネルの一部のリリースはサポートされないリリース候補である可能性があります。現在のリリースが一般利用公開リリースの場合、**candidate-4.3** チャネルから **fast-4.3** チャネルに切り換えることができます。**fast-4.3** チャネルから **stable-4.3** チャネルに常に切り換えることができますが、現在のリリースが **fast-4.3** に最近プロモートされた場合、リリースが **stable-4.3** にプロモートされるまでに最長 1 日分の遅延が生じる可能性があります。現在のリリースを含まないチャネルに変更すると、アラートが表示され、更新は推奨されません。ただし、元のチャネルにいつでも安全に戻ることができます。

3.4. CLI を使用したクラスターの更新

更新が利用可能な場合、OpenShift CLI (**oc**) を使用してクラスターを更新できます。

利用可能な OpenShift Container Platform アドバイザリーおよび更新については、カスタマーポータルの エラータのセクションを参照してください。

前提条件

- お使いの更新バージョンのバージョンに一致する OpenShift CLI (**oc**) をインストールします。
- **cluster-admin** 権限を持つユーザーとしてクラスターにログインします。

- **jq** パッケージをインストールします。

手順

1. クラスターが利用可能であることを確認します。

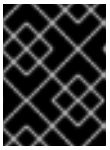
```
$ oc get clusterversion

NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version   4.3.0     True        False         158m    Cluster version is 4.3.0
```

2. 現在の更新チャネル情報を確認し、チャネルが **stable-4.3**に設定されていることを確認します。

```
$ oc get clusterversion -o json|jq ".items[0].spec"

{
  "channel": "stable-4.3",
  "clusterID": "990f7ab8-109b-4c95-8480-2bd1deec55ff",
  "upstream": "https://api.openshift.com/api/upgrades_info/v1/graph"
}
```



重要

実稼働クラスターの場合、**stable-*** チャネルにサブスクライブする必要があります。

3. 利用可能な更新を確認し、適用する必要のある更新のバージョン番号をメモします。

```
$ oc adm upgrade

Cluster version is 4.1.0

Updates:

VERSION IMAGE
4.1.2   quay.io/openshift-release-dev/ocp-release@sha256:9c5f0df8b192a0d7b46cd5f6a4da2289c155fd5302dec7954f8f06c878160b8b
```

4. 更新を適用します。

- 最新バージョンに更新するには、以下を実行します。

```
$ oc adm upgrade --to-latest=true (1)
```

- 特定のバージョンに更新するには、以下を実行します。

```
$ oc adm upgrade --to=<version> (1)
```

(1) **<version>** は、直前のコマンドの出力から得られる更新バージョンです。

5. クラスターバージョン Operator を確認します。

```
$ oc get clusterversion -o json|jq ".items[0].spec"

{
  "channel": "stable-4.3",
  "clusterID": "990f7ab8-109b-4c95-8480-2bd1deec55ff",
  "desiredUpdate": {
    "force": false,
    "image": "quay.io/openshift-release-dev/ocp-
release@sha256:9c5f0df8b192a0d7b46cd5f6a4da2289c155fd5302dec7954f8f06c878160b8b",

    "version": "4.3.1" (1)
  },
  "upstream": "https://api.openshift.com/api/upgrades_info/v1/graph"
}
```

(1) **desiredUpdate** スタンザの **version** 番号が指定した値と一致する場合、更新は進行中です。

6. クラスターバージョン履歴で、更新のステータスをモニターします。すべてのオブジェクトの更新が終了するまでに時間がかかる可能性があります。

```
$ oc get clusterversion -o json|jq ".items[0].status.history"

[
  {
    "completionTime": null,
    "image": "quay.io/openshift-release-dev/ocp-
release@sha256:9c5f0df8b192a0d7b46cd5f6a4da2289c155fd5302dec7954f8f06c878160b8b",

    "startedTime": "2019-06-19T20:30:50Z",
    "state": "Partial",
    "verified": true,
    "version": "4.1.2"
  },
  {
    "completionTime": "2019-06-19T20:30:50Z",
    "image": "quay.io/openshift-release-dev/ocp-
release@sha256:b8307ac0f3ec4ac86c3f3b52846425205022da52c16f56ec31cbe428501001d6
",
    "startedTime": "2019-06-19T17:38:10Z",
    "state": "Completed",
    "verified": false,
    "version": "4.1.0"
  }
]
```

履歴には、クラスターに適用された最新バージョンの一覧が含まれます。この値は、CVO が更新を適用する際に更新されます。この一覧は日付順に表示され、最新の更新は一覧の先頭に表示されます。履歴の更新には、ロールアウトが完了した場合には **Completed** と表示され、更新が失敗したか、または完了しなかった場合には **Partial** と表示されます。

重要

アップグレードに失敗する場合、Operator は停止し、失敗しているコンポーネントのステータスを報告します。クラスターの以前のバージョンへのロールバックはサポートされていません。アップグレードできない場合は、Red Hat サポートにお問い合わせください。

7. 更新が完了したら、クラスターのバージョンが新たなバージョンに更新されていることを確認できます。

```
$ oc get clusterversion

NAME      VERSION     AVAILABLE   PROGRESSING   SINCE     STATUS
version   4.1.2       True        False         2m        Cluster version is 4.1.2
```

8. OpenShift Container Platform 4.3.3 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。これは以下のコマンドで確認できます。

```
$ for I in $(oc get ns -o jsonpath='{range .items[*]} {.metadata.name}{"\n"} {end}'); \
    do oc delete pods --all -n $I; \
    sleep 1; \
    done
```

注記

OpenShift Container Platform 4.3.5 の時点で、サービス CA は 自動的にローテーションされるため、すべての Pod を再起動する必要があります。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。

第4章 RHEL コンピュートマシンを含むクラスターの更新

OpenShift Container Platform クラスターの更新またはアップグレードを実行できます。クラスターに Red Hat Enterprise Linux (RHEL) マシンが含まれる場合は、それらのマシンを更新するために追加の手順を実行する必要があります。

4.1. 前提条件

- **admin** 権限を持つユーザーとしてクラスターにアクセスできること。「Using RBAC to define and apply permissions」を参照してください。
- アップグレードが失敗し、クラスターを直前の状態に復元する必要がある場合に最新の etcd バックアップがあること。

重要

OpenShift Container Platform 4.2 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。

これは、OpenShift Container Platform 4.3.5 の時点でサービス CA が自動的にローテーションされるためです。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。

