



OpenShift Container Platform 4.5

レジストリー

OpenShift Container Platform のレジストリーの設定

OpenShift Container Platform 4.5 レジストリー

OpenShift Container Platform のレジストリーの設定

法律上の通知

Copyright © 2021 Red Hat, Inc.

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

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

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

概要

本書では、OpenShift Container Platform の内部レジストリーを設定し、管理する方法について説明します。また、OpenShift Container Platform に関連付けられたレジストリーの概要も提供します。

目次

第1章 イメージレジストリー	3
1.1. 統合 OPENSIFT CONTAINER PLATFORM レジストリー	3
第2章 OPENSIFT CONTAINER PLATFORM のイメージレジストリー OPERATOR	4
2.1. クラウドプラットフォームおよび OPENSTACK のイメージレジストリー	4
2.2. ベアメタルおよび VSPHERE のイメージレジストリー	4
2.3. イメージレジストリー OPERATOR の設定パラメーター	5
2.4. イメージレジストリーのデフォルトルートのカスタムリソース定義 (CRD、CUSTOM RESOURCE DEFINITION) で有効にする	6
2.5. イメージレジストリーアクセス用の追加のトラストストアの設定	6
2.6. イメージレジストリー OPERATOR のストレージの認証情報の設定	7
2.7. 追加リソース	7
第3章 レジストリーのセットアップおよび設定	8
3.1. AWS のユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定	8
3.2. GCP のユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定	10
3.3. AZURE ユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定	11
3.4. ベアメタルのレジストリーの設定	12
3.5. VSPHERE のレジストリーの設定	16
第4章 レジストリーオプション	21
4.1. 統合 OPENSIFT CONTAINER PLATFORM レジストリー	21
4.2. サードパーティーレジストリー	21
4.3. RED HAT QUAY レジストリー	21
4.4. 認証で有効にされる RED HAT レジストリー	22
第5章 レジストリーへのアクセス	23
5.1. 前提条件	23
5.2. クラスタからレジストリーへの直接アクセス	23
5.3. レジストリー POD のステータスの確認	25
5.4. レジストリーログの表示	25
5.5. レジストリーメトリクスへのアクセス	26
5.6. 追加リソース	27
第6章 レジストリーの公開	28
6.1. セキュアなレジストリーの手動による公開	28

第1章 イメージレジストリー

1.1. 統合 OPENSIFT CONTAINER PLATFORM レジストリー

OpenShift Container Platform は、クラスター上の標準ワークロードとして実行される組み込みコンテナイメージレジストリーを提供します。このレジストリーはインフラストラクチャー Operator によって設定され、管理されます。これはユーザーがワークロードを実行するイメージを管理するために追加設定なしで使用できるソリューションを提供し、既存のクラスターインフラストラクチャーの上部で実行されます。このレジストリーは、他のクラスターワークロードのようにスケールアップまたはスケールダウンでき、特定のインフラストラクチャーのプロビジョニングを必要としません。さらに、これはクラスターのユーザー認証および認可システムに統合されるため、イメージを作成し、取得するためのアクセスは、イメージリソースでユーザーのパーミッションを定義することによって制御できることを意味します。

通常、レジストリーはクラスター上にビルドされたイメージの公開ターゲットとして、またクラスター上で実行されるワークロードのイメージのソースとして使用されます。新規イメージがレジストリーにプッシュされると、クラスターにはその新規イメージについて通知され、他のコンポーネントは更新されたイメージに応答し、これを使用できます。

イメージデータは2つの場所に保存されます。実際のイメージデータは、クラウドストレージまたはファイルシステムボリュームなどの設定可能なストレージの場所に格納されます。標準のクラスター API によって公開され、アクセス制御を実行するために使用されるイメージメタデータは、標準的な API リソース、とくにイメージおよびイメージストリームとして保存されます。

追加リソース

- [OpenShift Container Platform のイメージレジストリー Operator](#)

第2章 OPENSIFT CONTAINER PLATFORM のイメージレジストリー OPERATOR

2.1. クラウドプラットフォームおよび OPENSTACK のイメージレジストリー

イメージレジストリー Operator は、OpenShift Container Platform レジストリーの単一インスタンスをインストールし、レジストリーストレージのセットアップを含む、レジストリーのすべての設定を管理します。



注記

ストレージは、AWS、GCP、Azure または OpenStack にインストーラーでプロビジョニングされるインフラストラクチャークラスターをインストールする場合にのみ自動的に設定されます。

コントロールプレーンのデプロイ後、Operator はクラスターで検出される設定に基づいてデフォルトの **configs.imageregistry.operator.openshift.io** リソースインスタンスを作成します。

完全な **configs.imageregistry.operator.openshift.io** リソースを定義するのに利用できる情報が十分でない場合、その不完全なリソースが定義され、Operator は足りない情報を示す情報を使ってリソースのステータスを更新します。

イメージレジストリー Operator は **openshift-image-registry** namespace で実行され、その場所のレジストリーインスタンスも管理します。レジストリーのすべての設定およびワークロードリソースはその namespace に置かれます。



重要

プルーナーを管理するためのイメージレジストリー Operator の動作は、イメージレジストリー Operator の **ClusterOperator** オブジェクトで指定される **managementState** とは独立しています。イメージレジストリー Operator が **Managed** の状態ではない場合、イメージプルーナーは **Pruning** カスタムリソースによって設定され、管理できます。

