OpenShift Container Platform 4.4

移行

OpenShift Container Platform 4 への移行

Last Updated: 2023-04-01

OpenShift Container Platform 4.4 移行

OpenShift Container Platform 4 への移行

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

本書では、OpenShift Container Platform クラスターをバージョン 3 からバージョン 4 に移行する方法と、OpenShift Container Platform 4 の以前のリリースから最新バージョンへの移行方法について説明します。

目次

第1章 OPENSHIFT CONTAINER PLATFORM 3 からの移行

1.1. OPENSHIFT CONTAINER PLATFORM 3 から 4 への移行について

OpenShift Container Platform 4 には、自己管理型の柔軟で自動化されたクラスターを実現する新規のテクノロジーおよび機能が含まれています。OpenShift Container Platform 4 クラスターがデプロイされ、管理される方法は OpenShift Container Platform 3 とは大きく異なります。

OpenShift Container Platform 3 から OpenShift Container Platform 4 に正常に移行するには、以下の情報を確認することが重要になります。

移行の計画

OpenShift Container Platform のバージョン 3 と 4 の相違点について確認します。移行の前に、ストレージ、ネットワーク、ロギング、セキュリティー、およびモニターリングに関する考慮事項について確認し、準備済みであることを確認します。

移行の実行

移行を実行するためのツールについて確認し、これらを使用します。

- アプリケーションのワークロードを移行するための Cluster Application Migration (CAM) ツール
- コントロールプレーンを移行するための Control Plane Migration Assistant (CPMA)

1.2. 移行の計画

OpenShift Container Platform 4.4 への移行を実行する前に、十分な時間を取って移行を適切に計画できるようにしてください。OpenShift Container Platform 4 ではアーキテクチャーの変更および拡張機能が導入されるため、OpenShift Container Platform 3 クラスターの管理に使用した手順は OpenShift Container Platform 4 で適用されない可能性があります。

注記

この計画ガイドでは、OpenShift Container Platform 3.11 から OpenShift Container Platform 4.4 への移行を前提としています。

本書では、最も重要な OpenShift Container Platform 3 と OpenShift Container Platform 4 の相違点と、最も注目すべき 移行に関する考慮事項 についての概要を説明します。OpenShift Container Platform 4 クラスターの設定についての詳細は、OpenShift Container Platform ドキュメントの該当するセクションを参照してください。新規機能および他の重要な技術上の変更点についての詳細は、OpenShift Container Platform 4.4 リリースノート を参照してください。

既存の OpenShift Container Platform 3 クラスターを OpenShift Container Platform 4 にアップグレードすることはできません。新規の OpenShift Container Platform 4 インストールで開始する必要があります。コントロールプレーンの設定およびアプリケーションのワークロードの移行に役立つツールを使用できます。

1.2.1. OpenShift Container Platform 3 と OpenShift Container Platform 4 の比較

OpenShift Container Platform 3 では、管理者は Red Hat Enterprise Linux (RHEL) ホストを個別にデプロイし、その後に OpenShift Container Platform をこれらのホストにインストールし、クラスターを作成しました。管理者は、これらのホストを適切に設定し、更新を実行する必要があります。

OpenShift Container Platform 4 では、これまでとは大きく異なる方法で OpenShift Container Platform クラスターがデプロイされ、管理されるようになりました。OpenShift Container Platform 4 には、Operator、MachineSet、および Red Hat Enterprise Linux CoreOS (RHCOS) などの、クラスターの操作に対するコアとなる新たなテクノロジーおよび機能が含まれます。このテクノロジーの移行により、クラスターは管理者が以前に実行していた一部の機能を自己管理できるようになります。また、プラットフォームの安定性と一貫性を確保し、インストールおよびスケーリングを単純化することが可能です。

詳細は、OpenShift Container Platform アーキテクチャー を参照してください。

1.2.1.1. アーキテクチャーの違い

イミュータブルなインフラストラクチャー
OpenShift Container Platform 4 は、コンテナー化されたアプリケーションを実行するために設計された Red Hat Enterprise Linux CoreOS (RHCOS) を使用し、効率的なインストール、Operator ベースの管理、および単純化されたアップグレードを可能にします。RHCOS は、RHEL のようなカスタマイズ可能なオペレーティングシステムではなく、イミュータブルなコンテナーホストです。RHCOS により、OpenShift Container Platform 4 は基礎となるコンテナーホストのデプロイメントを管理し、自動化できます。RHCOS は OpenShift Container Platform の一部です。これは、すべてがコンテナー内で実行され、OpenShift Container Platform を使用してデプロイされることを意味します。

OpenShift Container Platform 4 では、コントロールプレーンノードは RHCOS を実行する必要があるため、コントロールプレーンのフルスタック自動化が維持されます。これにより、OpenShift Container Platform 3 よりも簡単に更新がロールアウトされ、アップグレードが簡単になります。

詳細は、Red Hat Enterprise Linux CoreOS を参照してください。

Operator
Operator は、Kubernetes アプリケーションをパッケージ化し、デプロイし、管理する方法です。Operator は、ソフトウェアの他の部分を実行する際の運用上の複雑さを軽減します。Operator は環境を監視し、現在の状態を使用してリアルタイムの意思決定を行います。高度な Operator は、自動的にアップグレードし、障害に自動的に対応するように設計されています。

詳細は、Operator について を参照してください。

1.2.1.2. インストールおよび更新の違い

インストールプロセス
OpenShift Container Platform 3.11 をインストールするには、Red Hat Enterprise Linux (RHEL) ホストを準備し、クラスターが必要とする設定値をすべて設定してから、Ansible Playbook を実行してクラスターをインストールし、セットアップする必要がありました。

OpenShift Container Platform 4.4 では、OpenShift インストールプログラムを使用してクラスターに必要なリソースの最小セットを作成できます。クラスターの実行後に、Operator を使用してクラスターをさらに設定し、新規サービスをインストールすることができます。初回の起動後に、Red Hat Enterprise Linux CoreOS (RHCOS) システムは、OpenShift Container Platform クラスターで実行される Machine Config Operator (MCO) によって管理されます。

詳細は、インストールプロセス を参照してください。

RHEL ワーカーマシンを OpenShift Container Platform 4.4 クラスターに追加する場合、Ansible Playbook を使用して、クラスターの実行後に RHEL ワーカーマシンを追加します。詳細は、RHEL コンピュートマシンの OpenShift Container Platform クラスターへの追加 を参照してください。

インフラストラクチャーオプション
OpenShift Container Platform 3.11 では、ユーザーが準備し、維持する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必要があります。OpenShift Container Platform 4 では、独自のインフラストラクチャーを提供するだけでなく、OpenShift Container Platform インストールプログラムがプロビジョニングし、クラスターが維持するインフラストラクチャーにクラスターをデプロイするオプションを提供します。

詳細は、OpenShift Container Platform インストールの概要 を参照してください。

クラスターのアップグレード
OpenShift Container Platform 3.11 では、Ansible Playbook を実行してクラスターをアップグレードします。OpenShift Container Platform 4.4 では、クラスターが、クラスターノードの Red Hat Enterprise Linux CoreOS (RHCOS) への更新を含む独自の更新を管理します。Web コンソールまたは OpenShift CLI から **oc adm upgrade** コマンドを使用することでクラスターを容易にアップグレードでき、Operator は自動的にアップグレードされます。OpenShift Container Platform 4.4 クラスターに Red Hat Enterprise Linux ワーカーマシンがある場合、Ansible Playbook を引き続き実行してそれらのワーカーマシンをアップグレードする必要があります。

詳細は、クラスターの更新 を参照してください。

1.2.2. 移行に関する考慮事項

OpenShift Container Platform 3.11 から OpenShift Container Platform 4 への移行に影響を与える可能性のある変更やその他の考慮事項を確認します。

1.2.2.1. ストレージに関する考慮事項

OpenShift Container Platform 3.11 から OpenShift Container Platform 4.4 に移行する際に、以下のストレージの変更を考慮してください。

ローカルボリュームの永続ストレージ
ローカルストレージは、OpenShift Container Platform 4.4 ではローカルストレージ Operator を使用する場合にのみサポートされます。OpenShift Container Platform 3.11 のローカルプロビジョナーメソッドの使用はサポートされません。

詳細は、ローカルボリュームを使用した永続ストレージ を参照してください。

FlexVolume 永続ストレージ
FlexVolume プラグインの場所が OpenShift Container Platform 3.11 で変更になりました。OpenShift Container Platform 4.4 の新しい場所は **/etc/kubernetes/kubelet-plugins/volume/exec** です。割り当て可能な FlexVolume プラグインはサポートされなくなりました。

詳細は、FlexVolume を使用した永続ストレージ を参照してください。

Container Storage Interface (CSI) 永続ストレージ
Container Storage Interface (CSI) を使用した永続ストレージは OpenShift Container Platform 3.11 ではテクノロジープレビュー として利用可能でした。CSI バージョン 1.1.0 は OpenShift Container Platform 4.4 で完全にサポートされていますが、これは CSI ドライバーに同梱されません。そのため、独自のドライバーをインストールする必要があります。

詳細は、Container Storage Interface (CSI) を使用した永続ストレージ を参照してください。

Red Hat OpenShift Container Storage
OpenShift Container Platform 3.11 で使用できる Red Hat OpenShift Container Storage 3 は、バッキングストレージとして Red Hat Gluster Storage を使用します。

OpenShift Container Platform 4 で使用できる Red Hat OpenShift Container Storage 4 は、バッキングストレージとして Red Hat Ceph Storage を使用します。

詳細は、Persistent storage using Red Hat OpenShift Container Storage および interoperability matrix の記事を参照してください。

サポートされていない永続ストレージオプション
OpenShift Container Platform 3.11 の以下の永続ストレージオプションのサポートが OpenShift Container Platform 4.4 で変更になりました。

- GlusterFS はサポート対象外になりました。
- スタンドアロン製品としての CephFS がサポートされなくなりました。
- スタンドアロン製品としての Ceph RBD がサポートされなくなりました。

OpenShift Container Platform 3.11 でこれらのいずれかを使用していた場合は、OpenShift Container Platform 4.4 で完全にサポートされる別の永続ストレージオプションを選択する必要があります。

詳細は、永続ストレージについて を参照してください。

1.2.2.2. ネットワークの考慮事項

OpenShift Container Platform 3.11 から OpenShift Container Platform 4.4 に移行する際に、考慮事項となる以下のネットワークの変更を確認してください。

ネットワーク分離モード
OpenShift Container Platform 3.11 では、ユーザーは **ovn-multitenant** を使用するように頻繁に切り替えましたが、デフォルトのネットワーク分離モードは **ovs-subnet** でした。OpenShift Container Platform 4.4 のデフォルトのネットワーク分離モードは NetworkPolicy になりました。

OpenShift Container Platform 3.11 クラスターで **ovs-subnet** または **ovs-multitenant** モードを使用していた場合、OpenShift Container Platform 4.4 クラスターでは NetworkPolicy モードに切り換えることが推奨されます。NetworkPolicy はアップストリームでサポートされ、より柔軟であり、**ovs-multitenant** が実行する機能を提供します。OpenShift Container Platform 4.4 で NetworkPolicy を使用する際に **ovs-multitenant** 動作を維持する必要がある場合、NetworkPolicy を使用したマルチテナント分離の設定 手順を実行します。

詳細は、ネットワークポリシーについて を参照してください。

ホスト間のトラフィックの暗号化
OpenShift Container Platform 3.11 では、IPsec を使用してホスト間のトラフィックを暗号化できます。OpenShift Container Platform 4.4 は IPsec をサポートしません。サービス間で相互 TLS を有効にするために、Red Hat OpenShift Service Mesh を使用することが推奨されます。

詳細は、Understanding Red Hat OpenShift Service Mesh を参照してください。

1.2.2.3. ロギングについての考慮事項

OpenShift Container Platform 3.11 から OpenShift Container Platform 4.4 に移行する際に、考慮事項となる以下のロギングの変更を確認してください。

クラスターロギングのデプロイ
OpenShift Container Platform 4 は、クラスターロギングのカスタムリソース (Custom Resource) を使用してクラスターロギングの単純なデプロイメントメカニズムを提供します。デプロイが完了すると、クラスターロギングは OpenShift Container Platform 3.11 の場合と同じように使用できます。

詳細は、クラスターロギングのデプロイおよび設定について を参照してください。

集計ロギングデータ

集計ロギングデータを OpenShift Container Platform 3.11 から新規の OpenShift Container Platform 4 クラスターに移行することはできません。

詳細は、クラスターロギングについて を参照してください。

1.2.2.4. セキュリティーに関する考慮事項

OpenShift Container Platform 3.11 から OpenShift Container Platform 4.4 へ移行する際に、考慮事項となる以下のセキュリティーの変更を確認してください。

検出エンドポイントへの認証されていないアクセス
OpenShift Container Platform 3.11 では、認証されていないユーザーは検出エンドポイント (例: **/api/*** および **/apis/***) にアクセスできました。セキュリティー上の理由から、検出エンドポイントへの認証されていないアクセスは OpenShift Container Platform 4.4 で許可されなくなりました。認証されていないアクセスを許可する必要がある場合は、必要に応じて RBAC を設定できます。 ただし、これにより内部クラスターコンポーネントが外部ネットワークに公開される可能性があるため、セキュリティー上の影響を考慮してください。

アイデンティティープロバイダー
アイデンティティープロバイダーの設定は、以下の主な変更点を含め、 OpenShift Container Platform 4 で変更されています。

- OpenShift Container Platform 4.4 の要求ヘッダーアイデンティティープロバイダーには相互 TLS が必要ですが、OpenShift Container Platform 3.11 ではこれは必要ではありませんでした。
- OpenID Connect アイデンティティープロバイダーの設定は OpenShift Container Platform 4.4 で単純化されています。OpenShift Container Platform 3.11 で以前に指定される必要のあったデータが、プロバイダーの **/.well-known/openid-configuration** エンドポイントから取得できるようになりました。

詳細は、アイデンティティープロバイダー設定について を参照してください。

1.2.2.5. モニターリングに関する考慮事項

OpenShift Container Platform 3.11 から OpenShift Container Platform 4.4 に移行する際に、考慮事項となる以下のモニターリングの変更を確認してください。

インフラストラクチャーの可用性についてのモニターリングアラート
モニターリング構造の可用性を確保するためにトリガーするデフォルトのアラートは OpenShift Container Platform 3.11 では **DeadMansSwitch** と呼ばれていました。この名前は OpenShift Container Platform 4 で **Watchdog** に変更されています。OpenShift Container Platform 3.11 で PagerDuty 統合をこのアラートでセットアップしている場合、OpenShift Container Platform 4 では **Watchdog** アラートについて PagerDuty 統合をセットアップする必要があります。

詳細は、カスタム Alertmanager 設定の適用 を参照してください。

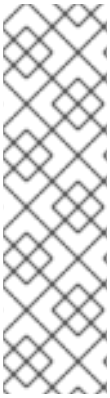
1.3. 移行ツールおよび前提条件

アプリケーションワークロードを、Cluster Application Migration (CAM) ツールを使用して OpenShift Container Platform 3.7、3.9、3.10、および 3.11 から OpenShift Container Platform 4.4 に移行できます。CAM ツールを使用すると、移行を制御し、アプリケーションのダウンタイムを最小限に抑えることができます。

Kubernetes カスタムリソースをベースとする CAM ツールの Web コンソールおよび API により、namespace の粒度でステートフルなアプリケーションワークロードを移行できます。

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するためにファイルシステムおよびスナップショットによるデータのコピー方法をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

移行フックを使用して、移行中の特定の時点で Ansible Playbook を実行できます。フックは移行計画の作成時に追加されます。



注記

サービスカタログは OpenShift Container Platform 4 では非推奨になっています。サービスカタログでプロビジョニングされたワークロードリソースを OpenShift Container Platform 3 から 4 に移行できますが、移行後にこれらのワークロードで **provision**、**deprovision**、または **update** などのサービスカタログのアクションを実行できません。

CAM ツールは移行できないサービスカタログリソース (**ClusterServiceClass**、**ServiceInstance**、または **ServiceBinding** など) についてのメッセージを表示します。

Control Plane Migration Assistant (CPMA) は、コントロールプレーンの移行に役立つ CLI ベースのツールです。CPMA は OpenShift Container Platform 3 設定ファイルを処理し、OpenShift Container Platform 4.4 Operator によって使用されるカスタムリソース (CR) マニフェストファイルを生成します。

重要

移行を開始する前に、移行計画 についての情報を確認してください。

1.3.1. 移行の前提条件

- **podman** がインストールされていること。
- ソースクラスターは OpenShift Container Platform 3.7、3.9、3.10、または 3.11 であること。
- ソースクラスターを最新の z-stream リリースにアップグレードすること。
- すべてのクラスターで **cluster-admin** 権限がある。
- ソースおよびターゲットクラスターには、レプリケーションリポジトリーへの無制限のネットワークアクセスがなければなりません。
- Migration コントローラーがインストールされているクラスターには、他のクラスターへの無制限のアクセスが必要です。
- アプリケーションが **openshift** namespace のイメージを使用する場合、イメージの必要なバージョンがターゲットクラスターに存在する必要があります。
  必要なイメージがない場合は、アプリケーションと互換性のある利用可能なバージョンを使用するように **imagestreamtags** 参照を更新する必要があります。**imagestreamtag** を更新できない場合、同等のイメージをアプリケーション namespace に手動でアップロードし、それらを参照するようにアプリケーションを更新できます。

以下の **imagestreamtags** は OpenShift Container Platform **4.2** から **削除** されています。

- **dotnet:1.0**、**dotnet:1.1**、 **dotnet:2.0**

- **dotnet-runtime:2.0**
- **mariadb:10.1**
- **mongodb:2.4**、**mongodb:2.6**
- **mysql:5.5**、**mysql:5.6**
- **nginx:1.8**
- **nodejs:0.10**、**nodejs:4**、**nodejs:6**
- **perl:5.16**、**perl:5.20**
- **php:5.5**、**php:5.6**
- **postgresql:9.2**、**postgresql:9.4**、**postgresql:9.5**
- **python:3.3**、**python:3.4**
- **ruby:2.0**, **ruby:2.2**

以下の **imagestreamtags** は OpenShift Container Platform **4.4** から **削除** されています。

- **dotnet: 2.2**
- **dotnet-runtime: 2.2**
- **nginx: 1.12**
- **nodejs: 8, 8-RHOAR, 10-SCL**
- **perl:5.24**
- **php: 7.0, 7.1**
- **redis: 3.2**

1.3.2. Cluster Application Migration ツールについて

Cluster Application Migration (CAM) ツールを使用すると、CAM Web コンソールまたは Kubernetes API を使用して Kubernetes リソース、永続ボリュームデータ、および内部コンテナーイメージを OpenShift Container Platform ソースクラスターから OpenShift Container Platform 4.4 ターゲットクラスターに移行できます。

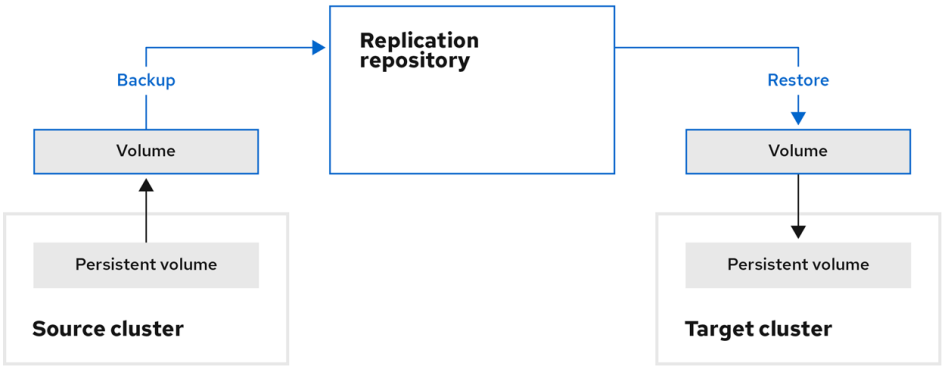
CAM Web コンソールを使用してアプリケーションを移行するには、以下の手順が必要です。

1. Cluster Application Migration Operator をすべてのクラスターにインストールします。
   インターネットアクセスが制限されているか、またはインターネットアクセスのない環境で Cluster Application Migration Operator をインストールできます。ソースおよびターゲットクラスターは、相互に対するネットワークアクセスおよびミラーレジストリーへのネットワークアクセスがなければなりません。

2. CAM ツールがデータ移行に使用する中間オブジェクトストレージであるレプリケーションリポジトリーを設定します。
   ソースおよびターゲットクラスターには、移行時にレプリケーションリポジトリーへのネットワークアクセスがなければなりません。制限された環境では、内部でホストされる S3 スト

レージリポジトリーを使用できます。プロキシーサーバーを使用する場合は、レプリケーションリポジトリーがホワイトリスト化されていることを確認する必要があります。

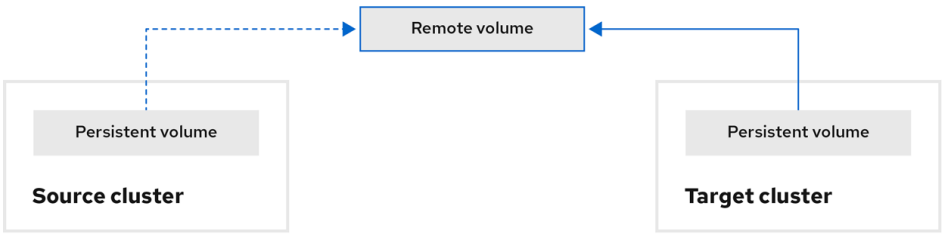
3. ソースクラスターを CAM Web コンソールに追加します。
4. レプリケーションリポジトリーを CAM Web コンソールに追加します。
5. 以下のデータ移行オプションのいずれかを使用して、移行計画を作成します。
   - **Copy**: CAM ツールは、データをソースクラスターからレプリケーションリポジトリーにコピーし、レプリケーションリポジトリーからターゲットクラスターにコピーします。



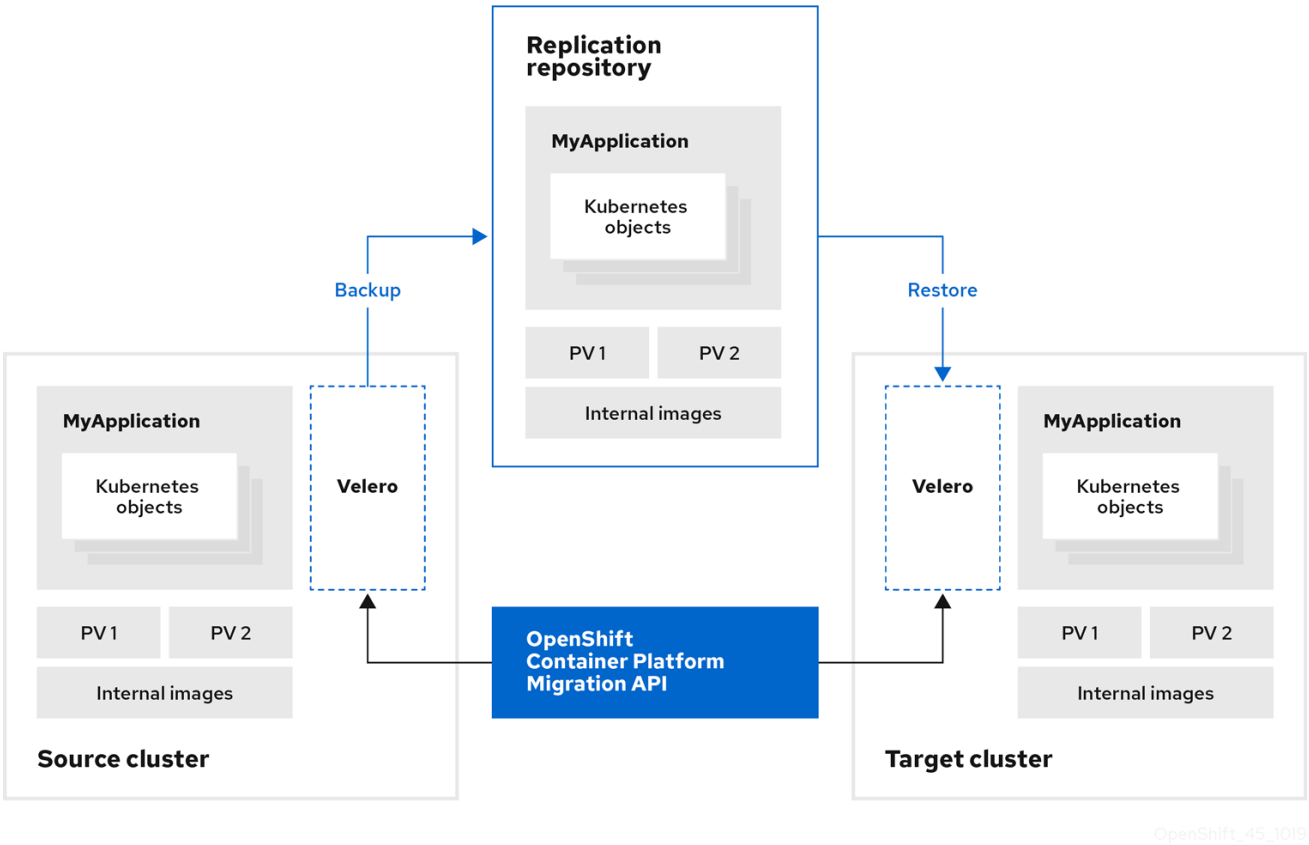
   - **Move**: CAM ツールはリモートボリューム (NFS など) をソースクラスターからアンマウントし、リモートボリュームをポイントするターゲットクラスターで PV リソースを作成し、その後にリモートボリュームをターゲットクラスターにマウントします。ターゲットクラスターで実行されているアプリケーションは、ソースクラスターが使用していたものと同じリモートボリュームを使用します。リモートボリュームは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。

注記

レプリケーションリポジトリーはこの図には表示されていませんが、実際の移行には必要になります。



6. 以下のオプションのいずれかを使用して、移行計画を実行します。
   - **Stage** (オプション) は、アプリケーションを停止せずにデータをターゲットクラスターにコピーします。
     ステージングは、移行前にほとんどのデータがターゲットにコピーされるように複数回実行することができます。これにより、実際の移行時間やアプリケーションのダウンタイムが最小限に抑えられます。
   - **Migrate** は、ソースクラスターでアプリケーションを停止し、ターゲットクラスターでそのリソースを再作成します。オプションで、アプリケーションを停止せずにワークロードを移行できます。



1.3.3. データのコピー方法

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するためにファイルシステムおよびスナップショットによるデータのコピー方法をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

1.3.3.1. ファイルシステムのコピー方法

CAM ツールは、データファイルをソースクラスターからレプリケーションリポジトリーにコピーし、そこからターゲットクラスターにコピーします。

表1.1 ファイルシステムのコピー方法の概要

利点	制限
• クラスターで複数の異なるストレージクラスを使用することが可能 • すべての S3 ストレージプロバイダーでサポートされている • チェックサムを使用したオプションのデータ検証	• スナップショットのコピー方法よりも遅い • オプションのデータ検証は、パフォーマンスを大幅に低下させます。

1.3.3.2. スナップショットのコピー方法

CAM ツールは、ソースクラスターのデータのスナップショットを、レプリケーションリポジトリーとして設定されたクラウドプロバイダーのオブジェクトストレージにコピーします。データはターゲットクラスターで復元されます。

AWS、Google Cloud Provider、および Microsoft Azure は、スナップショットのコピー方法をサポートします。

表1.2 スナップショットのコピー方法の概要

利点	制限
• ファイルシステムのコピー方法よりも高速	• クラウドプロバイダーはスナップショットをサポートしている必要があります。 • クラスターは同じクラウドプロバイダーになければなりません。 • クラスターは、同じ場所またはリージョンにある必要があります。 • クラスターには同じストレージクラスがなければなりません。 • ストレージクラスにはスナップショットとの互換性がある必要があります。

1.3.4. 移行フックについて

移行フックを使用して、移行中の特定の時点で Ansible Playbook を実行できます。フックは移行計画の作成時に追加されます。

注記

Ansible Playbook を使用する必要がない場合は、カスタムコンテナーイメージを作成し、これを移行計画に追加することができます。

移行フックは、アプリケーションの休止状態のカスタマイズ、サポート外のデータタイプの手動の移行、および移行後のアプリケーションの更新などのタスクを実行します。

単一の移行フックは、以下の移行手順のいずれかでソースまたはターゲットクラスターで実行されます。

- PreBackup: バックアップタスクがソースクラスターで開始される前
- PostBackup: バックアップタスクがソースクラスターで完了した後
- PreRestore: 復元タスクがターゲットクラスターで開始される前
- PostRestore: 復元タスクがターゲットクラスターで完了した後
  1 つのフックをそれぞれの移行ステップに割り当て、単一の移行計画について最大 4 つのフックを割り当てることができます。

デフォルトの **hook-runner** イメージは **registry.redhat.io/rhcam-1-2/openshift-migration-hook-runner-rhel7** です。このイメージは Ansible Runner をベースとしており、Ansible Kubernetes リソース用の **python-openshift** および更新された **oc** バイナリーが含まれます。追加の Ansible モジュールまた

はツールで独自のフックイメージを作成することもできます。

Ansible Playbook はフックコンテナーに ConfigMap としてマウントされます。フックコンテナーは、指定されたサービスアカウントおよび namespace を使用してクラスターでジョブとして実行されます。ジョブは、初期の Pod がエビクトされるか、または強制終了される場合でも、デフォルトの **backoffLimit** (**6**) または正常に完了した状態に達するまで実行されます。

1.3.5. Control Plane Migration Assistant について

Control Plane Migration Assistant (CPMA) は、OpenShift Container Platform 3.7 (以降) から OpenShift Container Platform 4.4 へのコントロールプレーンの移行に役立つ CLI ベースのツールです。CPMA は OpenShift Container Platform 3 設定ファイルを処理し、OpenShift Container Platform 4.4 Operator によって使用されるカスタムリソース (CR) マニフェストファイルを生成します。

OpenShift Container Platform 3 および 4 には設定上の大きな違いがあるため、すべてのパラメーターが処理される訳ではありません。CPMA は、機能が完全にサポートされるか、部分的にサポートされるか、または全くサポートされないかについて記述するレポートを生成できます。

設定ファイル

CPMA は Kubernetes および OpenShift Container Platform API を使用して、OpenShift Container Platform 3 クラスターの以下の設定ファイルにアクセスします。

- マスター設定ファイル (デフォルト: **/etc/origin/master/master-config.yaml**)
- CRI-O 設定ファイル (デフォルト: **/etc/crio/crio.conf**)
- etcd 設定ファイル (デフォルト: **/etc/etcd/etcd.conf**)
- イメージレジストリーファイル (デフォルト: **/etc/containers/registries.conf**)
- 依存する設定ファイル:
  - パスワードファイル (HTPasswd など)
  - ConfigMap
  - シークレット

CR マニフェスト

CPMA は、以下の設定の CR マニフェストを生成します。

- API サーバー CA 証明書: **100_CPMA-cluster-config-APISecret.yaml**

注記

署名なしの API サーバー CA 証明書を使用している場合は、証明書をターゲットクラスターに手動で追加する必要があります。

- CRI-O: **100_CPMA-crio-config.yaml**
- クラスターリソースクォータ: **100_CPMA-cluster-quota-resource-x.yaml**
- プロジェクトリソースクォータ: **100_CPMA-resource-quota-x.yaml**

- 移植可能なイメージレジストリー (**/etc/registries/registries.conf**) および移植可能なイメージポリシー (**etc/origin/master/master-config.yam**): **100_CPMA-cluster-config-image.yaml**
- OAuth プロバイダー: **100_CPMA-cluster-config-oauth.yaml**
- プロジェクト設定: **100_CPMA-cluster-config-project.yaml**
- スケジューラー: **100_CPMA-cluster-config-scheduler.yaml**
- SDN: **100_CPMA-cluster-config-sdn.yaml**

1.4. クラスターアプリケーション移行ツールのデプロイおよびアップグレード

Cluster Application Migration Operator を OpenShift Container Platform 4.4 ターゲットクラスターおよび OpenShift Container Platform 3 ソースクラスターにインストールできます。Cluster Application Migration Operator は、デフォルトで Cluster Application Migration (CAM) ツールをターゲットクラスターにインストールします。

注記

オプション: Cluster Application Migration Operator を、CAM ツールを OpenShift Container Platform 3 クラスターまたはリモートクラスターに インストールするように設定できます。

制限された環境では、ローカルミラーレジストリーから Cluster Application Migration Operator をインストールできます。

クラスターに Cluster Application Migration Operator をインストールした後に、CAM ツールを起動できます。

1.4.1. Cluster Application Migration Operator のインストール

Cluster Application Migration Operator は Operator Lifecycle Manager (OLM) で OpenShift Container Platform 4.4 ターゲットクラスターにインストールすることも、手動で OpenShift Container Platform 3 ソースクラスターにインストールすることもできます。

1.4.1.1. Cluster Application Migration Operator の OpenShift Container Platform 4.4 ターゲットクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4.4 ターゲットクラスターに Cluster Application Migration Operator をインストールできます。

Cluster Application Migration Operator は、デフォルトで Cluster Application Migration ツールをターゲットクラスターにインストールします。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。

3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
   Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **Create** をクリックします。
8. **Workloads → Pods** をクリックし、Controller Manager、Migration UI、Restic、および Velero Pod が実行中であることを確認します。

1.4.1.2. OpenShift Container Platform 3 ソースクラスターへの Cluster Application Migration Operator のインストール

Cluster Application Migration Operator を手動で OpenShift Container Platform 3 ソースクラスターにインストールできます。

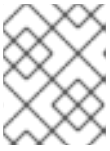
前提条件

- **registry.redhat.io** へのアクセス
- イメージを **registry.redhat.io** からプルするように設定された OpenShift Container Platform 3 クラスター
  イメージをプルするには、**imagestreamsecret** を作成し、これをクラスター内の各ノードにコピーする必要があります。

手順

1. Red Hat カスタマーポータルの認証情報を使用して **registry.redhat.io** にログインします。

```
$ sudo podman login registry.redhat.io
```



注記

システムがルートレス Podman コンテナー用に設定されている場合は、この手順に **sudo** は必要ありません。

2. **operator.yml** ファイルをダウンロードします。

```
$ sudo podman cp $(sudo podman create registry.redhat.io/rhcam-1-2/openshift-migration-rhel7-operator:v1.2):/operator.yml ./
```

