# Cisco HyperFlex CSI インターフェイス（Kubernetes 用）の構成



## 前提条件

次の前提条件は、Cisco HyperFlex CSI インテグレーションの構成の前に満たす必要があります。

HyperFlexクラスタ上：

- Cisco HyperFlex クラスタがインストールされ、HX 5.0(1a) 以降を実行している。
- インストールする前に、HX ConnectでiSCSIネットワークを設定します。iSCSIネットワークの設定の詳細については、『』『Cisco HyperFlex Data Platform Administration Guide、Release 5.0』を参照してください。

Kubernetesクラスタ上：

- HXCSI で進める前に、すべての Kubernetes ノードに `2.0.874-5ubuntu2.10` 以降のバージョンの `open-iscsi` パッケージがインストールされていることを確認します。これを行うには、`$iscsid-version` コマンドを実行します。

  `open-iscsi バージョン 2.0.874-5ubuntu2.10` をインストールするには、`apt-get install -y open-iscsi=2.0.874-5ubuntu2.10` コマンドを実行します。

- 各 Kubernetesノードに HX iSCSI ネットワーク上の専用インターフェイスがあるか、またはHX iSCSI ネットワークへのルーティング可能なアクセスがあることを確認します。
- `iscsid` がシステムのリブート時に開始されるようにするには、次のコマンドを実行します。

  ```
  sudo systemctl enable iscsid
  ```

`iscsid` ステータスが表示されます（例）。

```
$ sudo systemctl status iscsid

iscsid.service：iSCSIイニシエータ デーモン（iscid）

Loaded: loaded (/lib/systemd/system/iscid.service; enabled; vendor preset: enabled
```

- 各 Kubernetes プライマリ（「マスター」とも呼ばれる）ホストシステムに「`/etc/kubernetes/manifests/kube-controller-manager.yaml`」ファイル（`--disable-attach-detach-reconcile-sync=true` を含む）が含まれていることを確認します。
- ファイルの `-command` セクションに次のテキストを追加します。
  `--disable-attach-detach-reconcile-sync=true`

# 管理者ホスト

このガイドでは、管理者ホストは kubectl コマンドなどを Kubernetes クラスタに対して実行するための Linux ベースのシステムのことです。これは通常、Kubernetes クラスタの一部ではない別のシステム（VM）ですが、別のシステム（VM）をインストール/管理する必要がない場合は、管理者ホストとして Kubernetes ノードの 1 つを使用できます。

# Cisco HyperFlex CSI Integration for Kubernetes のインストール

Cisco HyperFlex CSI Integration をインストールするには、次の手順を記載されている順序で実行します。

## Cisco HyperFlex CSI バンドルのダウンロード

Cisco HyperFlex CSI バンドル（ファイル）をダウンロードするには、次の手順を実行します。

**ステップ 1** https://software.cisco.com に移動する

**ステップ 2** Cisco ID のクレデンシャルを使用してログインします。

**ステップ 3** **[ダウンロードとアップグレード（Download and Upgrade）]** セクションで、**[ソフトウェアのダウンロード（Software Download）]** を選択します。

**ステップ 4** **[製品の選択（Select a Product）]** 検索フィールドに、**`HyperFlex HX Data Platform`** と入力し、**Enter** をクリックします。

**ステップ 5** 左側の [リリース（Release）] ナビゲーション ウィンドウを使用して、クラスタで実行されている HyperFlex データ プラットフォーム ソフトウェアのバージョンを選択します。

Cisco HyperFlex Data Platform リリース 4.5(x) 以降では、Cisco HyperFlex CSI インテグレーションが必要です。

**ステップ 6** メインナビゲーション ペインで、「Cisco HyperFlex Kubernetes Container Storage Interface（HX-CSI）bundle（tar.gz）」ファイルをローカル マシンにダウンロードします。

以降、Cisco HyperFlex Kubernetes Container Storage Interface（HX-CSI）バンドル（tar.gz）ファイルを「Cisco HyperFlex CSI バンドル」と呼びます。