ただし、イメージレジストリー Operator の **managementState** は、デプロイされたイメージプルーナージョブの動作を変更します。

- **Managed:** イメージプルーナーの **--prune-registry** フラグは **true** に設定されません。
- **Removed:** イメージプルーナーの **--prune-registry** フラグは **false** に設定されます。つまり、これは etcd のイメージメタデータのためのブルーニングを実行します。
- **Unmanaged:** イメージプルーナーの **--prune-registry** フラグは **false** に設定されます。

2.2. ベアメタルおよび VSPHERE のイメージレジストリー

2.2.1. インストール時に削除されたイメージレジストリー

共有可能なオブジェクトストレージを提供しないプラットフォームでは、OpenShift イメージレジストリー Operator 自体が **Removed** としてブートストラップされます。これにより、**openshift-installer** がそれらのプラットフォームタイプでのインストールを完了できます。

インストール後に、イメージレジストリー Operator 設定を編集して **managementState** を **Removed** から **Managed** に切り替える必要があります。



注記

Prometheus コンソールは、以下のような **ImageRegistryRemoved** アラートを提供しません。

"Image Registry has been removed. **ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected. Please configure storage and update the config to **Managed** state by editing `configs.imageregistry.operator.openshift.io`."

2.3. イメージレジストリー OPERATOR の設定パラメーター

`configs.imageregistry.operator.openshift.io` リソースは以下の設定パラメーターを提供します。

パラメーター	説明
managementState	<p>Managed: Operator は、設定リソースが更新されるとレジストリーを更新します。</p> <p>Unmanaged: Operator は設定リソースへの変更を無視します。</p> <p>Removed: Operator はレジストリーインスタンスを取り除き、Operator がプロビジョニングしたすべてのストレージを削除します。</p>
logging	レジストリーインスタンスの loglevel を設定します。
httpSecret	デフォルトで生成されるアップロードのセキュリティーを保護するためにレジストリーに必要な値。
proxy	マスター API およびアップストリームレジストリーの呼び出し時に使用されるプロキシを定義します。
storage	StorageType: レジストリーストレージを設定するための詳細。たとえば、S3 バケットの位置情報 (coordinate) など。通常はデフォルトで設定されます。
readOnly	レジストリーインスタンスが新規イメージのプッシュや既存イメージの削除の試行を拒否するかどうかを示します。
requests	API 要求の制限の詳細。指定されたレジストリーインスタンスが追加リソースをキューに入れる前に処理する並列要求の数を制御します。
defaultRoute	外部ルートがデフォルトのホスト名を使用して定義されるかどうかを決定します。これが有効にされている場合、ルートは re-encrypt 暗号を使用します。デフォルトは <code>false</code> に設定されます。

パラメーター	説明
routes	作成する追加ルートの配列。ルートにホスト名および証明書を指定します。
replicas	レジストリーのレプリカ数。

2.4. イメージレジストリーのデフォルトルートのカスタムリソース定義 (CRD、CUSTOM RESOURCE DEFINITION) で有効にする

OpenShift Container Platform では、**Registry Operator** はレジストリー機能を制御します。Operator は、**configs.imageregistry.operator.openshift.io** カスタムリソース定義 (CRD) で定義されます。

イメージレジストリーのデフォルトルートを自動的に有効にする必要がある場合には、イメージレジストリー Operator CRD のパッチを適用します。

手順

- イメージレジストリー Operator CRD にパッチを適用します。

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --type merge -p '{"spec": {"defaultRoute": true}}'
```

2.5. イメージレジストリーアクセス用の追加のトラストストアの設定

image.config.openshift.io/cluster カスタムリソースには、イメージレジストリーのアクセス時に信頼される追加の認証局が含まれる設定マップへの参照を含めることができます。

前提条件

- 認証局 (CA) は PEM でエンコードされている必要があります。

手順

設定マップを **openshift-config** namespace に作成し、その名前を **image.config.openshift.io** カスタムリソースの **AdditionalTrustedCA** で使用し、追加の CA を指定することができます。

設定マップキーは、この CA が信頼されるポートを持つレジストリーのホスト名であり、base64 エンコード証明書が信頼する追加の各レジストリー CA についての値になります。

イメージレジストリー CA の設定マップの例

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-registry-ca
data:
  registry.example.com: |
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
  registry-with-port.example.com..5000: | 1
```

```
-----BEGIN CERTIFICATE-----
```

```
...
```

```
-----END CERTIFICATE-----
```

- 1 レジストリーにポートがある場合 (例: **registry-with-port.example.com:5000**)、**「:」** は **..** に置き換える必要があります。

以下の手順で追加の CA を設定することができます。

1. 追加の CA を設定するには、以下を実行します。

```
$ oc create configmap registry-config --from-file=<external_registry_address>=ca.crt -n
openshift-config
```

```
$ oc edit image.config.openshift.io cluster
```

```
spec:
  additionalTrustedCA:
    name: registry-config
```

2.6. イメージレジストリー OPERATOR のストレージの認証情報の設定

configs.imageregistry.operator.openshift.io および ConfigMap リソースのほかにも、**openshift-image-registry** namespace 内の別のシークレットリソースによってストレージの認証情報の設定が Operator に提供されます。

image-registry-private-configuration-user シークレットは、ストレージのアクセスおよび管理に必要な認証情報を提供します。これは、デフォルト認証情報が見つからない場合に Operator によって使用されるデフォルト認証情報を上書きします。

手順

- 必要なキーが含まれる OpenShift Container Platform シークレットを作成します。

```
$ oc create secret generic image-registry-private-configuration-user --from-file=KEY1=value1
--from-literal=KEY2=value2 --namespace openshift-image-registry
```

2.7. 追加リソース

- [AWS のユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定](#)
- [GCP のユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定](#)
- [Azure ユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定](#)
- [ベアメタルのレジストリーの設定](#)
- [vSphere のレジストリーの設定](#)