3. **controller-3.yml** ファイルをダウンロードします。

```
$ sudo podman cp $(sudo podman create registry.redhat.io/rhcam-1-2/openshift-migration-rhel7-operator:v1.2):/controller-3.yml ./
```

4. OpenShift Container Platform 3 クラスターにログインします。

5. クラスターが **registry.redhat.io** で認証できることを確認します。

```
$ oc run test --image registry.redhat.io/ubi8 --command sleep infinity
```

6. Cluster Application Migration Operator CR オブジェクトを作成します。

```
$ oc create -f operator.yml
```

出力は以下のようになります。

```
namespace/openshift-migration created
rolebinding.rbac.authorization.k8s.io/system:deployers created
serviceaccount/migration-operator created
customresourcedefinition.apiextensions.k8s.io/migrationcontrollers.migration.openshift.io
created
role.rbac.authorization.k8s.io/migration-operator created
rolebinding.rbac.authorization.k8s.io/migration-operator created
clusterrolebinding.rbac.authorization.k8s.io/migration-operator created
deployment.apps/migration-operator created
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-builders" already exists (1)
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-pullers" already exists
```

(1) **Error from server (AlreadyExists)** メッセージは無視できます。これらは、Cluster Applicatino Migration Operator が以降のリリースで提供されるリリースを OpenShift Container Platform 3 の以前のバージョン用に作成することによって生じます。

7. Migration コントローラー CR オブジェクトを作成します。

```
$ oc create -f controller-3.yml
```

8. Velero および Restic Pod が実行されていることを確認します。

```
$ oc get pods -n openshift-migration
```

1.4.2. 制限された環境での Cluster Application Migration Operator のインストール

Cluster Application Migration Operator は Operator Lifecycle Manager (OLM) で OpenShift Container Platform 4.4 ターゲットクラスターにインストールすることも、手動で OpenShift Container Platform 3 ソースクラスターにインストールすることもできます。

OpenShift Container Platform 4.4 では、カスタム Operator カタログイメージをビルドし、これをローカルミラーイメージレジストリーにプッシュし、OLM をローカルレジストリーから Cluster Application Migration Operator をインストールするように設定できます。**mapping.txt** ファイルは、**oc adm catalog mirror** コマンドの実行時に作成されます。

OpenShift Container Platform 3 クラスターでは、Operator イメージに基づいてマニフェストファイルを作成し、ファイルをローカルイメージレジストリーを参照するように編集できます。マニフェストファイルの **image** の値は、**mapping.txt** ファイルの **sha256** の値を使用します。次に、ローカルイメージを使用して Cluster Application Migration Operator を作成できます。

追加リソース

- ネットワークが制限された環境での Operator Lifecycle Manager の使用

1.4.2.1. Operator カタログイメージのビルド

クラスター管理者は、Operator Lifecycle Manager (OLM) によって使用されるカスタム Operator カタログイメージをビルドし、Docker v2-2 をサポートするコンテナーイメージレジストリーにそのイメージをプッシュできます。ネットワークが制限された環境のクラスターの場合、このレジストリーには、ネットワークが制限されたインストールで作成されたミラーレジストリーなど、クラスターにネットワークアクセスのあるレジストリーを使用できます。

重要

OpenShift Container Platform クラスターの内部レジストリーはターゲットレジストリーとして使用できません。これは、ミラーリングプロセスで必要となるタグを使わないプッシュをサポートしないためです。

以下の例では、お使いのネットワークとインターネットの両方にアクセスできるミラーレジストリーを使用することを前提としています。

前提条件

- ネットワークアクセスが無制限の Linux ワークステーション
- **oc** version 4.3.5+
- **podman** version 1.4.4+
- Docker v2-2 をサポートするミラーレジストリーへのアクセス
- プライベートレジストリーを使用している場合、後続の手順で使用するために **REG_CREDS** 環境変数をレジストリー認証情報のファイルパスに設定します。たとえば **podman** CLI の場合は、以下のようになります。

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

- quay.io アカウントがアクセスできるプライベート namespace を使用している場合、Quay 認証トークンを設定する必要があります。quay.io 認証情報を使用してログイン API に対して要求を行うことにより、**--auth-token** フラグで使用できる **AUTH_TOKEN** 環境変数を設定します。

```
$ AUTH_TOKEN=$(curl -sH "Content-Type: application/json" \
    -XPOST https://quay.io/cnr/api/v1/users/login -d '
    {
        "user": {
            "username": """<quay_username>""",
            "password": """<quay_password>"""
        }
    }' | jq -r '.token')
```

手順

1. ネットワークアクセスが無制限のワークステーションで、ターゲットミラーレジストリーを使用して認証を行います。

```
$ podman login <registry_host_name>
```

また、ビルド時にベースイメージをプルできるように、**registry.redhat.io** で認証します。

```
$ podman login registry.redhat.io
```

2. quay.io から **redhat-operators** カタログをベースにカタログイメージをビルドし、そのイメージにタグを付け、ミラーレジストリーにプッシュします。

```
$ oc adm catalog build \
    --appregistry-org redhat-operators \ 1
    --from=registry.redhat.io/openshift4/ose-operator-registry:v4.4 \ 2
    --filter-by-os="linux/amd64" \ 3
    --to=<registry_host_name>:<port>/olm/redhat-operators:v1 \ 4
    [-a ${REG_CREDS}] \ 5
    [--insecure] \ 6
    [--auth-token "${AUTH_TOKEN}"] 7

INFO[0013] loading Bundles
dir=/var/folders/st/9cskxqs53ll3wdn434vw4cd80000gn/T/300666084/manifests-829192605
...
Pushed sha256:f73d42950021f9240389f99ddc5b0c7f1b533c054ba344654ff1edaf6bf827e3
to example_registry:5000/olm/redhat-operators:v1
```

1. App Registry インスタンスからのプルに使用する組織 (namespace)。
2. ターゲット OpenShift Container Platform クラスターのメジャーバージョンおよびマイナーバージョンに一致するタグを使用して、**--from** を **ose-operator-registry** ベースイメージに設定します。
3. **--filter-by-os** を、ターゲットの OpenShift Container Platform クラスターと一致する必要のある、ベースイメージに使用するオペレーティングシステムおよびアーキテクチャーに設定します。使用できる値は、**linux/amd64**、**linux/ppc64le**、および **linux/s390x** です。
4. カタログイメージに名前を付け、**v1** などのタグを追加します。
5. オプション: 必要な場合は、レジストリー認証情報ファイルの場所を指定します。
6. オプション: ターゲットレジストリーの信頼を設定しない場合は、**--insecure** フラグを追加します。
7. オプション: 公開されていない他のアプリケーションレジストリーカタログが使用されている場合、Quay 認証トークンを指定します。

無効なマニフェストが Red Hat のカタログに誤って導入される可能性があります。これが実際に生じる場合には、以下のようなエラーが表示される可能性があります。

```
...
INFO[0014] directory
dir=/var/folders/st/9cskxqs53ll3wdn434vw4cd80000gn/T/300666084/manifests-829192605
file=4.2 load=package
```

```
W1114 19:42:37.876180   34665 builder.go:141] error building database: error loading
package into db: fuse-camel-k-operator.v7.5.0 specifies replacement that couldn't be found
Uploading ... 244.9kB/s
```

通常、これらのエラーは致命的なエラーではなく、該当する Operator パッケージにインストールする予定の Operator やその依存関係が含まれない場合、それらを無視することができます。

1.4.2.2. ネットワークが制限された環境向けの OperatorHub の設定

クラスター管理者は、カスタム Operator カタログイメージを使用し、OLM および OperatorHub をネットワークが制限された環境でローカルコンテンツを使用するように設定できます。この例では、以前にビルドされ、サポートされているレジストリーにプッシュされたカスタム **redhat-operators** カタログイメージを使用します。

前提条件

- ネットワークアクセスが無制限の Linux ワークステーション
- サポートされているレジストリーにプッシュされるカスタム Operator カタログイメージ
- **oc** version 4.3.5+
- **podman** version 1.4.4+
- Docker v2-2 をサポートするミラーレジストリーへのアクセス
- プライベートレジストリーを使用している場合、後続の手順で使用するために **REG_CREDS** 環境変数をレジストリー認証情報のファイルパスに設定します。たとえば **podman** CLI の場合は、以下のようになります。

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

手順

1. **disableAllDefaultSources: true** を仕様に追加してデフォルトの OperatorSource を無効にします。

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

これにより、OpenShift Container Platform のインストール時にデフォルトで設定されるデフォルトの OperatorSource が無効になります。

2. **oc adm catalog mirror** コマンドは、カスタム Operator カタログイメージのコンテンツを抽出し、ミラーリングに必要なマニフェストを生成します。以下のいずれかを選択できます。

   - コマンドのデフォルト動作で、マニフェストの生成後にすべてのイメージコンテンツをミラーレジストリーに自動的にミラーリングできるようにします。または、
   - **--manifests-only** フラグを追加して、ミラーリングに必要なマニフェストのみを生成しますが、これにより、イメージコンテンツがレジストリーに自動的にミラーリングされる訳ではありません。これは、ミラーリングする内容を確認するのに役立ちます。また、コンテンツのサブセットのみが必要な場合に、マッピングの一覧に変更を加えることができます。次に、そのファイルを **oc image mirror** コマンドで使用し、後のステップでイメージの変更済みの一覧をミラーリングできます。

ネットワークアクセスが無制限のワークステーションで、以下のコマンドを実行します。

```
$ oc adm catalog mirror \
    <registry_host_name>:<port>/olm/redhat-operators:v1 \ 1
    <registry_host_name>:<port> \
    [-a ${REG_CREDS}] \ 2
    [--insecure] \ 3
    [--filter-by-os="<os>/<arch>"] \ 4
    [--manifests-only] 5
```

1. Operator カタログイメージを指定します。
2. オプション: 必要な場合は、レジストリー認証情報ファイルの場所を指定します。
3. オプション: ターゲットレジストリーの信頼を設定しない場合は、**--insecure** フラグを追加します。
4. オプション: カタログは複数のアーキテクチャーおよびオペレーティングシステムをサポートするイメージを参照する可能性があるため、アーキテクチャーおよびオペレーティングシステムでフィルターして、一 致するイメージのみをミラーリングするようにできます。使用できる値は、**linux/amd64**、**linux/ppc64le**、および **linux/s390x** です。
5. オプション: ミラーリングに必要なマニフェストのみを生成し、実際にはイメージコンテンツをレジストリーにミラーリングしません。

出力例

```
using database path mapping: /:/tmp/190214037
wrote database to /tmp/190214037
using database at: /tmp/190214037/bundles.db 1
...
```

1. コマンドで生成される一時的なデータベース。

コマンドの実行後に、**<image_name>-manifests/** ディレクトリーが現在のディレクトリーに作成され、以下のファイルが生成されます。

- これにより、**imageContentSourcePolicy.yaml** ファイルは ImageContentSourcePolicy オブジェクトを定義します。このオブジェクトは、このオブジェクトは、ノードを Operator マニフェストおよびミラーリングされたレジストリーに保存されるイメージ参照間で変換できるように設定します。
- **mapping.txt** ファイルには、すべてのソースイメージが含まれ、これはそれらのイメージをターゲットレジストリー内のどこにマップするかを示します。このファイルは **oc image mirror** コマンドと互換性があり、ミラーリング設定をさらにカスタマイズするために使用できます。

3. 直前の手順で **--manifests-only** フラグを使用して、コンテンツのサブセットのみをミラーリングする場合は、以下を実行します。
    a. **mapping.txt** ファイルのイメージの一覧を仕様に変更します。ミラーリングするイメージのサブセットの名前とバージョンが不明な場合は、以下の手順で確認します。

i. **oc adm catalog mirror** コマンドで生成された一時的なデータベースに対して **sqlite3** ツールを実行し、一般的な検索クエリーに一致するイメージの一覧を取得します。出力は、後に **mapping.txt** ファイルを編集する方法を通知するのに役立ちます。
たとえば、**clusterlogging.4.3** の文字列のようなイメージの一覧を取得するには、以下を実行します。

```
$ echo "select * from related_image \
    where operatorbundle_name like 'clusterlogging.4.3%';" \
    | sqlite3 -line /tmp/190214037/bundles.db 1
```

1 **oc adm catalog mirror** コマンドの直前の出力を参照し、データベースファイルのパスを見つけます。

出力例

```
image = registry.redhat.io/openshift4/ose-logging-
kibana5@sha256:aa4a8b2a00836d0e28aa6497ad90a3c116f135f382d8211e3c55f34f
b36dfe61
operatorbundle_name = clusterlogging.4.3.33-202008111029.p0

image = registry.redhat.io/openshift4/ose-oauth-
proxy@sha256:6b4db07f6e6c962fc96473d86c44532c93b146bbefe311d0c348117bf75
9c506
operatorbundle_name = clusterlogging.4.3.33-202008111029.p0
...
```

ii. 直前の手順で取得した結果を使用して **mapping.txt** ファイルを編集し、ミラーリングする必要のあるイメージのサブセットのみを追加します。
たとえば、前述の出力例の **image** 値を使用して、**mapping.txt** ファイルに以下の一致する行が存在することを確認できます。

mapping.txt の一致するイメージマッピング。

```
registry.redhat.io/openshift4/ose-logging-
kibana5@sha256:aa4a8b2a00836d0e28aa6497ad90a3c116f135f382d8211e3c55f34f
b36dfe61=<registry_host_name>:<port>/openshift4-ose-logging-kibana5:a767c8f0
registry.redhat.io/openshift4/ose-oauth-
proxy@sha256:6b4db07f6e6c962fc96473d86c44532c93b146bbefe311d0c348117bf75
9c506=<registry_host_name>:<port>/openshift4-ose-oauth-proxy:3754ea2b
```

この例では、これらのイメージのみをミラーリングする場合に、**mapping.txt** ファイルの他のすべてのエントリーを削除し、上記の 2 行のみを残します。

b. ネットワークアクセスが無制限のワークステーション上で、変更した **mapping.txt** ファイルを使用し、 **oc image mirror** コマンドを使用してイメージをレジストリーにミラーリングします。

```
$ oc image mirror \
    [-a ${REG_CREDS}] \
    -f ./redhat-operators-manifests/mapping.txt
```

4. ImageContentSourcePolicy を適用します。

```
$ oc apply -f ./redhat-operators-manifests/imageContentSourcePolicy.yaml
```

5. カタログイメージを参照する CatalogSource オブジェクトを作成します。

   a. 仕様を以下のように変更し、これを **catalogsource.yaml** ファイルとして保存します。

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: my-operator-catalog
  namespace: openshift-marketplace
spec:
  sourceType: grpc
  image: <registry_host_name>:<port>/olm/redhat-operators:v1 (1)
  displayName: My Operator Catalog
  publisher: grpc
```

   (1) Operator カタログイメージを指定します。

   b. このファイルを使用して CatalogSource オブジェクトを作成します。

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

6. 以下のリソースが正常に作成されていることを確認します。

   a. Pod を確認します。

```
$ oc get pods -n openshift-marketplace
```

   出力例

```
NAME                                    READY   STATUS    RESTARTS  AGE
my-operator-catalog-6njx6               1/1     Running   0         28s
marketplace-operator-d9f549946-96sgr    1/1     Running   0         26h
```

   b. CatalogSource を確認します。

```
$ oc get catalogsource -n openshift-marketplace
```

   出力例

```
NAME                  DISPLAY               TYPE PUBLISHER  AGE
my-operator-catalog   My Operator Catalog   grpc           5s
```

   c. PackageManifest を確認します。

```
$ oc get packagemanifest -n openshift-marketplace
```

   出力例

```
NAME    CATALOG             AGE
etcd    My Operator Catalog  34s
```

ネットワークが制限された環境の OpenShift Container Platform クラスター Web コンソールで、**OperatorHub** ページから Operator をインストールできます。

1.4.2.3. 制限された環境での Cluster Application Migration Operator の OpenShift Container Platform 4.4 ターゲットクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4.4 ターゲットクラスターに Cluster Application Migration Operator をインストールできます。

Cluster Application Migration Operator は、デフォルトで Cluster Application Migration ツールをターゲットクラスターにインストールします。

前提条件

- カスタム Operator カタログを作成し、これをミラーレジストリーにプッシュしていること。
- ミラーレジストリーから Cluster Application Migration Operator をインストールするように OLM を設定していること。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
   Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **Create** をクリックします。
8. **Workloads → Pods** をクリックし、Controller Manager、Migration UI、Restic、および Velero Pod が実行中であることを確認します。

1.4.2.4. 制限された環境での Cluster Application Migration Operator の OpenShift Container Platform 3 ソースクラスターへのインストール

Cluster Application Migration Operator イメージに基づいてマニフェストファイルを作成し、マニフェストを編集してローカルイメージレジストリーを参照できます。次に、ローカルイメージを使用して Cluster Application Migration Operator を OpenShift Container Platform 3 ソースクラスターに作成できます。

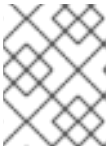
前提条件

- **registry.redhat.io** へのアクセス
- ネットワークアクセスが無制限の Linux ワークステーション
- Docker v2-2 をサポートするミラーレジストリー
- ミラーレジストリーにプッシュされるカスタム Operator カタログ

手順

1. ネットワークアクセスが無制限のワークステーションで、Red Hat カスタマーポータルの認証情報を使用して **registry.redhat.io** にログインします。

```
$ sudo podman login registry.redhat.io
```



注記

システムがルートレス Podman コンテナー用に設定されている場合は、この手順に **sudo** は必要ありません。

2. **operator.yml** ファイルをダウンロードします。

```
$ sudo podman cp $(sudo podman create registry.redhat.io/rhcam-1-2/openshift-migration-rhel7-operator:v1.2):/operator.yml ./
```

3. **controller-3.yml** ファイルをダウンロードします。

```
$ sudo podman cp $(sudo podman create registry.redhat.io/rhcam-1-2/openshift-migration-rhel7-operator:v1.2):/controller-3.yml ./
```

4. OpenShift Container Platform 4 クラスターで **oc adm catalog mirror** の実行時に作成された **mapping.txt** ファイルから Operator イメージの値を取得します。

```
$ grep openshift-migration-rhel7-operator ./mapping.txt | grep rhcam-1-2
```

出力には、**registry.redhat.io** イメージとミラーレジストリーイメージ間のマッピングが表示されます。

```
registry.redhat.io/rhcam-1-2/openshift-migration-rhel7-operator@sha256:468a6126f73b1ee12085ca53a312d1f96ef5a2ca03442bcb63724af5e2614e8a=<registry.apps.example.com>/rhcam-1-2/openshift-migration-rhel7-operator
```

5. **operator.yml** ファイルの **image** および **REGISTRY** の値を更新します。

```
containers:
  - name: ansible
    image: <registry.apps.example.com>/rhcam-1-2/openshift-migration-rhel7-operator@sha256:
<468a6126f73b1ee12085ca53a312d1f96ef5a2ca03442bcb63724af5e2614e8a> (1)
...
  - name: operator
    image: <registry.apps.example.com>/rhcam-1-2/openshift-migration-rhel7-operator@sha256:
```

```
<468a6126f73b1ee12085ca53a312d1f96ef5a2ca03442bcb63724af5e2614e8a> (2)
...
    env:
    - name: REGISTRY
      value: <registry.apps.example.com> (3)
```

(1) (2) ミラーレジストリー、および **mapping.txt** ファイルで Operator イメージの **sha256** の値を指定します。

(3) ミラーレジストリーを指定します。

6. OpenShift Container Platform 3 クラスターにログインします。
7. Cluster Application Migration Operator CR オブジェクトを作成します。

```
$ oc create -f operator.yml
```

出力は以下のようになります。

```
namespace/openshift-migration created
rolebinding.rbac.authorization.k8s.io/system:deployers created
serviceaccount/migration-operator created
customresourcedefinition.apiextensions.k8s.io/migrationcontrollers.migration.openshift.io
created
role.rbac.authorization.k8s.io/migration-operator created
rolebinding.rbac.authorization.k8s.io/migration-operator created
clusterrolebinding.rbac.authorization.k8s.io/migration-operator created
deployment.apps/migration-operator created
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-builders" already exists (1)
Error from server (AlreadyExists): error when creating "./operator.yml":
rolebindings.rbac.authorization.k8s.io "system:image-pullers" already exists
```

(1) **Error from server (AlreadyExists)** メッセージは無視できます。これらは、Cluster Applicatino Migration Operator が以降のリリースで提供されるリリースを OpenShift Container Platform 3 の以前のバージョン用に作成することによって生じます。

8. Migration コントローラー CR オブジェクトを作成します。

```
$ oc create -f controller-3.yml
```

9. Velero および Restic Pod が実行されていることを確認します。

```
$ oc get pods -n openshift-migration
```

1.4.3. CAM Web コンソールの起動

ブラウザーで CAM Web コンソールを起動できます。

手順

1. CAM ツールがインストールされている OpenShift Container Platform クラスターにログインします。
2. 以下のコマンドを入力して CAM Web コンソール URL を取得します。

```
$ oc get -n openshift-migration route/migration -o go-template='https://{{ .spec.host }}'
```

出力は **https://migration-openshift-migration.apps.cluster.openshift.com** のようになります。

3. ブラウザーを起動し、CAM Web コンソールに移動します。

注記

Cluster Application Migration Operator のインストール後すぐに CAM Web コンソールにアクセスしようとする場合、Operator は依然としてクラスターを設定しているため、コンソールが読み込まれないことがあります。数分待機した後に再試行します。

4. 自己署名 CA 証明書を使用している場合、ソースクラスターの API サーバーの CA 証明書を受け入れることを求めるプロンプトが出されます。Web ページは、残りの証明書を受け入れるプロセスについて説明します。
5. OpenShift Container Platform の **ユーザー名** および **パスワード** を使用してログインします。

1.4.4. Cluster Application Migration ツールのアップグレード

ソースおよびターゲットクラスターで Cluster Application Migration (CAM) ツールをアップグレードできます。

CAM 1.1 から 1.2 にアップグレードする場合、CAM Web コンソールでサービスアカウントトークンを更新する必要があります。

1.4.4.1. OpenShift Container Platform 4 クラスターでの CAM ツールのアップグレード

Operator Lifecycle Manager を使用して OpenShift Container Platform 4 クラスターで CAM ツールをアップグレードできます。

Cluster Application Migration Operator にサブスクライブする際に **自動** 承認オプションを選択した場合、CAM ツールは自動的に更新されます。

以下の手順では、**手動** 承認オプションを **自動** に変更するか、またはリリースチャネルを変更できます。

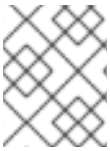
手順

1. OpenShift Container Platform コンソールで、**Operators** > **Installed Operators** に移動します。
2. **Cluster Application Migration Operator** をクリックします。
3. **Subscription** タブで、**Approval** オプションを **Automatic** に変更します。
4. オプション: **チャネル** を編集します。

サブスクリプションを更新すると、更新された Cluster Application Migration Operator がデプロイされ、CAM ツールコンポーネントが更新されます。

1.4.4.2. OpenShift Container Platform 3 クラスターでの CAM ツールのアップグレード

最新の **operator.yml** ファイルをダウンロードし、既存の Cluster Application Migration Operator CR オブジェクトを置き換えて、OpenShift Container Platform 3 クラスターで CAM ツールをアップグレードできます。



注記

namespace を削除し、再作成する場合、CAM Web コンソールでクラスターのサービスアカウントトークンを更新する必要があります。

手順

1. Red Hat カスタマーポータルの認証情報を使用して **registry.redhat.io** にログインします。

```
$ sudo podman login registry.redhat.io
```

2. 最新の **operator.yml** ファイルをダウンロードします。

```
$ sudo podman cp $(sudo podman create registry.redhat.io/rhcam-1-2/openshift-migration-rhel7-operator:v1.2):/operator.yml ./
```

3. OpenShift Container Platform 3 クラスターにログインします。
4. 更新された Cluster Application Migration Operator CR オブジェクトをデプロイします。

```
$ oc replace -f operator.yml
```

5. Restic Pod を取得します。

```
$ oc get pod -n openshift-migration | grep restic
```

6. Restic Pod を削除して、アップグレードが再起動時に適用されるようにします。

```
$ oc delete pod <restic_pod>
```

1.4.4.3. サービスアカウントトークンの更新

CAM 1.1 から 1.2 にアップグレードする場合、CAM Web コンソールでサービスアカウントトークンを更新する必要があります。

CAM 1.1 は **mig** サービスアカウントを使用し、CAM 1.2 は **migration-controller** サービスアカウントを使用します。

手順

1. クラスターにログインし、**migration-controller** サービスアカウントトークンを取得します。

```
$ oc sa get-token -n openshift-migration migration-controller
```

2. CAM Web コンソールにログインし、**Clusters** をクリックします。
3. クラスターの Options メニュー  をクリックし、**Edit** を選択します。
4. 新たなトークンを **Service account token** フィールドにコピーします。
5. **Update cluster** をクリックしてから、**Close** をクリックします。
クラスターのサービスアカウントトークンが更新されます。

1.5. レプリケーションリポジトリーの設定

オブジェクトストレージをレプリケーションリポジトリーとして使用するように設定する必要があります。Cluster Application Migration ツールは、データをソースクラスターからレプリケーションリポジトリーにコピーしてから、レプリケーションリポジトリーからターゲットクラスターにコピーします。

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するために ファイルシステムおよびスナップショットによるデータのコピー方法 をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

以下のストレージプロバイダーがサポートされています。

- Multi-Cloud Object Gateway (MCG)
- Amazon Web Services (AWS) S3
- Google Cloud Provider (GCP)
- Microsoft Azure
- 汎用 S3 オブジェクトストレージ (例: Minio または Ceph S3)

ソースおよびターゲットクラスターには、移行時にレプリケーションリポジトリーへのネットワークアクセスがなければなりません。

制限された環境では、内部でホストされるレプリケーションリポジトリーを作成できます。プロキシーサーバーを使用する場合は、レプリケーションリポジトリーがホワイトリスト化されていることを確認する必要があります。

1.5.1. Multi-Cloud Object Gateway ストレージバケットをレプリケーションリポジトリーとして設定する

OpenShift Container Storage Operator をインストールし、Multi-Cloud Object Gateway (MCG) ストレージバケットをレプリケーションリポジトリーとして設定できます。

1.5.1.1. OpenShift Container Storage Operator のインストール

OpenShift Container Storage Operator は、OperatorHub からインストールできます。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。

2. **Filter by keyword** (この場合は、**OCS**) を使用し、 **OpenShift Container Storage Operator** を見つけます。

3. **OpenShift Container Storage Operator** を選択し、**Install** をクリックします。

4. **Update Channel**、**Installation Mode**、および **Approval Strategy** を選択します。

5. **Subscribe** をクリックします。
Installed Operators ページで、**OpenShift Container Storage Operator** は、**Succeeded** のステータスと共に **openshift-storage** プロジェクトに表示されます。

1.5.1.2. Multi-Cloud Object Gateway ストレージバケットの作成

Multi-Cloud Object Gateway (MCG) ストレージバケットのカスタムリソース (CR) を作成できます。

手順

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

```
$ oc login
```

2. **NooBaa** CR 設定ファイル **noobaa.yml** を以下の内容で作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: NooBaa
metadata:
  name: noobaa
  namespace: openshift-storage
spec:
 dbResources:
   requests:
     cpu: 0.5 (1)
     memory: 1Gi
 coreResources:
   requests:
     cpu: 0.5 (2)
     memory: 1Gi
```

(1) (2) 非常に小規模なクラスターの場合、**cpu** の値を **0.1** に変更できます。

3. **NooBaa** オブジェクトを作成します。

```
$ oc create -f noobaa.yml
```

4. 以下の内容で、**BackingStore** CR 設定ファイル **bs.yml** を作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: BackingStore
metadata:
  finalizers:
  - noobaa.io/finalizer
  labels:
    app: noobaa
  name: mcg-pv-pool-bs
```

```
  namespace: openshift-storage
spec:
  pvPool:
    numVolumes: 3 (1)
    resources:
      requests:
        storage: 50Gi (2)
    storageClass: gp2 (3)
  type: pv-pool
```

(1) PV プール内のボリューム数を指定します。

(2) ボリュームのサイズを指定します。

(3) ストレージクラスを作成します。

5. **BackingStore** オブジェクトを作成します。

```
$ oc create -f bs.yml
```

6. 以下の内容で **BucketClass** CR 設定ファイル **bc.yml** を作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: BucketClass
metadata:
  labels:
    app: noobaa
  name: mcg-pv-pool-bc
  namespace: openshift-storage
spec:
  placementPolicy:
    tiers:
    - backingStores:
      - mcg-pv-pool-bs
      placement: Spread
```

7. **BucketClass** オブジェクトを作成します。

```
$ oc create -f bc.yml
```

8. 以下の内容で **ObjectBucketClaim** CR 設定ファイル **obc.yml** を作成します。

```
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: migstorage
  namespace: openshift-storage
spec:
  bucketName: migstorage (1)
  storageClassName: openshift-storage.noobaa.io
  additionalConfig:
    bucketclass: mcg-pv-pool-bc
```

1 レプリケーションリポジトリーを CAM Web コンソールに追加するために使用するバケット名を記録します。

9. **ObjectBucketClaim** オブジェクトを作成します。

```
$ oc create -f obc.yml
```

10. リソース作成プロセスを監視し、**ObjectBucketClaim** ステータスが **Bound** であることを確認します。

```
$ watch -n 30 'oc get -n openshift-storage objectbucketclaim migstorage -o yaml'
```

このプロセスには、5 分から 10 分の時間がかかる場合があります。

11. 以下の値を取得して記録します。この値は、レプリケーションリポジトリーを CAM Web コンソールに追加する際に必要になります。

- S3 エンドポイント:

```
$ oc get route -n openshift-storage s3
```

- S3 プロバイダーアクセスキー:

```
$ oc get secret -n openshift-storage migstorage -o go-template='{{
.data.AWS_ACCESS_KEY_ID }}' | base64 -d
```

- S3 プロバイダーシークレットアクセスキー:

```
$ oc get secret -n openshift-storage migstorage -o go-template='{{
.data.AWS_SECRET_ACCESS_KEY }}' | base64 -d
```

1.5.2. AWS S3 ストレージバケットをレプリケーションリポジトリーとして設定する

AWS S3 ストレージバケットをレプリケーションリポジトリーとして設定できます。

前提条件

- AWS S3 ストレージバケットは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。
- AWS CLI がインストールされていること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - EC2 Elastic Block Storage (EBS) にアクセスできる必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. AWS S3 バケットを作成します。

```
$ aws s3api create-bucket \
    --bucket <bucket_name> \ 1
    --region <bucket_region> 2
```

1 S3 バケット名を指定します。

2 S3 バケットリージョンを指定します (例: **us-east-1**)。

2. IAM ユーザー **velero** を作成します。

```
$ aws iam create-user --user-name velero
```

3. EC2 EBS スナップショットポリシーを作成します。

```
$ cat > velero-ec2-snapshot-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes",
                "ec2:DescribeSnapshots",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:CreateSnapshot",
                "ec2:DeleteSnapshot"
            ],
            "Resource": "*"
        }
    ]
}
EOF
```

4. 1 つまたはすべての S3 バケットの AWS S3 アクセスポリシーを作成します。

```
$ cat > velero-s3-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket_name>/*" 1
            ]
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket_name>" 2
            ]
        }
    ]
}
EOF
```

1 2 単一の S3 バケットへのアクセスを付与するには、バケット名を指定します。すべての AWS S3 バケットへのアクセスを付与するには、バケット名の代わりに * を指定します。

```
"Resource": [
    "arn:aws:s3:::*"
```

5. EC2 EBS ポリシーを **velero** に割り当てます。

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero-ebs \
  --policy-document file://velero-ec2-snapshot-policy.json
```

6. AWS S3 ポリシーを **velero** に割り当てます。

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero-s3 \
  --policy-document file://velero-s3-policy.json
```

7. **velero** のアクセスキーを作成します。

```
$ aws iam create-access-key --user-name velero
{
  "AccessKey": {
        "UserName": "velero",
        "Status": "Active",
        "CreateDate": "2017-07-31T22:24:41.576Z",
        "SecretAccessKey": <AWS_SECRET_ACCESS_KEY>, 1
        "AccessKeyId": <AWS_ACCESS_KEY_ID> 2
    }
}
```

1 2 AWS リポジトリーを CAM Web コンソールに追加するために **AWS_SECRET_ACCESS_KEY** および **AWS_ACCESS_KEY_ID** を記録します。

1.5.3. Google Cloud Provider ストレージバケットをレプリケーションリポジトリーとして設定する

Google Cloud Provider (GCP) ストレージバケットをレプリケーションリポジトリーとして設定できます。

前提条件

- GCP ストレージバケットは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。
- gsutil がインストールされていること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. **gsutil init** を実行してログインします。

```
$ gsutil init
Welcome! This command will take you through the configuration of gcloud.

Your current configuration has been set to: [default]

To continue, you must login. Would you like to login (Y/n)?
```

2. **BUCKET** 変数を設定します。

```
$ BUCKET=<bucket_name> 1
```

1 バケット名を指定します。

3. ストレージバケットを作成します。

```
$ gsutil mb gs://$BUCKET/
```

4. **PROJECT_ID** 変数をアクティブなプロジェクトに設定します。

```
$ PROJECT_ID=$(gcloud config get-value project)
```

5. **velero** サービスアカウントを作成します。

```
$ gcloud iam service-accounts create velero \
    --display-name "Velero Storage"
```

6. **SERVICE_ACCOUNT_EMAIL** 変数をサービスアカウントのメールアドレスに設定します。

```
$ SERVICE_ACCOUNT_EMAIL=$(gcloud iam service-accounts list \
  --filter="displayName:Velero Storage" \
  --format 'value(email)')
```

7. パーミッションをサービスアカウントに付与します。

```
$ ROLE_PERMISSIONS=(
    compute.disks.get
    compute.disks.create
    compute.disks.createSnapshot
    compute.snapshots.get
    compute.snapshots.create
    compute.snapshots.useReadOnly
    compute.snapshots.delete
    compute.zones.get
)

gcloud iam roles create velero.server \
    --project $PROJECT_ID \
    --title "Velero Server" \
    --permissions "$(IFS=","; echo "${ROLE_PERMISSIONS[*]}")"

gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member serviceAccount:$SERVICE_ACCOUNT_EMAIL \
    --role projects/$PROJECT_ID/roles/velero.server

gsutil iam ch serviceAccount:$SERVICE_ACCOUNT_EMAIL:objectAdmin gs://${BUCKET}
```