**ステップ 7** 管理者ホストで、`hxcsi` という名前の新しいディレクトリを作成します。

**例**：

```
administrator-host:~$ mkdir hxcsi
```

**ステップ 8** セキュア コピー（scp）またはその他の優先ファイル転送方式を使用して、ダウンロードしたCisco HyperFlex CSI バンドルをローカル マシンから管理者ホストの「hxcsi」ディレクトリに転送（移動またはコピー）します。結果は次のようになります。

**例**：

```
administrator-host:hxcsi$ ls

hxcsi-1.2.2a-626.tar.gz
```

### 次のタスク

Cisco HyperFlex CSI バンドルのオープンと抽出

# Cisco HyperFlex CSI バンドルのオープンと抽出

Cisco HyperFlex CSI バンドルを開くには、次の手順を実行します。

### 始める前に

Cisco HyperFlex CSI バンドルをダウンロードします。

`tar` コマンドを使用して、HyperFlex CSI バンドル（.tar.gz ファイル）をアーカイブ解除します。

**例**：

```
administrator-host:hxcsi$ tar -xf hxcsi-1.2.2a-626.tar.gz
```

完了すると、次のディレクトリ構造が存在します。

- **サンプル（ディレクトリ）**：HXCSIインテグレーションを使用するためのサンプルYAMLファイルが含まれています。

- **イメージ（ディレクトリ）**：HXCSI インテグレーション用の HXCSI docker コンテナ イメージが含まれます。これには、Provisioner、Attacher、Node-driver、および Resizer の基本 CSI イメージも含まれます。
- **setup（ディレクトリ）**：HXCSI 統合を展開するためのセットアップ スクリプトが含まれています。
- **support（ディレクトリ）**：デバッグに役立つログを収集するためのスクリプトが含まれています。
- **hxcsi-1.2.1.tgz（ファイル）**：これは、このリリースの HXCSI の HELM チャートパッケージ です。

### 例

```
administrator-host:hxcsi$ ls -l
total 133196
-rw-r--r--  1 ubuntu ubuntu      6791 May 10 11:23 hxcsi-1.2.1.tgz
drwxr-xr-x  2 ubuntu ubuntu      4096 May 10 11:23 support
drwxr-xr-x  2 ubuntu ubuntu      4096 May 10 11:23 setup
drwxr-xr-x  2 ubuntu ubuntu      4096 May 10 11:23 images
drwxr-xr-x 11 ubuntu ubuntu      4096 May 10 11:23 examples
```

### 次のタスク

Cisco HyperFlex CSI コンテナ イメージのアップロード

# Cisco HyperFlex CSI コンテナ イメージのアップロード

Cisco HyperFlex CSI インテグレーション コンポーネントは、Cisco HyperFlex CSI バンドルの「images」ディレクトリで提供される単一のコンテナ イメージから展開されます。hxcsi コンテナ イメージは、同じディレクトリ内の他の 4 つのベース CSI イメージを活用します。コンテナ イメージを展開する前に、Kubernetes クラスタ ワーカー ノードで実行されている Docker にアクセス可能な場所にコンテナ イメージを移動します。

## 各 Kubernetes ワーカー ノードに直接 Cisco HyperFlex CSI コンテナ イメージを手動でインポート

Cisco HyperFlex CSI コンテナ イメージを各 Kubernetes ワーカー ノードに直接追加するには、次の手順を実行します。

### 始める前に

Cisco HyperFlex CSI バンドルを開きます。

**ステップ 1** 管理者ホストで、「images」ディレクトリにある Cisco HyperFlex CSI コンテナ イメージ（.tar）ファイルを各 Kubernetes ワーカー ノードの /tmp ディレクトリにコピーします。

例：

```
administrator-host:hxcsi$ scp ./images/hxcsi-1.2.3a-659.tar k8s-worker1:/tmp

administrator-host:hxcsi$ scp ./images/hxcsi-1.2.3a-659.tar k8s-worker2:/tmp

administrator-host:hxcsi$ scp ./images/hxcsi-1.2.3a-659.tar k8s-workerN:/tmp
```

**ステップ 2** 他の基本 CSI コンテナイメージファイルを各 Kubernetes ノードにコピーします。

`csi-attacher-3.0.2-cisco1.tar`、`csi-node-driver-registrar-2.0.1-cisco1.tar`、`csi-resizer-1.0.1-cisco1.tar`、`csi-provisioner-2.0.4-cisco1.tar`