第3章 レジストリーのセットアップおよび設定

3.1. AWS のユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定

3.1.1. イメージレジストリー Operator のシークレットの設定

`configs.imageregistry.operator.openshift.io` および ConfigMap リソースのほかにも、`openshift-image-registry` namespace 内の別のシークレットリソースによって設定が Operator に提供されます。

`image-registry-private-configuration-user` シークレットは、ストレージのアクセスおよび管理に必要な認証情報を提供します。これは、デフォルト認証情報が見つからない場合に Operator によって使用されるデフォルト認証情報を上書きします。

AWS ストレージの S3 の場合、シークレットには以下のキーが含まれることが予想されます。

- `REGISTRY_STORAGE_S3_ACCESSKEY`
- `REGISTRY_STORAGE_S3_SECRETKEY`

手順

- 必要なキーが含まれる OpenShift Container Platform シークレットを作成します。

```
$ oc create secret generic image-registry-private-configuration-user --from-literal=REGISTRY_STORAGE_S3_ACCESSKEY=myaccesskey --from-literal=REGISTRY_STORAGE_S3_SECRETKEY=mysecretkey --namespace openshift-image-registry
```

3.1.2. ユーザーによってプロビジョニングされるインフラストラクチャーでの AWS のレジストリーストレージの設定

インストール時に、Amazon S3 バケットを作成するにはクラウド認証情報を使用でき、レジストリー Operator がストレージを自動的に設定します。

レジストリー Operator が S3 バケットを作成できず、ストレージを自動的に設定する場合、以下の手順により S3 バケットを作成し、ストレージを設定することができます。

前提条件

- ユーザーによってプロビジョニングされるインフラストラクチャーでの AWS 上のクラスター。
- Amazon S3 ストレージの場合、シークレットには以下のキーが含まれることが予想されます。
 - `REGISTRY_STORAGE_S3_ACCESSKEY`
 - `REGISTRY_STORAGE_S3_SECRETKEY`

手順

レジストリー Operator が S3 バケットを作成できず、ストレージを自動的に設定する場合は、以下の手順を使用してください。

1. [バケットライフサイクルポリシー](#) を設定し、1日以上経過している未完了のマルチパートアップロードを中止します。
2. `configs.imageregistry.operator.openshift.io/cluster` にストレージ設定を入力します。

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

設定例

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```



警告

AWS でレジストリーイメージのセキュリティーを保護するには、S3 バケットに対して [パブリックアクセスのブロック](#) を実行します。

3.1.3. AWS S3 のイメージレジストリー Operator 設定パラメーター

以下の設定パラメーターは AWS S3 レジストリーストレージで利用できます。

ImageRegistryConfigStorageS3 は、バックエンドストレージに AWS S3 サービスを使用するようにレジストリーを設定する情報を保持します。詳細は、[S3 ストレージドライバーのドキュメント](#) を参照してください。

パラメーター	説明
bucket	バケットは、レジストリーのデータを保存するバケット名です。これはオプションであり、指定されていない場合は生成されます。
region	リージョンはバケットが存在する AWS リージョンです。これはオプションであり、インストール済みの AWS リージョンに基づいて設定されます。
regionEndpoint	RegionEndpoint は、S3 互換のストレージサービスのエンドポイントです。これは、指定されるリージョンに応じてオプションおよびデフォルトになります。
virtualHostedStyle	VirtualHosted は、カスタム RegionEndpoint で S3 仮想ホストスタイルのバケットパスの使用を有効にします。これはオプションであり、デフォルトは false です。 このパラメーターを設定して、OpenShift Container Platform を非表示のリージョンにデプロイします。

パラメーター	説明
encrypt	encrypt は、イメージが暗号化された形式で保存されるかどうかを指定します。これはオプションであり、デフォルトは false です。
keyID	KeyID は、暗号化に使用する KMS キー ID です。これはオプションです。encrypt は true である必要があります。そうでない場合、このパラメーターは無視されます。
ImageRegistryConfigStorageS3CloudFront	CloudFront は Amazon Cloudfront をレジストリーでストレージミドルウェアとして設定します。これはオプションです。

3.2. GCP のユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定

3.2.1. イメージレジストリー Operator のシークレットの設定

configs.imageregistry.operator.openshift.io および ConfigMap リソースのほかにも、**openshift-image-registry** namespace 内の別のシークレットリソースによって設定が Operator に提供されます。

image-registry-private-configuration-user シークレットは、ストレージのアクセスおよび管理に必要な認証情報を提供します。これは、デフォルト認証情報が見つからない場合に Operator によって使用されるデフォルト認証情報を上書きします。

GCP ストレージ上の GCS の場合、シークレットには、GCP が提供する認証情報ファイルの内容に相当するキーが含まれることが予想されます。

- **REGISTRY_STORAGE_GCS_KEYFILE**

手順

- 必要なキーが含まれる OpenShift Container Platform シークレットを作成します。

```
$ oc create secret generic image-registry-private-configuration-user --from-file=REGISTRY_STORAGE_GCS_KEYFILE=<path_to_keyfile> --namespace openshift-image-registry
```

3.2.2. ユーザーによってプロビジョニングされるインフラストラクチャーでの GCP のレジストリーストレージ

ストレージメディアは手動で設定し、レジストリーのカスタムリソース (CR) で設定を行う必要があります。

前提条件

- ユーザーによってプロビジョニングされるインフラストラクチャーのある GCP 上のクラスター。
- GCP のレジストリーストレージを設定するには、レジストリー Operator クラウド認証情報を指定する必要があります。