8. サービスアカウントのキーを現在のディレクトリーにある **credentials-velero** ファイルに保存します。

```
$ gcloud iam service-accounts keys create credentials-velero \
  --iam-account $SERVICE_ACCOUNT_EMAIL
```

1.5.4. Microsoft Azure Blob ストレージコンテナーをレプリケーションリポジトリーとして設定

Microsoft Azure Blob ストレージコンテナーをレプリケーションリポジトリーとして設定できます。

前提条件

- Azure ストレージアカウント があること。
- Azure CLI がインストールされていること。
- Azure Blob ストレージコンテナーがソースクラスターおよびターゲットクラスターからアクセスできること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。

- ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. **AZURE_RESOURCE_GROUP** 変数を設定します。

```
$ AZURE_RESOURCE_GROUP=Velero_Backups
```

2. Azure リソースグループを作成します。

```
$ az group create -n $AZURE_RESOURCE_GROUP --location <CentralUS> 1
```

1 場所を指定します。

3. **AZURE_STORAGE_ACCOUNT_ID** 変数を設定します。

```
$ AZURE_STORAGE_ACCOUNT_ID=velerobackups
```

4. Azure ストレージアカウントを作成します。

```
$ az storage account create \
  --name $AZURE_STORAGE_ACCOUNT_ID \
  --resource-group $AZURE_RESOURCE_GROUP \
  --sku Standard_GRS \
  --encryption-services blob \
  --https-only true \
  --kind BlobStorage \
  --access-tier Hot
```

5. **BLOB_CONTAINER** 変数を設定します。

```
$ BLOB_CONTAINER=velero
```

6. Azure Blob ストレージコンテナーを作成します。

```
$ az storage container create \
  -n $BLOB_CONTAINER \
  --public-access off \
  --account-name $AZURE_STORAGE_ACCOUNT_ID
```

7. **velero** のサービスプリンシパルおよび認証情報を作成します。

```
$ AZURE_SUBSCRIPTION_ID=`az account list --query '[?isDefault].id' -o tsv`
$ AZURE_TENANT_ID=`az account list --query '[?isDefault].tenantId' -o tsv`
$ AZURE_CLIENT_SECRET=`az ad sp create-for-rbac --name "velero" --role "Contributor" --
query 'password' -o tsv`
$ AZURE_CLIENT_ID=`az ad sp list --display-name "velero" --query '[0].appId' -o tsv`
```

8. サービスプリンシパルの認証情報を **credentials-velero** ファイルに保存します。

```
$ cat << EOF  > ./credentials-velero
AZURE_SUBSCRIPTION_ID=${AZURE_SUBSCRIPTION_ID}
```

```
AZURE_TENANT_ID=${AZURE_TENANT_ID}
AZURE_CLIENT_ID=${AZURE_CLIENT_ID}
AZURE_CLIENT_SECRET=${AZURE_CLIENT_SECRET}
AZURE_RESOURCE_GROUP=${AZURE_RESOURCE_GROUP}
AZURE_CLOUD_NAME=AzurePublicCloud
EOF
```

1.6. CAM WEB コンソールを使用したアプリケーションの移行

アプリケーションワークロードの移行は、クラスターおよびレプリケーションリポジトリーを CAM Web コンソールに追加することによって実行できます。次に、移行計画を作成し、これを実行できます。

クラスターまたはレプリケーションリポジトリーのセキュリティーを自己署名証明書で保護する場合、CA 証明書バンドルファイルを作成するか、または SSL 検証を無効にできます。

1.6.1. CA 証明書バンドルファイルの作成

自己署名証明書を使用してクラスターまたはレプリケーションリポジトリーのセキュリティーを保護する場合、証明書の検証は **Certificate signed by unknown authority** というエラーメッセージを出して失敗する可能性があります。

カスタム CA 証明書バンドルファイルを作成し、クラスターまたはレプリケーションリポジトリーの追加時に CAM Web コンソールでこれをアップロードできます。

手順

リモートエンドポイントから CA 証明書をダウンロードし、これを CA バンドルファイルとして保存します。

```
$ echo -n | openssl s_client -connect <host_FQDN>:<port> \ 1
  | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > <ca_bundle.cert> 2
```

1. エンドポイントのホスト FQDN およびポートを指定します (例: **api.my-cluster.example.com:6443**)。
2. CA バンドルファイルの名前を指定します。

1.6.2. CAM Web コンソールへのクラスターの追加

クラスターを CAM Web コンソールに追加できます。

前提条件

Azure スナップショットを使用してデータをコピーする場合:

- ソースクラスターの追加時に Azure リソースグループ名を指定する必要があります。
- ソースおよびターゲットクラスターは同じ Azure リソースグループにあり、同じ場所にある必要があります。

手順

1. クラスターにログインします。

2. サービスアカウントトークンを取得します。

```
$ oc sa get-token migration-controller -n openshift-migration
eyJhbGciOiJSUzI1NiIsImtpZCI6IiJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwi
a3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJtaWciLCJrdWJlcm5ldGVz
LmlvL3NlcnZpY2VhY2NvdW50L3NlY3JldC5uYW1lIjoibWlnLXRva2VuLWs4dDJyIiwia3ViZXJuZ
XRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6Im1pZyIsImt1YmV
ybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50LnVpZCI6ImE1YjFiYWM
wLWMxYmYtMTFlOS05Y2NiLTAyOWRmODYwYjMwOCIsInN1YiI6InN5c3RlbTpzZXJ2aWNlY
WNjb3VudDptaWc6bWlnIn0.xqeeAINK7UXpdRqAtOj70qhBJPeMwmgLomV9iFxr5RoqUgKchZ
RG2J2rkqmPm6vr7K-
cm7ibD1IBpdQJCcVDuoHYsFgV4mp9vgOfn9osSDp2TGikwNz4Az95e81xnjVUmzh-
NjDsEpw71DH92iHV_xt2sTwtzftS49LpPW2LjrV0evtNBP_t_RfskdArt5VSv25eORl7zScqfe1CiM
kcVbf2UqACQjo3LbkpfN26HAioO2oH0ECPiRzT0Xyh-KwFutJLS9Xgghyw-
LD9kPKcE_xbbJ9Y4Rqajh7WdPYuB0Jd9DPVrslmzK-F6cgHHYoZEv0SvLQi-
PO0rpDrcjOEQQ
```

3. CAM Web コンソールにログインします。

4. **Clusters** セクションで、**Add cluster** をクリックします。

5. 以下のフィールドに値を入力します。

   - **Cluster name**: 小文字 (**a-z**) および数字 (**0-9**) を含めることができます。空白文字や国際文字を含めることはできません。
   - **Url**: クラスターの API サーバーの URL です (例: **https://<master1.example.com>:8443**)。
   - **Service account token**: ソースクラスターから取得される文字列。
   - **Azure cluster**: オプション。Azure スナップショットを使用してデータをコピーする場合はこれを選択します。
   - **Azure resource group**: このフィールドは、**Azure cluster** にチェックを付けると表示されます。
   - カスタム CA バンドルを使用する場合は、**Browse** をクリックし、CA バンドルファイルを参照します。

6. **Add cluster** をクリックします。
   クラスターが **Clusters** セクションに表示されます。

1.6.3. CAM Web コンソールへのレプリケーションリポジトリーの追加

CAM Web コンソールには、オブジェクトストレージバケットをレプリケーションリポジトリーとして追加できます。

前提条件

- データを移行するには、オブジェクトストレージバケットを設定する必要があります。

手順

1. CAM Web コンソールにログインします。

2. **Replication repositories** セクションで、**Add repository** をクリックします。

3. **Storage provider type** を選択し、以下のフィールドに入力します。
   - **AWS** (AWS S3、MCG、および汎用 S3 プロバイダーの場合):
     - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
     - **S3 bucket name**: 作成した S3 バケットの名前を指定します。
     - **S3 bucket region**: S3 バケットリージョンを指定します。AWS S3 の場合に**必須**です。**Optional** (他の S3 プロバイダーの場合)。
     - **S3 endpoint**: バケットではなく S3 サービスの URL を指定します (例: **https://<s3-storage.apps.cluster.com>**)。汎用 S3 プロバイダーの場合は**必須**です。**https://** 接頭辞を使用する必要があります。
     - **S3 provider access key**: AWS には **<AWS_SECRET_ACCESS_KEY>** を指定するか、または MCG には S3 プロバイダーアクセスキーを指定します。
     - **S3 provider secret access key**: AWS には **<AWS_ACCESS_KEY_ID>** を指定するか、または MCG には S3 プロバイダーシークレットアクセスキーを指定します。
     - **Require SSL verification**: 汎用 S3 プロバイダーを使用している場合は、このチェックボックスをクリアします。
     - カスタム CA バンドルを使用する場合は、**Browse** をクリックし、Base64 でエンコードされた CA バンドルファイルを参照します。
   - **GCP**:
     - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
     - **GCP bucket name**: GCP バケットの名前を指定します。
     - **GCP credential JSON blob**: **credentials-velero** ファイルに文字列を指定します。
   - **Azure**:
     - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
     - **Azure resource group**: Azure Blob ストレージのリソースグループを指定します。
     - **Azure storage account name**: Azure Blob ストレージアカウント名を指定します。
     - **Azure credentials - INI file contents** **credentials-velero** ファイルに文字列を指定します。
4. **Add repository** をクリックし、接続の検証を待機します。
5. **Close** をクリックします。
   新規リポジトリーが **Replication repositories** セクションに表示されます。

1.6.4. 大規模な移行の場合の移行計画の制限の変更

大規模な移行の移行計画の制限を変更することができます。

重要

移行の失敗を防ぐために、まず変更についてのテストをご使用の環境で実行する必要があります。

単一の移行計画には以下のデフォルト制限があります。

- 10 namespace
  この制限を超えると、CAM Web コンソールは **Namespace limit exceeded**エラーを表示し、移行計画を作成することができません。

- 100 Pod
  Pod の制限を超える場合、CAM Web コンソールには、以下の例と同様の警告メッセージが表示されます。**Plan has been validated with warning condition(s).See warning message.Pod limit: 100 exceeded, found: 104**

- 100 永続ボリューム
  永続ボリュームの制限を超過すると、CAM Web コンソールには同様の警告メッセージが表示されます。

手順

1. Migration コントローラー CR を編集します。

```
$ oc get migrationcontroller -n openshift-migration
NAME AGE
migration-controller 5d19h

$ oc edit migrationcontroller -n openshift-migration
```

2. 以下のパラメーターを更新します。

```
...
migration_controller: true

# This configuration is loaded into mig-controller, and should be set on the
# cluster where `migration_controller: true`
mig_pv_limit: 100
mig_pod_limit: 100
mig_namespace_limit: 10
...
```

1.6.5. CAM Web コンソールでの移行計画の作成

CAM Web コンソールで移行計画を作成できます。

前提条件

- CAM Web コンソールには以下が含まれている必要があります。
  - ソースクラスター
  - CAM ツールのインストール時に自動的に追加されるターゲットクラスター

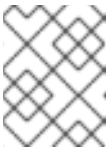
- レプリケーションリポジトリー

- ソースおよびターゲットクラスターには、相互に対するネットワークアクセスやレプリケーションリポジトリーへのネットワークアクセスがなければなりません。

- スナップショットを使用してデータをコピーする場合、ソースおよびターゲットクラスターは、同じクラウドプロバイダー (AWS、GCP、または Azure) および同じリージョンで実行される必要があります。

手順

1. CAM Web コンソールにログインします。
2. **Plans** セクションで、**Add plan** をクリックします。
3. **Plan name** を入力し、**Next** をクリックします。
   Plan name には、最大 253 文字の小文字の英数字 (**a-z、0-9**) を含めることができます。スペースまたはアンダースコア (_) を含めることはできません。
4. **Source cluster** を選択します。
5. **Target cluster**を選択します。
6. **Replication repository** を選択します。
7. 移行するプロジェクトを選択し、**Next** をクリックします。
8. PV の **Copy** または **Move** を選択します。
   - **Copy** は、ソースクラスターの PV のデータをレプリケーションリポジトリーにコピーしてから、これを同様の特徴のある新規に作成された PV で復元します。
     オプション: **Verify copy** を選択して、ファイルシステムメソッドでコピーされたデータを検証できます。このオプションを使用すると、各ソースファイルのチェックサムが生成され、復元後のチェックが実行されるため、パフォーマンスが大幅に低下します。
   - **Move** は、ソースクラスターからリモートボリューム (例: NFS) をアンマウントし、リモートボリュームをポイントするターゲットクラスターで PV リソースを作成し、その後にリモートボリュームをターゲットクラスターにマウントします。ターゲットクラスターで実行されているアプリケーションは、ソースクラスターが使用していたものと同じリモートボリュームを使用します。リモートボリュームは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。
9. **Next** をクリックします。
10. PV の **Copy method** を選択します。
    - **Snapshot** は、クラウドプロバイダーのスナップショット機能を使用してディスクのバックアップおよび復元を行います。この場合、**ファイルシステム**を使用する場合よりもはるかに高速になります。



注記

ストレージおよびクラスターは同じリージョンにあり、ストレージクラスには互換性がなければなりません。

- **ファイルシステム**は、データファイルをソースディスクから新規に作成されたターゲットディスクにコピーします。

11. PV の **Storage class** を選択します。
Filesystem のコピー方法を選択した場合、移行時にストレージクラスを変更できます。たとえば、Red Hat Gluster Storage または NFS ストレージから Red Hat Ceph Storage に変更できます。

12. **Next** をクリックします。

13. 移行フックを追加する必要がある場合は、**Add Hook** をクリックして、以下の手順を実行します。

    a. フックの名前を指定します。

    b. **Ansible playbook** を選択して独自の Playbook を使用するか、または別の言語で作成されがフックに **Custom container image** を選択します。

    c. **Browse** をクリックして Playbook をアップロードします。

    d. オプション: デフォルトの Ansible ランタイムイメージを使用していない場合は、カスタムの Ansible イメージを指定します。

    e. フックを実行する必要のあるクラスターを指定します。

    f. サービスアカウント名を指定します。

    g. namespace を指定します。

    h. フックを実行する必要のある移行手順を選択します。

    - PreBackup: バックアップタスクがソースクラスターで開始される前
    - PostBackup: バックアップタスクがソースクラスターで完了した後
    - PreRestore: 復元タスクがターゲットクラスターで開始される前
    - PostRestore: 復元タスクがターゲットクラスターで完了した後

14. **Add** をクリックします。
移行計画に最大 4 つのフックを追加し、それぞれのフックを異なる移行手順に割り当てることができます。

15. **Finish** をクリックします。

16. **Close** をクリックします。
移行計画は **Plans** セクションに表示されます。

1.6.6. CAM Web コンソールでの移行計画の実行

CAM Web コンソールで作成した移行計画を使用してアプリケーションとデータをステージングしたり、移行したりできます。

前提条件

CAM Web コンソールには以下が含まれている必要があります。

- ソースクラスター

- CAM ツールのインストール時に自動的に追加されるターゲットクラスター
- レプリケーションリポジトリー
- 有効な移行計画

手順

1. ターゲットクラスターの CAM Web コンソールにログインします。
2. 移行計画を選択します。
3. **Stage** をクリックし、アプリケーションを停止せずにソースクラスターからターゲットクラスターにデータをコピーします。
   実際の移行時間を短縮するには、**Stage** を複数回実行することができます。
4. アプリケーションのワークロードを移行する準備ができたら、**Migrate** をクリックします。
   Migrate は、ソースクラスターでアプリケーションワークロードを停止し、ターゲットクラスターでそのリソースを再作成します。
5. オプション: **Migrate** ウィンドウで **Do not stop applications on the source cluster during migration** を選択できます。
6. **Migrate** をクリックします。
7. オプション: 進行中の移行を停止するには、Options メニュー  をクリックし、**Cancel** を選択します。
8. 移行が完了したら、アプリケーションが OpenShift Container Platform Web コンソールで正常に移行されていることを確認します。
   a. **Home → Projects** をクリックします。
   b. 移行されたプロジェクトをクリックしてそのステータスを表示します。
   c. **Routes** セクションで **Location** をクリックし、アプリケーションが機能していることを確認します (該当する場合)。
   d. **Workloads → Pods** をクリックし、Pod が移行した namespace で実行されていることを確認します。
   e. **Storage → Persistent volumes** をクリックして、移行した永続ボリュームが正常にプロビジョニングされていることを確認します。

1.7. CONTROL PLANE MIGRATION ASSISTANT (CPMA) でのコントロールプレーン設定の移行

Control Plane Migration Assistant (CPMA) は、OpenShift Container Platform 3.7 (以降) から OpenShift Container Platform 4.4 へのコントロールプレーンの移行に役立つ CLI ベースのツールです。CPMA は OpenShift Container Platform 3 設定ファイルを処理し、OpenShift Container Platform 4.4 Operator によって使用されるカスタムリソース (CR) マニフェストファイルを生成します。

1.7.1. Control Plane Migration Assistant のインストール

Red Hat カスタマーポータルから Control Plane Migration Assistant (CPMA) バイナリーファイルをダウンロードし、Linux、MacOSX、Windows オペレーティングシステム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 **Downloads → Red Hat OpenShift Container Platform** に移動します。
2. **Download Red Hat OpenShift Container Platform**ページで、**Product Variant** 一覧から **Red Hat OpenShift Container Platform** を選択します。
3. **Version** 一覧から **CPMA 1.0 for RHEL 7**を選択します。このバイナリーは、RHEL 7 および RHEL 8 で機能します。
4. **Download Now** をクリックして Linux または MacOSX の場合は **cpma** をダウンロードし、Windows の場合は **cpma.exe** をダウンロードします。
5. Linux または MacOSX の場合は **$PATH** と定義されたディレクトリーに、Windows の場合は **%PATH%** と定義されたディレクトリーにファイルを保存します。
6. Linux の場合は、ファイルを実行可能にします。

```
$ sudo chmod +x cpma
```

1.7.2. Control Plane Migration Assistant の使用

Control Plane Migration Assistant (CPMA) は、OpenShift Container Platform 4.4 Operator によって使用される CR マニフェストを生成し、どの OpenShift Container Platform 3 機能が完全に、または部分的にサポートされるか、または全くサポートされないかを示すレポートを生成します。

CPMA はリモートモードで実行でき、SSH を使用するか、またはローカルモードで、ソースクラスターの設定ファイルのローカルコピーを使用して、ソースクラスターの設定ファイルを取得します。

前提条件

- ソースクラスターは OpenShift Container Platform 3.7 以降であること。
- ソースクラスターは、最新の同期リリースに対して更新される必要があります。
- 診断エラーや警告がないことを確認するために、環境ヘルスチェックをソースクラスターで実行する必要があります。
- CPMA バイナリーは実行可能である必要があります。
- ソースクラスターの **cluster-admin** 権限がなければなりません。

手順

1. OpenShift Container Platform 3 クラスターにログインします。

```
$ oc login https://<master1.example.com> 1
```

1 OpenShift Container Platform 3 マスターノード。Kubernetes および OpenShift Container Platform API のトークンを受信するには、クラスターにログインしている必要があります。

2. CPMA を実行します。以下の例にあるように、各プロンプトで入力が求められます。

```
$ cpma --manifests=false 1
? Do you wish to save configuration for future use? true
? What will be the source for OCP3 config files? Remote host 2
? Path to crio config file /etc/crio/crio.conf
? Path to etcd config file /etc/etcd/etcd.conf
? Path to master config file /etc/origin/master/master-config.yaml
? Path to node config file /etc/origin/node/node-config.yaml
? Path to registries config file /etc/containers/registries.conf
? Do wish to find source cluster using KUBECONFIG or prompt it? KUBECONFIG
? Select cluster obtained from KUBECONFIG contexts master1-example-com:443
? Select master node master1.example.com
? SSH login root 3
? SSH Port 22
? Path to private SSH key /home/user/.ssh/openshift_key
? Path to application data, skip to use current directory .
INFO[29 Aug 19 00:07 UTC] Starting manifest and report generation
INFO[29 Aug 19 00:07 UTC] Transform:Starting for - API
INFO[29 Aug 19 00:07 UTC] APITransform::Extract
INFO[29 Aug 19 00:07 UTC] APITransform::Transform:Reports
INFO[29 Aug 19 00:07 UTC] Transform:Starting for - Cluster
INFO[29 Aug 19 00:08 UTC] ClusterTransform::Transform:Reports
INFO[29 Aug 19 00:08 UTC] ClusterReport::ReportQuotas
INFO[29 Aug 19 00:08 UTC] ClusterReport::ReportPVs
INFO[29 Aug 19 00:08 UTC] ClusterReport::ReportNamespaces
INFO[29 Aug 19 00:08 UTC] ClusterReport::ReportNodes
INFO[29 Aug 19 00:08 UTC] ClusterReport::ReportRBAC
INFO[29 Aug 19 00:08 UTC] ClusterReport::ReportStorageClasses
INFO[29 Aug 19 00:08 UTC] Transform:Starting for - Crio
INFO[29 Aug 19 00:08 UTC] CrioTransform::Extract
WARN[29 Aug 19 00:08 UTC] Skipping Crio: No configuration file available
INFO[29 Aug 19 00:08 UTC] Transform:Starting for - Docker
INFO[29 Aug 19 00:08 UTC] DockerTransform::Extract
INFO[29 Aug 19 00:08 UTC] DockerTransform::Transform:Reports
INFO[29 Aug 19 00:08 UTC] Transform:Starting for - ETCD
INFO[29 Aug 19 00:08 UTC] ETCDTransform::Extract
INFO[29 Aug 19 00:08 UTC] ETCDTransform::Transform:Reports
INFO[29 Aug 19 00:08 UTC] Transform:Starting for - OAuth
INFO[29 Aug 19 00:08 UTC] OAuthTransform::Extract
INFO[29 Aug 19 00:08 UTC] OAuthTransform::Transform:Reports
INFO[29 Aug 19 00:08 UTC] Transform:Starting for - SDN
INFO[29 Aug 19 00:08 UTC] SDNTransform::Extract
INFO[29 Aug 19 00:08 UTC] SDNTransform::Transform:Reports
INFO[29 Aug 19 00:08 UTC] Transform:Starting for - Image
INFO[29 Aug 19 00:08 UTC] ImageTransform::Extract
INFO[29 Aug 19 00:08 UTC] ImageTransform::Transform:Reports
INFO[29 Aug 19 00:08 UTC] Transform:Starting for - Project
INFO[29 Aug 19 00:08 UTC] ProjectTransform::Extract
INFO[29 Aug 19 00:08 UTC] ProjectTransform::Transform:Reports
INFO[29 Aug 19 00:08 UTC] Flushing reports to disk
INFO[29 Aug 19 00:08 UTC] Report:Added: report.json
INFO[29 Aug 19 00:08 UTC] Report:Added: report.html
INFO[29 Aug 19 00:08 UTC] Successfully finished transformations
```

1. **--manifests=false**: CR マニフェストを生成しない
2. **Remote host**: リモートモード
3. **SSH login**: SSH ユーザーには、設定ファイルにアクセスできるように OpenShift Container Platform 3 クラスターで **sudo** パーミッションが必要です。

CPMA は、出力ディレクトリーを指定しなかった場合に、現在のディレクトリーに以下のファイルおよびディレクトリーを作成します。

- **cpma.yaml** ファイル: CPMA の実行時に指定した設定オプション
- **master1.example.com**/: マスターノードからの設定ファイル
- **report.json**: JSON でエンコードされたレポート
- **report.html**: HTML でエンコードされたレポート

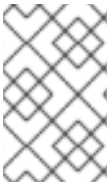
3. ブラウザーで **report.html** ファイルを開き、CPMA レポートを表示します。
4. CR マニフェストを生成する場合は、以下の例のように CR マニフェストを OpenShift Container Platform 4.4 クラスターに適用します。

```
$ oc apply -f 100_CPMA-cluster-config-secret-htpasswd-secret.yaml
```

1.8. トラブルシューティング

移行用のカスタムリソース (CR) を表示し、ログをダウンロードして失敗した移行の失敗についてのトラブルシューティングを行うことができます。

移行の失敗時にアプリケーションが停止した場合は、データ破損を防ぐために手作業でアプリケーションをロールバックする必要があります。

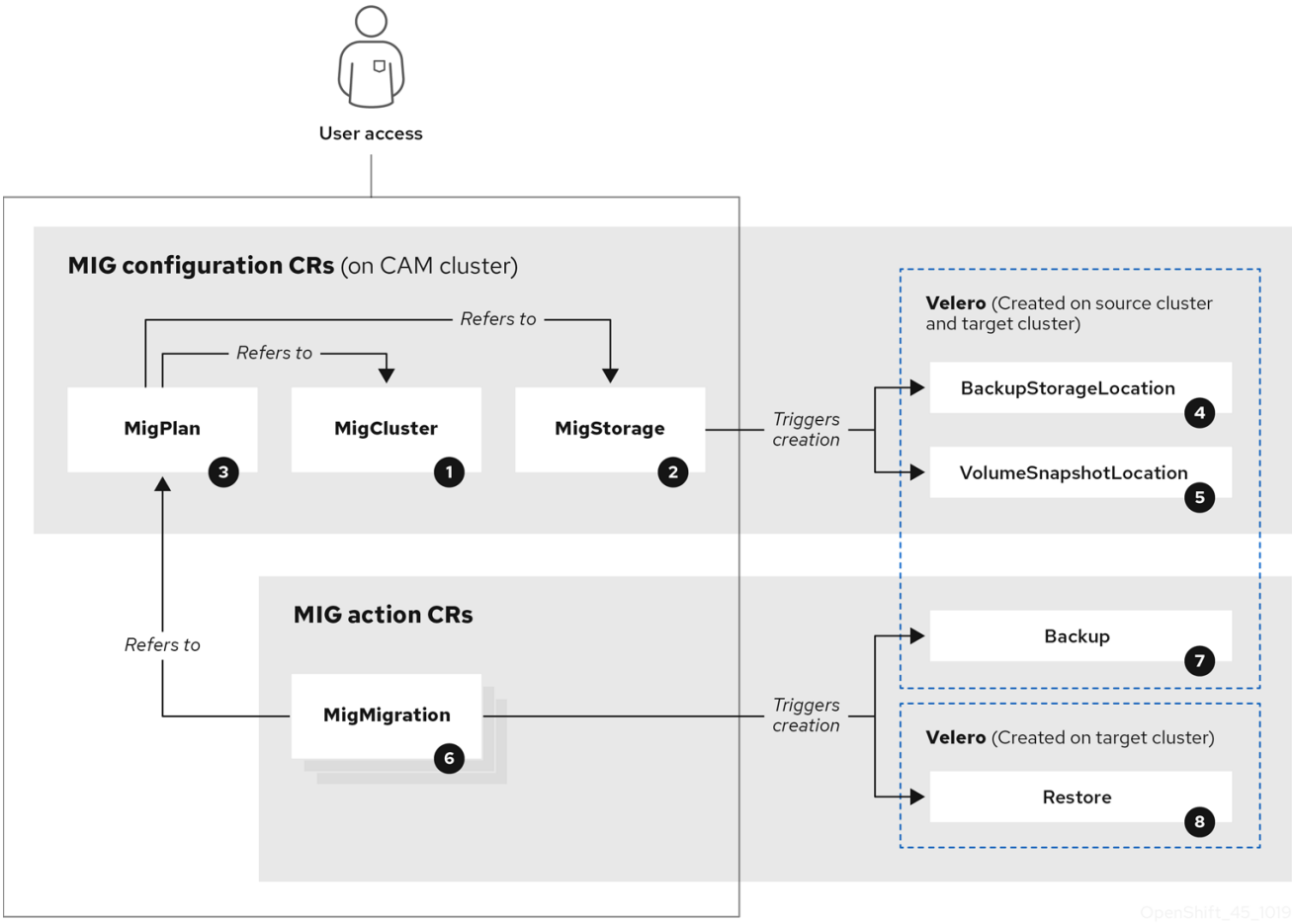


注記

移行時にアプリケーションが停止しなかった場合には、手動のロールバックは必要ありません。元のアプリケーションがソースクラスター上で依然として実行されているためです。

1.8.1. 移行カスタムリソースの表示

Cluster Application Migration (CAM) ツールは以下の カスタムリソース (Custom Resource、CR) を作成します。



1 MigCluster (設定、CAM クラスター): クラスター定義

2 MigStorage (設定、CAM クラスター): ストレージ定義

3 MigPlan (設定、CAM クラスター): 移行計画

MigPlan CR は移行されるソースおよびターゲットクラスター、リポジトリー、および namespace を記述します。これは 0、1 または多数の MigMigration CR に関連付けられます。

注記

MigPlan CR を削除すると、関連付けられた MigMigration CR が削除されます。

4 BackupStorageLocation (設定、CAM クラスター): Velero バックアップオブジェクトの場所

5 VolumeSnapshotLocation (設定、CAM クラスター): Velero ボリュームスナップショットの場所

6 MigMigration (アクション、CAM クラスター): 移行、移行時に作成される

MigMigration CR は、データのステージングまたは移行を実行するたびに作成されます。各 MigMigration CR は MigPlan CR に関連付けられます。

7 Backup (アクション、ソースクラスター): 移行計画の実行時に、MigMigration CR は各ソースクラスターに 2 つの Velero バックアップ CR を作成します。

- Kubernetes オブジェクトのバックアップ CR #1
- PV データのバックアップ CR #2

8 Restore (アクション、ターゲットクラスター): 移行計画の実行時に、MigMigration CR はターゲットクラスターに 2 つの Velero リスト CR を作成します。

- PV データのリストア CR #1 (バックアップ CR #2 の使用)
- Kubernetes オブジェクトの復元 CR #2 (バックアップ CR #1 の使用)

手順

1. CR 名を取得します。

```
$ oc get <migration_cr> -n openshift-migration 1
```

1 移行 CR を指定します (例: **migmigration**)。

出力は以下の例のようになります。

```
NAME                                   AGE
88435fe0-c9f8-11e9-85e6-5d593ce65e10   6m42s
```

2. CR を表示します。

```
$ oc describe <migration_cr> <88435fe0-c9f8-11e9-85e6-5d593ce65e10> -n openshift-
migration
```

出力は以下の例のようになります。

MigMigration の例

```
name:         88435fe0-c9f8-11e9-85e6-5d593ce65e10
namespace:    openshift-migration
labels:       <none>
annotations:  touch: 3b48b543-b53e-4e44-9d34-33563f0f8147
apiVersion:  migration.openshift.io/v1alpha1
kind:         MigMigration
metadata:
  creationTimestamp:  2019-08-29T01:01:29Z
  generation:         20
  resourceVersion:    88179
  selfLink:           /apis/migration.openshift.io/v1alpha1/namespaces/openshift-
migration/migmigrations/88435fe0-c9f8-11e9-85e6-5d593ce65e10
  uid:                8886de4c-c9f8-11e9-95ad-0205fe66cbb6
spec:
  migPlanRef:
    name:         socks-shop-mig-plan
    namespace:    openshift-migration
```

```
  quiescePods:  true
  stage:        false
status:
  conditions:
    category:            Advisory
    durable:             True
    lastTransitionTime:  2019-08-29T01:03:40Z
    message:               The migration has completed successfully.
    reason:              Completed
    status:              True
    type:                Succeeded
  phase:                 Completed
  startTimestamp:        2019-08-29T01:01:29Z
events:                  <none>
```

Velero バックアップ CR #2 の例 (PV データ)

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.105.179:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-44dd3bd5-c9f8-11e9-95ad-
0205fe66cbb6
  creationTimestamp: "2019-08-29T01:03:15Z"
  generateName: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-
  generation: 1
  labels:
    app.kubernetes.io/part-of: migration
    migmigration: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
    migration-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
    velero.io/storage-location: myrepo-vpzq9
  name: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-59gb7
  namespace: openshift-migration
  resourceVersion: "87313"
  selfLink: /apis/velero.io/v1/namespaces/openshift-migration/backups/88435fe0-c9f8-11e9-85e6-
5d593ce65e10-59gb7
  uid: c80dbbc0-c9f8-11e9-95ad-0205fe66cbb6
spec:
  excludedNamespaces: []
  excludedResources: []
  hooks:
    resources: []
  includeClusterResources: null
  includedNamespaces:
  - sock-shop
  includedResources:
  - persistentvolumes
  - persistentvolumeclaims
  - namespaces
  - imagestreams
  - imagestreamtags
  - secrets
  - configmaps
```

```
  - pods
  labelSelector:
    matchLabels:
      migration-included-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
  storageLocation: myrepo-vpzq9
  ttl: 720h0m0s
  volumeSnapshotLocations:
  - myrepo-wv6fx
status:
  completionTimestamp: "2019-08-29T01:02:36Z"
  errors: 0
  expiration: "2019-09-28T01:02:35Z"
  phase: Completed
  startTimestamp: "2019-08-29T01:02:35Z"
  validationErrors: null
  version: 1
  volumeSnapshotsAttempted: 0
  volumeSnapshotsCompleted: 0
  warnings: 0
```

Velero リストア CR #2 の例 (Kubernetes リソース)

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.90.187:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-36f54ca7-c925-11e9-825a-
06fa9fb68c88
  creationTimestamp: "2019-08-28T00:09:49Z"
  generateName: e13a1b60-c927-11e9-9555-d129df7f3b96-
  generation: 3
  labels:
    app.kubernetes.io/part-of: migration
    migmigration: e18252c9-c927-11e9-825a-06fa9fb68c88
    migration-final-restore: e18252c9-c927-11e9-825a-06fa9fb68c88
  name: e13a1b60-c927-11e9-9555-d129df7f3b96-gb8nx
  namespace: openshift-migration
  resourceVersion: "82329"
  selfLink: /apis/velero.io/v1/namespaces/openshift-migration/restores/e13a1b60-c927-11e9-9555-
d129df7f3b96-gb8nx
  uid: 26983ec0-c928-11e9-825a-06fa9fb68c88
spec:
  backupName: e13a1b60-c927-11e9-9555-d129df7f3b96-sz24f
  excludedNamespaces: null
  excludedResources:
  - nodes
  - events
  - events.events.k8s.io
  - backups.velero.io
  - restores.velero.io
  - resticrepositories.velero.io
  includedNamespaces: null
  includedResources: null
```

```
  namespaceMapping: null
  restorePVs: true
status:
  errors: 0
  failureReason: ""
  phase: Completed
  validationErrors: null
  warnings: 15
```

1.8.2. 移行ログのダウンロード

CAM Web コンソールで Velero、Restic、および Migration コントローラーログをダウンロードして、移行の失敗についてのトラブルシューティングを行うことができます。