4.2. OPENSHIFT CONTAINER PLATFORM の更新サービスについて

OpenShift Container Platform の更新サービスとは、OpenShift Container Platform と Red Hat Enterprise Linux CoreOS (RHCOS) の両方に OTA(over-the-air) 更新を提供するホスト型サービスです。コンポーネント Operator のグラフ、または **頂点** とそれらを結ぶ **辺** を含む図表が提示されます。グラフの辺は、どのバージョンであれば安全に更新できるかを示します。 頂点は、管理されたクラスターコンポーネントの想定された状態を特定する更新のペイロードです。

クラスター内の Cluster Version Operator (CVO) は、OpenShift Container Platform の更新サービスをチェックして、グラフの現在のコンポーネントバージョンとグラフの情報に基づき、有効な更新および更新パスを確認します。ユーザーが更新をリクエストすると、OpenShift Container Platform CVO はその更新のリリースイメージを使ってクラスターをアップグレードします。リリースアーティファクトは、コンテナーイメージとして Quay でホストされます。

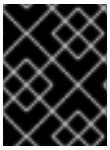
OpenShift Container Platform 更新サービスが互換性のある更新のみを提供できるようにするために、自動化を支援するリリース検証 Pipeline が使用されます。それぞれのリリースアーティファクトについて、他のコンポーネントパッケージだけでなくサポートされているクラウドプラットフォームおよびシステムアーキテクチャーとの互換性の有無が検証されます。Pipeline がリリースの適合性を確認した後に、OpenShift Container Platform 更新サービスは更新が利用可能であることを通知します。

重要

更新サービスが有効な更新をすべて表示するために、更新サービスが表示しないバージョンへの更新を強制することはできません。

連続更新モードでは、2 つのコントローラーが実行されます。1 つのコントローラーはペイロードマニ

フェストを絶えず更新し、それらをクラスターに適用し、Operator が利用可能か、アップグレード中か、または失敗しているかに応じて Operator の制御されたロールアウトのステータスを出力します。2 つ目のコントローラーは OpenShift Container Platform 更新サービスをポーリングして、更新が利用可能かどうかを判別します。



重要

クラスターを以前のバージョンに戻すこと、つまりロールバックはサポートされていません。サポートされているのは、新規バージョンへのアップグレードのみです。

アップグレードプロセスで、Machine Config Operator (MCO) は新規設定をクラスターマシンに適用します。これは、マシン設定プールの **maxUnavailable** フィールドによって指定されるノードの数を分離し、それらを利用不可としてマークします。デフォルトで、この値は **1** に設定されます。次に、新しい設定を適用して、マシンを再起動します。Red Hat Enterprise Linux (RHEL) マシンをワーカーとして使用する場合、まず OpenShift API をそれらのマシンで更新する必要があるため、MCO はそれらのマシンで kubelet を更新しません。新規バージョンの仕様は古い kubelet に適用されるため、RHEL マシンを **Ready** 状態に戻すことができません。マシンが利用可能になるまで更新を完了することはできません。ただし、利用不可のノードの最大数は、その数のマシンがサービス停止状態のマシンとして分離されても通常のクラスター操作が継続できるようにするために設定されます。

追加リソース

- 「管理外の Operator のサポートポリシー」

4.3. OPENSHIFT CONTAINER PLATFORM アップグレードチャネルおよびリリース

OpenShift Container Platform 4.1 で、Red Hat はクラスターのアップグレードの適切なリリースバージョンを推奨するためにチャネルという概念を導入しました。アップグレードのペースを制御することで、これらのアップグレードチャネルからアップグレードストラテジーを選択することができます。アップグレードチャネルは OpenShift Container Platform のマイナーバージョンに関連付けられます。たとえば、OpenShift Container Platform 4.3 アップグレードチャネルには 4.4 リリースへのアップグレードが含まれることはありません。このストラテジーにより、管理者は OpenShift Container Platform の次のマイナーバージョンへのアップグレードに関して明確な決定を行うことができます。アップグレードチャネルはリリースの選択のみを制御し、インストールするクラスターのバージョンには影響を与えません。 OpenShift Container Platform の特定のバージョンの **openshift-install** バイナリーファイルは常に該当バージョンをインストールします。

OpenShift Container Platform 4.3 は以下のアップグレードチャネルを提供します。

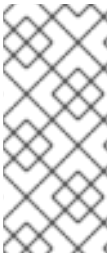
- **candidate-4.3**
- **fast-4.3**
- **stable-4.3**

candidate-4.3 チャネル

candidate-4.3 チャネルには、z-stream (4.3.z) リリースの候補となるビルドが含まれます。リリース候補には、製品のすべての機能が含まれますが、それらがサポートされる訳ではありません。リリース候補を使用して機能の受け入れテストを実行し、OpenShift Container Platform の次のバージョンへの対応を支援します。リリース候補は、名前に **-rc** を含まないものを含め、候補チャネルで利用可能なビルドを指します。候補チャネルでバージョンが利用可能になると、さらに品質のチェックが行われます。品質基準を満たす場合には、これは **fast-4.3** または **stable-4.3** チャネルにプロモートされます。このストラテジーにより、特定のリリースが **candidate-4.3** チャネルと **fast-4.3** または **stable-4.3** チャネルの

両方で利用可能な場合、そのリリースは Red Hat でサポートされるバージョンということになります。**candidate-4.3** チャネルには、いずれのチャネルでも推奨されている更新のないリリースバージョンを含めることができます。

candidate-4.3 チャネルを使用して、OpenShift Container Platform の直前のマイナーバージョンからアップグレードできます。



注記

リリース候補は夜間ビルドとは異なります。夜間ビルドは各種機能への早期アクセスのために利用できますが、夜間ビルドへの/からの更新は推奨されておらず、サポートもされていません。夜間ビルドはいずれのアップグレードチャネルでも利用できません。ビルドについての詳細は、OpenShift Container Platform の リリースのステータス を参照できます。

fast-4.3 チャネル

fast-4.3 チャネルは、Red Hat が一般公開リリースとして指定のバージョンを宣言するとすぐに新しいバージョンで更新されます。そのため、これらのリリースは完全にサポートされ、実稼働用の品質があり、これらのリリースのプロモート元の **candidate-4.3** チャネルのリリース候補として利用可能であった間のパフォーマンスにも問題がなかったリリースです。リリースは **fast-4.3** チャネルに表示されてからしばらくすると、**stable-4.3** チャネルに追加されます。リリースは **fast-4.3** チャネルに表示される前に、**stable-4.3** チャネルに表示されることはありません。

fast-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

stable-4.3 チャネル

fast-4.3 チャネルにはエラータの公開後すぐにリリースが組み込まれ、リリースの **stable-4.3** チャネルへの追加は遅延します。この期間中、接続環境のカスタマープログラム(Connected Customer Program) に関わる Red Hat SRE チーム、Red Hat サポートサービス、および実稼働前および実稼働環境からリリースの安定性についてのデータが収集されます。

stable-4.3 チャネルを使用して、OpenShift Container Platform の以前のマイナーバージョンからのアップグレードを実行できます。