- GCP ストレージ上の GCS の場合、シークレットには、GCP が提供する認証情報ファイルの内容に相当するキーが含まれることが予想されます。
 - **REGISTRY_STORAGE_GCS_KEYFILE**

3.2.3. GCP GCS のイメージレジストリー Operator 設定パラメーター。

手順

以下の設定パラメーターは、GCP GCS レジストリーストレージに利用できます。

パラメーター	説明
bucket	バケットは、レジストリーのデータを保存するバケット名です。これはオプションであり、指定されていない場合は生成されます。
region	リージョンは、バケットが存在する GCS の場所です。これはオプションであり、インストールされている GCS リージョンに基づいて設定されます。
projectID	ProjectID は、このバケットが関連付けられる必要がある GCP プロジェクトのプロジェクト ID です。これはオプションです。
keyID	KeyID は、暗号化に使用する KMS キー ID です。バケットは GCP でデフォルトで暗号化されているため、これはオプションになります。これにより、カスタム暗号化キーを使用できます。

3.3. AZURE ユーザーによってプロビジョニングされるインフラストラクチャーのレジストリーの設定

3.3.1. イメージレジストリー Operator のシークレットの設定

configs.imageregistry.operator.openshift.io および ConfigMap リソースのほかにも、**openshift-image-registry** namespace 内の別のシークレットリソースによって設定が Operator に提供されます。

image-registry-private-configuration-user シークレットは、ストレージのアクセスおよび管理に必要な認証情報を提供します。これは、デフォルト認証情報が見つからない場合に Operator によって使用されるデフォルト認証情報を上書きします。

Azure レジストリーストレージの場合、シークレットには、Azure が提供する認証情報ファイルの内容に相当する値を持つキーが含まれることが予想されます。

- **REGISTRY_STORAGE_AZURE_ACCOUNTKEY**

手順

- 必要なキーが含まれる OpenShift Container Platform シークレットを作成します。

```
$ oc create secret generic image-registry-private-configuration-user --from-literal=REGISTRY_STORAGE_AZURE_ACCOUNTKEY=<accountkey> --namespace openshift-image-registry
```

3.3.2. Azure の場合のレジストリーストレージの設定

インストール時に、Azure Blob Storage を作成するにはクラウド認証情報を使用でき、レジストリー Operator がストレージを自動的に設定します。

前提条件

- ユーザーによってプロビジョニングされるインフラストラクチャーでの Azure 上のクラスター。
- Azure のレジストリーストレージを設定するには、レジストリー Operator クラウド認証情報を指定する必要があります。
- AWS ストレージの場合、シークレットには1つのキーが含まれることが予想されます。
 - **REGISTRY_STORAGE_AZURE_ACCOUNTKEY**

手順

1. [Azure ストレージコンテナ](#) を作成します。
2. **configs.imageregistry.operator.openshift.io/cluster** にストレージ設定を入力します。

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

設定例

```
storage:
  azure:
    accountName: <storage-account-name>
    container: <container-name>
```

3.4. ベアメタルのレジストリーの設定

3.4.1. インストール時に削除されたイメージレジストリー

共有可能なオブジェクトストレージを提供しないプラットフォームでは、OpenShift イメージレジストリー Operator 自体が **Removed** としてブートストラップされます。これにより、**openshift-installer** がそれらのプラットフォームタイプでのインストールを完了できます。

インストール後に、イメージレジストリー Operator 設定を編集して **managementState** を **Removed** から **Managed** に切り替える必要があります。



注記

Prometheus コンソールは、以下のような **ImageRegistryRemoved** アラートを提供しません。

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

3.4.2. イメージレジストリーの管理状態の変更

イメージレジストリーを起動するには、イメージレジストリー Operator 設定の **managementState** を **Removed** から **Managed** に変更する必要があります。

手順

- **ManagementState** イメージレジストリー Operator 設定を **Removed** から **Managed** に変更します。以下は例になります。

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

3.4.3. イメージレジストリーストレージの設定

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

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

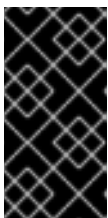
アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

3.4.4. ベアメタルの場合のレジストリーストレージの設定

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

前提条件

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



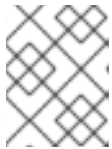
重要

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

- 100Gi の容量が必要です。

手順

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

**注記**

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

2. レジストリー Pod がないことを確認します。

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

**注記**

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

3. レジストリー設定を確認します。

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

出力例

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

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

4. **clusteroperator** ステータスを確認します。

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

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

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

手順

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

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

**警告**

実稼働用以外のクラスターにのみこのオプションを設定します。

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

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

数分待機した後に、このコマンドを再び実行します。

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

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

次に、行を変更します。

```
managementState: Removed
```

次のように変更してください。

```
managementState: Managed
```

3.4.6. ベアメタルの場合のブロックレジストリーストレージの設定

イメージレジストリーがクラスター管理者によるアップグレード時にブロックストレージタイプを使用できるようにするには、**Recreate** ロールアウトストラテジーを使用できます。



重要

ブロックストレージボリュームはサポートされますが、実稼働クラスターでのイメージレジストリーと併用することは推奨されません。レジストリーに複数のレプリカを含めることができないため、ブロックストレージにレジストリーが設定されているインストールに高可用性はありません。

手順

1. イメージレジストリーストレージをブロックストレージタイプとして設定するには、レジストリーが **Recreate** ロールアウトストラテジーを使用し、1つの (1) レプリカのみで実行されるように、レジストリーにパッチを適用します。

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. ブロックストレージデバイスの PV をプロビジョニングし、そのボリュームの PVC を作成します。要求されたブロックボリュームは ReadWriteOnce (RWO) アクセスモードを使用します。
3. 正しい PVC を参照するようにレジストリー設定を編集します。