手順

1. CAM Web コンソールにログインします。
2. **Plans** をクリックし、 移行計画の一覧を表示します。
3. 特定の移行計画の **Options** メニュー  をクリックし、**Logs** を選択します。
4. **Download Logs** をクリックし、すべてのクラスターの Migration コントローラー、Velero、および Restic のログをダウンロードします。
5. 特定のログをダウンロードするには、以下を実行します。
   a. ログオプションを指定します。
      - **Cluster**: ソース、ターゲット、または CAM ホストクラスターを選択します。
      - **Log source**: **Velero**、**Restic**、または **Controller** を選択します。
      - **Pod source**: Pod 名を選択します (例: **controller-manager-78c469849c-v6wcf**)。選択したログが表示されます。

        選択した内容を変更することで、ログ選択の設定をクリアできます。

   b. **Download Selected** をクリックし、選択したログをダウンロードします。

オプションで、以下の例にあるように CLI を使用してログにアクセスできます。

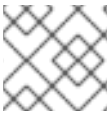
```
$ oc get pods -n openshift-migration | grep controller
controller-manager-78c469849c-v6wcf          1/1     Running     0          4h49m

$ oc logs controller-manager-78c469849c-v6wcf -f -n openshift-migration
```

1.8.3. 非推奨の API GroupVersionKinds の更新

OpenShift Container Platform 4.4 では、OpenShift Container Platform 3.x によって使用される一部の API **GroupVersionKinds** (GVK) が 非推奨 になりました。

ソースクラスターが非推奨の GVK を使用する場合、移行計画の作成時に、**Some namespaces contain GVKs incompatible with destination cluster** という警告が表示されます。**See details** をクリックして namespace と互換性のない GVK を表示します。



注記

この警告は移行をブロックしません。

移行時に、非推奨の GVK は Kubernetes オブジェクトの Velero バックアップカスタムリソース (CR)#1 に保存されます。バックアップ CR をダウンロードし、非推奨の GVK **yaml** ファイルを展開し、これらを **oc convert** コマンドで更新できます。次に、ターゲットクラスターで更新された GVK を作成します。

手順

1. 移行計画を実行します。
2. MigPlan CR を表示します。

```
$ oc describe migplan <migplan_name> -n openshift-migration 1
```

1 移行計画の名前を指定します。

出力は以下の例のようになります。

```
metadata:
  ...
  uid: 79509e05-61d6-11e9-bc55-02ce4781844a 1
status:
  ...
  conditions:
  - category: Warn
    lastTransitionTime: 2020-04-30T17:16:23Z
    message: 'Some namespaces contain GVKs incompatible with destination cluster.
      See: `incompatibleNamespaces` for details'
    status: "True"
    type: GVKsIncompatible
  incompatibleNamespaces:
  - gvks:
    - group: batch
      kind: cronjobs 2
      version: v2alpha1
    - group: batch
      kind: scheduledjobs 3
      version: v2alpha1
```

1 MigPlan UID を記録します。

2 3 非推奨の GVK を記録します。

3. MigPlan UID に関連付けられた MigMigration 名を取得します。

```
$ oc get migmigration -o json | jq -r '.items[] | select(.metadata.ownerReferences[].uid=="
<migplan_uid>") | .metadata.name' 1
```

1 MigPlan UID を指定します。

4. MigMigration 名に関連付けられた MigMigration UID を取得します。

```
$ oc get migmigration <migmigration_name> -o jsonpath='{.metadata.uid}' 1
```

1 MigMigration 名を指定します。

5. MigMigration UID に関連付けられた Velero バックアップ名を取得します。

```
$ oc get backup.velero.io --selector migration-initial-backup="<migmigration_uid>" -o
jsonpath={.items[*].metadata.name} 1
```

1 MigMigration UID を指定します。

6. Velero バックアップの内容をローカルマシンにダウンロードします。

- AWS S3 の場合:

```
$ aws s3 cp s3://<bucket_name>/velero/backups/<backup_name> <backup_local_dir> --
recursive 1
```

1 バケット、バックアップ名、およびローカルバックアップのディレクトリー名を指定します。

- GCP の場合:

```
$ gsutil cp gs://<bucket_name>/velero/backups/<backup_name> <backup_local_dir> --
recursive 1
```

1 バケット、バックアップ名、およびローカルバックアップのディレクトリー名を指定します。

- Azure の場合:

```
$ azcopy copy
'https://velerobackups.blob.core.windows.net/velero/backups/<backup_name>'
'<backup_local_dir>' --recursive 1
```

1 バックアップ名とローカルバックアップのディレクトリー名を指定します。

7. Velero バックアップアーカイブファイルを展開します。

```
$ tar -xfv <backup_local_dir>/<backup_name>.tar.gz -C <backup_local_dir>
```

8. 非推奨となったそれぞれの GVK でオフラインモードで **oc convert** を実行します。

```
$ oc convert -f <backup_local_dir>/resources/<gvk>.json 1
```

1 非推奨の GVK を指定します。

9. ターゲットクラスターで変換された GVK を作成します。

```
$ oc create -f <gvk>.json 1
```

1 変換された GVK を指定します。

1.8.4. エラーメッセージ

1.8.4.1. Velero Pod ログの Restic タイムアウトエラーメッセージ

Restic のタイムアウトにより移行が失敗する場合、以下のエラーが Velero Pod ログに表示されます。

```
level=error msg="Error backing up item" backup=velero/monitoring error="timed out waiting for all
PodVolumeBackups to complete"
error.file="/go/src/github.com/heptio/velero/pkg/restic/backupper.go:165"
error.function="github.com/heptio/velero/pkg/restic.(*backupper).BackupPodVolumes" group=v1
```

restic_timeout のデフォルト値は 1 時間です。大規模な移行では、このパラメーターの値を大きくすることができます。値を高くすると、エラーメッセージが返されるタイミングが遅れる可能性があることに注意してください。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** に移動します。
2. **Cluster Application Migration Operator** をクリックします。
3. **MigrationController** タブで、**migration-controller** をクリックします。
4. **YAML** タブで、以下のパラメーター値を更新します。

```
spec:
  restic_timeout: 1h 1
```

1 有効な単位は **h** (時間)、**m** (分)、および **s** (秒) です (例: **3h30m15s**)。

5. **Save** をクリックします。

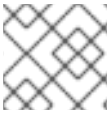
1.8.4.2. MigMigration カスタムリソースの ResticVerifyErrors

ファイルシステムデータのコピー方法を使用して PV を移行する際にデータ検証が失敗すると、以下のエラーが MigMigration カスタムリソース (CR) に表示されます。

```
status:
  conditions:
  - category: Warn
    durable: true
    lastTransitionTime: 2020-04-16T20:35:16Z
    message: There were verify errors found in 1 Restic volume restores. See restore `<registry-
example-migration-rvwcm>`
      for details (1)
    status: "True"
    type: ResticVerifyErrors (2)
```

(1) エラーメッセージはリストア CR 名を識別します。

(2) **ResticErrors** も表示されます。**ResticErrors** は、検証エラーが含まれる一般的なエラーの警告です。



注記

データ検証エラーによって移行プロセスが失敗することはありません。

ターゲットクラスターのリストア CR を確認して、データ検証エラーのソースを特定できます。

手順

1. ターゲットクラスターにログインします。
2. リストア CR を表示します。

```
$ oc describe <registry-example-migration-rvwcm> -n openshift-migration
```

出力では、**PodVolumeRestore** エラーのある PV を特定できます。

```
status:
  phase: Completed
  podVolumeRestoreErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
  podVolumeRestoreResticErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
```

3. **PodVolumeRestore** CR を表示します。

```
$ oc describe <migration-example-rvwcm-98t49>
```

出力はエラーをログに記録した Restic Pod を特定します。

```
  completionTimestamp: 2020-05-01T20:49:12Z
  errors: 1
  resticErrors: 1
```

```
...
resticPod: <restic-nr2v5>
```

4. Restic Pod ログを表示します。

```
$ oc logs -f restic-nr2v5
```

1.8.5. 移行の手動ロールバック

移行の失敗時にアプリケーションが停止された場合は、PV でのデータの破損を防ぐために手動でこれをロールバックする必要があります。

移行時にアプリケーションが停止しなかった場合には、この手順は必要ありません。元のアプリケーションがソースクラスター上で依然として実行されているためです。

手順

1. ターゲットクラスター上で、移行したプロジェクトに切り替えます。

```
$ oc project <project>
```

2. デプロイされたリソースを取得します。

```
$ oc get all
```

3. デプロイされたリソースを削除し、アプリケーションがターゲットクラスターで実行されておらず、PVC 上にあるデータにアクセスできるようにします。

```
$ oc delete <resource_type>
```

4. これを削除せずにデーモンセットを停止するには、YAML ファイルで **nodeSelector** を更新します。

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: hello-daemonset
spec:
  selector:
     matchLabels:
       name: hello-daemonset
  template:
    metadata:
      labels:
        name: hello-daemonset
    spec:
      nodeSelector:
        role: worker 1
```

1 いずれのノードにも存在しない **nodeSelector** 値を指定します。

5. 不要なデータが削除されるように、各 PV の回収ポリシーを更新します。移行時に、バインド

された PV の回収ポリシーは **Retain** であり、アプリケーションがソースクラスターから削除される際にデータの損失が生じないようにされます。ロールバック時にこれらの PV を削除できます。

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0001
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain (1)
  ...
status:
  ...
```

(1) **Recycle** または **Delete** を指定します。

6. ソースクラスターで、移行したプロジェクトに切り替えます。

```
$ oc project <project_name>
```

7. プロジェクトのデプロイされたリソースを取得します。

```
$ oc get all
```

8. デプロイされた各リソースのレプリカを開始します。

```
$ oc scale --replicas=1 <resource_type>/<resource_name>
```

9. 手順の実行時に変更した場合は、 **DaemonSet** リソースの **nodeSelector** を元の値に更新します。

1.8.6. カスタマーサポートケース用のデータの収集

カスタマーサポートケースを作成する場合、 **openshift-migration-must-gather-rhel8** イメージを使用して **must-gather** ツールを実行し、クラスターについての情報を収集し、これを Red Hat カスタマーポータル にアップロードできます。

openshift-migration-must-gather-rhel8 イメージは、デフォルトの **must-gather** イメージで収集されないログおよびカスタムリソースデータを収集します。

手順

1. **must-gather** データを保存するディレクトリーに移動します。
2. **oc adm must-gather** コマンドを実行します。

```
$ oc adm must-gather --image=registry.redhat.io/rhcam-1-2/openshift-migration-must-gather-rhel8
```

must-gather ツールはクラスター情報を収集し、これを **must-gather.local.<uid>** ディレクトリーに保存します。

3. 認証キーおよびその他の機密情報を **must-gather** データから削除します。

4. **must-gather.local.<uid>** ディレクトリーの内容を含むアーカイブファイルを作成します。

```
$ tar cvaf must-gather.tar.gz must-gather.local.<uid>/
```

圧縮ファイルを Red Hat カスタマーポータル 上のサポートケースに添付します。

1.8.7. 既知の問題

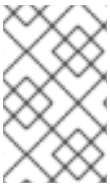
本リリースには、以下の既知の問題があります。

- 移行時に、Cluster Application Migration (CAM) ツールは以下の namespace アノテーションを保持します。
  - **openshift.io/sa.scc.mcs**
  - **openshift.io/sa.scc.supplemental-groups**
  - **openshift.io/sa.scc.uid-range**
    これらのアノテーションは UID 範囲を保持し、コンテナーがターゲットクラスターのファイルシステムのパーミッションを保持できるようにします。移行された UID が、ターゲットクラスターの既存の namespace または今後の namespace 内の UID を重複させるリスクがあります。(BZ#1748440)
- AWS バケットが CAM Web コンソールに追加された後に削除される場合、MigStorage CR は更新されないため、そのステータスは **True** のままになります。(BZ#1738564)
- ほとんどのクラスタースコープのリソースは CAM ツールで処理されません。アプリケーションがクラスタースコープのリソースを必要とする場合、ターゲットクラスターでそれらを手動で作成する必要がある場合があります。
- 移行に失敗すると、移行計画は休止状態の Pod のカスタム PV 設定を保持しません。移行を手動でロールバックし、移行計画を削除し、PV 設定で新たな移行計画を作成する必要があります。(BZ#1784899)
- Restic のタイムアウトにより大規模な移行が失敗する場合は、Migration コントローラー CR の **restic_timeout** パラメーターの値 (デフォルト: **1h**) を増やすことができます。
- ファイルシステムのコピー方法で移行される PV のデータ検証オプションを選択すると、パフォーマンスは大幅に遅くなります。Velero は各ファイルのチェックサムを生成し、ファイルを復元する際に確認します。
- 現在のリリース (CAM 1.2) では、ソースクラスターで使用される特定の API の **GroupVersionKinds** (GVK) が非推奨になっているため、OpenShift Container Platform 3.7 から 4.4 に移行することはできません。移行後に GVK を手動で更新できます。(BZ#1817251)
- OpenShift Container Platform 3 クラスターに CAM 1.2 をインストールできない場合は、現在の **operator.yml** ファイルをダウンロードし、この問題を修正します。(BZ#1843059)

第2章 OPENSHIFT CONTAINER PLATFORM 4.1 からの移行

2.1. 移行ツールおよび前提条件

アプリケーションワークロードを、Cluster Application Migration (CAM) ツールを使用して OpenShift Container Platform 4.1 から OpenShift Container Platform 4.4 に移行できます。CAM ツールを使用すると、移行を制御し、アプリケーションのダウンタイムを最小限に抑えることができます。



注記

ソースおよびターゲットクラスターが正しく設定されている場合、同じバージョンの OpenShift Container Platform クラスター間で移行できます (4.1 から 4.1 への移行など)。

Kubernetes カスタムリソースをベースとする CAM ツールの Web コンソールおよび API により、namespace の粒度でステートフルおよびステートレスのアプリケーションワークロードを移行できます。

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するためにファイルシステムおよびスナップショットによるデータのコピー方法をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

移行フックを使用して、移行中の特定の時点で Ansible Playbook を実行できます。フックは移行計画の作成時に追加されます。

2.1.1. 移行の前提条件

- ソースクラスターを最新の z-stream リリースにアップグレードすること。
- すべてのクラスターで **cluster-admin** 権限がある。
- ソースおよびターゲットクラスターには、レプリケーションリポジトリーへの無制限のネットワークアクセスがなければなりません。
- Migration コントローラーがインストールされているクラスターには、他のクラスターへの無制限のアクセスが必要です。
- アプリケーションが **openshift** namespace のイメージを使用する場合、イメージの必要なバージョンがターゲットクラスターに存在する必要があります。
  必要なイメージがない場合は、アプリケーションと互換性のある利用可能なバージョンを使用するように **imagestreamtags** 参照を更新する必要があります。**imagestreamtag** を更新できない場合、同等のイメージをアプリケーション namespace に手動でアップロードし、それらを参照するようにアプリケーションを更新できます。

以下の **imagestreamtags** は OpenShift Container Platform **4.2** から **削除** されています。

- **dotnet:1.0**、**dotnet:1.1**、 **dotnet:2.0**
- **dotnet-runtime:2.0**
- **mariadb:10.1**
- **mongodb:2.4**、**mongodb:2.6**
- **mysql:5.5**、**mysql:5.6**

- **nginx:1.8**
- **nodejs:0.10**、**nodejs:4**、**nodejs:6**
- **perl:5.16**、**perl:5.20**
- **php:5.5**、**php:5.6**
- **postgresql:9.2**、**postgresql:9.4**、**postgresql:9.5**
- **python:3.3**、**python:3.4**
- **ruby:2.0**, **ruby:2.2**

以下の **imagestreamtags** は OpenShift Container Platform **4.4** から **削除** されています。

- **dotnet: 2.2**
- **dotnet-runtime: 2.2**
- **nginx: 1.12**
- **nodejs: 8, 8-RHOAR, 10-SCL**
- **perl:5.24**
- **php: 7.0, 7.1**
- **redis: 3.2**

2.1.2. Cluster Application Migration ツールについて

Cluster Application Migration (CAM) ツールを使用すると、CAM Web コンソールまたは Kubernetes API を使用して Kubernetes リソース、永続ボリュームデータ、および内部コンテナーイメージを OpenShift Container Platform ソースクラスターから OpenShift Container Platform 4.4 ターゲットクラスターに移行できます。

CAM Web コンソールを使用してアプリケーションを移行するには、以下の手順が必要です。

1. Cluster Application Migration Operator をすべてのクラスターにインストールします。
   インターネットアクセスが制限されているか、またはインターネットアクセスのない環境で Cluster Application Migration Operator をインストールできます。ソースおよびターゲットクラスターは、相互に対するネットワークアクセスおよびミラーレジストリーへのネットワークアクセスがなければなりません。

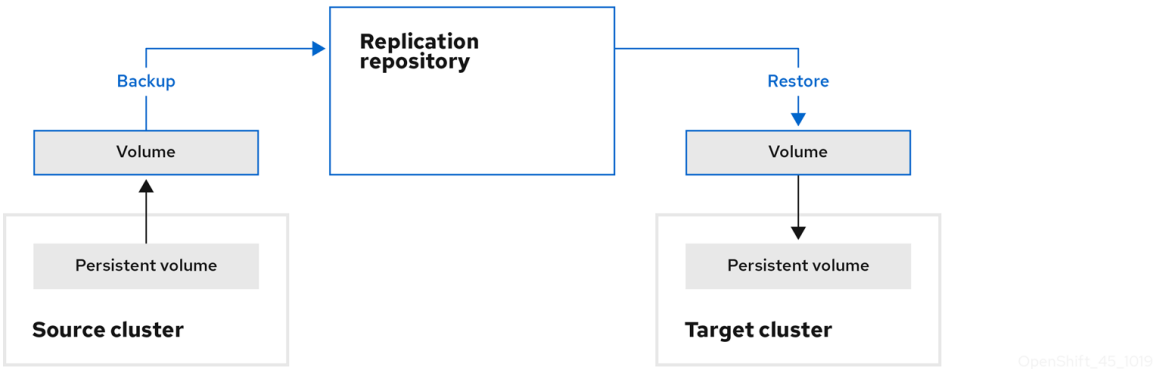
2. CAM ツールがデータ移行に使用する中間オブジェクトストレージであるレプリケーションリポジトリーを設定します。
   ソースおよびターゲットクラスターには、移行時にレプリケーションリポジトリーへのネットワークアクセスがなければなりません。制限された環境では、内部でホストされる S3 ストレージリポジトリーを使用できます。プロキシーサーバーを使用する場合は、レプリケーションリポジトリーがホワイトリスト化されていることを確認する必要があります。

3. ソースクラスターを CAM Web コンソールに追加します。

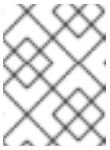
4. レプリケーションリポジトリーを CAM Web コンソールに追加します。

5. 以下のデータ移行オプションのいずれかを使用して、移行計画を作成します。

- **Copy**: CAM ツールは、データをソースクラスターからレプリケーションリポジトリーにコピーし、レプリケーションリポジトリーからターゲットクラスターにコピーします。



- **Move**: CAM ツールはリモートボリューム (NFS など) をソースクラスターからアンマウントし、リモートボリュームをポイントするターゲットクラスターで PV リソースを作成し、その後にリモートボリュームをターゲットクラスターにマウントします。ターゲットクラスターで実行されているアプリケーションは、ソースクラスターが使用していたものと同じリモートボリュームを使用します。リモートボリュームは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。



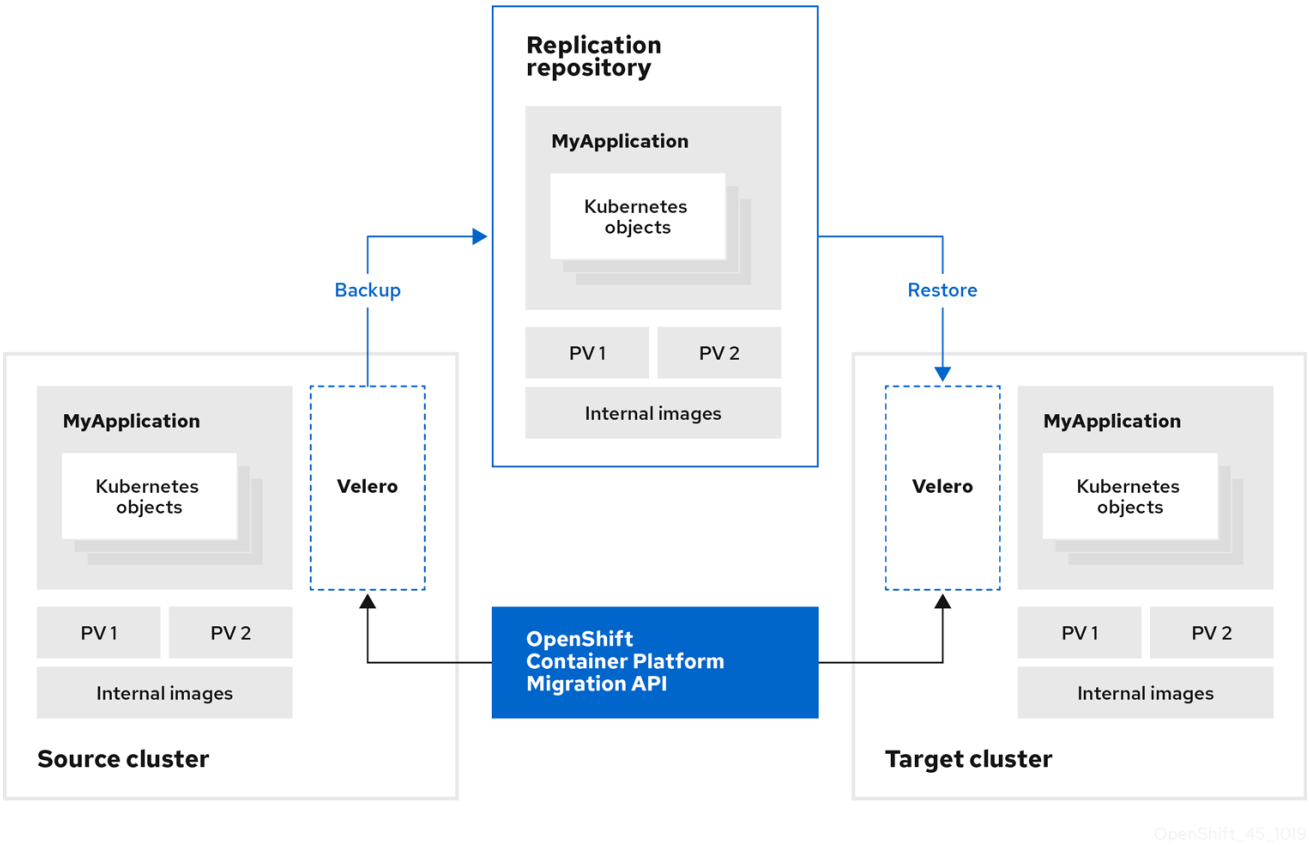
注記

レプリケーションリポジトリーはこの図には表示されていませんが、実際の移行には必要になります。

6. 以下のオプションのいずれかを使用して、移行計画を実行します。

- **Stage** (オプション) は、アプリケーションを停止せずにデータをターゲットクラスターにコピーします。
ステージングは、移行前にほとんどのデータがターゲットにコピーされるように複数回実行することができます。これにより、実際の移行時間やアプリケーションのダウンタイムが最小限に抑えられます。

- **Migrate** は、ソースクラスターでアプリケーションを停止し、ターゲットクラスターでそのリソースを再作成します。オプションで、アプリケーションを停止せずにワークロードを移行できます。



2.1.3. データのコピー方法

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するためにファイルシステムおよびスナップショットによるデータのコピー方法をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

2.1.3.1. ファイルシステムのコピー方法

CAM ツールは、データファイルをソースクラスターからレプリケーションリポジトリーにコピーし、そこからターゲットクラスターにコピーします。

表2.1 ファイルシステムのコピー方法の概要

利点	制限
• クラスターで複数の異なるストレージクラスを使用することが可能 • すべての S3 ストレージプロバイダーでサポートされている • チェックサムを使用したオプションのデータ検証	• スナップショットのコピー方法よりも遅い • オプションのデータ検証は、パフォーマンスを大幅に低下させます。

2.1.3.2. スナップショットのコピー方法

CAM ツールは、ソースクラスターのデータのスナップショットを、レプリケーションリポジトリーとして設定されたクラウドプロバイダーのオブジェクトストレージにコピーします。データはターゲットクラスターで復元されます。

AWS、Google Cloud Provider、および Microsoft Azure は、スナップショットのコピー方法をサポートします。

表2.2 スナップショットのコピー方法の概要

利点	制限
● ファイルシステムのコピー方法よりも高速	● クラウドプロバイダーはスナップショットをサポートしている必要があります。 ● クラスターは同じクラウドプロバイダーになければなりません。 ● クラスターは、同じ場所またはリージョンにある必要があります。 ● クラスターには同じストレージクラスがなければなりません。 ● ストレージクラスにはスナップショットとの互換性がある必要があります。

2.1.4. 移行フックについて

移行フックを使用して、移行中の特定の時点で Ansible Playbook を実行できます。フックは移行計画の作成時に追加されます。

注記

Ansible Playbook を使用する必要がない場合は、カスタムコンテナーイメージを作成し、これを移行計画に追加することができます。

移行フックは、アプリケーションの休止状態のカスタマイズ、サポート外のデータタイプの手動の移行、および移行後のアプリケーションの更新などのタスクを実行します。

単一の移行フックは、以下の移行手順のいずれかでソースまたはターゲットクラスターで実行されます。

- PreBackup: バックアップタスクがソースクラスターで開始される前
- PostBackup: バックアップタスクがソースクラスターで完了した後
- PreRestore: 復元タスクがターゲットクラスターで開始される前
- PostRestore: 復元タスクがターゲットクラスターで完了した後
  1 つのフックをそれぞれの移行ステップに割り当て、単一の移行計画について最大 4 つのフックを割り当てることができます。

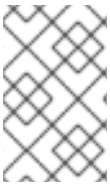
デフォルトの **hook-runner** イメージは **registry.redhat.io/rhcam-1-2/openshift-migration-hook-runner-rhel7** です。このイメージは Ansible Runner をベースとしており、Ansible Kubernetes リソース用の **python-openshift** および更新された **oc** バイナリーが含まれます。追加の Ansible モジュールまた

はツールで独自のフックイメージを作成することもできます。

Ansible Playbook はフックコンテナーに ConfigMap としてマウントされます。フックコンテナーは、指定されたサービスアカウントおよび namespace を使用してクラスターでジョブとして実行されます。ジョブは、初期の Pod がエビクトされるか、または強制終了される場合でも、デフォルトの **backoffLimit** (**6**) または正常に完了した状態に達するまで実行されます。

2.2. クラスターアプリケーション移行ツールのデプロイおよびアップグレード

OLM を使用して OpenShift Container Platform 4.4 ターゲットクラスターおよび 4.1 ソースクラスターに Cluster Application Migration Operator をインストールできます。Cluster Application Migration Operator は、デフォルトで Cluster Application Migration (CAM) ツールをターゲットクラスターにインストールします。



注記

オプション: Cluster Application Migration Operator を、CAM ツールを OpenShift Container Platform 3 クラスターまたはリモートクラスターに インストールするように設定できます。

制限された環境では、ローカルミラーレジストリーから Cluster Application Migration Operator をインストールできます。

クラスターに Cluster Application Migration Operator をインストールした後に、CAM ツールを起動できます。

2.2.1. Cluster Application Migration Operator のインストール

Cluster Application Migration Operator は Operation Lifecycle Manager (OLM) で OpenShift Container Platform 4.4 ターゲットクラスターにインストールすることも、OpenShift Container Platform 4.1 ソースクラスターにインストールすることもできます。

2.2.1.1. Cluster Application Migration Operator の OpenShift Container Platform 4.4 ターゲットクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4.4 ターゲットクラスターに Cluster Application Migration Operator をインストールできます。

Cluster Application Migration Operator は、デフォルトで Cluster Application Migration ツールをターゲットクラスターにインストールします。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。

Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。

5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **Create** をクリックします。
8. **Workloads → Pods** をクリックし、Controller Manager、Migration UI、Restic、および Velero Pod が実行中であることを確認します。

2.2.1.2. OpenShift Container Platform 4.1 ソースクラスターへの CAM Operator のインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4 ソースクラスターに Cluster Application Migration Operator をインストールできます。

手順

1. OpenShift Container Platform Web コンソールで、**Catalog → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **migration_controller** および **migration_ui** パラメーターを **false** に設定し、**deprecated_cors_configuration: true** パラメーターを **spec** スタンザに追加します。

```
spec:
  ...
  migration_controller: false
  migration_ui: false
  ...
  deprecated_cors_configuration: true
```

8. **Create** をクリックします。
9. **Workloads → Pods** をクリックし、Restic および Velero Pod が実行されていることを確認します。

2.2.2. 制限された環境での Cluster Application Migration Operator のインストール

OpenShift Container Platform 4 のカスタム Operator カタログイメージをビルドし、これをローカルミラーイメージレジストリーにプッシュし、Operator Lifecycle Manager をローカルレジストリーから Cluster Application Migration Operator をインストールするように設定できます。

追加リソース

- ネットワークが制限された環境での Operator Lifecycle Manager の使用

2.2.2.1. Operator カタログイメージのビルド

クラスター管理者は、Operator Lifecycle Manager (OLM) によって使用されるカスタム Operator カタログイメージをビルドし、Docker v2-2 をサポートするコンテナーイメージレジストリーにそのイメージをプッシュできます。ネットワークが制限された環境のクラスターの場合、このレジストリーには、ネットワークが制限されたインストールで作成されたミラーレジストリーなど、クラスターにネットワークアクセスのあるレジストリーを使用できます。

重要

OpenShift Container Platform クラスターの内部レジストリーはターゲットレジストリーとして使用できません。これは、ミラーリングプロセスで必要となるタグを使わないプッシュをサポートしないためです。

以下の例では、お使いのネットワークとインターネットの両方にアクセスできるミラーレジストリーを使用することを前提としています。

前提条件

- ネットワークアクセスが無制限の Linux ワークステーション
- **oc** version 4.3.5+
- **podman** version 1.4.4+
- Docker v2-2 をサポートするミラーレジストリーへのアクセス
- プライベートレジストリーを使用している場合、後続の手順で使用するために **REG_CREDS** 環境変数をレジストリー認証情報のファイルパスに設定します。たとえば **podman** CLI の場合は、以下のようになります。

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

- quay.io アカウントがアクセスできるプライベート namespace を使用している場合、Quay 認証トークンを設定する必要があります。quay.io 認証情報を使用してログイン API に対して要求を行うことにより、**--auth-token** フラグで使用できる **AUTH_TOKEN** 環境変数を設定します。

```
$ AUTH_TOKEN=$(curl -sH "Content-Type: application/json" \
    -XPOST https://quay.io/cnr/api/v1/users/login -d '
    {
        "user": {
            "username": """<quay_username>""",
            "password": """<quay_password>"""
        }
    }' | jq -r '.token')
```

手順

1. ネットワークアクセスが無制限のワークステーションで、ターゲットミラーレジストリーを使用して認証を行います。

```
$ podman login <registry_host_name>
```

また、ビルド時にベースイメージをプルできるように、**registry.redhat.io** で認証します。

```
$ podman login registry.redhat.io
```

2. quay.io から **redhat-operators** カタログをベースにカタログイメージをビルドし、そのイメージにタグを付け、ミラーレジストリーにプッシュします。

```
$ oc adm catalog build \
    --appregistry-org redhat-operators \ 1
    --from=registry.redhat.io/openshift4/ose-operator-registry:v4.4 \ 2
    --filter-by-os="linux/amd64" \ 3
    --to=<registry_host_name>:<port>/olm/redhat-operators:v1 \ 4
    [-a ${REG_CREDS}] \ 5
    [--insecure] \ 6
    [--auth-token "${AUTH_TOKEN}"] 7

INFO[0013] loading Bundles
dir=/var/folders/st/9cskxqs53ll3wdn434vw4cd80000gn/T/300666084/manifests-829192605
...
Pushed sha256:f73d42950021f9240389f99ddc5b0c7f1b533c054ba344654ff1edaf6bf827e3
to example_registry:5000/olm/redhat-operators:v1
```

1. App Registry インスタンスからのプルに使用する組織 (namespace)。
2. ターゲット OpenShift Container Platform クラスターのメジャーバージョンおよびマイナーバージョンに一致するタグを使用して、**--from** を **ose-operator-registry** ベースイメージに設定します。
3. **--filter-by-os** を、ターゲットの OpenShift Container Platform クラスターと一致する必要のある、ベースイメージに使用するオペレーティングシステムおよびアーキテクチャーに設定します。使用できる値は、**linux/amd64**、**linux/ppc64le**、および **linux/s390x** です。
4. カタログイメージに名前を付け、**v1** などのタグを追加します。
5. オプション: 必要な場合は、レジストリー認証情報ファイルの場所を指定します。
6. オプション: ターゲットレジストリーの信頼を設定しない場合は、**--insecure** フラグを追加します。
7. オプション: 公開されていない他のアプリケーションレジストリーカタログが使用されている場合、Quay 認証トークンを指定します。

無効なマニフェストが Red Hat のカタログに誤って導入される可能性があります。これが実際に生じる場合には、以下のようなエラーが表示される可能性があります。

```
...
INFO[0014] directory
```

```
dir=/var/folders/st/9cskxqs53ll3wdn434vw4cd80000gn/T/300666084/manifests-829192605
file=4.2 load=package
W1114 19:42:37.876180   34665 builder.go:141] error building database: error loading
package into db: fuse-camel-k-operator.v7.5.0 specifies replacement that couldn't be found
Uploading ... 244.9kB/s
```

通常、これらのエラーは致命的なエラーではなく、該当する Operator パッケージにインストールする予定の Operator やその依存関係が含まれない場合、それらを無視することができます。

2.2.2.2. ネットワークが制限された環境向けの OperatorHub の設定

クラスター管理者は、カスタム Operator カタログイメージを使用し、OLM および OperatorHub をネットワークが制限された環境でローカルコンテンツを使用するように設定できます。この例では、以前にビルドされ、サポートされているレジストリーにプッシュされたカスタム **redhat-operators** カタログイメージを使用します。