アップグレードバージョンパス

OpenShift Container Platform では、インストールされた OpenShift Container Platform のバージョンと、次のリリースにアクセスするために選択したチャネル内のパスの確認を可能にするアップグレード推奨サービスが提供されます。**fast-4.3** チャネルでは以下を確認できます。

- 4.3.0
- 4.3.1
- 4.3.3
- 4.3.4

このサービスは、テスト済みの重大な問題のないアップグレードのみを推奨します。クラスターが 4.3.1 にあり、OpenShift Container Platform が 4.3.4 を提案している場合、4.3.1 から 4.3.4 に更新しても問題がありません。パッチの連続する番号のみに依存しないようにしてください。たとえば、この例では 4.3.2 はチャネルで利用可能な状態ではなく、これまで利用可能になったことがありません。更新サービスは、既知の脆弱性を含む OpenShift Container Platform のバージョンへの更新を提案しません。

更新の安定性は、チャネルによって異なります。**candidate-4.3** チャネルに更新についての推奨があるからといって、その更新が必ずしもサポートされる訳ではありません。つまり、更新について深刻な問

題がまだ検出されていないものの、この更新の安定性についての提案を導くようなトラフィックの安定性はとくに確認されていない可能性があります。**fast-4.3** または **stable-4.3** チャネルに更新の推奨がある場合は、更新がそれぞれのチャネルにある限り完全にサポートされることを示します。リリースがチャネルから削除されることは決してありませんが、深刻な問題を示す更新の推奨はすべてのチャネルから削除されます。更新の推奨が削除された後に開始された更新はサポートされない可能性があります。

Red Hat は最終的には、**fast-4.3** または **stable-4.3** チャネルのサポートされるリリースから 4.3.z の最新リリースへのサポートされる更新パスを提供します。ただし、問題のあるリリースからの安全なパスが構築され、検証される間に遅延が生じる可能性があります。

高速かつ安定したチャネルの使用およびストラテジー
fast-4.3 および **stable-4.3** チャネルでは、一般公開リリースが利用可能になり次第これを受信するか、または Red Hat がそれらの更新のロールアウトを制御するようにするかを選択することができます。問題がロールアウト時またはロールアウト後に検出される場合、該当バージョンへのアップグレードは **fast-4.3** および **stable-4.3** チャネルの両方でブロックされ、新たに推奨されるアップグレード先の新規バージョンが導入される可能性があります。

fast-4.3 チャネルで実稼働前のシステムを設定し、**stable-4.3** チャネルで実稼働システムを設定してから Red Hat の接続環境のカスタマープログラム(Connected Customer Program) に参加することで、お客様のプロセスを改善することができます。Red Hat はこのプログラムを使用して、ご使用の特定のハードウェアおよびソフトウェア設定に対する更新の影響の有無を確認します。今後のリリースでは、更新が **fast-4.3** から **stable-4.3** チャネルに移行するペースが改善されるか、変更される可能性があります。

ネットワークが制限された環境のクラスター
OpenShift Container Platform クラスターのコンテナーイメージを独自に管理する場合には、製品リリースに関連する Red Hat エラータを確認し、アップグレードへの影響に関するコメントに留意する必要があります。アップグレード時に、インターフェースにこれらのバージョン間の切り替えについての警告が表示される場合があります。そのため、これらの警告を無視するかどうかを決める前に適切なバージョンを選択していることを確認する必要があります。

チャネル間の切り替え
stable-4.3 チャネルから **fast-4.3** チャネルに切り換える場合も、クラスターは引き続きサポートされます。**candidate-4.3** チャネルに常に切り替えることはできますが、そのチャネルの一部のリリースはサポートされないリリース候補である可能性があります。現在のリリースが一般利用公開リリースの場合、**candidate-4.3** チャネルから **fast-4.3** チャネルに切り換えることができます。**fast-4.3** チャネルから **stable-4.3** チャネルに常に切り換えることができますが、現在のリリースが **fast-4.3** に最近プロモートされた場合、リリースが **stable-4.3** にプロモートされるまでに最長 1 日分の遅延が生じる可能性があります。現在のリリースを含まないチャネルに変更すると、アラートが表示され、更新は推奨されません。ただし、元のチャネルにいつでも安全に戻ることができます。

4.4. WEB コンソールを使用したクラスターの更新

更新が利用可能な場合、Web コンソールからクラスターを更新できます。

利用可能な OpenShift Container Platform アドバイザリーおよび更新については、カスタマーポータルの エラータのセクションを参照してください。

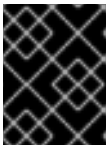
前提条件

- **admin** 権限を持つユーザーとして Web コンソールにアクセスできること。

手順

1. Web コンソールから、**Administration** > **Cluster Settings** をクリックし、 **Overview** タブの内容を確認します。

2. 実稼働クラスターの場合、**CHANNEL** が **stable-4.3** などの現在のマイナーバージョンの、更新する必要のあるバージョンの正しいチャネルに設定されていることを確認します。



重要

実稼働クラスターの場合、stable-* または fast-* チャネルにサブスクライブする必要があります。

- **UPDATE STATUS** が **Updates Available** ではない場合、クラスターをアップグレードすることはできません。

- **DESIRED VERSION** は、クラスターが実行されているか、または更新されるクラスターのバージョンを示します。

3. **Updates Available**をクリックし、更新するバージョンで、利用可能な最新バージョンを選択し、**Update** をクリックします。**UPDATE STATUS** は **Updating** に切り替わり、**Cluster Operators** タブで Operator のアップグレードの進捗を確認できます。

4. OpenShift Container Platform 4.2 以前から本リリースにアップグレードする場合、アップグレードの完了後にすべての Pod を再起動する必要があります。OpenShift CLI（**oc**）を必要とする以下のコマンドを使用してこれを実行できます。

```
$ for I in $(oc get ns -o jsonpath='{range .items[*]} {.metadata.name}{"\n"} {end}'); \
    do oc delete pods --all -n $I; \
    sleep 1; \
    done
```