3.4.7. 追加リソース

ベアメタルの場合のレジストリーストレージの設定方法についての詳細は、「[Recommended configurable storage technology](#)」を参照してください。

3.5. VSPHERE のレジストリーの設定

3.5.1. インストール時に削除されたイメージレジストリー

共有可能なオブジェクトストレージを提供しないプラットフォームでは、OpenShift イメージレジストリー Operator 自体が **Removed** としてブートストラップされます。これにより、**openshift-installer** がそれらのプラットフォームタイプでのインストールを完了できます。

インストール後に、イメージレジストリー Operator 設定を編集して **managementState** を **Removed** から **Managed** に切り替える必要があります。



注記

Prometheus コンソールは、以下のような **ImageRegistryRemoved** アラートを提供しません。

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

3.5.2. イメージレジストリーの管理状態の変更

イメージレジストリーを起動するには、イメージレジストリー Operator 設定の **managementState** を **Removed** から **Managed** に変更する必要があります。

手順

- **ManagementState** イメージレジストリー Operator 設定を **Removed** から **Managed** に変更します。以下は例になります。

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

3.5.2.1. イメージレジストリーストレージの設定

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

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

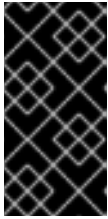
アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

3.5.3. VMware vSphere のレジストリーストレージの設定

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

前提条件

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



重要

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

- 「100Gi」の容量が必要です。



重要

テストにより、NFS サーバーを RHEL でコアサービスのストレージバックエンドとして使用することに関する問題が検出されています。これには、OpenShift Container レジストリーおよび Quay、メトリクスストレージの Prometheus、およびロギングストレージの Elasticsearch が含まれます。そのため、コアサービスで使用される PV をサポートするために RHEL NFS を使用することは推奨されていません。

他の NFS の実装ではこれらの問題が検出されない可能性があります。OpenShift Container Platform コアコンポーネントに対して実施された可能性のあるテストに関する詳細情報は、個別の NFS 実装ベンダーにお問い合わせください。

手順

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



注記

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

2. レジストリー Pod がないことを確認します。

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



注記

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

3. レジストリー設定を確認します。

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

出力例

```
storage:
  pvc:
    claim: ❶
```

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

4. **clusteroperator** ステータスを確認します。

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

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

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

手順

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

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



警告

実稼働用以外のクラスターにのみこのオプションを設定します。

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

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

数分待機した後に、このコマンドを再び実行します。

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

- 以下を実行します。

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

次に、行を変更します。

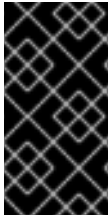
```
managementState: Removed
```

次のように変更してください。

```
managementState: Managed
```

3.5.5. VMware vSphere のブロックレジストリーストレージの設定

イメージレジストリーがクラスター管理者によるアップグレード時に vSphere Virtual Machine Disk (VMDK) などのブロックストレージタイプを使用できるようにするには、**Recreate** ロールアウトストラテジーを使用できます。



重要

ブロックストレージボリュームはサポートされますが、実稼働クラスターでのイメージレジストリーと併用することは推奨されません。レジストリーに複数のレプリカを含めることができないため、ブロックストレージにレジストリーが設定されているインストールに高可用性はありません。

手順

1. イメージレジストリーストレージをブロックストレージタイプとして設定するには、レジストリーが **Recreate** ロールアウトストラテジーを使用し、**1** レプリカのみで実行されるように、レジストリーにパッチを適用します。

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate", "replicas": 1}}'
```

2. ブロックストレージデバイスの PV をプロビジョニングし、そのボリュームの PVC を作成します。要求されたブロックボリュームは ReadWriteOnce (RWO) アクセスモードを使用します。
 - a. 以下の内容で **pvc.yaml** ファイルを作成して VMware vSphere **PersistentVolumeClaim** オブジェクトを定義します。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage ❶
spec:
  accessModes:
  - ReadWriteOnce ❷
resources:
  requests:
    storage: 100Gi ❸
```

- ❶ **PersistentVolumeClaim** オブジェクトを表す一意の名前。
- ❷ **PersistentVolumeClaim** のアクセスモード。**ReadWriteOnce** では、ボリュームは単一ノードによって読み取り/書き込みパーミッションでマウントできます。
- ❸ **PersistentVolumeClaim** のサイズ。

- b. ファイルから **PersistentVolumeClaim** オブジェクトを作成します。

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

- 正しい PVC を参照するようにレジストリー設定を編集します。

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

出力例

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

- ❶ カスタム PVC を作成すると、**image-registry-storage** PVC のデフォルトの自動作成の **claim** フィールドを空のままにすることができます。

正しい PVC を参照するようにレジストリーストレージを設定する方法については、「[vSphere のレジストリーの設定](#)」を参照してください。

3.5.6. 追加リソース

vSphere の場合のレジストリーストレージの設定方法についての詳細は、「[Recommended configurable storage technology](#)」を参照してください。

第4章 レジストリーオプション

OpenShift Container Platform はイメージをソースコードからビルドし、それらをデプロイし、それらのライフサイクルを管理できます。これを可能にするため、OpenShift Container Platform は内部の統合コンテナイメージレジストリーを提供しています。このレジストリーは OpenShift Container Platform 環境にデプロイでき、ここからイメージをローカルで管理できます。