前提条件

- ネットワークアクセスが無制限の Linux ワークステーション
- サポートされているレジストリーにプッシュされるカスタム Operator カタログイメージ
- **oc** version 4.3.5+
- **podman** version 1.4.4+
- Docker v2-2 をサポートするミラーレジストリーへのアクセス
- プライベートレジストリーを使用している場合、後続の手順で使用するために **REG_CREDS** 環境変数をレジストリー認証情報のファイルパスに設定します。たとえば **podman** CLI の場合は、以下のようになります。

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

手順

1. **disableAllDefaultSources: true** を仕様に追加してデフォルトの OperatorSource を無効にします。

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

これにより、OpenShift Container Platform のインストール時にデフォルトで設定されるデフォルトの OperatorSource が無効になります。

2. **oc adm catalog mirror** コマンドは、カスタム Operator カタログイメージのコンテンツを抽出し、ミラーリングに必要なマニフェストを生成します。以下のいずれかを選択できます。
   - コマンドのデフォルト動作で、マニフェストの生成後にすべてのイメージコンテンツをミラーレジストリーに自動的にミラーリングできるようにします。または、
   - **--manifests-only** フラグを追加して、ミラーリングに必要なマニフェストのみを生成しますが、これにより、イメージコンテンツがレジストリーに自動的にミラーリングされる訳ではありません。これは、ミラーリングする内容を確認するのに役立ちます。また、コン

テンツのサブセットのみが必要な場合に、マッピングの一覧に変更を加えることができます。次に、そのファイルを **oc image mirror** コマンドで使用し、後のステップでイメージの変更済みの一覧をミラーリングできます。

ネットワークアクセスが無制限のワークステーションで、以下のコマンドを実行します。

```
$ oc adm catalog mirror \
    <registry_host_name>:<port>/olm/redhat-operators:v1 \ 1
    <registry_host_name>:<port> \
    [-a ${REG_CREDS}] \ 2
    [--insecure] \ 3
    [--filter-by-os="<os>/<arch>"] \ 4
    [--manifests-only] 5
```

1. Operator カタログイメージを指定します。
2. オプション: 必要な場合は、レジストリー認証情報ファイルの場所を指定します。
3. オプション: ターゲットレジストリーの信頼を設定しない場合は、**--insecure** フラグを追加します。
4. オプション: カタログは複数のアーキテクチャーおよびオペレーティングシステムをサポートするイメージを参照する可能性があるため、アーキテクチャーおよびオペレーティングシステムでフィルターして、一 致するイメージのみをミラーリングするようにできます。使用できる値は、**linux/amd64**、**linux/ppc64le**、および **linux/s390x** です。
5. オプション: ミラーリングに必要なマニフェストのみを生成し、実際にはイメージコンテンツをレジストリーにミラーリングしません。

出力例

```
using database path mapping: /:/tmp/190214037
wrote database to /tmp/190214037
using database at: /tmp/190214037/bundles.db 1
...
```

1. コマンドで生成される一時的なデータベース。

コマンドの実行後に、**<image_name>-manifests/** ディレクトリーが現在のディレクトリーに作成され、以下のファイルが生成されます。

- これにより、**imageContentSourcePolicy.yaml** ファイルは ImageContentSourcePolicy オブジェクトを定義します。このオブジェクトは、このオブジェクトは、ノードを Operator マニフェストおよびミラーリングされたレジストリーに保存されるイメージ参照間で変換できるように設定します。
- **mapping.txt** ファイルには、すべてのソースイメージが含まれ、これはそれらのイメージをターゲットレジストリー内のどこにマップするかを示します。このファイルは **oc image mirror** コマンドと互換性があり、ミラーリング設定をさらにカスタマイズするために使用できます。

3. 直前の手順で **--manifests-only** フラグを使用して、コンテンツのサブセットのみをミラーリングする場合は、以下を実行します。

a. **mapping.txt** ファイルのイメージの一覧を仕様に変更します。ミラーリングするイメージのサブセットの名前とバージョンが不明な場合は、以下の手順で確認します。

   i. **oc adm catalog mirror** コマンドで生成された一時的なデータベースに対して **sqlite3** ツールを実行し、一般的な検索クエリーに一致するイメージの一覧を取得します。出力は、後に **mapping.txt** ファイルを編集する方法を通知するのに役立ちます。
   たとえば、**clusterlogging.4.3** の文字列のようなイメージの一覧を取得するには、以下を実行します。

```
$ echo "select * from related_image \
    where operatorbundle_name like 'clusterlogging.4.3%';" \
    | sqlite3 -line /tmp/190214037/bundles.db 1
```

   1 **oc adm catalog mirror** コマンドの直前の出力を参照し、データベースファイルのパスを見つけます。

   出力例

```
image = registry.redhat.io/openshift4/ose-logging-
kibana5@sha256:aa4a8b2a00836d0e28aa6497ad90a3c116f135f382d8211e3c55f34f
b36dfe61
operatorbundle_name = clusterlogging.4.3.33-202008111029.p0

image = registry.redhat.io/openshift4/ose-oauth-
proxy@sha256:6b4db07f6e6c962fc96473d86c44532c93b146bbefe311d0c348117bf75
9c506
operatorbundle_name = clusterlogging.4.3.33-202008111029.p0
...
```

   ii. 直前の手順で取得した結果を使用して **mapping.txt** ファイルを編集し、ミラーリングする必要のあるイメージのサブセットのみを追加します。
   たとえば、前述の出力例の **image** 値を使用して、**mapping.txt** ファイルに以下の一致する行が存在することを確認できます。

   mapping.txt の一致するイメージマッピング。

```
registry.redhat.io/openshift4/ose-logging-
kibana5@sha256:aa4a8b2a00836d0e28aa6497ad90a3c116f135f382d8211e3c55f34f
b36dfe61=<registry_host_name>:<port>/openshift4-ose-logging-kibana5:a767c8f0
registry.redhat.io/openshift4/ose-oauth-
proxy@sha256:6b4db07f6e6c962fc96473d86c44532c93b146bbefe311d0c348117bf75
9c506=<registry_host_name>:<port>/openshift4-ose-oauth-proxy:3754ea2b
```

   この例では、これらのイメージのみをミラーリングする場合に、**mapping.txt** ファイルの他のすべてのエントリーを削除し、上記の 2 行のみを残します。

b. ネットワークアクセスが無制限のワークステーション上で、変更した **mapping.txt** ファイルを使用し、 **oc image mirror** コマンドを使用してイメージをレジストリーにミラーリングします。

```
$ oc image mirror \
    [-a ${REG_CREDS}] \
    -f ./redhat-operators-manifests/mapping.txt
```

4. ImageContentSourcePolicy を適用します。

```
$ oc apply -f ./redhat-operators-manifests/imageContentSourcePolicy.yaml
```

5. カタログイメージを参照する CatalogSource オブジェクトを作成します。

   a. 仕様を以下のように変更し、これを **catalogsource.yaml** ファイルとして保存します。

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: my-operator-catalog
  namespace: openshift-marketplace
spec:
  sourceType: grpc
  image: <registry_host_name>:<port>/olm/redhat-operators:v1 (1)
  displayName: My Operator Catalog
  publisher: grpc
```

   (1) Operator カタログイメージを指定します。

   b. このファイルを使用して CatalogSource オブジェクトを作成します。

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

6. 以下のリソースが正常に作成されていることを確認します。

   a. Pod を確認します。

```
$ oc get pods -n openshift-marketplace
```

   出力例

```
NAME                                READY   STATUS    RESTARTS  AGE
my-operator-catalog-6njx6           1/1     Running   0         28s
marketplace-operator-d9f549946-96sgr  1/1     Running   0         26h
```

   b. CatalogSource を確認します。

```
$ oc get catalogsource -n openshift-marketplace
```

   出力例

```
NAME              DISPLAY               TYPE PUBLISHER  AGE
my-operator-catalog   My Operator Catalog   grpc          5s
```

   c. PackageManifest を確認します。

```
$ oc get packagemanifest -n openshift-marketplace
```

   出力例

```
NAME    CATALOG             AGE
etcd    My Operator Catalog  34s
```

ネットワークが制限された環境の OpenShift Container Platform クラスター Web コンソールで、**OperatorHub** ページから Operator をインストールできます。

2.2.2.3. 制限された環境での Cluster Application Migration Operator の OpenShift Container Platform 4.4 ターゲットクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4.4 ターゲットクラスターに Cluster Application Migration Operator をインストールできます。

Cluster Application Migration Operator は、デフォルトで Cluster Application Migration ツールをターゲットクラスターにインストールします。

前提条件

- カスタム Operator カタログを作成し、これをミラーレジストリーにプッシュしていること。
- ミラーレジストリーから Cluster Application Migration Operator をインストールするように OLM を設定していること。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
   Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **Create** をクリックします。
8. **Workloads → Pods** をクリックし、Controller Manager、Migration UI、Restic、および Velero Pod が実行中であることを確認します。

2.2.2.4. 制限された環境での Cluster Application Migration Operator の OpenShift Container Platform 4.1 ソースクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4 ソースクラスターに Cluster Application Migration Operator をインストールできます。

前提条件

- カスタム Operator カタログを作成し、これをミラーレジストリーにプッシュしていること。

- ミラーレジストリーから Cluster Application Migration Operator をインストールするように OLM を設定していること。

手順

1. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
2. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
3. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
   Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
4. **Cluster Application Migration Operator** をクリックします。
5. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
6. **Create** をクリックします。

2.2.3. CAM Web コンソールの起動

ブラウザーで CAM Web コンソールを起動できます。

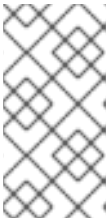
手順

1. CAM ツールがインストールされている OpenShift Container Platform クラスターにログインします。
2. 以下のコマンドを入力して CAM Web コンソール URL を取得します。

```
$ oc get -n openshift-migration route/migration -o go-template='https://{{ .spec.host }}'
```

   出力は **https://migration-openshift-migration.apps.cluster.openshift.com** のようになります。

3. ブラウザーを起動し、CAM Web コンソールに移動します。



注記

Cluster Application Migration Operator のインストール後すぐに CAM Web コンソールにアクセスしようとする場合、Operator は依然としてクラスターを設定しているため、コンソールが読み込まれないことがあります。数分待機した後に再試行します。

4. 自己署名 CA 証明書を使用している場合、ソースクラスターの API サーバーの CA 証明書を受け入れることを求めるプロンプトが出されます。Web ページは、残りの証明書を受け入れるプロセスについて説明します。
5. OpenShift Container Platform の **ユーザー名** および **パスワード** を使用してログインします。

2.2.4. Cluster Application Migration ツールのアップグレード

ソースおよびターゲットクラスターで Cluster Application Migration (CAM) ツールをアップグレードできます。

CAM 1.1 から 1.2 にアップグレードする場合、CAM Web コンソールでサービスアカウントトークンを更新する必要があります。

2.2.4.1. OpenShift Container Platform 4 クラスターでの CAM ツールのアップグレード

Operator Lifecycle Manager を使用して OpenShift Container Platform 4 クラスターで CAM ツールをアップグレードできます。

Cluster Application Migration Operator にサブスクライブする際に **自動** 承認オプションを選択した場合、CAM ツールは自動的に更新されます。

以下の手順では、**手動** 承認オプションを **自動** に変更するか、またはリリースチャネルを変更できます。

手順

1. OpenShift Container Platform コンソールで、**Operators** > **Installed Operators** に移動します。
2. **Cluster Application Migration Operator** をクリックします。
3. **Subscription** タブで、**Approval** オプションを **Automatic** に変更します。
4. オプション: **チャネル** を編集します。
   サブスクリプションを更新すると、更新された Cluster Application Migration Operator がデプロイされ、CAM ツールコンポーネントが更新されます。

2.2.4.2. サービスアカウントトークンの更新

CAM 1.1 から 1.2 にアップグレードする場合、CAM Web コンソールでサービスアカウントトークンを更新する必要があります。

CAM 1.1 は **mig** サービスアカウントを使用し、CAM 1.2 は **migration-controller** サービスアカウントを使用します。

手順

1. クラスターにログインし、**migration-controller** サービスアカウントトークンを取得します。

   ```
   $ oc sa get-token -n openshift-migration migration-controller
   ```

2. CAM Web コンソールにログインし、**Clusters** をクリックします。
3. クラスターの Options メニュー  をクリックし、**Edit** を選択します。
4. 新たなトークンを **Service account token** フィールドにコピーします。
5. **Update cluster** をクリックしてから、**Close** をクリックします。
   クラスターのサービスアカウントトークンが更新されます。

2.3. レプリケーションリポジトリーの設定

オブジェクトストレージをレプリケーションリポジトリーとして使用するように設定する必要があります。Cluster Application Migration ツールは、データをソースクラスターからレプリケーションリポジトリーにコピーしてから、レプリケーションリポジトリーからターゲットクラスターにコピーします。

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するために ファイルシステムおよびスナップショットによるデータのコピー方法 をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

以下のストレージプロバイダーがサポートされています。

- Multi-Cloud Object Gateway (MCG)
- Amazon Web Services (AWS) S3
- Google Cloud Provider (GCP)
- Microsoft Azure
- 汎用 S3 オブジェクトストレージ (例: Minio または Ceph S3)

ソースおよびターゲットクラスターには、移行時にレプリケーションリポジトリーへのネットワークアクセスがなければなりません。

制限された環境では、内部でホストされるレプリケーションリポジトリーを作成できます。プロキシーサーバーを使用する場合は、レプリケーションリポジトリーがホワイトリスト化されていることを確認する必要があります。

2.3.1. Multi-Cloud Object Gateway ストレージバケットをレプリケーションリポジトリーとして設定する

OpenShift Container Storage Operator をインストールし、Multi-Cloud Object Gateway (MCG) ストレージバケットをレプリケーションリポジトリーとして設定できます。

2.3.1.1. OpenShift Container Storage Operator のインストール

OpenShift Container Storage Operator は、OperatorHub からインストールできます。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** (この場合は、**OCS**) を使用し、 **OpenShift Container Storage Operator** を見つけます。
3. **OpenShift Container Storage Operator** を選択し、**Install** をクリックします。
4. **Update Channel**、**Installation Mode**、および **Approval Strategy** を選択します。
5. **Subscribe** をクリックします。
   Installed Operators ページで、**OpenShift Container Storage Operator** は、**Succeeded** のステータスと共に **openshift-storage** プロジェクトに表示されます。

2.3.1.2. Multi-Cloud Object Gateway ストレージバケットの作成

Multi-Cloud Object Gateway (MCG) ストレージバケットのカスタムリソース (CR) を作成できます。

手順

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

```
$ oc login
```

2. **NooBaa** CR 設定ファイル **noobaa.yml** を以下の内容で作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: NooBaa
metadata:
  name: noobaa
  namespace: openshift-storage
spec:
 dbResources:
   requests:
     cpu: 0.5 (1)
     memory: 1Gi
 coreResources:
   requests:
     cpu: 0.5 (2)
     memory: 1Gi
```

(1) (2) 非常に小規模なクラスターの場合、**cpu** の値を **0.1** に変更できます。

3. **NooBaa** オブジェクトを作成します。

```
$ oc create -f noobaa.yml
```

4. 以下の内容で、**BackingStore** CR 設定ファイル **bs.yml** を作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: BackingStore
metadata:
  finalizers:
  - noobaa.io/finalizer
  labels:
    app: noobaa
  name: mcg-pv-pool-bs
  namespace: openshift-storage
spec:
  pvPool:
    numVolumes: 3 (1)
    resources:
      requests:
        storage: 50Gi (2)
    storageClass: gp2 (3)
  type: pv-pool
```

1. PV プール内のボリューム数を指定します。
2. ボリュームのサイズを指定します。
3. ストレージクラスを作成します。

5. **BackingStore** オブジェクトを作成します。

```
$ oc create -f bs.yml
```

6. 以下の内容で **BucketClass** CR 設定ファイル **bc.yml** を作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: BucketClass
metadata:
  labels:
    app: noobaa
  name: mcg-pv-pool-bc
  namespace: openshift-storage
spec:
  placementPolicy:
    tiers:
    - backingStores:
      - mcg-pv-pool-bs
      placement: Spread
```

7. **BucketClass** オブジェクトを作成します。

```
$ oc create -f bc.yml
```

8. 以下の内容で **ObjectBucketClaim** CR 設定ファイル **obc.yml** を作成します。

```
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: migstorage
  namespace: openshift-storage
spec:
  bucketName: migstorage 1
  storageClassName: openshift-storage.noobaa.io
  additionalConfig:
    bucketclass: mcg-pv-pool-bc
```

1. レプリケーションリポジトリーを CAM Web コンソールに追加するために使用するバケット名を記録します。

9. **ObjectBucketClaim** オブジェクトを作成します。

```
$ oc create -f obc.yml
```

10. リソース作成プロセスを監視し、**ObjectBucketClaim** ステータスが **Bound** であることを確認します。

-

```
$ watch -n 30 'oc get -n openshift-storage objectbucketclaim migstorage -o yaml'
```

このプロセスには、5 分から 10 分の時間がかかる場合があります。

11. 以下の値を取得して記録します。この値は、レプリケーションリポジトリーを CAM Web コンソールに追加する際に必要になります。

    - S3 エンドポイント:

      ```
      $ oc get route -n openshift-storage s3
      ```

    - S3 プロバイダーアクセスキー:

      ```
      $ oc get secret -n openshift-storage migstorage -o go-template='{{
      .data.AWS_ACCESS_KEY_ID }}' | base64 -d
      ```

    - S3 プロバイダーシークレットアクセスキー:

      ```
      $ oc get secret -n openshift-storage migstorage -o go-template='{{
      .data.AWS_SECRET_ACCESS_KEY }}' | base64 -d
      ```

2.3.2. AWS S3 ストレージバケットをレプリケーションリポジトリーとして設定する

AWS S3 ストレージバケットをレプリケーションリポジトリーとして設定できます。

前提条件

- AWS S3 ストレージバケットは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。
- AWS CLI がインストールされていること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - EC2 Elastic Block Storage (EBS) にアクセスできる必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. AWS S3 バケットを作成します。

   ```
   $ aws s3api create-bucket \
     --bucket <bucket_name> \ 1
     --region <bucket_region> 2
   ```

   1 S3 バケット名を指定します。

   2 S3 バケットリージョンを指定します (例: **us-east-1**)。

2. IAM ユーザー **velero** を作成します。

```
$ aws iam create-user --user-name velero
```

3. EC2 EBS スナップショットポリシーを作成します。

```
$ cat > velero-ec2-snapshot-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes",
                "ec2:DescribeSnapshots",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:CreateSnapshot",
                "ec2:DeleteSnapshot"
            ],
            "Resource": "*"
        }
    ]
}
EOF
```

4. 1 つまたはすべての S3 バケットの AWS S3 アクセスポリシーを作成します。

```
$ cat > velero-s3-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket_name>/*" (1)
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket_name>" (2)
            ]
```

```
        }
    ]
}
EOF
```

1 2 単一の S3 バケットへのアクセスを付与するには、バケット名を指定します。すべての AWS S3 バケットへのアクセスを付与するには、バケット名の代わりに * を指定します。

```
"Resource": [
    "arn:aws:s3:::*"
```

5. EC2 EBS ポリシーを **velero** に割り当てます。

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero-ebs \
  --policy-document file://velero-ec2-snapshot-policy.json
```

6. AWS S3 ポリシーを **velero** に割り当てます。

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero-s3 \
  --policy-document file://velero-s3-policy.json
```

7. **velero** のアクセスキーを作成します。

```
$ aws iam create-access-key --user-name velero
{
  "AccessKey": {
        "UserName": "velero",
        "Status": "Active",
        "CreateDate": "2017-07-31T22:24:41.576Z",
        "SecretAccessKey": <AWS_SECRET_ACCESS_KEY>, 1
        "AccessKeyId": <AWS_ACCESS_KEY_ID> 2
    }
}
```

1 2 AWS リポジトリーを CAM Web コンソールに追加するために **AWS_SECRET_ACCESS_KEY** および **AWS_ACCESS_KEY_ID** を記録します。

2.3.3. Google Cloud Provider ストレージバケットをレプリケーションリポジトリーとして設定する

Google Cloud Provider (GCP) ストレージバケットをレプリケーションリポジトリーとして設定できます。

前提条件

- GCP ストレージバケットは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。

- gsutil がインストールされていること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. **gsutil init** を実行してログインします。

```
$ gsutil init
Welcome! This command will take you through the configuration of gcloud.

Your current configuration has been set to: [default]

To continue, you must login. Would you like to login (Y/n)?
```

2. **BUCKET** 変数を設定します。

```
$ BUCKET=<bucket_name> 1
```

1 バケット名を指定します。

3. ストレージバケットを作成します。

```
$ gsutil mb gs://$BUCKET/
```

4. **PROJECT_ID** 変数をアクティブなプロジェクトに設定します。

```
$ PROJECT_ID=$(gcloud config get-value project)
```

5. **velero** サービスアカウントを作成します。

```
$ gcloud iam service-accounts create velero \
    --display-name "Velero Storage"
```

6. **SERVICE_ACCOUNT_EMAIL** 変数をサービスアカウントのメールアドレスに設定します。

```
$ SERVICE_ACCOUNT_EMAIL=$(gcloud iam service-accounts list \
  --filter="displayName:Velero Storage" \
  --format 'value(email)')
```

7. パーミッションをサービスアカウントに付与します。

```
$ ROLE_PERMISSIONS=(
    compute.disks.get
    compute.disks.create
    compute.disks.createSnapshot
    compute.snapshots.get
```

```
    compute.snapshots.create
    compute.snapshots.useReadOnly
    compute.snapshots.delete
    compute.zones.get
)

gcloud iam roles create velero.server \
    --project $PROJECT_ID \
    --title "Velero Server" \
    --permissions "$(IFS=","; echo "${ROLE_PERMISSIONS[*]}")"

gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member serviceAccount:$SERVICE_ACCOUNT_EMAIL \
    --role projects/$PROJECT_ID/roles/velero.server

gsutil iam ch serviceAccount:$SERVICE_ACCOUNT_EMAIL:objectAdmin gs://${BUCKET}
```

8. サービスアカウントのキーを現在のディレクトリーにある **credentials-velero** ファイルに保存します。

```
$ gcloud iam service-accounts keys create credentials-velero \
  --iam-account $SERVICE_ACCOUNT_EMAIL
```

2.3.4. Microsoft Azure Blob ストレージコンテナーをレプリケーションリポジトリーとして設定

Microsoft Azure Blob ストレージコンテナーをレプリケーションリポジトリーとして設定できます。

前提条件

- Azure ストレージアカウント があること。
- Azure CLI がインストールされていること。
- Azure Blob ストレージコンテナーがソースクラスターおよびターゲットクラスターからアクセスできること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. **AZURE_RESOURCE_GROUP** 変数を設定します。

```
$ AZURE_RESOURCE_GROUP=Velero_Backups
```

2. Azure リソースグループを作成します。

```
$ az group create -n $AZURE_RESOURCE_GROUP --location <CentralUS> 1
```

1 場所を指定します。

3. **AZURE_STORAGE_ACCOUNT_ID** 変数を設定します。

```
$ AZURE_STORAGE_ACCOUNT_ID=velerobackups
```

4. Azure ストレージアカウントを作成します。

```
$ az storage account create \
  --name $AZURE_STORAGE_ACCOUNT_ID \
  --resource-group $AZURE_RESOURCE_GROUP \
  --sku Standard_GRS \
  --encryption-services blob \
  --https-only true \
  --kind BlobStorage \
  --access-tier Hot
```

5. **BLOB_CONTAINER** 変数を設定します。

```
$ BLOB_CONTAINER=velero
```

6. Azure Blob ストレージコンテナーを作成します。

```
$ az storage container create \
  -n $BLOB_CONTAINER \
  --public-access off \
  --account-name $AZURE_STORAGE_ACCOUNT_ID
```

7. **velero** のサービスプリンシパルおよび認証情報を作成します。

```
$ AZURE_SUBSCRIPTION_ID=`az account list --query '[?isDefault].id' -o tsv`
$ AZURE_TENANT_ID=`az account list --query '[?isDefault].tenantId' -o tsv`
$ AZURE_CLIENT_SECRET=`az ad sp create-for-rbac --name "velero" --role "Contributor" --
query 'password' -o tsv`
$ AZURE_CLIENT_ID=`az ad sp list --display-name "velero" --query '[0].appId' -o tsv`
```

8. サービスプリンシパルの認証情報を **credentials-velero** ファイルに保存します。

```
$ cat << EOF  > ./credentials-velero
AZURE_SUBSCRIPTION_ID=${AZURE_SUBSCRIPTION_ID}
AZURE_TENANT_ID=${AZURE_TENANT_ID}
AZURE_CLIENT_ID=${AZURE_CLIENT_ID}
AZURE_CLIENT_SECRET=${AZURE_CLIENT_SECRET}
AZURE_RESOURCE_GROUP=${AZURE_RESOURCE_GROUP}
AZURE_CLOUD_NAME=AzurePublicCloud
EOF
```

2.4. CAM WEB コンソールを使用したアプリケーションの移行

アプリケーションワークロードの移行は、クラスターおよびレプリケーションリポジトリーを CAM Web コンソールに追加することによって実行できます。次に、移行計画を作成し、これを実行できます。

クラスターまたはレプリケーションリポジトリーのセキュリティーを自己署名証明書で保護する場合、CA 証明書バンドルファイルを作成するか、または SSL 検証を無効にできます。

2.4.1. CA 証明書バンドルファイルの作成

自己署名証明書を使用してクラスターまたはレプリケーションリポジトリーのセキュリティーを保護する場合、証明書の検証は **Certificate signed by unknown authority** というエラーメッセージを出して失敗する可能性があります。

カスタム CA 証明書バンドルファイルを作成し、クラスターまたはレプリケーションリポジトリーの追加時に CAM Web コンソールでこれをアップロードできます。

手順

リモートエンドポイントから CA 証明書をダウンロードし、これを CA バンドルファイルとして保存します。

```
$ echo -n | openssl s_client -connect <host_FQDN>:<port> \ 1
  | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > <ca_bundle.cert> 2
```

1 エンドポイントのホスト FQDN およびポートを指定します (例: **api.my-cluster.example.com:6443**)。

2 CA バンドルファイルの名前を指定します。

2.4.2. CAM Web コンソールへのクラスターの追加

クラスターを CAM Web コンソールに追加できます。

前提条件

Azure スナップショットを使用してデータをコピーする場合:

- ソースクラスターの追加時に Azure リソースグループ名を指定する必要があります。
- ソースおよびターゲットクラスターは同じ Azure リソースグループにあり、同じ場所にある必要があります。

手順

1. クラスターにログインします。
2. サービスアカウントトークンを取得します。

```
$ oc sa get-token migration-controller -n openshift-migration
eyJhbGciOiJSUzI1NiIsImtpZCI6IiJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwi
a3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJtaWciLCJrdWJlcm5ldGVz
LmlvL3NlcnZpY2VhY2NvdW50L3NlY3JldC5uYW1lIjoibWlnLXRva2VuLWs4dDJyIiwia3ViZXJuZ
XRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6Im1pZyIsImt1YmV
ybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50LnVpZCI6ImE1YjFiYWM
wLWMxYmYtMTFlOS05Y2NiLTAyOWRmODYwYjMwOCIsInN1YiI6InN5c3RlbTpzZXJ2aWNlY
WNjb3VudDptaWc6bWlnIn0.xqeeAINK7UXpdRqAtOj70qhBJPeMwmgLomV9iFxr5RoqUgKchZ
RG2J2rkqmPm6vr7K-
cm7ibD1IBpdQJCcVDuoHYsFgV4mp9vgOfn9osSDp2TGikwNz4Az95e81xnjVUmzh-
NjDsEpw71DH92iHV_xt2sTwtzftS49LpPW2LjrV0evtNBP_t_RfskdArt5VSv25eORl7zScqfe1CiM
```

```
kcVbf2UqACQjo3LbkpfN26HAioO2oH0ECPiRzT0Xyh-KwFutJLS9Xgghyw-
LD9kPKcE_xbbJ9Y4Rqajh7WdPYuB0Jd9DPVrslmzK-F6cgHHYoZEv0SvLQi-
PO0rpDrcjOEQQ
```

3. CAM Web コンソールにログインします。
4. **Clusters** セクションで、**Add cluster** をクリックします。
5. 以下のフィールドに値を入力します。
   - **Cluster name**: 小文字 (**a-z**) および数字 (**0-9**) を含めることができます。空白文字や国際文字を含めることはできません。
   - **Url**: クラスターの API サーバーの URL です (例: **https://<master1.example.com>:8443**)。
   - **Service account token**: ソースクラスターから取得される文字列。
   - **Azure cluster**: オプション。Azure スナップショットを使用してデータをコピーする場合はこれを選択します。
   - **Azure resource group**: このフィールドは、**Azure cluster** にチェックを付けると表示されます。
   - カスタム CA バンドルを使用する場合は、**Browse** をクリックし、CA バンドルファイルを参照します。
6. **Add cluster** をクリックします。
   クラスターが **Clusters** セクションに表示されます。

2.4.3. CAM Web コンソールへのレプリケーションリポジトリーの追加

CAM Web コンソールには、オブジェクトストレージバケットをレプリケーションリポジトリーとして追加できます。

前提条件

- データを移行するには、オブジェクトストレージバケットを設定する必要があります。

手順

1. CAM Web コンソールにログインします。
2. **Replication repositories** セクションで、**Add repository** をクリックします。
3. **Storage provider type** を選択し、以下のフィールドに入力します。
   - **AWS** (AWS S3、MCG、および汎用 S3 プロバイダーの場合):
     - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
     - **S3 bucket name**: 作成した S3 バケットの名前を指定します。
     - **S3 bucket region**: S3 バケットリージョンを指定します。AWS S3 の場合に**必須**です。**Optional** (他の S3 プロバイダーの場合)。

- **S3 endpoint**: バケットではなく S3 サービスの URL を指定します (例: **https://<s3-storage.apps.cluster.com>**)。汎用 S3 プロバイダーの場合は**必須**です。**https://** 接頭辞を使用する必要があります。
- **S3 provider access key**: AWS には **<AWS_SECRET_ACCESS_KEY>** を指定するか、または MCG には S3 プロバイダーアクセスキーを指定します。
- **S3 provider secret access key**: AWS には **<AWS_ACCESS_KEY_ID>** を指定するか、または MCG には S3 プロバイダーシークレットアクセスキーを指定します。
- **Require SSL verification**: 汎用 S3 プロバイダーを使用している場合は、このチェックボックスをクリアします。
- カスタム CA バンドルを使用する場合は、**Browse** をクリックし、Base64 でエンコードされた CA バンドルファイルを参照します。

- **GCP**:
  - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
  - **GCP bucket name**: GCP バケットの名前を指定します。
  - **GCP credential JSON blob**: **credentials-velero** ファイルに文字列を指定します。
- **Azure**:
  - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
  - **Azure resource group**: Azure Blob ストレージのリソースグループを指定します。
  - **Azure storage account name**: Azure Blob ストレージアカウント名を指定します。
  - **Azure credentials - INI file contents** **credentials-velero** ファイルに文字列を指定します。

4. **Add repository** をクリックし、接続の検証を待機します。
5. **Close** をクリックします。
   新規リポジトリーが **Replication repositories** セクションに表示されます。

2.4.4. 大規模な移行の場合の移行計画の制限の変更

大規模な移行の移行計画の制限を変更することができます。

重要

移行の失敗を防ぐために、まず変更についてのテストをご使用の環境で実行する必要があります。

単一の移行計画には以下のデフォルト制限があります。

- 10 namespace
  この制限を超えると、CAM Web コンソールは **Namespace limit exceeded**エラーを表示し、移行計画を作成することができません。

- 100 Pod
  Pod の制限を超える場合、CAM Web コンソールには、以下の例と同様の警告メッセージが表示されます。**Plan has been validated with warning condition(s).See warning message.Pod limit: 100 exceeded, found: 104**

- 100 永続ボリューム
  永続ボリュームの制限を超過すると、CAM Web コンソールには同様の警告メッセージが表示されます。

手順

1. Migration コントローラー CR を編集します。

```
$ oc get migrationcontroller -n openshift-migration
NAME AGE
migration-controller 5d19h

$ oc edit migrationcontroller -n openshift-migration
```

2. 以下のパラメーターを更新します。

```
...
migration_controller: true

# This configuration is loaded into mig-controller, and should be set on the
# cluster where `migration_controller: true`
mig_pv_limit: 100
mig_pod_limit: 100
mig_namespace_limit: 10
...
```

2.4.5. CAM Web コンソールでの移行計画の作成

CAM Web コンソールで移行計画を作成できます。

前提条件

- CAM Web コンソールには以下が含まれている必要があります。
  - ソースクラスター
  - CAM ツールのインストール時に自動的に追加されるターゲットクラスター
  - レプリケーションリポジトリー
- ソースおよびターゲットクラスターには、相互に対するネットワークアクセスやレプリケーションリポジトリーへのネットワークアクセスがなければなりません。
- スナップショットを使用してデータをコピーする場合、ソースおよびターゲットクラスターは、同じクラウドプロバイダー (AWS、GCP、または Azure) および同じリージョンで実行される必要があります。

手順

1. CAM Web コンソールにログインします。

2. **Plans** セクションで、**Add plan** をクリックします。

3. **Plan name** を入力し、**Next** をクリックします。
Plan name には、最大 253 文字の小文字の英数字 (**a-z、0-9**) を含めることができます。スペースまたはアンダースコア (_) を含めることはできません。

4. **Source cluster** を選択します。

5. **Target cluster**を選択します。

6. **Replication repository** を選択します。

7. 移行するプロジェクトを選択し、**Next** をクリックします。

8. PV の **Copy** または **Move** を選択します。

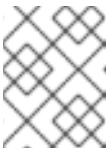
   - **Copy** は、ソースクラスターの PV のデータをレプリケーションリポジトリーにコピーしてから、これを同様の特徴のある新規に作成された PV で復元します。
   オプション: **Verify copy** を選択して、ファイルシステムメソッドでコピーされたデータを検証できます。このオプションを使用すると、各ソースファイルのチェックサムが生成され、復元後のチェックが実行されるため、パフォーマンスが大幅に低下します。

   - **Move** は、ソースクラスターからリモートボリューム (例: NFS) をアンマウントし、リモートボリュームをポイントするターゲットクラスターで PV リソースを作成し、その後にリモートボリュームをターゲットクラスターにマウントします。ターゲットクラスターで実行されているアプリケーションは、ソースクラスターが使用していたものと同じリモートボリュームを使用します。リモートボリュームは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。