注記

OpenShift Container Platform 4.3.5 の時点で、サービス CA は 自動的にローテーションされるため、すべての Pod を再起動する必要があります。サービス CA はアップグレード時にローテーションされ、その後再起動して、以前のサービス CA の期限が切れる前にすべてのサービスが新しいサービス CA を使用していることを確認する必要があります。

この 1 回で実行される手動による再起動の後、後続のアップグレードおよびローテーションにより、サービス CA の期限が切れる前に手動の介入なしの再起動が行われます。

5. 更新が完了し、Cluster Version Operator が利用可能な更新を更新したら、追加の更新が現在のチャネルで利用可能かどうかを確認します。

- 更新が利用可能な場合は、更新ができなくなるまで、現在のチャネルでの更新を継続します。

- 利用可能な更新がない場合は、**CHANNEL** を次のマイナーバージョンの stable-* または fast-* チャネルに切り替え、そのチャネルで必要なバージョンに更新します。

必要なバージョンに達するまで、いくつかの中間更新を実行する必要がある場合があります。

注記

Red Hat Enterprise Linux (RHEL) ワーカーマシンを含むクラスターを更新する場合、それらのワーカーは、更新プロセス時に一時的に使用できなくなります。クラスターの更新の終了において各 RHEL マシンがのステートが **NotReady** になる際に、アップグレード Playbook を各 RHEL マシンに対して実行する必要があります。

4.5. (オプション) RHEL マシンで ANSIBLE タスクを実行するためのフックの追加

OpenShift Container Platform の更新時に **フック** を使用し、RHEL コンピュートマシンで Ansible タスクを実行できます。

4.5.1. アップグレード用の Ansible Hook について

OpenShift Container Platform の更新時に **フック** を使用し、特定操作の実行中に Red Hat Enterprise Linux (RHEL) ノードでカスタムタスクを実行できます。フックを使用して、特定の更新タスクの前後に実行するタスクを定義するファイルを指定できます。OpenShift Container Platform クラスターで RHEL コンピュートノードを更新する際に、フックを使用してカスタムインフラストラクチャーを検証したり、変更したりすることができます。

フックが失敗すると操作も失敗するため、フックはべき等性があるか、または複数回実行でき、同じ結果を出せるように設計する必要があります。

フックには以下のような重要な制限があります。まず、フックには定義された、 またはバージョン付けされたインターフェースがありません。フックは内部の **openshift-ansible** 変数を使用できますが、これらの変数は今後の OpenShift Container Platform のリリースで変更されるか、または削除される予定です。次に、フックにはエラー処理機能がないため、フックにエラーが生じると更新プロセスが中止されます。エラーの発生時には、まず問題に対応してからアップグレードを再び開始する必要があります。

4.5.2. Ansible インベントリーファイルでのフックを使用する設定

Red Hat Enterprise Linux (RHEL) コンピュートマシン (ワーカーマシンとしても知られている) の更新時に使用するフックを、**all:vars** セクションの下にある **hosts** インベントリーファイルで定義します。

前提条件

- RHEL コンピュートマシンクラスターの追加に使用したマシンへのアクセスがあること。RHEL マシンを定義する **hosts** Ansible インベントリーファイルにアクセスできる必要があります。

手順

1. フックの設計後に、フック用に Ansible タスクを定義する YAML ファイルを作成します。このファイルは、以下に示すように一連のタスクで構成される必要があり、Playbook にすることはできません。

```
---
# Trivial example forcing an operator to acknowledge the start of an upgrade
# file=/home/user/openshift-ansible/hooks/pre_compute.yml

- name: note the start of a compute machine update
  debug:
```

```
      msg: "Compute machine upgrade of {{ inventory_hostname }} is about to start"

- name: require the user agree to start an upgrade
  pause:
      prompt: "Press Enter to start the compute machine update"
```

2. **hosts** Ansible インベントリーファイルを変更してフックファイルを指定します。フックファイルは、以下に示すように **[all:vars]** セクションのパラメーター値として指定されます。

   インベントリーファイルのフック定義の例

   ```
   [all:vars]
   openshift_node_pre_upgrade_hook=/home/user/openshift-ansible/hooks/pre_node.yml
   openshift_node_post_upgrade_hook=/home/user/openshift-ansible/hooks/post_node.yml
   ```

   フックへのパスでの曖昧さを避けるために、それらの定義では相対パスの代わりに絶対パスを使用します。

4.5.3. RHEL コンピュートマシンで利用できるフック

Red Hat Enterprise Linux (RHEL) コンピュートマシンを OpenShift Container Platform クラスターで更新する際に、以下のフックを使用できます。

フック名	説明
openshift_node_pre_cordon_hook	• 各ノードの遮断 (cordon) **前** に実行されます。 • このフックは **各ノード** に対して連続して実行されます。 • タスクが異なるホストに対して実行される必要がある場合、そのタスクは **delegate_to** または **local_action** を使用する必要があります。
openshift_node_pre_upgrade_hook	• 各ノードの遮断 (cordon) **後**、更新 **前** に実行されます。 • このフックは **各ノード** に対して連続して実行されます。 • タスクが異なるホストに対して実行される必要がある場合、そのタスクは **delegate_to** または **local_action** を使用する必要があります。

フック名	説明
openshift_node_pre_uncordon_hook	• 各ノードの更新 **後**、遮断の解除 (uncordon) **前に** 実行されます。 • このフックは **各ノード** に対して連続して実行されます。 • タスクが異なるホストに対して実行される必要がある場合、そのタスクは **delegate_to** または **local_action** を使用する必要があります。
openshift_node_post_upgrade_hook	• 各ノードの遮断の解除 (uncordon) **後** に実行されます。これは、**最後の** ノード更新アクションになります。 • このフックは **各ノード** に対して連続して実行されます。 • タスクが異なるホストに対して実行される必要がある場合、そのタスクは **delegate_to** または **local_action** を使用する必要があります。

4.6. クラスター内の RHEL コンピュートマシンの更新

クラスターの更新後は、クラスター内の Red Hat Enterprise Linux (RHEL) コンピュートマシンを更新する必要があります。

前提条件

- クラスターが更新されていること。

重要

RHEL マシンには、更新プロセスを完了するためにクラスターで生成されるアセットが必要になるため、クラスターを更新してから、クラスター内の RHEL コンピュートマシンを更新する必要があります。

- RHEL コンピュートマシンクラスターの追加に使用したマシンへのアクセスがあること。RHEL マシンを定義する **hosts** Ansible インベントリーファイルおよび **upgrade** Playbook にアクセスできる必要があります。

手順

1. ホストで firewalld を停止し、無効にします。

```
# systemctl disable --now firewalld.service
```