4.1. 統合 OPENSIFT CONTAINER PLATFORM レジストリー

OpenShift Container Platform は、クラスター上の標準ワークロードとして実行される組み込みコンテナイメージレジストリーを提供します。このレジストリーはインフラストラクチャー Operator によって設定され、管理されます。これはユーザーがワークロードを実行するイメージを管理するために追加設定なしで使用できるソリューションを提供し、既存のクラスターインフラストラクチャーの上部で実行されます。このレジストリーは、他のクラスターワークロードのようにスケールアップまたはスケールダウンでき、特定のインフラストラクチャーのプロビジョニングを必要としません。さらに、これはクラスターのユーザー認証および認可システムに統合されるため、イメージを作成し、取得するためのアクセスは、イメージリソースでユーザーのパーミッションを定義することによって制御できることを意味します。

通常、レジストリーはクラスター上にビルドされたイメージの公開ターゲットとして、またクラスター上で実行されるワークロードのイメージのソースとして使用されます。新規イメージがレジストリーにプッシュされると、クラスターにはその新規イメージについて通知され、他のコンポーネントは更新されたイメージに応答し、これを使用できます。

イメージデータは2つの場所に保存されます。実際のイメージデータは、クラウドストレージまたはファイルシステムボリュームなどの設定可能なストレージの場所に格納されます。標準のクラスター API によって公開され、アクセス制御を実行するために使用されるイメージメタデータは、標準的な API リソース、とくにイメージおよびイメージストリームとして保存されます。

4.2. サードパーティーレジストリー

OpenShift Container Platform はサードパーティーレジストリーからのイメージを使用してコンテナを作成できますが、これらのレジストリーは統合 OpenShift Container Platform レジストリーと同じイメージ通知のサポートを提供する訳ではありません。このため、OpenShift Container Platform はイメージストリームの作成時にリモートレジストリーからタグをフェッチします。

フェッチされたタグの更新は、**oc import-image <stream>** を実行するだけで簡単に実行できます。新規イメージが検出されると、以前に記述されたビルドとデプロイメントの応答が生じます。

4.2.1. 認証

OpenShift Container Platform はユーザーが指定する認証情報を使用してプライベートイメージリポジトリにアクセスするためにレジストリーと通信できます。これにより、OpenShift Container Platform はイメージのプッシュ/プルをプライベートリポジトリへ/から実行できます。

4.3. RED HAT QUAY レジストリー

エンタープライズ向けの高品質なコンテナイメージレジストリーを必要とされる場合、Red Hat Quay をホストされたサービスとして、また独自のデータセンターやクラウド環境にインストールするソフトウェアとしてご利用いただけます。Red Hat Quay の高度なレジストリーには、geo レプリケーション、イメージのスキャニング、およびイメージのロールバック機能が含まれます。

Quay.io サイトにアクセスし、独自のホストされる Quay レジストリーアカウントをセットアップします。その後、Quay チュートリアルに従って Quay レジストリーにログインし、イメージの管理を開始します。

Red Hat Quay レジストリーへのアクセスは、任意のリモートコンテナイメージレジストリーと同様に OpenShift Container Platform から実行できます。

4.4. 認証で有効にされる RED HAT レジストリー

Red Hat Ecosystem Catalog のコンテナイメージのセクションで利用可能なすべてのコンテナイメージはイメージレジストリーの **registry.redhat.io** でホストされます。

レジストリー **registry.redhat.io** では、イメージおよび OpenShift Container Platform でホストされるコンテンツへのアクセスに認証が必要です。新規レジストリーへの移行後も、既存レジストリーはしばらく利用可能になります。



注記

OpenShift Container Platform はイメージを **registry.redhat.io** からプルするため、これを使用できるようにクラスターを設定する必要があります。

新規レジストリーは、以下の方法を使用して認証に標準の OAuth メカニズムを使用します。

- **認証トークン**。管理者によって生成されるこれらのトークンは、システムにコンテナイメージレジストリーに対する認証機能を付与するサービスアカウントです。サービスアカウントはユーザーアカウントの変更による影響を受けないため、トークンの認証方法は信頼性があり、回復性があります。これは、実稼働クラスター用にサポートされている唯一の認証オプションです。
- **Web ユーザー名およびパスワード**。これは、**access.redhat.com** などのリソースへのログインに使用する標準的な認証情報のセットです。OpenShift Container Platform でこの認証方法を使用することはできますが、これは実稼働デプロイメントではサポートされません。この認証方法の使用は、OpenShift Container Platform 外のスタンドアロンのプロジェクトに制限されます。

ユーザー名およびパスワード、または認証トークンのいずれかの認証情報を使用して **podman login** を使用し、新規レジストリーのコンテンツにアクセスします。

すべてのイメージストリームは、インストールプルシークレットを使用して認証を行う新規レジストリーを参照します。

認証情報は以下のいずれかの場所に配置する必要があります。

- **openshift namespace**。OpenShift namespace のイメージストリームがインポートできるように、認証情報は **openshift namespace** になければなりません。
- **ホスト**。Kubernetes でイメージをプルする際にホストの認証情報を使用するため、認証情報はホスト上になければなりません。

第5章 レジストリーへのアクセス

ログおよびメトリクスの表示やレジストリーのセキュリティー保護および公開などの、レジストリーへのアクセスについての各種の方法について、以下のセクションを参照してください。

レジストリーに直接アクセスし、**podman** コマンドを起動することが可能です。これにより、**podman push** や **podman pull** などの操作で統合レジストリーへ/からイメージを直接プッシュまたはプルすることができます。これを実行するには、**oc login** コマンドを使ってレジストリーにログインしている必要があります。実行できる操作は、以下のセクションで説明されているようにユーザーが持つパーミッションによって異なります。