9. **Next** をクリックします。

10. PV の **Copy method** を選択します。

    - **Snapshot** は、クラウドプロバイダーのスナップショット機能を使用してディスクのバックアップおよび復元を行います。この場合、**ファイルシステム**を使用する場合よりもはるかに高速になります。

      注記

      ストレージおよびクラスターは同じリージョンにあり、ストレージクラスには互換性がなければなりません。

    - **ファイルシステム**は、データファイルをソースディスクから新規に作成されたターゲットディスクにコピーします。

11. PV の **Storage class** を選択します。
Filesystem のコピー方法を選択した場合、移行時にストレージクラスを変更できます。たとえば、Red Hat Gluster Storage または NFS ストレージから Red Hat Ceph Storage に変更できます。

12. **Next** をクリックします。

13. 移行フックを追加する必要がある場合は、**Add Hook** をクリックして、以下の手順を実行します。

    a. フックの名前を指定します。

b. **Ansible playbook** を選択して独自の Playbook を使用するか、または別の言語で作成されがフックに **Custom container image** を選択します。

c. **Browse** をクリックして Playbook をアップロードします。

d. オプション: デフォルトの Ansible ランタイムイメージを使用していない場合は、カスタムの Ansible イメージを指定します。

e. フックを実行する必要のあるクラスターを指定します。

f. サービスアカウント名を指定します。

g. namespace を指定します。

h. フックを実行する必要のある移行手順を選択します。

- PreBackup: バックアップタスクがソースクラスターで開始される前
- PostBackup: バックアップタスクがソースクラスターで完了した後
- PreRestore: 復元タスクがターゲットクラスターで開始される前
- PostRestore: 復元タスクがターゲットクラスターで完了した後

14. **Add** をクリックします。
移行計画に最大 4 つのフックを追加し、それぞれのフックを異なる移行手順に割り当てることができます。

15. **Finish** をクリックします。

16. **Close** をクリックします。
移行計画は **Plans** セクションに表示されます。

2.4.6. CAM Web コンソールでの移行計画の実行

CAM Web コンソールで作成した移行計画を使用してアプリケーションとデータをステージングしたり、移行したりできます。

前提条件

CAM Web コンソールには以下が含まれている必要があります。

- ソースクラスター
- CAM ツールのインストール時に自動的に追加されるターゲットクラスター
- レプリケーションリポジトリー
- 有効な移行計画

手順

1. ターゲットクラスターの CAM Web コンソールにログインします。
2. 移行計画を選択します。

3. **Stage** をクリックし、アプリケーションを停止せずにソースクラスターからターゲットクラスターにデータをコピーします。
実際の移行時間を短縮するには、**Stage** を複数回実行することができます。

4. アプリケーションのワークロードを移行する準備ができたら、**Migrate** をクリックします。
Migrate は、ソースクラスターでアプリケーションワークロードを停止し、ターゲットクラスターでそのリソースを再作成します。

5. オプション: **Migrate** ウィンドウで **Do not stop applications on the source cluster during migration** を選択できます。

6. **Migrate** をクリックします。

7. オプション: 進行中の移行を停止するには、Options メニュー  をクリックし、**Cancel** を選択します。

8. 移行が完了したら、アプリケーションが OpenShift Container Platform Web コンソールで正常に移行されていることを確認します。

   a. **Home → Projects** をクリックします。

   b. 移行されたプロジェクトをクリックしてそのステータスを表示します。

   c. **Routes** セクションで **Location** をクリックし、アプリケーションが機能していることを確認します (該当する場合)。

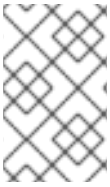
   d. **Workloads → Pods** をクリックし、Pod が移行した namespace で実行されていることを確認します。

   e. **Storage → Persistent volumes** をクリックして、移行した永続ボリュームが正常にプロビジョニングされていることを確認します。

2.5. トラブルシューティング

移行用のカスタムリソース (CR) を表示し、ログをダウンロードして失敗した移行の失敗についてのトラブルシューティングを行うことができます。

移行の失敗時にアプリケーションが停止した場合は、データ破損を防ぐために手作業でアプリケーションをロールバックする必要があります。

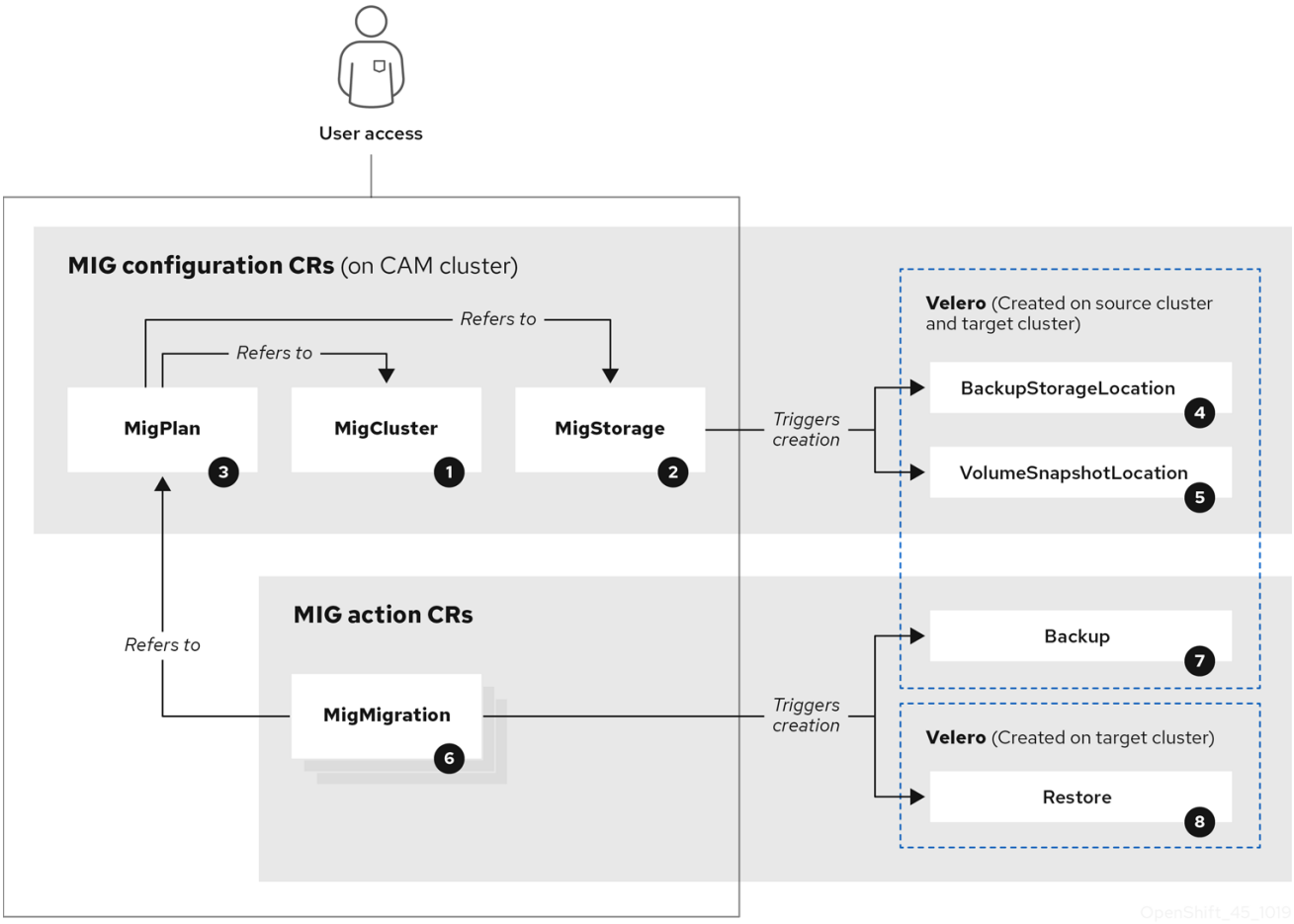


注記

移行時にアプリケーションが停止しなかった場合には、手動のロールバックは必要ありません。元のアプリケーションがソースクラスター上で依然として実行されているためです。

2.5.1. 移行カスタムリソースの表示

Cluster Application Migration (CAM) ツールは以下の カスタムリソース (Custom Resource、CR) を作成します。

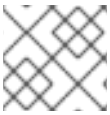


1 MigCluster (設定、CAM クラスター): クラスター定義

2 MigStorage (設定、CAM クラスター): ストレージ定義

3 MigPlan (設定、CAM クラスター): 移行計画

MigPlan CR は移行されるソースおよびターゲットクラスター、リポジトリー、および namespace を記述します。これは 0、1 または多数の MigMigration CR に関連付けられます。



注記

MigPlan CR を削除すると、関連付けられた MigMigration CR が削除されます。

4 BackupStorageLocation (設定、CAM クラスター): Velero バックアップオブジェクトの場所

5 VolumeSnapshotLocation (設定、CAM クラスター): Velero ボリュームスナップショットの場所

6 MigMigration (アクション、CAM クラスター): 移行、移行時に作成される

MigMigration CR は、データのステージングまたは移行を実行するたびに作成されます。各 MigMigration CR は MigPlan CR に関連付けられます。

7 Backup (アクション、ソースクラスター): 移行計画の実行時に、MigMigration CR は各ソースクラスターに 2 つの Velero バックアップ CR を作成します。

- Kubernetes オブジェクトのバックアップ CR #1
- PV データのバックアップ CR #2

8 Restore (アクション、ターゲットクラスター): 移行計画の実行時に、MigMigration CR はターゲットクラスターに 2 つの Velero リスト CR を作成します。

- PV データのリストア CR #1 (バックアップ CR #2 の使用)
- Kubernetes オブジェクトの復元 CR #2 (バックアップ CR #1 の使用)

手順

1. CR 名を取得します。

```
$ oc get <migration_cr> -n openshift-migration 1
```

1 移行 CR を指定します (例: **migmigration**)。

出力は以下の例のようになります。

```
NAME                                   AGE
88435fe0-c9f8-11e9-85e6-5d593ce65e10   6m42s
```

2. CR を表示します。

```
$ oc describe <migration_cr> <88435fe0-c9f8-11e9-85e6-5d593ce65e10> -n openshift-migration
```

出力は以下の例のようになります。

MigMigration の例

```
name:         88435fe0-c9f8-11e9-85e6-5d593ce65e10
namespace:    openshift-migration
labels:       <none>
annotations:  touch: 3b48b543-b53e-4e44-9d34-33563f0f8147
apiVersion:  migration.openshift.io/v1alpha1
kind:         MigMigration
metadata:
  creationTimestamp:  2019-08-29T01:01:29Z
  generation:         20
  resourceVersion:    88179
  selfLink:           /apis/migration.openshift.io/v1alpha1/namespaces/openshift-migration/migmigrations/88435fe0-c9f8-11e9-85e6-5d593ce65e10
  uid:                8886de4c-c9f8-11e9-95ad-0205fe66cbb6
spec:
  migPlanRef:
    name:         socks-shop-mig-plan
    namespace:    openshift-migration
```

```
  quiescePods:  true
  stage:        false
status:
  conditions:
    category:            Advisory
    durable:             True
    lastTransitionTime:  2019-08-29T01:03:40Z
    message:             The migration has completed successfully.
    reason:              Completed
    status:              True
    type:                Succeeded
  phase:                 Completed
  startTimestamp:        2019-08-29T01:01:29Z
events:                  <none>
```

Velero バックアップ CR #2 の例 (PV データ)

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.105.179:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-44dd3bd5-c9f8-11e9-95ad-
0205fe66cbb6
  creationTimestamp: "2019-08-29T01:03:15Z"
  generateName: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-
  generation: 1
  labels:
    app.kubernetes.io/part-of: migration
    migmigration: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
    migration-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
    velero.io/storage-location: myrepo-vpzq9
  name: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-59gb7
  namespace: openshift-migration
  resourceVersion: "87313"
  selfLink: /apis/velero.io/v1/namespaces/openshift-migration/backups/88435fe0-c9f8-11e9-85e6-
5d593ce65e10-59gb7
  uid: c80dbbc0-c9f8-11e9-95ad-0205fe66cbb6
spec:
  excludedNamespaces: []
  excludedResources: []
  hooks:
    resources: []
  includeClusterResources: null
  includedNamespaces:
  - sock-shop
  includedResources:
  - persistentvolumes
  - persistentvolumeclaims
  - namespaces
  - imagestreams
  - imagestreamtags
  - secrets
  - configmaps
```

```
  - pods
  labelSelector:
    matchLabels:
      migration-included-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
  storageLocation: myrepo-vpzq9
  ttl: 720h0m0s
  volumeSnapshotLocations:
  - myrepo-wv6fx
status:
  completionTimestamp: "2019-08-29T01:02:36Z"
  errors: 0
  expiration: "2019-09-28T01:02:35Z"
  phase: Completed
  startTimestamp: "2019-08-29T01:02:35Z"
  validationErrors: null
  version: 1
  volumeSnapshotsAttempted: 0
  volumeSnapshotsCompleted: 0
  warnings: 0
```

Velero リストア CR #2 の例 (Kubernetes リソース)

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.90.187:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-36f54ca7-c925-11e9-825a-
06fa9fb68c88
  creationTimestamp: "2019-08-28T00:09:49Z"
  generateName: e13a1b60-c927-11e9-9555-d129df7f3b96-
  generation: 3
  labels:
    app.kubernetes.io/part-of: migration
    migmigration: e18252c9-c927-11e9-825a-06fa9fb68c88
    migration-final-restore: e18252c9-c927-11e9-825a-06fa9fb68c88
  name: e13a1b60-c927-11e9-9555-d129df7f3b96-gb8nx
  namespace: openshift-migration
  resourceVersion: "82329"
  selfLink: /apis/velero.io/v1/namespaces/openshift-migration/restores/e13a1b60-c927-11e9-9555-
d129df7f3b96-gb8nx
  uid: 26983ec0-c928-11e9-825a-06fa9fb68c88
spec:
  backupName: e13a1b60-c927-11e9-9555-d129df7f3b96-sz24f
  excludedNamespaces: null
  excludedResources:
  - nodes
  - events
  - events.events.k8s.io
  - backups.velero.io
  - restores.velero.io
  - resticrepositories.velero.io
  includedNamespaces: null
  includedResources: null
```

```
  namespaceMapping: null
  restorePVs: true
status:
  errors: 0
  failureReason: ""
  phase: Completed
  validationErrors: null
  warnings: 15
```

2.5.2. 移行ログのダウンロード

CAM Web コンソールで Velero、Restic、および Migration コントローラーログをダウンロードして、移行の失敗についてのトラブルシューティングを行うことができます。

手順

1. CAM Web コンソールにログインします。
2. **Plans** をクリックし、 移行計画の一覧を表示します。
3. 特定の移行計画の **Options** メニュー  をクリックし、**Logs** を選択します。
4. **Download Logs** をクリックし、すべてのクラスターの Migration コントローラー、Velero、および Restic のログをダウンロードします。
5. 特定のログをダウンロードするには、以下を実行します。
   a. ログオプションを指定します。
      - **Cluster**: ソース、ターゲット、または CAM ホストクラスターを選択します。
      - **Log source**: **Velero**、**Restic**、または **Controller** を選択します。
      - **Pod source**: Pod 名を選択します (例: **controller-manager-78c469849c-v6wcf**)。選択したログが表示されます。

        選択した内容を変更することで、ログ選択の設定をクリアできます。

   b. **Download Selected** をクリックし、選択したログをダウンロードします。

オプションで、以下の例にあるように CLI を使用してログにアクセスできます。

```
$ oc get pods -n openshift-migration | grep controller
controller-manager-78c469849c-v6wcf          1/1     Running     0          4h49m

$ oc logs controller-manager-78c469849c-v6wcf -f -n openshift-migration
```

2.5.3. エラーメッセージ

2.5.3.1. Velero Pod ログの Restic タイムアウトエラーメッセージ

Restic のタイムアウトにより移行が失敗する場合、以下のエラーが Velero Pod ログに表示されます。

```
level=error msg="Error backing up item" backup=velero/monitoring error="timed out waiting for all
PodVolumeBackups to complete"
error.file="/go/src/github.com/heptio/velero/pkg/restic/backupper.go:165"
error.function="github.com/heptio/velero/pkg/restic.(*backupper).BackupPodVolumes" group=v1
```

restic_timeout のデフォルト値は 1 時間です。大規模な移行では、このパラメーターの値を大きくすることができます。値を高くすると、エラーメッセージが返されるタイミングが遅れる可能性があることに注意してください。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** に移動します。
2. **Cluster Application Migration Operator** をクリックします。
3. **MigrationController** タブで、**migration-controller** をクリックします。
4. **YAML** タブで、以下のパラメーター値を更新します。

```
spec:
  restic_timeout: 1h (1)
```

(1) 有効な単位は **h** (時間)、**m** (分)、および **s** (秒) です (例: **3h30m15s**)。

5. **Save** をクリックします。

2.5.3.2. MigMigration カスタムリソースの ResticVerifyErrors

ファイルシステムデータのコピー方法を使用して PV を移行する際にデータ検証が失敗すると、以下のエラーが MigMigration カスタムリソース (CR) に表示されます。

```
status:
  conditions:
  - category: Warn
    durable: true
    lastTransitionTime: 2020-04-16T20:35:16Z
    message: There were verify errors found in 1 Restic volume restores. See restore `<registry-
example-migration-rvwcm>`
      for details (1)
    status: "True"
    type: ResticVerifyErrors (2)
```

(1) エラーメッセージはリストア CR 名を識別します。

(2) **ResticErrors** も表示されます。**ResticErrors** は、検証エラーが含まれる一般的なエラーの警告です。

注記

データ検証エラーによって移行プロセスが失敗することはありません。

ターゲットクラスターのリストア CR を確認して、データ検証エラーのソースを特定できます。

手順

1. ターゲットクラスターにログインします。
2. リストア CR を表示します。

```
$ oc describe <registry-example-migration-rvwcm> -n openshift-migration
```

出力では、**PodVolumeRestore** エラーのある PV を特定できます。

```
status:
  phase: Completed
  podVolumeRestoreErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
  podVolumeRestoreResticErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
```

3. **PodVolumeRestore** CR を表示します。

```
$ oc describe <migration-example-rvwcm-98t49>
```

出力はエラーをログに記録した Restic Pod を特定します。

```
  completionTimestamp: 2020-05-01T20:49:12Z
  errors: 1
  resticErrors: 1
  ...
  resticPod: <restic-nr2v5>
```

4. Restic Pod ログを表示します。

```
$ oc logs -f restic-nr2v5
```

2.5.4. 移行の手動ロールバック

移行の失敗時にアプリケーションが停止された場合は、PV でのデータの破損を防ぐために手動でこれをロールバックする必要があります。

移行時にアプリケーションが停止しなかった場合には、この手順は必要ありません。元のアプリケーションがソースクラスター上で依然として実行されているためです。

手順

1. ターゲットクラスター上で、移行したプロジェクトに切り替えます。

```
$ oc project <project>
```

2. デプロイされたリソースを取得します。

```
$ oc get all
```

3. デプロイされたリソースを削除し、アプリケーションがターゲットクラスターで実行されておらず、PVC 上にあるデータにアクセスできるようにします。

```
$ oc delete <resource_type>
```

4. これを削除せずにデーモンセットを停止するには、YAML ファイルで **nodeSelector** を更新します。

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: hello-daemonset
spec:
  selector:
     matchLabels:
       name: hello-daemonset
  template:
    metadata:
      labels:
        name: hello-daemonset
    spec:
      nodeSelector:
        role: worker 1
```

1 いずれのノードにも存在しない **nodeSelector** 値を指定します。

5. 不要なデータが削除されるように、各 PV の回収ポリシーを更新します。移行時に、バインドされた PV の回収ポリシーは **Retain** であり、アプリケーションがソースクラスターから削除される際にデータの損失が生じないようにされます。ロールバック時にこれらの PV を削除できます。

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0001
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain 1
  ...
status:
  ...
```

1 **Recycle** または **Delete** を指定します。

6. ソースクラスターで、移行したプロジェクトに切り替えます。

```
$ oc project <project_name>
```

7. プロジェクトのデプロイされたリソースを取得します。

```
$ oc get all
```

8. デプロイされた各リソースのレプリカを開始します。

```
$ oc scale --replicas=1 <resource_type>/<resource_name>
```

9. 手順の実行時に変更した場合は、 **DaemonSet** リソースの **nodeSelector** を元の値に更新します。

2.5.5. カスタマーサポートケース用のデータの収集

カスタマーサポートケースを作成する場合、 **openshift-migration-must-gather-rhel8** イメージを使用して **must-gather** ツールを実行し、クラスターについての情報を収集し、これを Red Hat カスタマーポータル にアップロードできます。

openshift-migration-must-gather-rhel8 イメージは、デフォルトの **must-gather** イメージで収集されないログおよびカスタムリソースデータを収集します。

手順

1. **must-gather** データを保存するディレクトリーに移動します。
2. **oc adm must-gather** コマンドを実行します。

```
$ oc adm must-gather --image=registry.redhat.io/rhcam-1-2/openshift-migration-must-gather-rhel8
```

must-gather ツールはクラスター情報を収集し、これを **must-gather.local.<uid>** ディレクトリーに保存します。

3. 認証キーおよびその他の機密情報を **must-gather** データから削除します。
4. **must-gather.local.<uid>** ディレクトリーの内容を含むアーカイブファイルを作成します。

```
$ tar cvaf must-gather.tar.gz must-gather.local.<uid>/
```

圧縮ファイルを Red Hat カスタマーポータル 上のサポートケースに添付します。

2.5.6. 既知の問題

本リリースには、以下の既知の問題があります。

- 移行時に、Cluster Application Migration (CAM) ツールは以下の namespace アノテーションを保持します。
  - **openshift.io/sa.scc.mcs**
  - **openshift.io/sa.scc.supplemental-groups**
  - **openshift.io/sa.scc.uid-range**

これらのアノテーションは UID 範囲を保持し、コンテナーがターゲットクラスターのファイルシステムのパーミッションを保持できるようにします。移行された UID が、ターゲットクラスターの既存の namespace または今後の namespace 内の UID を重複させるリスクがあります。(BZ#1748440)

- AWS バケットが CAM Web コンソールに追加された後に削除される場合、MigStorage CR は更新されないため、そのステータスは **True** のままになります。(BZ#1738564)
- ほとんどのクラスタースコープのリソースは CAM ツールで処理されません。アプリケーションがクラスタースコープのリソースを必要とする場合、ターゲットクラスターでそれらを手動で作成する必要がある場合があります。
- 移行に失敗すると、移行計画は休止状態の Pod のカスタム PV 設定を保持しません。移行を手動でロールバックし、移行計画を削除し、PV 設定で新たな移行計画を作成する必要があります。(BZ#1784899)
- Restic のタイムアウトにより大規模な移行が失敗する場合は、Migration コントローラー CR の **restic_timeout** パラメーターの値 (デフォルト: **1h**) を増やすことができます。
- ファイルシステムのコピー方法で移行される PV のデータ検証オプションを選択すると、パフォーマンスは大幅に遅くなります。Velero は各ファイルのチェックサムを生成し、ファイルを復元する際に確認します。

第3章 OPENSHIFT CONTAINER PLATFORM 4.2 以降からの移行

3.1. 移行ツールおよび前提条件

アプリケーションワークロードを、Cluster Application Migration (CAM) ツールを使用して OpenShift Container Platform 4.2 以降から OpenShift Container Platform 4.4 に移行できます。CAM ツールを使用すると、移行を制御し、アプリケーションのダウンタイムを最小限に抑えることができます。

注記

ソースおよびターゲットクラスターが正しく設定されている場合、同じバージョンの OpenShift Container Platform クラスター間で移行できます (4.2 から 4.2 または 4.3 から 4.3 への移行など)。

Kubernetes カスタムリソースをベースとする CAM ツールの Web コンソールおよび API により、namespace の粒度でステートフルおよびステートレスのアプリケーションワークロードを移行できます。

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するためにファイルシステムおよびスナップショットによるデータのコピー方法をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

移行フックを使用して、移行中の特定の時点で Ansible Playbook を実行できます。フックは移行計画の作成時に追加されます。

3.1.1. 移行の前提条件

- ソースクラスターを最新の z-stream リリースにアップグレードすること。
- すべてのクラスターで **cluster-admin** 権限がある。
- ソースおよびターゲットクラスターには、レプリケーションリポジトリーへの無制限のネットワークアクセスがなければなりません。
- Migration コントローラーがインストールされているクラスターには、他のクラスターへの無制限のアクセスが必要です。
- アプリケーションが **openshift** namespace のイメージを使用する場合、イメージの必要なバージョンがターゲットクラスターに存在する必要があります。
  必要なイメージがない場合は、アプリケーションと互換性のある利用可能なバージョンを使用するように **imagestreamtags** 参照を更新する必要があります。**imagestreamtag** を更新できない場合、同等のイメージをアプリケーション namespace に手動でアップロードし、それらを参照するようにアプリケーションを更新できます。

以下の **imagestreamtags** は OpenShift Container Platform **4.2** から **削除** されています。

- **dotnet:1.0**、**dotnet:1.1**、 **dotnet:2.0**
- **dotnet-runtime:2.0**
- **mariadb:10.1**
- **mongodb:2.4**、**mongodb:2.6**
- **mysql:5.5**、**mysql:5.6**

- **nginx:1.8**
- **nodejs:0.10**、**nodejs:4**、**nodejs:6**
- **perl:5.16**、**perl:5.20**
- **php:5.5**、**php:5.6**
- **postgresql:9.2**、**postgresql:9.4**、**postgresql:9.5**
- **python:3.3**、**python:3.4**
- **ruby:2.0**, **ruby:2.2**

以下の **imagestreamtags** は OpenShift Container Platform **4.4** から **削除** されています。

- **dotnet: 2.2**
- **dotnet-runtime: 2.2**
- **nginx: 1.12**
- **nodejs: 8, 8-RHOAR, 10-SCL**
- **perl:5.24**
- **php: 7.0, 7.1**
- **redis: 3.2**

3.1.2. Cluster Application Migration ツールについて

Cluster Application Migration (CAM) ツールを使用すると、CAM Web コンソールまたは Kubernetes API を使用して Kubernetes リソース、永続ボリュームデータ、および内部コンテナーイメージを OpenShift Container Platform ソースクラスターから OpenShift Container Platform 4.4 ターゲットクラスターに移行できます。

CAM Web コンソールを使用してアプリケーションを移行するには、以下の手順が必要です。

1. Cluster Application Migration Operator をすべてのクラスターにインストールします。
   インターネットアクセスが制限されているか、またはインターネットアクセスのない環境で Cluster Application Migration Operator をインストールできます。ソースおよびターゲットクラスターは、相互に対するネットワークアクセスおよびミラーレジストリーへのネットワークアクセスがなければなりません。

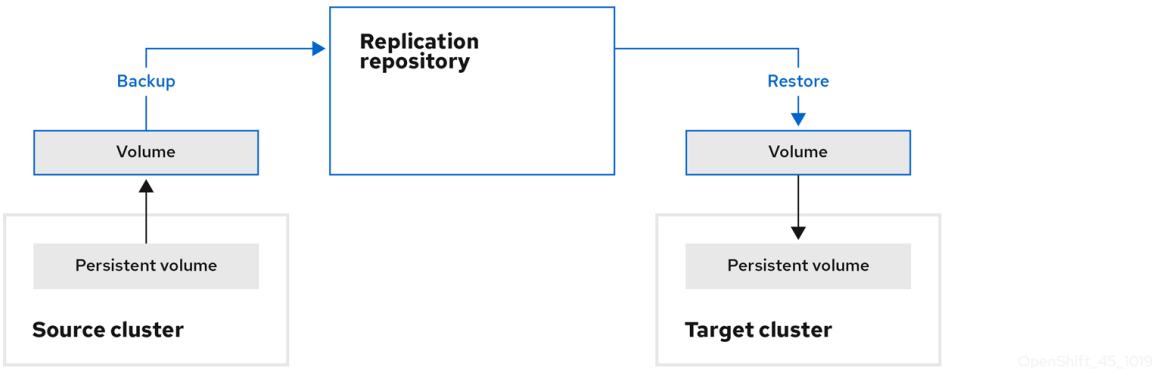
2. CAM ツールがデータ移行に使用する中間オブジェクトストレージであるレプリケーションリポジトリーを設定します。
   ソースおよびターゲットクラスターには、移行時にレプリケーションリポジトリーへのネットワークアクセスがなければなりません。制限された環境では、内部でホストされる S3 ストレージリポジトリーを使用できます。プロキシーサーバーを使用する場合は、レプリケーションリポジトリーがホワイトリスト化されていることを確認する必要があります。

3. ソースクラスターを CAM Web コンソールに追加します。

4. レプリケーションリポジトリーを CAM Web コンソールに追加します。

5. 以下のデータ移行オプションのいずれかを使用して、移行計画を作成します。

- **Copy**: CAM ツールは、データをソースクラスターからレプリケーションリポジトリーにコピーし、レプリケーションリポジトリーからターゲットクラスターにコピーします。



- **Move**: CAM ツールはリモートボリューム (NFS など) をソースクラスターからアンマウントし、リモートボリュームをポイントするターゲットクラスターで PV リソースを作成し、その後にリモートボリュームをターゲットクラスターにマウントします。ターゲットクラスターで実行されているアプリケーションは、ソースクラスターが使用していたものと同じリモートボリュームを使用します。リモートボリュームは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。

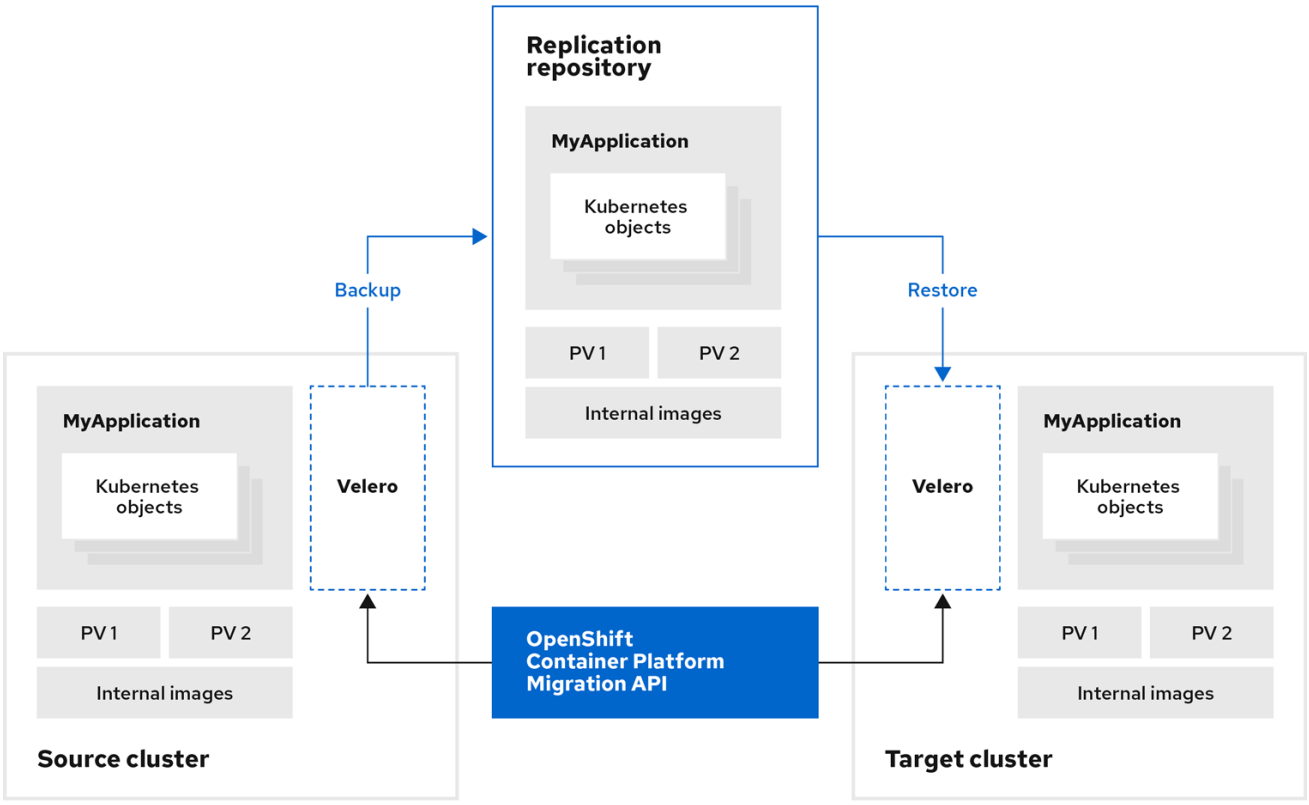
注記

レプリケーションリポジトリーはこの図には表示されていませんが、実際の移行には必要になります。

6. 以下のオプションのいずれかを使用して、移行計画を実行します。

- **Stage** (オプション) は、アプリケーションを停止せずにデータをターゲットクラスターにコピーします。
ステージングは、移行前にほとんどのデータがターゲットにコピーされるように複数回実行することができます。これにより、実際の移行時間やアプリケーションのダウンタイムが最小限に抑えられます。

- **Migrate** は、ソースクラスターでアプリケーションを停止し、ターゲットクラスターでそのリソースを再作成します。オプションで、アプリケーションを停止せずにワークロードを移行できます。



3.1.3. データのコピー方法

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するためにファイルシステムおよびスナップショットによるデータのコピー方法をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

3.1.3.1. ファイルシステムのコピー方法

CAM ツールは、データファイルをソースクラスターからレプリケーションリポジトリーにコピーし、そこからターゲットクラスターにコピーします。

表3.1 ファイルシステムのコピー方法の概要

利点	制限
• クラスターで複数の異なるストレージクラスを使用することが可能 • すべての S3 ストレージプロバイダーでサポートされている • チェックサムを使用したオプションのデータ検証	• スナップショットのコピー方法よりも遅い • オプションのデータ検証は、パフォーマンスを大幅に低下させます。

3.1.3.2. スナップショットのコピー方法

CAM ツールは、ソースクラスターのデータのスナップショットを、レプリケーションリポジトリーとして設定されたクラウドプロバイダーのオブジェクトストレージにコピーします。データはターゲットクラスターで復元されます。

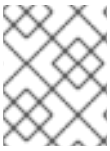
AWS、Google Cloud Provider、および Microsoft Azure は、スナップショットのコピー方法をサポートします。