注記

firewalld は、後で有効にすることはできません。これを実行する場合、ワーカー上の OpenShift Container Platform ログにはアクセスできません。

2. OpenShift Container Platform 4.3 で必要なリポジトリーを有効にします。

   a. Ansible Playbook を実行するマシンで、必要なリポジトリーを更新します。

```
# subscription-manager repos --disable=rhel-7-server-ansible-2.7-rpms  \
                           --disable=rhel-7-server-ose-4.2-rpms \
                           --enable=rhel-7-server-ansible-2.8-rpms \
                           --enable=rhel-7-server-ose-4.3-rpms
```

   b. Ansible Playbook を実行するマシンで、**openshift-ansible** を含む必要なパッケージを更新します。

```
# yum update openshift-ansible openshift-clients
```

   c. 各 RHEL コンピュートノードで、必要なリポジトリーを更新します。

```
# subscription-manager repos --disable=rhel-7-server-ose-4.2-rpms \
                           --enable=rhel-7-server-ose-4.3-rpms
```

3. RHEL ワーカーマシンを更新します。

   a. 現在のノードステータスを確認し、更新する RHEL ワーカーを判別します。

```
# oc get node
NAME                 STATUS                     ROLES    AGE    VERSION
mycluster-control-plane-0   Ready                      master   145m   v1.16.2
mycluster-control-plane-1   Ready                      master   145m   v1.16.2
mycluster-control-plane-2   Ready                      master   145m   v1.16.2
mycluster-rhel7-0           NotReady,SchedulingDisabled   worker   98m
v1.14.6+97c81d00e
mycluster-rhel7-1           Ready                      worker   98m    v1.14.6+97c81d00e
mycluster-rhel7-2           Ready                      worker   98m    v1.14.6+97c81d00e
mycluster-rhel7-3           Ready                      worker   98m    v1.14.6+97c81d00e
```

   ステータスが **NotReady,SchedulingDisabled** のマシンに留意してください。

   b. /**<path>**/**inventory**/**hosts** で Ansible インベントリーファイルを確認し、以下の例に示されるように、ステータスが **NotReady,SchedulingDisabled** のマシンのみが **[workers]** セクションに一覧表示されるようにそのコンテンツを更新します。

```
[all:vars]
ansible_user=root
#ansible_become=True

openshift_kubeconfig_path="~/.kube/config"

[workers]
mycluster-rhel7-0.example.com
```

c. **openshift-ansible** ディレクトリーに切り替え、**upgrade** Playbook を実行します。

```
$ cd /usr/share/ansible/openshift-ansible
$ ansible-playbook -i /<path>/inventory/hosts playbooks/upgrade.yml 1
```

1 **<path>** については、作成した Ansible インベントリーファイルへのパスを指定します。

4. 直前の手順で実行したプロセスに従って、クラスター内の各 RHEL ワーカーマシンを更新します。

5. すべてのワーカーを更新したら、すべてのクラスターノードが新規バージョンに更新されていることを確認します。

```
# oc get node
NAME                    STATUS                        ROLES    AGE    VERSION
mycluster-control-plane-0   Ready                        master   145m   v1.16.2
mycluster-control-plane-1   Ready                        master   145m   v1.16.2
mycluster-control-plane-2   Ready                        master   145m   v1.16.2
mycluster-rhel7-0           NotReady,SchedulingDisabled   worker   98m    v1.16.2
mycluster-rhel7-1           Ready                        worker   98m    v1.16.2
mycluster-rhel7-2           Ready                        worker   98m    v1.16.2
mycluster-rhel7-3           Ready                        worker   98m    v1.16.2
```

第5章 ネットワークが制限された環境でのクラスターの更新

oc コマンドラインインターフェース (CLI) を使用してネットワークが制限された OpenShift Container Platform クラスターをアップグレードできます。

ネットワークが制限された環境とは、クラスターノードがインターネットにアクセスできない環境のことです。このため、レジストリーにはインストールイメージを設定する必要があります。レジストリーホストがインターネットとクラスターの両方にアクセスできない場合、その環境から切断されたファイルシステムにイメージをミラーリングし、そのホストまたはリムーバブルメディアを非接続環境に置きます。ローカルコンテナーレジストリーがミラーレジストリーのホストに接続されている場合、リリースイメージをローカルレジストリーに直接プッシュできます。

複数のクラスターがネットワークが制限されたネットワークに存在する場合、必要なリリースイメージを単一のコンテナーイメージレジストリーにミラーリングし、そのレジストリーを使用してすべてのクラスターを更新します。

前提条件

- 必要なコンテナーイメージを取得するためのインターネットへのアクセスがあること。
- イメージをプッシュおよびプルするために、ネットワークが制限された環境でコンテナーレジストリーへの書き込みアクセスがあること。コンテナーレジストリーは Docker レジストリー API v2 と互換性がある必要があります。
- **oc** コマンドツールインターフェース (CLI) ツールがインストールされていること。
- **admin** 権限を持つユーザーとしてクラスターにアクセスできること。「Using RBAC to define and apply permissions」を参照してください。
- アップグレードが失敗し、クラスターを直前の状態に復元する必要がある場合に最新の etcd バックアップがあること。

5.1. ミラーホストの準備

ミラー手順を実行する前に、ホストを準備して、コンテンツを取得し、リモートの場所にプッシュできるようにする必要があります。

5.1.1. バイナリーのダウンロードによる CLI のインストール

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

重要

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.3 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。ネットワークが制限された環境でクラスターをアップグレードする場合は、アップグレードする予定の **oc** バージョンをインストールします。

5.1.1.1. Linux への CLI のインストール

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

手順

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

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

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

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

5.1.1.2. Windows での CLI のインストール

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

手順

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

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

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

5.1.1.3. macOS への CLI のインストール

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

手順

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

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

5.2. イメージのミラーリングを可能にする認証情報の設定

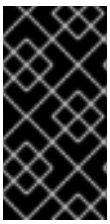
Red Hat からミラーへのイメージのミラーリングを可能にするコンテナーイメージレジストリーの認証情報ファイルを作成します。

警告

クラスターのインストール時に、このイメージレジストリー認証情報ファイルをプルシークレットとして使用しないでください。クラスターのインストール時にこのファイルを指定すると、クラスター内のすべてのマシンにミラーレジストリーへの書き込みアクセスが付与されます。

警告

このプロセスでは、ミラーレジストリーのコンテナーイメージレジストリーへの書き込みアクセスがあり、認証情報をレジストリープルシークレットに追加する必要があります。



重要

クラスターのインストール時に、このイメージレジストリー認証情報ファイルをプルシークレットとして使用しないでください。クラスターのインストール時にこのファイルを指定すると、クラスター内のすべてのマシンにミラーレジストリーへの書き込みアクセスが付与されます。