5.1. 前提条件

- アイデンティティプロバイダー (IDP) を設定しておく必要があります。
- **podman pull** コマンドを使用する場合などにイメージをプルするには、ユーザーに **registry-viewer** ロールがなければなりません。このロールを追加するには、以下を実行します。

```
$ oc policy add-role-to-user registry-viewer <user_name>
```

- イメージの書き出しやプッシュを実行するには (**podman push** コマンドを使用する場合など)、ユーザーに **registry-editor** ロールが必要です。このロールを追加するには、以下を実行します。

```
$ oc policy add-role-to-user registry-editor <user_name>
```

5.2. クラスターからレジストリーへの直接アクセス

クラスター内からレジストリーにアクセスすることができます。

手順

内部ルートを使用して、クラスターからレジストリーにアクセスします。

1. ノードのアドレスを取得することにより、ノードにアクセスします。

```
$ oc get nodes
```

```
$ oc debug nodes/<node_address>
```

2. ノード上で **oc** や **podman** などのツールへのアクセスを有効にするには、以下のコマンドを実行します。

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

3. アクセストークンを使用してコンテナイメージレジストリーにログインします。

```
sh-4.2# oc login -u kubeadmin -p <password_from_install_log> https://api-int.
<cluster_name>.<base_domain>:6443
```

```
sh-4.2# podman login -u kubeadmin -p $(oc whoami -t) image-registry.openshift-image-
registry.svc:5000
```

以下のようなログインを確認するメッセージが表示されるはずです。

```
Login Succeeded!
```



注記

ユーザー名には任意の値を指定でき、トークンには必要な情報がすべて含まれます。コロンが含まれるユーザー名を指定すると、ログインに失敗します。

イメージレジストリー Operator はルートを作成するため、**default-route-openshift-image-registry.<cluster_name>** のようになります。

- レジストリーに対して **podman pull** および **podman push** 操作を実行します。



重要

任意のイメージをプルできますが、**system:registry** ロールを追加している場合は、各自のプロジェクトにあるレジストリーにのみイメージをプッシュすることができます。

次の例では、以下を使用します。

コンポーネント	値
<registry_ip>	172.30.124.220
<port>	5000
<project>	openshift
<image>	image
<tag>	省略 (デフォルトは latest)

- 任意のイメージをプルします。

```
$ podman pull name.io/image
```

- 新規イメージに **<registry_ip>:<port>/<project>/<image>** 形式でタグ付けします。プロジェクト名は、イメージを正しくレジストリーに配置し、これに後でアクセスできるようにするために OpenShift Container Platform のプル仕様に表示される必要があります。

```
$ podman tag name.io/image image-registry.openshift-image-registry.svc:5000/openshift/image
```



注記

指定されたプロジェクトについて **system:image-builder** ロールを持っている必要があります。このロールにより、ユーザーはイメージの書き出しやプッシュを実行できます。そうでない場合は、次の手順の **podman push** は失敗します。これをテストするには、新規プロジェクトを作成し、イメージをプッシュできます。

- c. 新しくタグ付けされたイメージをレジストリーにプッシュします。

```
$ podman push image-registry.openshift-image-registry.svc:5000/openshift/image
```

5.3. レジストリー POD のステータスの確認

クラスター管理者は、**openshift-image-registry** プロジェクトで実行されているイメージレジストリー Pod を一覧表示し、それらのステータスを確認できます。

前提条件

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift CLI (**oc**) がインストールされていること。

手順

1. **openshift-image-registry** プロジェクトの Pod を一覧表示し、それらのステータスを表示します。

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

出力例

```
NAME READY STATUS RESTARTS AGE
cluster-image-registry-operator-764bd7f846-qqtph 1/1 Running 0 78m
image-registry-79fb4469f6-llrln 1/1 Running 0 77m
node-ca-hjksc 1/1 Running 0 73m
node-ca-tftj6 1/1 Running 0 77m
node-ca-wb6ht 1/1 Running 0 77m
node-ca-zvt9q 1/1 Running 0 74m
```

5.4. レジストリーログの表示

oc logs コマンドを使用してレジストリーのログを表示することができます。

手順

1. デプロイメントで **oc logs** コマンドを使用して、コンテナイメージレジストリーのログを表示します。

```
$ oc logs deployments/image-registry -n openshift-image-registry
```

出力例

```

2015-05-01T19:48:36.300593110Z time="2015-05-01T19:48:36Z" level=info
msg="version=v2.0.0+unknown"
2015-05-01T19:48:36.303294724Z time="2015-05-01T19:48:36Z" level=info msg="redis not
configured" instance.id=9ed6c43d-23ee-453f-9a4b-031fea646002
2015-05-01T19:48:36.303422845Z time="2015-05-01T19:48:36Z" level=info msg="using
inmemory layerinfo cache" instance.id=9ed6c43d-23ee-453f-9a4b-031fea646002
2015-05-01T19:48:36.303433991Z time="2015-05-01T19:48:36Z" level=info msg="Using
OpenShift Auth handler"
2015-05-01T19:48:36.303439084Z time="2015-05-01T19:48:36Z" level=info msg="listening
on :5000" instance.id=9ed6c43d-23ee-453f-9a4b-031fea646002

```

5.5. レジストリーメトリクスへのアクセス

OpenShift Container レジストリーは、[Prometheus メトリクス](#) のエンドポイントを提供します。Prometheus はスタンドアロンのオープンソースのシステムモニタリングおよびアラートツールキットです。