**ステップ 3** 各 Kubernetes ワーカーノードで、`docker load --input` コマンドを使用して Cisco HyperFlex CSI コンテナイメージをロードします。

例：

```
k8s-worker1:/tmp# docker load –input ./hxcsi-1.2.3a-659.tar
Loaded image: hxcsi:hxcsi-1.2.3a-659

k8s-worker2:/tmp# docker load –input ./hxcsi-1.2.3a-659.tar Loaded image: hxcsi:hxcsi-1.2.2a-626

k8s-workerN:/tmp# docker load –input ./hxcsi-1.2.3a-659.tar Loaded image: hxcsi:hxcsi-1.2.2a-626
```

**ステップ 4** Docker によって他の基本 CSI コンテナイメージファイルが各 Kubernetes ノードにロードされます。

`csi-attacher-3.2.1-cisco1.tar`、`csi-node-driver-registrar-2.2.0-cisco1.tar`、`csi-resizer-1.2.0-cisco1.tar`、`csi-provisioner-2.2.1-cisco1.tar`

### 次のタスク

Cisco HyperFlex CSI をインストールします。

# hxcsi-setup ユーティリティを使用した HXCSI の展開

Cisco HyperFlex CSI インテグレーションを展開するには、`hxcsi-setup` スクリプトを実行する必要があります。`hxcsi-setup` スクリプトは「setup」ディレクトリにあり、必要な YAML ファイルまたはヘルム チャートを自動的に生成して、Kubernetes クラスタに適用（送信）して、Cisco HyperFlex CSI コンポーネントを展開します。

次の表に、`hxcsi-setup` コマンドで指定できるパラメータを示します。

表 *1 : hxcsi-setup* のパラメータ

| パラメータ | 必須またはオプション | 説明 |
|---|---|---|
| `client-id` | オプション | テナントのクライアント ID。<br>（注） 複数の Kubernetes クラスタを作成して、同じ HX クラスタからストレージを要求できます。「clientId」パラメータは、これらのクライアント/テナントそれぞれのストレージ割り当ての分離に役立ちます。 |
| `-cluster-name` | Required | この特定の Kubernetes クラスタを一意に識別する名前を指定します。 |
| `-helm-chart` | オプション | ヘルムインストールのヘルム チャートを生成します（デフォルトは YAML ファイルを生成します） |
| `-hx-csi-image string` | Required | Cisco HyperFlex CSI コンテナ イメージの名前と場所。これにより、Cisco HyperFlex CSI コンテナ イメージを取得する場所が Kubernetes に通知されます。<br>[1] |
| `-iscsi-url string` | Required | HyperFlex クラスタの eth-iscsi1:0 インターフェイスの HyperFlex iSCSI クラスタ IP アドレス。詳細については、前提条件 （1 ページ）を参照してください。 |
| `-output-dir string` | オプション | 出力ディレクトリ（デフォルトは「./hxcsi-deploy/」） |
| `-password string` | Required<br>（最初に入力しない場合は、プロンプトが表示されます） | HX クラスタ API へのパスワード |
| `-token string` | オプション | サービス認証トークン。hxcsi-setup を呼び出す前に、アウトオブバンドでトークンを作成できます。 |

| パラメータ | 必須またはオプション | 説明 |
|---|---|---|
| `-url string` | Required | クラスタ管理IPアドレス。このIPはボリューム プロビジョニングとして使用されます。 |
| `-username string` | Required | HX クラスタ API のユーザ名（つまり、「admin」） |
| `-docker-registry` | オプション | Docker レジストリ名（例: mydockerhub.com/hx-docker） |

[1] Cisco HyperFlex CSI コンテナイメージが各 Kubernetes ワーカーノードの Docker に直接インポートされた場合、このパラメータの形式は **<repository_name>:<tag>** のように入力する必要があります。

### 始める前に

Cisco HyperFlex CSI コンテナイメージとベース CSI コンテナイメージをアップロードします。

---

管理者ホストで、「setup」ディレクトリで `hxcsi-setup` コマンドを使用して、必要な Cisco HyperFlex CSI 展開ファイルを作成します。

（注） **hxcsi-setup** コマンドを実行する前に、url および iscsi-url パラメータで指定された IP が Kubernetes ノードから到達可能であることを確認します。