前提条件

- ネットワークが制限された環境で使用するミラーレジストリーを設定していること。
- イメージをミラーリングするミラーレジストリー上のイメージリポジトリーの場所を特定している。
- イメージのイメージリポジトリーへのアップロードを許可するミラーレジストリーアカウントをプロビジョニングしている。

手順

インストールホストで以下の手順を実行します。

1. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから **registry.redhat.io** プルシークレットをダウンロードし、これを **.json** ファイルに保存します。
2. ミラーレジストリーの base64 でエンコードされたユーザー名およびパスワードまたはトークンを生成します。

```
$ echo -n '<user_name>:<password>' | base64 -w0 (1)
BGVtbYk3ZHAtqXs=
```

(1) **<user_name>** および **<password>** については、レジストリーに設定したユーザー名およびパスワードを指定します。

3. JSON 形式でプルシークレットのコピーを作成します。

```
$ cat ./pull-secret.text | jq .  > <path>/<pull-secret-file> (1)
```

(1) プルシークレットを保存するフォルダーへのパスおよび作成する JSON ファイルの名前を指定します。

ファイルの内容は以下の例のようになります。

```
{
  "auths": {
    "cloud.openshift.com": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "quay.io": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
```

```
    }
  }
}
```

4. 新規ファイルを編集し、レジストリーについて記述するセクションをこれに追加します。

```
 "auths": {
  "<mirror_registry>": { 1
    "auth": "<credentials>", 2
    "email": "you@example.com"
 },
```

1 **<mirror_registry>** については、レジストリードメイン名と、ミラーレジストリーがコンテンツを提供するために使用するポートをオプションで指定します。例: **registry.example.com** または **registry.example.com:5000**

2 **<credentials>** については、ミラーレジストリーの base64 でエンコードされたユーザー名およびパスワードを指定します。

ファイルは以下の例のようになります。

```
{
  "auths": {
    "<mirror_registry>": {
      "auth": "<credentials>",
      "email": "you@example.com"
    },
    "cloud.openshift.com": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "quay.io": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    }
  }
}
```

1. 新規ファイルを編集し、レジストリーについて記述するセクションをこれに追加します。

```
  "auths": {
...
    "<mirror_registry>": { 1
      "auth": "<credentials>", 2
```

```
      "email": "you@example.com"
    },
...
```

1. **<mirror_registry>** については、レジストリードメイン名と、ミラーレジストリーがコンテンツを提供するために使用するポートをオプションで指定します。例: **registry.example.com** または **registry.example.com:5000**

2. **<credentials>** については、ミラーレジストリーの base64 でエンコードされたユーザー名およびパスワードを指定します。

ファイルは以下の例のようになります。

```
{
  "auths": {
    "cloud.openshift.com": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "quay.io": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    "<mirror_registry>": {
      "auth": "<credentials>",
      "email": "you@example.com"
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    }
  }
}
```

5.3. OPENSHIFT CONTAINER PLATFORM イメージリポジトリーのミラーリング

ネットワークが制限された環境でプロビジョニングするインフラストラクチャーのクラスターをアップグレードする前に、必要なコンテナーイメージをその環境にミラーリングする必要があります。この手順を無制限のネットワークで使用して、クラスターが外部コンテンツにちて組織の制御の条件を満たすコンテナーイメージのみを使用するようにすることもできます。

手順

1. OpenShift Container Platform のアップグレードパス を確認し、現在のクラスターバージョンと意図されるクラスターバージョン間にアップグレードパスがあることを確認します。

2. 必要な環境変数を設定します。

```
$ OCP_RELEASE=<release_version> 1
$ LOCAL_REGISTRY='<local_registry_host_name>:<local_registry_host_port>' 2
$ LOCAL_REPOSITORY='<repository_name>' 3
$ PRODUCT_REPO='openshift-release-dev' 4
$ LOCAL_SECRET_JSON='<path_to_pull_secret>' 5
$ RELEASE_NAME='ocp-release' 6
$ ARCHITECTURE=<server_architecture> 7
$ REMOVABLE_MEDIA_PATH=<path> 8
```

1. **<release_version>** について、インストールする OpenShift Container Platform のバージョンに対応するタグを指定します (例: **4.5.0**)。
2. **<local_registry_host_name>** については、ミラーレジストリーのレジストリードメイン名を指定し、**<local_registry_host_port>** については、コンテンツの送信に使用するポートを指定します。
3. **<repository_name>**については、**ocp4/openshift4**などのレジストリーに作成するリポジトリーの名前を指定します。
4. ミラーリングするリポジトリー。実稼働環境のリリースの場合には、**openshift-release-dev**を指定する必要があります。
5. **<path_to_pull_secret>**については、作成したミラーレジストリーのプルシークレットおよびファイル名の絶対パスを指定します。
6. 実稼働環境のリリースについては、**ocp-release**を指定する必要があります。
7. **<server_architecture>** について、サーバーのアーキテクチャー (**x86_64** など) を指定します。
8. **<path>** について、ミラーリングされたイメージをホストするためのディレクトリーへのパスを指定します。

3. ミラーリングするイメージおよび設定マニフェストを確認します。

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --to-
dir=${REMOVABLE_MEDIA_PATH}/mirror
quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-${ARCHITECTURE}
--dry-run
```

4. バージョンイメージを内部コンテナーレジストリーにミラーリングします。

- ミラーホストがインターネットにアクセスできない場合は、以下の操作を実行します。

  i. リムーバブルメディアをインターネットに接続しているシステムに接続します。

  ii. イメージおよび設定マニフェストをリムーバブルメディア上のディレクトリーにミラーリングします。

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --to-
dir=${REMOVABLE_MEDIA_PATH}/mirror
quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE}
```

iii. メディアをネットワークが制限された環境に移し、イメージをローカルコンテナーレジストリーにアップロードします。

```
$ oc image mirror  -a ${LOCAL_SECRET_JSON} --from-
dir=${REMOVABLE_MEDIA_PATH}/mirror
'file://openshift/release:${OCP_RELEASE}*'
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}
```