メトリクスは、レジストリーエンドポイントの `/extensions/v2/metrics` パスに公開されます。

手順

メトリクスクエリーの実行またはクラスターロールの使用という、メトリクスにアクセスするための2つの方法を使用できます。

メトリクスクエリー

1. 以下のようにメトリクスクエリーを実行します。

```

$ curl --insecure -s -u <user>:<secret> \ 1
https://image-registry.openshift-image-registry.svc:5000/extensions/v2/metrics | grep
imageregistry | head -n 20

```

出力例

```

# HELP imageregistry_build_info A metric with a constant '1' value labeled by major, minor,
git commit & git version from which the image registry was built.
# TYPE imageregistry_build_info gauge
imageregistry_build_info{gitCommit="9f72191",gitVersion="v3.11.0+9f72191-135-
dirty",major="3",minor="11+"} 1
# HELP imageregistry_digest_cache_requests_total Total number of requests without scope
to the digest cache.
# TYPE imageregistry_digest_cache_requests_total counter
imageregistry_digest_cache_requests_total{type="Hit"} 5
imageregistry_digest_cache_requests_total{type="Miss"} 24
# HELP imageregistry_digest_cache_scoped_requests_total Total number of scoped
requests to the digest cache.
# TYPE imageregistry_digest_cache_scoped_requests_total counter
imageregistry_digest_cache_scoped_requests_total{type="Hit"} 33
imageregistry_digest_cache_scoped_requests_total{type="Miss"} 44
# HELP imageregistry_http_in_flight_requests A gauge of requests currently being served by
the registry.
# TYPE imageregistry_http_in_flight_requests gauge
imageregistry_http_in_flight_requests 1
# HELP imageregistry_http_request_duration_seconds A histogram of latencies for requests
to the registry.

```

```
# TYPE imageregistry_http_request_duration_seconds summary
imageregistry_http_request_duration_seconds{method="get",quantile="0.5"} 0.01296087
imageregistry_http_request_duration_seconds{method="get",quantile="0.9"} 0.014847248
imageregistry_http_request_duration_seconds{method="get",quantile="0.99"} 0.015981195
imageregistry_http_request_duration_seconds_sum{method="get"} 12.260727916000022
```

- ① **<user>** は任意ですが、**<secret>** はレジストリー設定で指定された値と一致していなければなりません。

クラスターロール

1. メトリクスにアクセスするために必要なクラスターロールがない場合、これを作成します。

```
$ cat <<EOF | oc create -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: prometheus-scraper
rules:
- apiGroups:
  - image.openshift.io
  resources:
  - registry/metrics
  verbs:
  - get
EOF
```

2. このロールをユーザーに追加し、以下のコマンドを実行します。

```
$ oc adm policy add-cluster-role-to-user prometheus-scraper <username>
```

3. クラスターロールを使用してメトリクスにアクセスします。設定ファイルのメトリクスに対応する部分は以下のようになります。

```
openshift:
  version: 1.0
  metrics:
    enabled: true
...
```

5.6. 追加リソース

- プロジェクトの Pod が別のプロジェクトのイメージを参照できるようにする方法についての詳細は、[Pod の複数のプロジェクト間でのイメージの参照を許可する方法](#)について参照してください。
- **kubeadmin** は削除されるまでレジストリーにアクセスできます。詳細は、「[Removing the kubeadmin user](#)」を参照してください。
- アイデンティティプロバイダーの設定についての詳細は、「[Understanding identity provider configuration](#)」を参照してください。

第6章 レジストリーの公開

デフォルトで、OpenShift Container Platform レジストリーのセキュリティは、TLS 経由でトラフィックを送信できるようにクラスターのインストール時に保護されます。以前のバージョンの OpenShift Container Platform とは異なり、レジストリーはインストール時にクラスター外に公開されません。

6.1. セキュアなレジストリーの手動による公開

クラスター内から OpenShift Container Platform レジストリーにログインするのではなく、外部からレジストリーにアクセスできるように、このレジストリーをルートに公開します。この方法を使うと、ルートアドレスを使ってクラスターの外部からレジストリーにログインし、ルートのホストを使ってイメージにタグ付けしたり、イメージをプッシュしたりできます。

前提条件:

- 以下の前提条件は自動的に実行されます。
 - レジストリー Operator のデプロイ。
 - Ingress Operator のデプロイ。

手順

`configs.imageregistry.operator.openshift.io` リソースで `DefaultRoute` パラメーターを使用するか、またはカスタムルートを使用してルートを開示することができます。

`DefaultRoute` を使用してレジストリーを開示するには、以下を実行します。

1. `DefaultRoute` を `True` に設定します。

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --patch '{"spec": {"defaultRoute":true}}' --type=merge
```

2. `podman` でログインします。

```
$ HOST=$(oc get route default-route -n openshift-image-registry --template='{{ .spec.host }}')
```

```
$ podman login -u kubeadmin -p $(oc whoami -t) --tls-verify=false $HOST 1
```

- 1** `--tls-verify=false` は、ルートのクラスターのデフォルト証明書が信頼されない場合に必要になります。Ingress Operator で、信頼されるカスタム証明書をデフォルト証明書として設定できます。

カスタムルートを使用してレジストリーを開示するには、以下を実行します。

1. ルートの TLS キーでシークレットを作成します。

```
$ oc create secret tls public-route-tls \
  -n openshift-image-registry \
  --cert=</path/to/tls.crt> \
  --key=</path/to/tls.key>
```


この手順はオプションです。シークレットを作成しない場合、ルートは Ingress Operator からデフォルトの TLS 設定を使用します。

2. レジストリー Operator では、以下ようになります。

```
spec:
  routes:
  - name: public-routes
    hostname: myregistry.mycorp.organization
    secretName: public-route-tls
  ...
```



注記

レジストリーのルートのカスタム TLS 設定を指定している場合は **secretName** のみを設定します。