表3.2 スナップショットのコピー方法の概要

利点	制限
• ファイルシステムのコピー方法よりも高速	• クラウドプロバイダーはスナップショットをサポートしている必要があります。 • クラスターは同じクラウドプロバイダーになければなりません。 • クラスターは、同じ場所またはリージョンにある必要があります。 • クラスターには同じストレージクラスがなければなりません。 • ストレージクラスにはスナップショットとの互換性がある必要があります。

3.1.4. 移行フックについて

移行フックを使用して、移行中の特定の時点で Ansible Playbook を実行できます。フックは移行計画の作成時に追加されます。



注記

Ansible Playbook を使用する必要がない場合は、カスタムコンテナーイメージを作成し、これを移行計画に追加することができます。

移行フックは、アプリケーションの休止状態のカスタマイズ、サポート外のデータタイプの手動の移行、および移行後のアプリケーションの更新などのタスクを実行します。

単一の移行フックは、以下の移行手順のいずれかでソースまたはターゲットクラスターで実行されます。

- PreBackup: バックアップタスクがソースクラスターで開始される前
- PostBackup: バックアップタスクがソースクラスターで完了した後
- PreRestore: 復元タスクがターゲットクラスターで開始される前
- PostRestore: 復元タスクがターゲットクラスターで完了した後
  1 つのフックをそれぞれの移行ステップに割り当て、単一の移行計画について最大 4 つのフックを割り当てることができます。

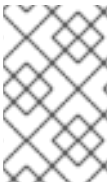
デフォルトの **hook-runner** イメージは **registry.redhat.io/rhcam-1-2/openshift-migration-hook-runner-rhel7** です。このイメージは Ansible Runner をベースとしており、Ansible Kubernetes リソース用の **python-openshift** および更新された **oc** バイナリーが含まれます。追加の Ansible モジュールまた

はツールで独自のフックイメージを作成することもできます。

Ansible Playbook はフックコンテナーに ConfigMap としてマウントされます。フックコンテナーは、指定されたサービスアカウントおよび namespace を使用してクラスターでジョブとして実行されます。ジョブは、初期の Pod がエビクトされるか、または強制終了される場合でも、デフォルトの **backoffLimit** (**6**) または正常に完了した状態に達するまで実行されます。

3.2. クラスターアプリケーション移行ツールのデプロイおよびアップグレード

OLM を使用して OpenShift Container Platform 4.4 ターゲットクラスターおよび 4.2 ソースクラスターに Cluster Application Migration Operator をインストールできます。Cluster Application Migration Operator は、デフォルトで Cluster Application Migration (CAM) ツールをターゲットクラスターにインストールします。



注記

オプション: Cluster Application Migration Operator を、CAM ツールを OpenShift Container Platform 3 クラスターまたはリモートクラスターに インストールするように設定できます。

制限された環境では、ローカルミラーレジストリーから Cluster Application Migration Operator をインストールできます。

クラスターに Cluster Application Migration Operator をインストールした後に、CAM ツールを起動できます。

3.2.1. Cluster Application Migration Operator のインストール

Cluster Application Migration Operator は Operation Lifecycle Manager (OLM) で OpenShift Container Platform 4.4 ターゲットクラスターにインストールすることも、OpenShift Container Platform 4.1 ソースクラスターにインストールすることもできます。

3.2.1.1. Cluster Application Migration Operator の OpenShift Container Platform 4.4 ターゲットクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4.4 ターゲットクラスターに Cluster Application Migration Operator をインストールできます。

Cluster Application Migration Operator は、デフォルトで Cluster Application Migration ツールをターゲットクラスターにインストールします。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。

Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。

5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **Create** をクリックします。
8. **Workloads → Pods** をクリックし、Controller Manager、Migration UI、Restic、および Velero Pod が実行中であることを確認します。

3.2.1.2. OpenShift Container Platform 4.2 ソースクラスターへの Cluster Application Migration Operator のインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4 ソースクラスターに Cluster Application Migration Operator をインストールできます。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **migration_controller** および **migration_ui** パラメーターを **spec** スタンザの **false** に設定します。

```
spec:
  ...
  migration_controller: false
  migration_ui: false
  ...
```

8. **Create** をクリックします。
9. **Workloads → Pods** をクリックし、Restic および Velero Pod が実行されていることを確認します。

3.2.2. 制限された環境での Cluster Application Migration Operator のインストール

OpenShift Container Platform 4 のカスタム Operator カタログイメージをビルドし、これをローカルミラーイメージレジストリーにプッシュし、OLM をローカルレジストリーから Operator をインストールするように設定できます。

追加リソース

- ネットワークが制限された環境での Operator Lifecycle Manager の使用

3.2.2.1. Operator カタログイメージのビルド

クラスター管理者は、Operator Lifecycle Manager (OLM) によって使用されるカスタム Operator カタログイメージをビルドし、Docker v2-2 をサポートするコンテナーイメージレジストリーにそのイメージをプッシュできます。ネットワークが制限された環境のクラスターの場合、このレジストリーには、ネットワークが制限されたインストールで作成されたミラーレジストリーなど、クラスターにネットワークアクセスのあるレジストリーを使用できます。

重要

OpenShift Container Platform クラスターの内部レジストリーはターゲットレジストリーとして使用できません。これは、ミラーリングプロセスで必要となるタグを使わないプッシュをサポートしないためです。

以下の例では、お使いのネットワークとインターネットの両方にアクセスできるミラーレジストリーを使用することを前提としています。

前提条件

- ネットワークアクセスが無制限の Linux ワークステーション
- **oc** version 4.3.5+
- **podman** version 1.4.4+
- Docker v2-2 をサポートするミラーレジストリーへのアクセス
- プライベートレジストリーを使用している場合、後続の手順で使用するために **REG_CREDS** 環境変数をレジストリー認証情報のファイルパスに設定します。たとえば **podman** CLI の場合は、以下のようになります。

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

- quay.io アカウントがアクセスできるプライベート namespace を使用している場合、Quay 認証トークンを設定する必要があります。quay.io 認証情報を使用してログイン API に対して要求を行うことにより、**--auth-token** フラグで使用できる **AUTH_TOKEN** 環境変数を設定します。

```
$ AUTH_TOKEN=$(curl -sH "Content-Type: application/json" \
    -XPOST https://quay.io/cnr/api/v1/users/login -d '
    {
        "user": {
            "username": """<quay_username>""",
            "password": """<quay_password>"""
        }
    }' | jq -r '.token')
```

手順

1. ネットワークアクセスが無制限のワークステーションで、ターゲットミラーレジストリーを使用して認証を行います。

```
$ podman login <registry_host_name>
```

また、ビルド時にベースイメージをプルできるように、**registry.redhat.io** で認証します。

```
$ podman login registry.redhat.io
```

2. quay.io から **redhat-operators** カタログをベースにカタログイメージをビルドし、そのイメージにタグを付け、ミラーレジストリーにプッシュします。

```
$ oc adm catalog build \
    --appregistry-org redhat-operators \ 1
    --from=registry.redhat.io/openshift4/ose-operator-registry:v4.4 \ 2
    --filter-by-os="linux/amd64" \ 3
    --to=<registry_host_name>:<port>/olm/redhat-operators:v1 \ 4
    [-a ${REG_CREDS}] \ 5
    [--insecure] \ 6
    [--auth-token "${AUTH_TOKEN}"] 7

INFO[0013] loading Bundles
dir=/var/folders/st/9cskxqs53ll3wdn434vw4cd80000gn/T/300666084/manifests-829192605
...
Pushed sha256:f73d42950021f9240389f99ddc5b0c7f1b533c054ba344654ff1edaf6bf827e3
to example_registry:5000/olm/redhat-operators:v1
```

1. App Registry インスタンスからのプルに使用する組織 (namespace)。
2. ターゲット OpenShift Container Platform クラスターのメジャーバージョンおよびマイナーバージョンに一致するタグを使用して、**--from** を **ose-operator-registry** ベースイメージに設定します。
3. **--filter-by-os** を、ターゲットの OpenShift Container Platform クラスターと一致する必要のある、ベースイメージに使用するオペレーティングシステムおよびアーキテクチャーに設定します。使用できる値は、**linux/amd64**、**linux/ppc64le**、および **linux/s390x** です。
4. カタログイメージに名前を付け、**v1** などのタグを追加します。
5. オプション: 必要な場合は、レジストリー認証情報ファイルの場所を指定します。
6. オプション: ターゲットレジストリーの信頼を設定しない場合は、**--insecure** フラグを追加します。
7. オプション: 公開されていない他のアプリケーションレジストリーカタログが使用されている場合、Quay 認証トークンを指定します。

無効なマニフェストが Red Hat のカタログに誤って導入される可能性があります。これが実際に生じる場合には、以下のようなエラーが表示される可能性があります。

```
...
INFO[0014] directory
```

```
dir=/var/folders/st/9cskxqs53ll3wdn434vw4cd80000gn/T/300666084/manifests-829192605
file=4.2 load=package
W1114 19:42:37.876180   34665 builder.go:141] error building database: error loading
package into db: fuse-camel-k-operator.v7.5.0 specifies replacement that couldn't be found
Uploading ... 244.9kB/s
```

通常、これらのエラーは致命的なエラーではなく、該当する Operator パッケージにインストールする予定の Operator やその依存関係が含まれない場合、それらを無視することができます。

3.2.2.2. ネットワークが制限された環境向けの OperatorHub の設定

クラスター管理者は、カスタム Operator カタログイメージを使用し、OLM および OperatorHub をネットワークが制限された環境でローカルコンテンツを使用するように設定できます。この例では、以前にビルドされ、サポートされているレジストリーにプッシュされたカスタム **redhat-operators** カタログイメージを使用します。

前提条件

- ネットワークアクセスが無制限の Linux ワークステーション
- サポートされているレジストリーにプッシュされるカスタム Operator カタログイメージ
- **oc** version 4.3.5+
- **podman** version 1.4.4+
- Docker v2-2 をサポートするミラーレジストリーへのアクセス
- プライベートレジストリーを使用している場合、後続の手順で使用するために **REG_CREDS** 環境変数をレジストリー認証情報のファイルパスに設定します。たとえば **podman** CLI の場合は、以下のようになります。

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

手順

1. **disableAllDefaultSources: true** を仕様に追加してデフォルトの OperatorSource を無効にします。

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

これにより、OpenShift Container Platform のインストール時にデフォルトで設定されるデフォルトの OperatorSource が無効になります。

2. **oc adm catalog mirror** コマンドは、カスタム Operator カタログイメージのコンテンツを抽出し、ミラーリングに必要なマニフェストを生成します。以下のいずれかを選択できます。
   - コマンドのデフォルト動作で、マニフェストの生成後にすべてのイメージコンテンツをミラーレジストリーに自動的にミラーリングできるようにします。または、
   - **--manifests-only** フラグを追加して、ミラーリングに必要なマニフェストのみを生成しますが、これにより、イメージコンテンツがレジストリーに自動的にミラーリングされる訳ではありません。これは、ミラーリングする内容を確認するのに役立ちます。また、コン

テンツのサブセットのみが必要な場合に、マッピングの一覧に変更を加えることができます。次に、そのファイルを **oc image mirror** コマンドで使用し、後のステップでイメージの変更済みの一覧をミラーリングできます。

ネットワークアクセスが無制限のワークステーションで、以下のコマンドを実行します。

```
$ oc adm catalog mirror \
    <registry_host_name>:<port>/olm/redhat-operators:v1 \ 1
    <registry_host_name>:<port> \
    [-a ${REG_CREDS}] \ 2
    [--insecure] \ 3
    [--filter-by-os="<os>/<arch>"] \ 4
    [--manifests-only] 5
```

1. Operator カタログイメージを指定します。
2. オプション: 必要な場合は、レジストリー認証情報ファイルの場所を指定します。
3. オプション: ターゲットレジストリーの信頼を設定しない場合は、**--insecure** フラグを追加します。
4. オプション: カタログは複数のアーキテクチャーおよびオペレーティングシステムをサポートするイメージを参照する可能性があるため、アーキテクチャーおよびオペレーティングシステムでフィルターして、一 致するイメージのみをミラーリングするようにできます。使用できる値は、**linux/amd64**、**linux/ppc64le**、および **linux/s390x** です。
5. オプション: ミラーリングに必要なマニフェストのみを生成し、実際にはイメージコンテンツをレジストリーにミラーリングしません。

出力例

```
using database path mapping: /:/tmp/190214037
wrote database to /tmp/190214037
using database at: /tmp/190214037/bundles.db 1
...
```

1. コマンドで生成される一時的なデータベース。

コマンドの実行後に、**<image_name>-manifests/** ディレクトリーが現在のディレクトリーに作成され、以下のファイルが生成されます。

- これにより、**imageContentSourcePolicy.yaml** ファイルは ImageContentSourcePolicy オブジェクトを定義します。このオブジェクトは、このオブジェクトは、ノードを Operator マニフェストおよびミラーリングされたレジストリーに保存されるイメージ参照間で変換できるように設定します。
- **mapping.txt** ファイルには、すべてのソースイメージが含まれ、これはそれらのイメージをターゲットレジストリー内のどこにマップするかを示します。このファイルは **oc image mirror** コマンドと互換性があり、ミラーリング設定をさらにカスタマイズするために使用できます。

3. 直前の手順で **--manifests-only** フラグを使用して、コンテンツのサブセットのみをミラーリングする場合は、以下を実行します。

a. **mapping.txt** ファイルのイメージの一覧を仕様に変更します。ミラーリングするイメージのサブセットの名前とバージョンが不明な場合は、以下の手順で確認します。

   i. **oc adm catalog mirror** コマンドで生成された一時的なデータベースに対して **sqlite3** ツールを実行し、一般的な検索クエリーに一致するイメージの一覧を取得します。出力は、後に **mapping.txt** ファイルを編集する方法を通知するのに役立ちます。
   たとえば、**clusterlogging.4.3** の文字列のようなイメージの一覧を取得するには、以下を実行します。

```
$ echo "select * from related_image \
    where operatorbundle_name like 'clusterlogging.4.3%';" \
    | sqlite3 -line /tmp/190214037/bundles.db 1
```

   1 **oc adm catalog mirror** コマンドの直前の出力を参照し、データベースファイルのパスを見つけます。

   出力例

```
image = registry.redhat.io/openshift4/ose-logging-
kibana5@sha256:aa4a8b2a00836d0e28aa6497ad90a3c116f135f382d8211e3c55f34f
b36dfe61
operatorbundle_name = clusterlogging.4.3.33-202008111029.p0

image = registry.redhat.io/openshift4/ose-oauth-
proxy@sha256:6b4db07f6e6c962fc96473d86c44532c93b146bbefe311d0c348117bf75
9c506
operatorbundle_name = clusterlogging.4.3.33-202008111029.p0
...
```

   ii. 直前の手順で取得した結果を使用して **mapping.txt** ファイルを編集し、ミラーリングする必要のあるイメージのサブセットのみを追加します。
   たとえば、前述の出力例の **image** 値を使用して、**mapping.txt** ファイルに以下の一致する行が存在することを確認できます。

   mapping.txt の一致するイメージマッピング。

```
registry.redhat.io/openshift4/ose-logging-
kibana5@sha256:aa4a8b2a00836d0e28aa6497ad90a3c116f135f382d8211e3c55f34f
b36dfe61=<registry_host_name>:<port>/openshift4-ose-logging-kibana5:a767c8f0
registry.redhat.io/openshift4/ose-oauth-
proxy@sha256:6b4db07f6e6c962fc96473d86c44532c93b146bbefe311d0c348117bf75
9c506=<registry_host_name>:<port>/openshift4-ose-oauth-proxy:3754ea2b
```

   この例では、これらのイメージのみをミラーリングする場合に、**mapping.txt** ファイルの他のすべてのエントリーを削除し、上記の 2 行のみを残します。

b. ネットワークアクセスが無制限のワークステーション上で、変更した **mapping.txt** ファイルを使用し、 **oc image mirror** コマンドを使用してイメージをレジストリーにミラーリングします。

```
$ oc image mirror \
    [-a ${REG_CREDS}] \
    -f ./redhat-operators-manifests/mapping.txt
```

4. ImageContentSourcePolicy を適用します。

```
$ oc apply -f ./redhat-operators-manifests/imageContentSourcePolicy.yaml
```

5. カタログイメージを参照する CatalogSource オブジェクトを作成します。

   a. 仕様を以下のように変更し、これを **catalogsource.yaml** ファイルとして保存します。

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: my-operator-catalog
  namespace: openshift-marketplace
spec:
  sourceType: grpc
  image: <registry_host_name>:<port>/olm/redhat-operators:v1 (1)
  displayName: My Operator Catalog
  publisher: grpc
```

   (1) Operator カタログイメージを指定します。

   b. このファイルを使用して CatalogSource オブジェクトを作成します。

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

6. 以下のリソースが正常に作成されていることを確認します。

   a. Pod を確認します。

```
$ oc get pods -n openshift-marketplace
```

   出力例

```
NAME                                READY   STATUS    RESTARTS  AGE
my-operator-catalog-6njx6           1/1     Running   0         28s
marketplace-operator-d9f549946-96sgr  1/1     Running   0         26h
```

   b. CatalogSource を確認します。

```
$ oc get catalogsource -n openshift-marketplace
```

   出力例

```
NAME                  DISPLAY               TYPE PUBLISHER  AGE
my-operator-catalog   My Operator Catalog   grpc           5s
```

   c. PackageManifest を確認します。

```
$ oc get packagemanifest -n openshift-marketplace
```

   出力例

```
NAME    CATALOG             AGE
etcd    My Operator Catalog  34s
```

ネットワークが制限された環境の OpenShift Container Platform クラスター Web コンソールで、**OperatorHub** ページから Operator をインストールできます。

3.2.2.3. 制限された環境での Cluster Application Migration Operator の OpenShift Container Platform 4.4 ターゲットクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4.4 ターゲットクラスターに Cluster Application Migration Operator をインストールできます。

Cluster Application Migration Operator は、デフォルトで Cluster Application Migration ツールをターゲットクラスターにインストールします。

前提条件

- カスタム Operator カタログを作成し、これをミラーレジストリーにプッシュしていること。
- ミラーレジストリーから Cluster Application Migration Operator をインストールするように OLM を設定していること。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
   Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **Create** をクリックします。
8. **Workloads → Pods** をクリックし、Controller Manager、Migration UI、Restic、および Velero Pod が実行中であることを確認します。

3.2.2.4. 制限された環境での Cluster Application Migration Operator の OpenShift Container Platform 4.2 ソースクラスターへのインストール

Operator Lifecycle Manager (OLM) を使用して OpenShift Container Platform 4 ソースクラスターに Cluster Application Migration Operator をインストールできます。

前提条件

- カスタム Operator カタログを作成し、これをミラーレジストリーにプッシュしていること。

- ミラーレジストリーから Cluster Application Migration Operator をインストールするように OLM を設定していること。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールド (この場合は **Migration**) を使用して **Cluster Application Migration Operator** を見つけます。
3. **Cluster Application Migration Operator** を選択し、**Install** をクリックします。
4. **Create Operator Subscription** ページで、**Subscribe** をクリックします。
   Installed Operators ページで、**Cluster Application Migration Operator** は、**Succeeded** のステータスで **openshift-migration** プロジェクトに表示されます。
5. **Cluster Application Migration Operator** をクリックします。
6. **Provided APIs** の下で **Migration Controller** タイルを見つけ、**Create Instance** をクリックします。
7. **Create** をクリックします。

3.2.3. CAM Web コンソールの起動

ブラウザーで CAM Web コンソールを起動できます。

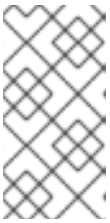
手順

1. CAM ツールがインストールされている OpenShift Container Platform クラスターにログインします。
2. 以下のコマンドを入力して CAM Web コンソール URL を取得します。

```
$ oc get -n openshift-migration route/migration -o go-template='https://{{ .spec.host }}'
```

   出力は **https://migration-openshift-migration.apps.cluster.openshift.com** のようになります。

3. ブラウザーを起動し、CAM Web コンソールに移動します。



注記

Cluster Application Migration Operator のインストール後すぐに CAM Web コンソールにアクセスしようとする場合、Operator は依然としてクラスターを設定しているため、コンソールが読み込まれないことがあります。数分待機した後に再試行します。

4. 自己署名 CA 証明書を使用している場合、ソースクラスターの API サーバーの CA 証明書を受け入れることを求めるプロンプトが出されます。Web ページは、残りの証明書を受け入れるプロセスについて説明します。
5. OpenShift Container Platform の **ユーザー名** および **パスワード** を使用してログインします。

3.2.4. Cluster Application Migration ツールのアップグレード

ソースおよびターゲットクラスターで Cluster Application Migration (CAM) ツールをアップグレードできます。

CAM 1.1 から 1.2 にアップグレードする場合、CAM Web コンソールでサービスアカウントトークンを更新する必要があります。

3.2.4.1. OpenShift Container Platform 4 クラスターでの CAM ツールのアップグレード

Operator Lifecycle Manager を使用して OpenShift Container Platform 4 クラスターで CAM ツールをアップグレードできます。

Cluster Application Migration Operator にサブスクライブする際に **自動** 承認オプションを選択した場合、CAM ツールは自動的に更新されます。

以下の手順では、**手動** 承認オプションを **自動** に変更するか、またはリリースチャネルを変更できます。

手順

1. OpenShift Container Platform コンソールで、**Operators** > **Installed Operators** に移動します。
2. **Cluster Application Migration Operator** をクリックします。
3. **Subscription** タブで、**Approval** オプションを **Automatic** に変更します。
4. オプション: **チャネル** を編集します。
   サブスクリプションを更新すると、更新された Cluster Application Migration Operator がデプロイされ、CAM ツールコンポーネントが更新されます。

3.2.4.2. サービスアカウントトークンの更新

CAM 1.1 から 1.2 にアップグレードする場合、CAM Web コンソールでサービスアカウントトークンを更新する必要があります。

CAM 1.1 は **mig** サービスアカウントを使用し、CAM 1.2 は **migration-controller** サービスアカウントを使用します。

手順

1. クラスターにログインし、**migration-controller** サービスアカウントトークンを取得します。

```
$ oc sa get-token -n openshift-migration migration-controller
```

2. CAM Web コンソールにログインし、**Clusters** をクリックします。
3. クラスターの Options メニュー  をクリックし、**Edit** を選択します。
4. 新たなトークンを **Service account token** フィールドにコピーします。
5. **Update cluster** をクリックしてから、**Close** をクリックします。
   クラスターのサービスアカウントトークンが更新されます。

3.3. レプリケーションリポジトリーの設定

オブジェクトストレージをレプリケーションリポジトリーとして使用するように設定する必要があります。Cluster Application Migration ツールは、データをソースクラスターからレプリケーションリポジトリーにコピーしてから、レプリケーションリポジトリーからターゲットクラスターにコピーします。

CAM ツールは、ソースクラスターからターゲットクラスターにデータを移行するために ファイルシステムおよびスナップショットによるデータのコピー方法 をサポートします。ご使用の環境に適した方法で、ストレージプロバイダーでサポートされる方法を選択できます。

以下のストレージプロバイダーがサポートされています。

- Multi-Cloud Object Gateway (MCG)
- Amazon Web Services (AWS) S3
- Google Cloud Provider (GCP)
- Microsoft Azure
- 汎用 S3 オブジェクトストレージ (例: Minio または Ceph S3)

ソースおよびターゲットクラスターには、移行時にレプリケーションリポジトリーへのネットワークアクセスがなければなりません。

制限された環境では、内部でホストされるレプリケーションリポジトリーを作成できます。プロキシーサーバーを使用する場合は、レプリケーションリポジトリーがホワイトリスト化されていることを確認する必要があります。

3.3.1. Multi-Cloud Object Gateway ストレージバケットをレプリケーションリポジトリーとして設定する

OpenShift Container Storage Operator をインストールし、Multi-Cloud Object Gateway (MCG) ストレージバケットをレプリケーションリポジトリーとして設定できます。

3.3.1.1. OpenShift Container Storage Operator のインストール

OpenShift Container Storage Operator は、OperatorHub からインストールできます。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** (この場合は、**OCS**) を使用し、 **OpenShift Container Storage Operator** を見つけます。
3. **OpenShift Container Storage Operator** を選択し、**Install** をクリックします。
4. **Update Channel**、**Installation Mode**、および **Approval Strategy** を選択します。
5. **Subscribe** をクリックします。
   Installed Operators ページで、**OpenShift Container Storage Operator** は、**Succeeded** のステータスと共に **openshift-storage** プロジェクトに表示されます。

3.3.1.2. Multi-Cloud Object Gateway ストレージバケットの作成

Multi-Cloud Object Gateway (MCG) ストレージバケットのカスタムリソース (CR) を作成できます。

手順

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

```
$ oc login
```

2. **NooBaa** CR 設定ファイル **noobaa.yml** を以下の内容で作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: NooBaa
metadata:
  name: noobaa
  namespace: openshift-storage
spec:
 dbResources:
   requests:
     cpu: 0.5 (1)
     memory: 1Gi
 coreResources:
   requests:
     cpu: 0.5 (2)
     memory: 1Gi
```

(1) (2) 非常に小規模なクラスターの場合、**cpu** の値を **0.1** に変更できます。

3. **NooBaa** オブジェクトを作成します。

```
$ oc create -f noobaa.yml
```

4. 以下の内容で、**BackingStore** CR 設定ファイル **bs.yml** を作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: BackingStore
metadata:
  finalizers:
  - noobaa.io/finalizer
  labels:
    app: noobaa
  name: mcg-pv-pool-bs
  namespace: openshift-storage
spec:
  pvPool:
    numVolumes: 3 (1)
    resources:
      requests:
        storage: 50Gi (2)
    storageClass: gp2 (3)
  type: pv-pool
```

1. PV プール内のボリューム数を指定します。
2. ボリュームのサイズを指定します。
3. ストレージクラスを作成します。

5. **BackingStore** オブジェクトを作成します。

```
$ oc create -f bs.yml
```

6. 以下の内容で **BucketClass** CR 設定ファイル **bc.yml** を作成します。

```
apiVersion: noobaa.io/v1alpha1
kind: BucketClass
metadata:
  labels:
    app: noobaa
  name: mcg-pv-pool-bc
  namespace: openshift-storage
spec:
  placementPolicy:
    tiers:
    - backingStores:
      - mcg-pv-pool-bs
      placement: Spread
```

7. **BucketClass** オブジェクトを作成します。

```
$ oc create -f bc.yml
```

8. 以下の内容で **ObjectBucketClaim** CR 設定ファイル **obc.yml** を作成します。

```
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: migstorage
  namespace: openshift-storage
spec:
  bucketName: migstorage 1
  storageClassName: openshift-storage.noobaa.io
  additionalConfig:
    bucketclass: mcg-pv-pool-bc
```

1. レプリケーションリポジトリーを CAM Web コンソールに追加するために使用するバケット名を記録します。

9. **ObjectBucketClaim** オブジェクトを作成します。

```
$ oc create -f obc.yml
```

10. リソース作成プロセスを監視し、**ObjectBucketClaim** ステータスが **Bound** であることを確認します。

-

```
$ watch -n 30 'oc get -n openshift-storage objectbucketclaim migstorage -o yaml'
```

このプロセスには、5 分から 10 分の時間がかかる場合があります。

11. 以下の値を取得して記録します。この値は、レプリケーションリポジトリーを CAM Web コンソールに追加する際に必要になります。

- S3 エンドポイント:

```
$ oc get route -n openshift-storage s3
```

- S3 プロバイダーアクセスキー:

```
$ oc get secret -n openshift-storage migstorage -o go-template='{{
.data.AWS_ACCESS_KEY_ID }}' | base64 -d
```

- S3 プロバイダーシークレットアクセスキー:

```
$ oc get secret -n openshift-storage migstorage -o go-template='{{
.data.AWS_SECRET_ACCESS_KEY }}' | base64 -d
```

3.3.2. AWS S3 ストレージバケットをレプリケーションリポジトリーとして設定する

AWS S3 ストレージバケットをレプリケーションリポジトリーとして設定できます。

前提条件

- AWS S3 ストレージバケットは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。
- AWS CLI がインストールされていること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - EC2 Elastic Block Storage (EBS) にアクセスできる必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. AWS S3 バケットを作成します。

```
$ aws s3api create-bucket \
    --bucket <bucket_name> \ 1
    --region <bucket_region> 2
```

1. S3 バケット名を指定します。
2. S3 バケットリージョンを指定します (例: **us-east-1**)。

2. IAM ユーザー **velero** を作成します。

```
$ aws iam create-user --user-name velero
```

3. EC2 EBS スナップショットポリシーを作成します。

```
$ cat > velero-ec2-snapshot-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes",
                "ec2:DescribeSnapshots",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:CreateSnapshot",
                "ec2:DeleteSnapshot"
            ],
            "Resource": "*"
        }
    ]
}
EOF
```

4. 1 つまたはすべての S3 バケットの AWS S3 アクセスポリシーを作成します。

```
$ cat > velero-s3-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket_name>/*" (1)
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket_name>" (2)
            ]
```

```
        }
    ]
}
EOF
```

1 2 単一の S3 バケットへのアクセスを付与するには、バケット名を指定します。すべての AWS S3 バケットへのアクセスを付与するには、バケット名の代わりに * を指定します。

```
"Resource": [
    "arn:aws:s3:::*"
```

5. EC2 EBS ポリシーを **velero** に割り当てます。

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero-ebs \
  --policy-document file://velero-ec2-snapshot-policy.json
```

6. AWS S3 ポリシーを **velero** に割り当てます。

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero-s3 \
  --policy-document file://velero-s3-policy.json
```

7. **velero** のアクセスキーを作成します。

```
$ aws iam create-access-key --user-name velero
{
  "AccessKey": {
        "UserName": "velero",
        "Status": "Active",
        "CreateDate": "2017-07-31T22:24:41.576Z",
        "SecretAccessKey": <AWS_SECRET_ACCESS_KEY>, 1
        "AccessKeyId": <AWS_ACCESS_KEY_ID> 2
    }
}
```

1 2 AWS リポジトリーを CAM Web コンソールに追加するために **AWS_SECRET_ACCESS_KEY** および **AWS_ACCESS_KEY_ID** を記録します。

3.3.3. Google Cloud Provider ストレージバケットをレプリケーションリポジトリーとして設定する

Google Cloud Provider (GCP) ストレージバケットをレプリケーションリポジトリーとして設定できます。

前提条件

- GCP ストレージバケットは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。

- gsutil がインストールされていること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. **gsutil init** を実行してログインします。

```
$ gsutil init
Welcome! This command will take you through the configuration of gcloud.