- ローカルコンテナーレジストリーがミラーホストに接続されている場合、リリースイメージをローカルレジストリーに直接プッシュできます。

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --
from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE} \
  --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}
```

5.4. イメージ CONFIGMAP の作成

クラスターを更新する前に、使用するリリースイメージの署名が含まれる ConfigMap を手動で作成する必要があります。この署名により、Cluster Version Operator (CVO) では、予想されるイメージと実際のイメージの署名を比較することでリリースイメージが変更されていないことを確認できます。

5.4.1. oc CLI の使用によるイメージ署名の検証用の ConfigMap の作成

クラスターを更新する前に、使用するリリースイメージの署名が含まれる ConfigMap を手動で作成する必要があります。この署名により、Cluster Version Operator (CVO) では、予想されるイメージと実際のイメージの署名を比較することでリリースイメージが変更されていないことを確認できます。

注記

バージョン 4.4.8 より前のリリースからアップグレードする場合は、この手順ではなく ConfigMap を作成するために手動の方法を使用する必要があります。この手順で使用するコマンドは、以前のバージョンの **oc** コマンドラインインターフェース (CLI) では提供されていません。

前提条件

- **oc** として知られる OpenShift コマンドラインインターフェース (CLI) のインストール (バージョン 4.4.8 以降)。

手順

1. mirror.openshift.com または Google Cloud Storage (GCS) のいずれかからアップグレードするバージョンのイメージ署名を取得します。
2. **oc** コマンドラインインターフェース (CLI) を使用して、アップグレードしているクラスターにログインします。
3. ミラーリングされたリリースイメージ署名 ConfigMap を接続されたクラスターに適用します。

```
$ oc apply -f <image_signature_file> 1
```

1 **<image_signature_file>** について、ファイルのパスおよび名前を指定します (例: **mirror/config/signature-sha256-81154f5c03294534.yaml**)。

5.4.2. イメージ署名 ConfigMap の手動での作成

イメージ署名 ConfigMap を作成し、更新するクラスターに適用します。

注記

クラスターを更新するたびに以下の手順を実行する必要があります。

手順

1. OpenShift Container Platform アップグレードパス についてのナレッジベースの記事を参照し、クラスターの有効なパスを判別します。
2. バージョンを **OCP_RELEASE_NUMBER** 環境変数に追加します。

```
$ OCP_RELEASE_NUMBER=<release_version> 1
```

1 **<release_version>** について、クラスターを更新する OpenShift Container Platform のバージョンに対応するタグを指定します (例: **4.4.0**)。

3. クラスターのシステムアーキテクチャーを **ARCHITECTURE** 環境変数に追加します。

```
$ ARCHITECTURE=<server_architecture> 1
```

server_architecture について、サーバーのアーキテクチャー (例: **x86_64**) を指定します。

4. Quay からリリースイメージダイジェストを取得します。

```
$ DIGEST="$(oc adm release info quay.io/openshift-release-dev/ocp-release:${OCP_RELEASE_NUMBER}-${ARCHITECTURE} | sed -n 's/Pull From: .*@//p')"
```

5. ダイジェストアルゴリズムを設定します。

```
$ DIGEST_ALGO="${DIGEST%%:*}"
```

6. ダイジェスト署名を設定します。

```
$ DIGEST_ENCODED="${DIGEST#*:}"
```

7. イメージ署名を mirror.openshift.com Web サイトから取得します。

```
$ SIGNATURE_BASE64=$(curl -s "https://mirror.openshift.com/pub/openshift-v4/signatures/openshift/release/${DIGEST_ALGO}=${DIGEST_ENCODED}/signature-1" | base64 -w0 && echo)
```

8. ConfigMap を作成します。

```
$ cat >checksum-${OCP_RELEASE_NUMBER}.yaml <<EOF
apiVersion: v1
kind: ConfigMap
metadata:
  name: release-image-${OCP_RELEASE_NUMBER}
  namespace: openshift-config-managed
  labels:
    release.openshift.io/verification-signatures: ""
binaryData:
  ${DIGEST_ALGO}-${DIGEST_ENCODED}: ${SIGNATURE_BASE64}
EOF
```

9. ConfigMap をクラスターに適用し、更新します。

```
$ oc apply -f checksum-${OCP_RELEASE_NUMBER}.yaml
```

5.5. ネットワークが制限された環境のクラスターのアップグレード

ネットワークが制限された環境のクラスターを、ダウンロードしたリリースイメージの OpenShift Container Platform バージョンに更新します。

前提条件

- 新規リリースのイメージをレジストリーに対してミラーリングしている。
- 新規リリースのリリースイメージ署名 ConfigMap をクラスターに適用している。
- イメージ署名 ConfigMap からリリースの sha256 合計値を取得している。
- **oc** として知られる OpenShift コマンドラインインターフェース (CLI) のインストール (バージョン 4.4.8 以降)。

手順

- クラスターを更新します。

```
$ oc adm upgrade --allow-explicit-upgrade --to-image
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}<sha256_sum_value> (1)
```

(1) **<sha256_sum_value>** 値は、イメージ署名 ConfigMap からのリリースの sha256 合計値です (例: **@sha256:81154f5c03294534e1eaf0319bef7a601134f891689ccede5d705ef659aa8c92**)。

ミラーレジストリーに **ImageContentSourcePolicy** を使用する場合、**LOCAL_REGISTRY** の代わりに正規レジストリー名を使用できます。

5.6. イメージレジストリーのリポジトリーミラーリングの設定

コンテナーレジストリーのリポジトリーミラーリングの設定により、以下が可能になります。

- ソースイメージのレジストリーのリポジトリーからイメージをプルする要求をリダイレクトするように OpenShift Container Platform クラスターを設定し、これをミラーリングされたイメージレジストリーのリポジトリーで解決できるようにします。

- 各ターゲットリポジトリーに対して複数のミラーリングされたリポジトリーを特定し、1つのミラーがダウンした場合に別のミラーを使用できるようにします。

以下は、OpenShift Container Platform のリポジトリーミラーリングの属性の一部です。

- イメージプルには、レジストリーのダウンタイムに対する回復性があります
- ネットワークが制限された環境のクラスターは、重要な場所 (quay.io など) からイメージをプルするように要求でき、会社のファイアウォールの背後にあるレジストリーが要求されたイメージを提供するようにできます。
- イメージのプル要求時にレジストリーへの接続が特定の順序で試行され、通常は永続レジストリーが最後に試行されます。
- 入力したミラー情報は、OpenShift Container Platform クラスターの全ノードの **/etc/containers/registries.conf** ファイルに追加されます。
- ノードがソースリポジトリーからイメージの要求を行うと、要求されたコンテンツを見つけるまで、ミラーリングされた各リポジトリーに対する接続を順番に試行します。すべてのミラーで障害が発生した場合、クラスターはソースリポジトリーに対して試行します。成功すると、イメージはノードにプルされます。

リポジトリーミラーリングのセットアップは次の方法で実行できます。

- OpenShift Container Platform インストール時: OpenShift Container Platform が必要とするコンテナーイメージをプルし、それらのイメージを会社のファイアウォールの内側に配置すると、制限されたネットワーク内にあるデータセンターに OpenShift Container Platform をインストールできます。詳細は、OpenShift Container Platform イメージレジストリーのミラーリングについて参照してください。
- OpenShift Container Platform のインストール後: OpenShift Container Platform インストール時にミラーリングを設定しなくても、**ImageContentSourcePolicy** オブジェクトを使用して後で設定することができます。

以下の手順では、インストール後のミラーを設定し、以下を識別する **ImageContentSourcePolicy** オブジェクトを作成します。

- ミラーリングするコンテナーイメージリポジトリーのソース
- ソースリポジトリーから要求されたコンテンツを提供する各ミラーリポジトリーの個別のエントリー。

前提条件

- **cluster-admin** ロールを持つユーザーとしてのクラスターへのアクセスがあること。

手順

1. ミラーリングされたリポジトリーを設定します。これを実行するには、以下のいずれかを行います。
   - 「Repository Mirroring in Red Hat Quay」で説明されているように、Red Hat Quay でミラーリングされたリポジトリーを設定します。Red Hat Quay を使用すると、あるリポジトリーから別のリポジトリーにイメージをコピーでき、これらのリポジトリーを一定期間繰り返し自動的に同期することもできます。

- **skopeo** などのツールを使用して、ソースディレクトリーからミラーリングされたリポジトリーにイメージを手動でコピーします。
  たとえば、Red Hat Enterprise Linux (RHEL 7 または RHEL 8) システムに skopeo RPM パッケージをインストールした後、以下の例に示すように **skopeo** コマンドを使用します。

```
$ skopeo copy \
  docker://registry.access.redhat.com/ubi8/ubi-minimal@sha256:5cfbaf45ca96806917830c183e9f37df2e913b187adb32e89fd83fa455ebaa6 \
  docker://example.io/ubi8/ubi-minimal
```

  この例では、**example.io** いう名前のコンテナーイメージレジストリーと **example** という名前のイメージリポジトリーがあり、そこに **registry.access.redhat.com** から **ubi8/ubi-minimal** イメージをコピーします。レジストリーを作成した後、OpenShift Container Platform クラスターを設定して、ソースリポジトリーで作成される要求をミラーリングされたリポジトリーにリダイレクトできます。

2. OpenShift Container Platform クラスターにログインします。

3. **ImageContentSourcePolicy** ファイル (例: **registryrepomirror.yaml**) を作成し、ソースとミラーを固有のレジストリー、およびリポジトリーのペアとイメージのものに置き換えます。

```
apiVersion: operator.openshift.io/v1alpha1
kind: ImageContentSourcePolicy
metadata:
  name: ubi8repo
spec:
  repositoryDigestMirrors:
  - mirrors:
    - example.io/example/ubi-minimal (1)
    source: registry.access.redhat.com/ubi8/ubi-minimal (2)
  - mirrors:
    - example.com/example/ubi-minimal
    source: registry.access.redhat.com/ubi8/ubi-minimal
```

(1) イメージレジストリーおよびリポジトリーの名前を示します。

(2) ミラーリングされているコンテンツが含まれるレジストリーおよびリポジトリーを示します。

4. 新しい **ImageContentSourcePolicy** を作成します。

```
$ oc create -f registryrepomirror.yaml
```

  ImageContentSourcePolicy が作成されると、新しい設定が各ノードにデプロイされ、ソースリポジトリーへの要求のためにミラーリングされたリポジトリーの使用をすぐに開始します。

5. ミラーリングされた設定が機能することを確認するには、ノードのいずれかに移動します。以下は例になります。

   a. ノードの一覧を表示します。

```
$ oc get node
```

出力例

```
NAME                         STATUS                     ROLES    AGE  VERSION
ip-10-0-137-44.ec2.internal  Ready                      worker   7m   v1.16.2
ip-10-0-138-148.ec2.internal  Ready                      master   11m  v1.16.2
ip-10-0-139-122.ec2.internal  Ready                      master   11m  v1.16.2
ip-10-0-147-35.ec2.internal  Ready,SchedulingDisabled   worker   7m   v1.16.2
ip-10-0-153-12.ec2.internal  Ready                      worker   7m   v1.16.2
ip-10-0-154-10.ec2.internal  Ready                      master   11m  v1.16.2
```

変更が適用されているため、各ワーカーノードのスケジューリングが無効にされていることを確認できます。

b. デバッグプロセスを開始します。

```
$ oc debug node/ip-10-0-147-35.ec2.internal
```

出力例

```
Starting pod/ip-10-0-147-35ec2internal-debug ...
To use host binaries, run `chroot /host`
```

c. ノードのファイルにアクセスします。

```
sh-4.2# chroot /host
```

d. **/etc/containers/registries.conf** ファイルをチェックして、変更が行われたことを確認します。

```
unqualified-search-registries = ["registry.access.redhat.com", "docker.io"]
[[registry]]
  location = "registry.access.redhat.com/ubi8/"
  insecure = false
  blocked = false
  mirror-by-digest-only = true
  prefix = ""

  [[registry.mirror]]
    location = "example.io/example/ubi8-minimal"
    insecure = false

  [[registry.mirror]]
    location = "example.com/example/ubi8-minimal"
    insecure = false
```

e. ソースからノードにイメージダイジェストをプルし、ミラーによって実際に解決されているかどうかを確認します。**ImageContentSourcePolicy** はイメージダイジェストのみをサポートし、イメージタグはサポートしません。

```
sh-4.2# podman pull --log-level=debug registry.access.redhat.com/ubi8/ubi-minimal@sha256:5cfbaf45ca96806917830c183e9f37df2e913b187adb32e89fd83fa455ebaa6
```

リポジトリーのミラーリングのトラブルシューティング

リポジトリーのミラーリング手順が説明どおりに機能しない場合は、リポジトリーミラーリングの動作方法についての以下の情報を使用して、問題のトラブルシューティングを行うことができます。

- 最初に機能するミラーは、プルされるイメージを指定するために使用されます。
- メインレジストリーは、他のミラーが機能していない場合にのみ使用されます。
- システムコンテキストによって、**Insecure** フラグがフォールバックとして使用されます。
- **/etc/containers/registries** ファイルの形式が最近変更されました。現在のバージョンはバージョン 2 で、TOML 形式です。