Your current configuration has been set to: [default]

To continue, you must login. Would you like to login (Y/n)?
```

2. **BUCKET** 変数を設定します。

```
$ BUCKET=<bucket_name> 1
```

1 バケット名を指定します。

3. ストレージバケットを作成します。

```
$ gsutil mb gs://$BUCKET/
```

4. **PROJECT_ID** 変数をアクティブなプロジェクトに設定します。

```
$ PROJECT_ID=$(gcloud config get-value project)
```

5. **velero** サービスアカウントを作成します。

```
$ gcloud iam service-accounts create velero \
    --display-name "Velero Storage"
```

6. **SERVICE_ACCOUNT_EMAIL** 変数をサービスアカウントのメールアドレスに設定します。

```
$ SERVICE_ACCOUNT_EMAIL=$(gcloud iam service-accounts list \
  --filter="displayName:Velero Storage" \
  --format 'value(email)')
```

7. パーミッションをサービスアカウントに付与します。

```
$ ROLE_PERMISSIONS=(
    compute.disks.get
    compute.disks.create
    compute.disks.createSnapshot
    compute.snapshots.get
```

```
    compute.snapshots.create
    compute.snapshots.useReadOnly
    compute.snapshots.delete
    compute.zones.get
)

gcloud iam roles create velero.server \
    --project $PROJECT_ID \
    --title "Velero Server" \
    --permissions "$(IFS=","; echo "${ROLE_PERMISSIONS[*]}")"

gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member serviceAccount:$SERVICE_ACCOUNT_EMAIL \
    --role projects/$PROJECT_ID/roles/velero.server

gsutil iam ch serviceAccount:$SERVICE_ACCOUNT_EMAIL:objectAdmin gs://${BUCKET}
```

8. サービスアカウントのキーを現在のディレクトリーにある **credentials-velero** ファイルに保存します。

```
$ gcloud iam service-accounts keys create credentials-velero \
  --iam-account $SERVICE_ACCOUNT_EMAIL
```

3.3.4. Microsoft Azure Blob ストレージコンテナーをレプリケーションリポジトリーとして設定

Microsoft Azure Blob ストレージコンテナーをレプリケーションリポジトリーとして設定できます。

前提条件

- Azure ストレージアカウント があること。
- Azure CLI がインストールされていること。
- Azure Blob ストレージコンテナーがソースクラスターおよびターゲットクラスターからアクセスできること。
- スナップショットのコピー方法を使用する場合は、以下の条件を満たす必要があります。
  - ソースおよびターゲットクラスターが同じリージョンにある必要があります。
  - ソースおよびターゲットクラスターには、同じストレージクラスがある必要があります。
  - ストレージクラスはスナップショットと互換性がある必要があります。

手順

1. **AZURE_RESOURCE_GROUP** 変数を設定します。

```
$ AZURE_RESOURCE_GROUP=Velero_Backups
```

2. Azure リソースグループを作成します。

```
$ az group create -n $AZURE_RESOURCE_GROUP --location <CentralUS> 1
```

1 場所を指定します。

3. **AZURE_STORAGE_ACCOUNT_ID** 変数を設定します。

```
$ AZURE_STORAGE_ACCOUNT_ID=velerobackups
```

4. Azure ストレージアカウントを作成します。

```
$ az storage account create \
  --name $AZURE_STORAGE_ACCOUNT_ID \
  --resource-group $AZURE_RESOURCE_GROUP \
  --sku Standard_GRS \
  --encryption-services blob \
  --https-only true \
  --kind BlobStorage \
  --access-tier Hot
```

5. **BLOB_CONTAINER** 変数を設定します。

```
$ BLOB_CONTAINER=velero
```

6. Azure Blob ストレージコンテナーを作成します。

```
$ az storage container create \
  -n $BLOB_CONTAINER \
  --public-access off \
  --account-name $AZURE_STORAGE_ACCOUNT_ID
```

7. **velero** のサービスプリンシパルおよび認証情報を作成します。

```
$ AZURE_SUBSCRIPTION_ID=`az account list --query '[?isDefault].id' -o tsv`
$ AZURE_TENANT_ID=`az account list --query '[?isDefault].tenantId' -o tsv`
$ AZURE_CLIENT_SECRET=`az ad sp create-for-rbac --name "velero" --role "Contributor" --
query 'password' -o tsv`
$ AZURE_CLIENT_ID=`az ad sp list --display-name "velero" --query '[0].appId' -o tsv`
```

8. サービスプリンシパルの認証情報を **credentials-velero** ファイルに保存します。

```
$ cat << EOF  > ./credentials-velero
AZURE_SUBSCRIPTION_ID=${AZURE_SUBSCRIPTION_ID}
AZURE_TENANT_ID=${AZURE_TENANT_ID}
AZURE_CLIENT_ID=${AZURE_CLIENT_ID}
AZURE_CLIENT_SECRET=${AZURE_CLIENT_SECRET}
AZURE_RESOURCE_GROUP=${AZURE_RESOURCE_GROUP}
AZURE_CLOUD_NAME=AzurePublicCloud
EOF
```

3.4. CAM WEB コンソールを使用したアプリケーションの移行

アプリケーションワークロードの移行は、クラスターおよびレプリケーションリポジトリーを CAM Web コンソールに追加することによって実行できます。次に、移行計画を作成し、これを実行できます。

クラスターまたはレプリケーションリポジトリーのセキュリティーを自己署名証明書で保護する場合、CA 証明書バンドルファイルを作成するか、または SSL 検証を無効にできます。

3.4.1. CA 証明書バンドルファイルの作成

自己署名証明書を使用してクラスターまたはレプリケーションリポジトリーのセキュリティーを保護する場合、証明書の検証は **Certificate signed by unknown authority** というエラーメッセージを出して失敗する可能性があります。

カスタム CA 証明書バンドルファイルを作成し、クラスターまたはレプリケーションリポジトリーの追加時に CAM Web コンソールでこれをアップロードできます。

手順

リモートエンドポイントから CA 証明書をダウンロードし、これを CA バンドルファイルとして保存します。

```
$ echo -n | openssl s_client -connect <host_FQDN>:<port> \ 1
  | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > <ca_bundle.cert> 2
```

1 エンドポイントのホスト FQDN およびポートを指定します (例: **api.my-cluster.example.com:6443**)。

2 CA バンドルファイルの名前を指定します。

3.4.2. CAM Web コンソールへのクラスターの追加

クラスターを CAM Web コンソールに追加できます。

前提条件

Azure スナップショットを使用してデータをコピーする場合:

- ソースクラスターの追加時に Azure リソースグループ名を指定する必要があります。
- ソースおよびターゲットクラスターは同じ Azure リソースグループにあり、同じ場所にある必要があります。

手順

1. クラスターにログインします。
2. サービスアカウントトークンを取得します。

```
$ oc sa get-token migration-controller -n openshift-migration
eyJhbGciOiJSUzI1NiIsImtpZCI6IiJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwi
a3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJtaWciLCJrdWJlcm5ldGVz
LmlvL3NlcnZpY2VhY2NvdW50L3NlY3JldC5uYW1lIjoibWlnLXRva2VuLWs4dDJyIiwia3ViZXJuZ
XRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6Im1pZyIsImt1YmV
ybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50LnVpZCI6ImE1YjFiYWM
wLWMxYmYtMTFlOS05Y2NiLTAyOWRmODYwYjMwOCIsInN1YiI6InN5c3RlbTpzZXJ2aWNlY
WNjb3VudDptaWc6bWlnIn0.xqeeAINK7UXpdRqAtOj70qhBJPeMwmgLomV9iFxr5RoqUgKchZ
RG2J2rkqmPm6vr7K-
cm7ibD1IBpdQJCcVDuoHYsFgV4mp9vgOfn9osSDp2TGikwNz4Az95e81xnjVUmzh-
NjDsEpw71DH92iHV_xt2sTwtzftS49LpPW2LjrV0evtNBP_t_RfskdArt5VSv25eORl7zScqfe1CiM
```

```
kcVbf2UqACQjo3LbkpfN26HAioO2oH0ECPiRzT0Xyh-KwFutJLS9Xgghyw-
LD9kPKcE_xbbJ9Y4Rqajh7WdPYuB0Jd9DPVrslmzK-F6cgHHYoZEv0SvLQi-
PO0rpDrcjOEQQ
```

3. CAM Web コンソールにログインします。
4. **Clusters** セクションで、**Add cluster** をクリックします。
5. 以下のフィールドに値を入力します。
   - **Cluster name**: 小文字 (**a-z**) および数字 (**0-9**) を含めることができます。空白文字や国際文字を含めることはできません。
   - **Url**: クラスターの API サーバーの URL です (例: **https://<master1.example.com>:8443**)。
   - **Service account token**: ソースクラスターから取得される文字列。
   - **Azure cluster**: オプション。Azure スナップショットを使用してデータをコピーする場合はこれを選択します。
   - **Azure resource group**: このフィールドは、**Azure cluster** にチェックを付けると表示されます。
   - カスタム CA バンドルを使用する場合は、**Browse** をクリックし、CA バンドルファイルを参照します。
6. **Add cluster** をクリックします。
   クラスターが **Clusters** セクションに表示されます。

3.4.3. CAM Web コンソールへのレプリケーションリポジトリーの追加

CAM Web コンソールには、オブジェクトストレージバケットをレプリケーションリポジトリーとして追加できます。

前提条件

- データを移行するには、オブジェクトストレージバケットを設定する必要があります。

手順

1. CAM Web コンソールにログインします。
2. **Replication repositories** セクションで、**Add repository** をクリックします。
3. **Storage provider type** を選択し、以下のフィールドに入力します。
   - **AWS** (AWS S3、MCG、および汎用 S3 プロバイダーの場合):
     - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
     - **S3 bucket name**: 作成した S3 バケットの名前を指定します。
     - **S3 bucket region**: S3 バケットリージョンを指定します。AWS S3 の場合に**必須**です。**Optional** (他の S3 プロバイダーの場合)。

- **S3 endpoint**: バケットではなく S3 サービスの URL を指定します (例: **https://<s3-storage.apps.cluster.com>**)。汎用 S3 プロバイダーの場合は**必須**です。**https://** 接頭辞を使用する必要があります。
- **S3 provider access key**: AWS には **<AWS_SECRET_ACCESS_KEY>** を指定するか、または MCG には S3 プロバイダーアクセスキーを指定します。
- **S3 provider secret access key**: AWS には **<AWS_ACCESS_KEY_ID>** を指定するか、または MCG には S3 プロバイダーシークレットアクセスキーを指定します。
- **Require SSL verification**: 汎用 S3 プロバイダーを使用している場合は、このチェックボックスをクリアします。
- カスタム CA バンドルを使用する場合は、**Browse** をクリックし、Base64 でエンコードされた CA バンドルファイルを参照します。

- **GCP**:
  - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
  - **GCP bucket name**: GCP バケットの名前を指定します。
  - **GCP credential JSON blob**: **credentials-velero** ファイルに文字列を指定します。
- **Azure**:
  - **Replication repository name**: CAM Web コンソールでレプリケーションリポジトリー名を指定します。
  - **Azure resource group**: Azure Blob ストレージのリソースグループを指定します。
  - **Azure storage account name**: Azure Blob ストレージアカウント名を指定します。
  - **Azure credentials - INI file contents** **credentials-velero** ファイルに文字列を指定します。

4. **Add repository** をクリックし、接続の検証を待機します。
5. **Close** をクリックします。
   新規リポジトリーが **Replication repositories** セクションに表示されます。

3.4.4. 大規模な移行の場合の移行計画の制限の変更

大規模な移行の移行計画の制限を変更することができます。

重要

移行の失敗を防ぐために、まず変更についてのテストをご使用の環境で実行する必要があります。

単一の移行計画には以下のデフォルト制限があります。

- 10 namespace
  この制限を超えると、CAM Web コンソールは **Namespace limit exceeded**エラーを表示し、移行計画を作成することができません。

- 100 Pod
  Pod の制限を超える場合、CAM Web コンソールには、以下の例と同様の警告メッセージが表示されます。**Plan has been validated with warning condition(s).See warning message.Pod limit: 100 exceeded, found: 104**

- 100 永続ボリューム
  永続ボリュームの制限を超過すると、CAM Web コンソールには同様の警告メッセージが表示されます。

手順

1. Migration コントローラー CR を編集します。

```
$ oc get migrationcontroller -n openshift-migration
NAME AGE
migration-controller 5d19h

$ oc edit migrationcontroller -n openshift-migration
```

2. 以下のパラメーターを更新します。

```
...
migration_controller: true

# This configuration is loaded into mig-controller, and should be set on the
# cluster where `migration_controller: true`
mig_pv_limit: 100
mig_pod_limit: 100
mig_namespace_limit: 10
...
```

3.4.5. CAM Web コンソールでの移行計画の作成

CAM Web コンソールで移行計画を作成できます。

前提条件

- CAM Web コンソールには以下が含まれている必要があります。
  - ソースクラスター
  - CAM ツールのインストール時に自動的に追加されるターゲットクラスター
  - レプリケーションリポジトリー
- ソースおよびターゲットクラスターには、相互に対するネットワークアクセスやレプリケーションリポジトリーへのネットワークアクセスがなければなりません。
- スナップショットを使用してデータをコピーする場合、ソースおよびターゲットクラスターは、同じクラウドプロバイダー (AWS、GCP、または Azure) および同じリージョンで実行される必要があります。

手順

1. CAM Web コンソールにログインします。

2. **Plans** セクションで、**Add plan** をクリックします。

3. **Plan name** を入力し、**Next** をクリックします。
Plan name には、最大 253 文字の小文字の英数字 (**a-z、0-9**) を含めることができます。スペースまたはアンダースコア (_) を含めることはできません。

4. **Source cluster** を選択します。

5. **Target cluster**を選択します。

6. **Replication repository** を選択します。

7. 移行するプロジェクトを選択し、**Next** をクリックします。

8. PV の **Copy** または **Move** を選択します。

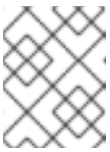
   - **Copy** は、ソースクラスターの PV のデータをレプリケーションリポジトリーにコピーしてから、これを同様の特徴のある新規に作成された PV で復元します。
   オプション: **Verify copy** を選択して、ファイルシステムメソッドでコピーされたデータを検証できます。このオプションを使用すると、各ソースファイルのチェックサムが生成され、復元後のチェックが実行されるため、パフォーマンスが大幅に低下します。

   - **Move** は、ソースクラスターからリモートボリューム (例: NFS) をアンマウントし、リモートボリュームをポイントするターゲットクラスターで PV リソースを作成し、その後にリモートボリュームをターゲットクラスターにマウントします。ターゲットクラスターで実行されているアプリケーションは、ソースクラスターが使用していたものと同じリモートボリュームを使用します。リモートボリュームは、ソースクラスターおよびターゲットクラスターからアクセスできる必要があります。

9. **Next** をクリックします。

10. PV の **Copy method** を選択します。

    - **Snapshot** は、クラウドプロバイダーのスナップショット機能を使用してディスクのバックアップおよび復元を行います。この場合、**ファイルシステム**を使用する場合よりもはるかに高速になります。

      注記

      ストレージおよびクラスターは同じリージョンにあり、ストレージクラスには互換性がなければなりません。

    - **ファイルシステム**は、データファイルをソースディスクから新規に作成されたターゲットディスクにコピーします。

11. PV の **Storage class** を選択します。
Filesystem のコピー方法を選択した場合、移行時にストレージクラスを変更できます。たとえば、Red Hat Gluster Storage または NFS ストレージから Red Hat Ceph Storage に変更できます。

12. **Next** をクリックします。

13. 移行フックを追加する必要がある場合は、**Add Hook** をクリックして、以下の手順を実行します。

    a. フックの名前を指定します。

b. **Ansible playbook** を選択して独自の Playbook を使用するか、または別の言語で作成されがフックに **Custom container image** を選択します。

c. **Browse** をクリックして Playbook をアップロードします。

d. オプション: デフォルトの Ansible ランタイムイメージを使用していない場合は、カスタムの Ansible イメージを指定します。

e. フックを実行する必要のあるクラスターを指定します。

f. サービスアカウント名を指定します。

g. namespace を指定します。

h. フックを実行する必要のある移行手順を選択します。

- PreBackup: バックアップタスクがソースクラスターで開始される前
- PostBackup: バックアップタスクがソースクラスターで完了した後
- PreRestore: 復元タスクがターゲットクラスターで開始される前
- PostRestore: 復元タスクがターゲットクラスターで完了した後

14. **Add** をクリックします。
移行計画に最大 4 つのフックを追加し、それぞれのフックを異なる移行手順に割り当てることができます。

15. **Finish** をクリックします。

16. **Close** をクリックします。
移行計画は **Plans** セクションに表示されます。

3.4.6. CAM Web コンソールでの移行計画の実行

CAM Web コンソールで作成した移行計画を使用してアプリケーションとデータをステージングしたり、移行したりできます。

前提条件

CAM Web コンソールには以下が含まれている必要があります。

- ソースクラスター
- CAM ツールのインストール時に自動的に追加されるターゲットクラスター
- レプリケーションリポジトリー
- 有効な移行計画

手順

1. ターゲットクラスターの CAM Web コンソールにログインします。
2. 移行計画を選択します。

3. **Stage** をクリックし、アプリケーションを停止せずにソースクラスターからターゲットクラスターにデータをコピーします。
   実際の移行時間を短縮するには、**Stage** を複数回実行することができます。

4. アプリケーションのワークロードを移行する準備ができたら、**Migrate** をクリックします。
   Migrate は、ソースクラスターでアプリケーションワークロードを停止し、ターゲットクラスターでそのリソースを再作成します。

5. オプション: **Migrate** ウィンドウで **Do not stop applications on the source cluster during migration** を選択できます。

6. **Migrate** をクリックします。

7. オプション: 進行中の移行を停止するには、Options メニュー  をクリックし、**Cancel** を選択します。

8. 移行が完了したら、アプリケーションが OpenShift Container Platform Web コンソールで正常に移行されていることを確認します。

   a. **Home → Projects** をクリックします。

   b. 移行されたプロジェクトをクリックしてそのステータスを表示します。

   c. **Routes** セクションで **Location** をクリックし、アプリケーションが機能していることを確認します (該当する場合)。

   d. **Workloads → Pods** をクリックし、Pod が移行した namespace で実行されていることを確認します。

   e. **Storage → Persistent volumes** をクリックして、移行した永続ボリュームが正常にプロビジョニングされていることを確認します。

3.5. トラブルシューティング

移行用のカスタムリソース (CR) を表示し、ログをダウンロードして失敗した移行の失敗についてのトラブルシューティングを行うことができます。

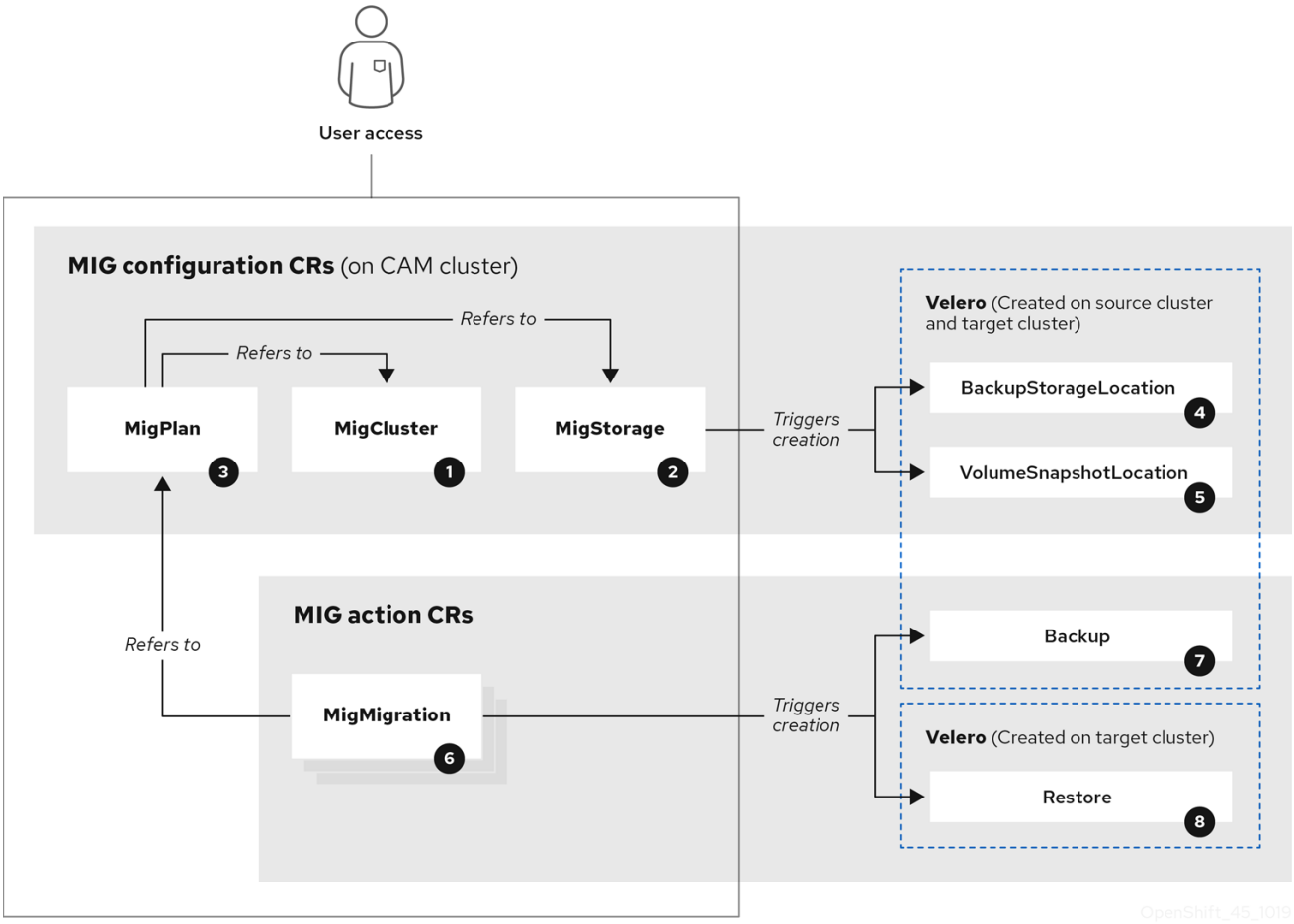
移行の失敗時にアプリケーションが停止した場合は、データ破損を防ぐために手作業でアプリケーションをロールバックする必要があります。

注記

移行時にアプリケーションが停止しなかった場合には、手動のロールバックは必要ありません。元のアプリケーションがソースクラスター上で依然として実行されているためです。

3.5.1. 移行カスタムリソースの表示

Cluster Application Migration (CAM) ツールは以下の カスタムリソース (Custom Resource、CR) を作成します。

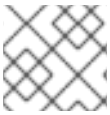


1 MigCluster (設定、CAM クラスター): クラスター定義

2 MigStorage (設定、CAM クラスター): ストレージ定義

3 MigPlan (設定、CAM クラスター): 移行計画

MigPlan CR は移行されるソースおよびターゲットクラスター、リポジトリー、および namespace を記述します。これは 0、1 または多数の MigMigration CR に関連付けられます。



注記

MigPlan CR を削除すると、関連付けられた MigMigration CR が削除されます。

4 BackupStorageLocation (設定、CAM クラスター): Velero バックアップオブジェクトの場所

5 VolumeSnapshotLocation (設定、CAM クラスター): Velero ボリュームスナップショットの場所

6 MigMigration (アクション、CAM クラスター): 移行、移行時に作成される

MigMigration CR は、データのステージングまたは移行を実行するたびに作成されます。各 MigMigration CR は MigPlan CR に関連付けられます。

7 Backup (アクション、ソースクラスター): 移行計画の実行時に、MigMigration CR は各ソースクラスターに 2 つの Velero バックアップ CR を作成します。

- Kubernetes オブジェクトのバックアップ CR #1
- PV データのバックアップ CR #2

8 Restore (アクション、ターゲットクラスター): 移行計画の実行時に、MigMigration CR はターゲットクラスターに 2 つの Velero リスト CR を作成します。

- PV データのリストア CR #1 (バックアップ CR #2 の使用)
- Kubernetes オブジェクトの復元 CR #2 (バックアップ CR #1 の使用)

手順

1. CR 名を取得します。

```
$ oc get <migration_cr> -n openshift-migration 1
```

1 移行 CR を指定します (例: **migmigration**)。

出力は以下の例のようになります。

```
NAME                                   AGE
88435fe0-c9f8-11e9-85e6-5d593ce65e10   6m42s
```

2. CR を表示します。

```
$ oc describe <migration_cr> <88435fe0-c9f8-11e9-85e6-5d593ce65e10> -n openshift-migration
```

出力は以下の例のようになります。

MigMigration の例

```
name:         88435fe0-c9f8-11e9-85e6-5d593ce65e10
namespace:    openshift-migration
labels:       <none>
annotations:  touch: 3b48b543-b53e-4e44-9d34-33563f0f8147
apiVersion:  migration.openshift.io/v1alpha1
kind:         MigMigration
metadata:
  creationTimestamp:  2019-08-29T01:01:29Z
  generation:         20
  resourceVersion:    88179
  selfLink:           /apis/migration.openshift.io/v1alpha1/namespaces/openshift-migration/migmigrations/88435fe0-c9f8-11e9-85e6-5d593ce65e10
  uid:                8886de4c-c9f8-11e9-95ad-0205fe66cbb6
spec:
  migPlanRef:
    name:         socks-shop-mig-plan
    namespace:    openshift-migration
```

```
  quiescePods:  true
  stage:        false
status:
  conditions:
    category:            Advisory
    durable:             True
    lastTransitionTime:  2019-08-29T01:03:40Z
    message:             The migration has completed successfully.
    reason:              Completed
    status:              True
    type:                Succeeded
  phase:                 Completed
  startTimestamp:        2019-08-29T01:01:29Z
events:                  <none>
```

Velero バックアップ CR #2 の例 (PV データ)

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.105.179:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-44dd3bd5-c9f8-11e9-95ad-
0205fe66cbb6
  creationTimestamp: "2019-08-29T01:03:15Z"
  generateName: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-
  generation: 1
  labels:
    app.kubernetes.io/part-of: migration
    migmigration: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
    migration-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
    velero.io/storage-location: myrepo-vpzq9
  name: 88435fe0-c9f8-11e9-85e6-5d593ce65e10-59gb7
  namespace: openshift-migration
  resourceVersion: "87313"
  selfLink: /apis/velero.io/v1/namespaces/openshift-migration/backups/88435fe0-c9f8-11e9-85e6-
5d593ce65e10-59gb7
  uid: c80dbbc0-c9f8-11e9-95ad-0205fe66cbb6
spec:
  excludedNamespaces: []
  excludedResources: []
  hooks:
    resources: []
  includeClusterResources: null
  includedNamespaces:
  - sock-shop
  includedResources:
  - persistentvolumes
  - persistentvolumeclaims
  - namespaces
  - imagestreams
  - imagestreamtags
  - secrets
  - configmaps
```

```
  - pods
  labelSelector:
    matchLabels:
      migration-included-stage-backup: 8886de4c-c9f8-11e9-95ad-0205fe66cbb6
  storageLocation: myrepo-vpzq9
  ttl: 720h0m0s
  volumeSnapshotLocations:
  - myrepo-wv6fx
status:
  completionTimestamp: "2019-08-29T01:02:36Z"
  errors: 0
  expiration: "2019-09-28T01:02:35Z"
  phase: Completed
  startTimestamp: "2019-08-29T01:02:35Z"
  validationErrors: null
  version: 1
  volumeSnapshotsAttempted: 0
  volumeSnapshotsCompleted: 0
  warnings: 0
```

Velero リストア CR #2 の例 (Kubernetes リソース)

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  annotations:
    openshift.io/migrate-copy-phase: final
    openshift.io/migrate-quiesce-pods: "true"
    openshift.io/migration-registry: 172.30.90.187:5000
    openshift.io/migration-registry-dir: /socks-shop-mig-plan-registry-36f54ca7-c925-11e9-825a-
06fa9fb68c88
  creationTimestamp: "2019-08-28T00:09:49Z"
  generateName: e13a1b60-c927-11e9-9555-d129df7f3b96-
  generation: 3
  labels:
    app.kubernetes.io/part-of: migration
    migmigration: e18252c9-c927-11e9-825a-06fa9fb68c88
    migration-final-restore: e18252c9-c927-11e9-825a-06fa9fb68c88
  name: e13a1b60-c927-11e9-9555-d129df7f3b96-gb8nx
  namespace: openshift-migration
  resourceVersion: "82329"
  selfLink: /apis/velero.io/v1/namespaces/openshift-migration/restores/e13a1b60-c927-11e9-9555-
d129df7f3b96-gb8nx
  uid: 26983ec0-c928-11e9-825a-06fa9fb68c88
spec:
  backupName: e13a1b60-c927-11e9-9555-d129df7f3b96-sz24f
  excludedNamespaces: null
  excludedResources:
  - nodes
  - events
  - events.events.k8s.io
  - backups.velero.io
  - restores.velero.io
  - resticrepositories.velero.io
  includedNamespaces: null
  includedResources: null
```

```
  namespaceMapping: null
  restorePVs: true
status:
  errors: 0
  failureReason: ""
  phase: Completed
  validationErrors: null
  warnings: 15
```

3.5.2. 移行ログのダウンロード

CAM Web コンソールで Velero、Restic、および Migration コントローラーログをダウンロードして、移行の失敗についてのトラブルシューティングを行うことができます。

手順

1. CAM Web コンソールにログインします。
2. **Plans** をクリックし、 移行計画の一覧を表示します。
3. 特定の移行計画の **Options** メニュー  をクリックし、**Logs** を選択します。
4. **Download Logs** をクリックし、すべてのクラスターの Migration コントローラー、Velero、および Restic のログをダウンロードします。
5. 特定のログをダウンロードするには、以下を実行します。
   a. ログオプションを指定します。
      - **Cluster**: ソース、ターゲット、または CAM ホストクラスターを選択します。
      - **Log source**: **Velero**、**Restic**、または **Controller** を選択します。
      - **Pod source**: Pod 名を選択します (例: **controller-manager-78c469849c-v6wcf**)。選択したログが表示されます。

        選択した内容を変更することで、ログ選択の設定をクリアできます。

   b. **Download Selected** をクリックし、選択したログをダウンロードします。

オプションで、以下の例にあるように CLI を使用してログにアクセスできます。

```
$ oc get pods -n openshift-migration | grep controller
controller-manager-78c469849c-v6wcf          1/1     Running     0          4h49m

$ oc logs controller-manager-78c469849c-v6wcf -f -n openshift-migration
```

3.5.3. エラーメッセージ

3.5.3.1. Velero Pod ログの Restic タイムアウトエラーメッセージ

Restic のタイムアウトにより移行が失敗する場合、以下のエラーが Velero Pod ログに表示されます。

```
level=error msg="Error backing up item" backup=velero/monitoring error="timed out waiting for all
PodVolumeBackups to complete"
error.file="/go/src/github.com/heptio/velero/pkg/restic/backupper.go:165"
error.function="github.com/heptio/velero/pkg/restic.(*backupper).BackupPodVolumes" group=v1
```

restic_timeout のデフォルト値は 1 時間です。大規模な移行では、このパラメーターの値を大きくすることができます。値を高くすると、エラーメッセージが返されるタイミングが遅れる可能性があることに注意してください。

手順

1. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** に移動します。
2. **Cluster Application Migration Operator** をクリックします。
3. **MigrationController** タブで、**migration-controller** をクリックします。
4. **YAML** タブで、以下のパラメーター値を更新します。

```
spec:
  restic_timeout: 1h (1)
```

(1) 有効な単位は **h** (時間)、**m** (分)、および **s** (秒) です (例: **3h30m15s**)。

5. **Save** をクリックします。

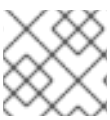
3.5.3.2. MigMigration カスタムリソースの ResticVerifyErrors

ファイルシステムデータのコピー方法を使用して PV を移行する際にデータ検証が失敗すると、以下のエラーが MigMigration カスタムリソース (CR) に表示されます。

```
status:
  conditions:
  - category: Warn
    durable: true
    lastTransitionTime: 2020-04-16T20:35:16Z
    message: There were verify errors found in 1 Restic volume restores. See restore `<registry-
example-migration-rvwcm>`
      for details (1)
    status: "True"
    type: ResticVerifyErrors (2)
```

(1) エラーメッセージはリストア CR 名を識別します。

(2) **ResticErrors** も表示されます。**ResticErrors** は、検証エラーが含まれる一般的なエラーの警告です。



注記

データ検証エラーによって移行プロセスが失敗することはありません。

ターゲットクラスターのリストア CR を確認して、データ検証エラーのソースを特定できます。

手順

1. ターゲットクラスターにログインします。
2. リストア CR を表示します。

```
$ oc describe <registry-example-migration-rvwcm> -n openshift-migration
```

出力では、**PodVolumeRestore** エラーのある PV を特定できます。

```
status:
  phase: Completed
  podVolumeRestoreErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
  podVolumeRestoreResticErrors:
  - kind: PodVolumeRestore
    name: <registry-example-migration-rvwcm-98t49>
    namespace: openshift-migration
```

3. **PodVolumeRestore** CR を表示します。

```
$ oc describe <migration-example-rvwcm-98t49>
```

出力はエラーをログに記録した Restic Pod を特定します。

```
  completionTimestamp: 2020-05-01T20:49:12Z
  errors: 1
  resticErrors: 1
  ...
  resticPod: <restic-nr2v5>
```

4. Restic Pod ログを表示します。

```
$ oc logs -f restic-nr2v5
```

3.5.4. 移行の手動ロールバック

移行の失敗時にアプリケーションが停止された場合は、PV でのデータの破損を防ぐために手動でこれをロールバックする必要があります。

移行時にアプリケーションが停止しなかった場合には、この手順は必要ありません。元のアプリケーションがソースクラスター上で依然として実行されているためです。

手順

1. ターゲットクラスター上で、移行したプロジェクトに切り替えます。

```
$ oc project <project>
```

2. デプロイされたリソースを取得します。

```
$ oc get all
```

3. デプロイされたリソースを削除し、アプリケーションがターゲットクラスターで実行されておらず、PVC 上にあるデータにアクセスできるようにします。

```
$ oc delete <resource_type>
```

4. これを削除せずにデーモンセットを停止するには、YAML ファイルで **nodeSelector** を更新します。

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: hello-daemonset
spec:
  selector:
     matchLabels:
       name: hello-daemonset
  template:
    metadata:
      labels:
        name: hello-daemonset
    spec:
      nodeSelector:
        role: worker 1
```

1 いずれのノードにも存在しない **nodeSelector** 値を指定します。

5. 不要なデータが削除されるように、各 PV の回収ポリシーを更新します。移行時に、バインドされた PV の回収ポリシーは **Retain** であり、アプリケーションがソースクラスターから削除される際にデータの損失が生じないようにされます。ロールバック時にこれらの PV を削除できます。

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0001
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain 1
  ...
status:
  ...
```

1 **Recycle** または **Delete** を指定します。

6. ソースクラスターで、移行したプロジェクトに切り替えます。

```
$ oc project <project_name>
```

7. プロジェクトのデプロイされたリソースを取得します。

```
$ oc get all
```

8. デプロイされた各リソースのレプリカを開始します。

```
$ oc scale --replicas=1 <resource_type>/<resource_name>
```

9. 手順の実行時に変更した場合は、 **DaemonSet** リソースの **nodeSelector** を元の値に更新します。

3.5.5. カスタマーサポートケース用のデータの収集

カスタマーサポートケースを作成する場合、 **openshift-migration-must-gather-rhel8** イメージを使用して **must-gather** ツールを実行し、クラスターについての情報を収集し、これを Red Hat カスタマーポータル にアップロードできます。

openshift-migration-must-gather-rhel8 イメージは、デフォルトの **must-gather** イメージで収集されないログおよびカスタムリソースデータを収集します。

手順

1. **must-gather** データを保存するディレクトリーに移動します。
2. **oc adm must-gather** コマンドを実行します。

```
$ oc adm must-gather --image=registry.redhat.io/rhcam-1-2/openshift-migration-must-gather-rhel8
```

must-gather ツールはクラスター情報を収集し、これを **must-gather.local.<uid>** ディレクトリーに保存します。

3. 認証キーおよびその他の機密情報を **must-gather** データから削除します。
4. **must-gather.local.<uid>** ディレクトリーの内容を含むアーカイブファイルを作成します。

```
$ tar cvaf must-gather.tar.gz must-gather.local.<uid>/
```

圧縮ファイルを Red Hat カスタマーポータル 上のサポートケースに添付します。

3.5.6. 既知の問題

本リリースには、以下の既知の問題があります。

- 移行時に、Cluster Application Migration (CAM) ツールは以下の namespace アノテーションを保持します。
  - **openshift.io/sa.scc.mcs**
  - **openshift.io/sa.scc.supplemental-groups**
  - **openshift.io/sa.scc.uid-range**

これらのアノテーションは UID 範囲を保持し、コンテナーがターゲットクラスターのファイルシステムのパーミッションを保持できるようにします。移行された UID が、ターゲットクラスターの既存の namespace または今後の namespace 内の UID を重複させるリスクがあります。(BZ#1748440)

- AWS バケットが CAM Web コンソールに追加された後に削除される場合、MigStorage CR は更新されないため、そのステータスは **True** のままになります。(BZ#1738564)
- ほとんどのクラススコープのリソースは CAM ツールで処理されません。アプリケーションがクラススコープのリソースを必要とする場合、ターゲットクラスターでそれらを手動で作成する必要がある場合があります。
- 移行に失敗すると、移行計画は休止状態の Pod のカスタム PV 設定を保持しません。移行を手動でロールバックし、移行計画を削除し、PV 設定で新たな移行計画を作成する必要があります。(BZ#1784899)
- Restic のタイムアウトにより大規模な移行が失敗する場合は、Migration コントローラー CR の **restic_timeout** パラメーターの値 (デフォルト: **1h**) を増やすことができます。
- ファイルシステムのコピー方法で移行される PV のデータ検証オプションを選択すると、パフォーマンスは大幅に遅くなります。Velero は各ファイルのチェックサムを生成し、ファイルを復元する際に確認します。
- 現在のリリース (CAM 1.2) では、イメージストリームが含まれる namespace を OpenShift Container Platform 4.4 以降から移行することはできません。以下のエラーメッセージが Velero Pod ログに表示されます: **Error restore nametags**(BZ#1848561)