**例**：

**すべてのイメージが dockerhub にアップロードされた場合**：次の例は、すべてのイメージが各 Kubernetes ノードからアクセス可能な dockerhub にアップロードされた場合を示しています。イメージ名は `hxcsi`、タグ名は `hxcsi-1.2.3a-659` です。

```
administrator-host:hxcsi$ ./setup/hxcsi-setup -cluster-name demo-hxcsi -clientId
 demo-client1 -hx-csi-image  hxcsi-1.2.3a-659 -iscsi-url 10.2.17.18 -url 10.2.17.13
-username admin -docker-registry dockerhub.cisco.com/hx-dev-docker

password for [admin] at [10.2.17.13]: *******
wrote config to hxcsi-deploy/hxcsi-config.yaml
wrote config to hxcsi-deploy/csi-attacher-hxcsi.yaml
wrote config to hxcsi-deploy/csi-nodeplugin-hxcsi.yaml
wrote config to hxcsi-deploy/csi-provisioner-hxcsi.yaml
wrote config to hxcsi-deploy/csi-attacher-rbac.yaml
wrote config to hxcsi-deploy/csi-nodeplugin-rbac.yaml
wrote config to hxcsi-deploy/csi-provisioner-rbac.yaml
wrote config to hxcsi-deploy/csi-resizer-rbac.yaml
wrote config to hxcsi-deploy/csi-resizer-hxcsi.yaml
```

**例**：

**すべてのイメージが各ノードにローカルにドッカーアップロードされている場合**：次に、各ノードにアップロードされている Cisco HyperFlex CSI コンテナイメージの展開例を示します。イメージ名は `hxcsi`、タグ名は `hxcsi-1.2.3a-659` です。この使用例は、中央の場所からイメージを取得するために使用される dockerhub がない場合に適用されます。

```
administrator-host:hxcsi$ ./setup/hxcsi-setup -cluster-name demo-hxcsi -clientId demo-client1
-hx-csi-image hxcsi:hxcsi-1.2.3a-659  -iscsi-url 10.2.17.18 -url 10.2.17.13 -username admin

password for [admin] at [10.2.17.13]: *******
wrote config to hxcsi-deploy/hxcsi-config.yaml
wrote config to hxcsi-deploy/csi-attacher-hxcsi.yaml
wrote config to hxcsi-deploy/csi-nodeplugin-hxcsi.yaml
wrote config to hxcsi-deploy/csi-provisioner-hxcsi.yaml
wrote config to hxcsi-deploy/csi-attacher-rbac.yaml
wrote config to hxcsi-deploy/csi-nodeplugin-rbac.yaml
wrote config to hxcsi-deploy/csi-provisioner-rbac.yaml
wrote config to hxcsi-deploy/csi-resizer-rbac.yaml
wrote config to hxcsi-deploy/csi-resizer-hxcsi.yaml
```

### 次のタスク

Cisco HyperFlex CSI コンポーネントの展開

# Cisco HyperFlex CSI コンポーネントを使用した HXCSI の展開

hxcsi-setup スクリプトを実行し、Cisco HyperFlex CSI 展開ファイルを生成すると、新しい「hxcsi-deploy」ディレクトリが管理者ホストに作成されます。

```
root@administrator-host:hxcsi$ ls
examples  hxcsi-1.2.2a-626.tar.gz  hxcsi-1.2.1.tgz  hxcsi-deploy  images  setup  support
```

### 始める前に

Cisco HyperFlex CSI 展開ファイルを作成します。

**ステップ 1** 管理者ホストで `kubectl create -f` コマンドを使用して、Cisco HyperFlex CSI コンポーネントを展開します。

**例**：

```
administrator-host:hxcsi$ kubectl create -f ./hxcsi-deploy/
service/csi-attacher-hxcsi created
statefulset.apps/csi-attacher-hxcsi created
serviceaccount/csi-attacher created
clusterrole.rbac.authorization.k8s.io/external-attacher-runner created
clusterrolebinding.rbac.authorization.k8s.io/csi-attacher-role created
daemonset.apps/csi-nodeplugin-hxcsi created
serviceaccount/csi-nodeplugin created
clusterrole.rbac.authorization.k8s.io/csi-nodeplugin created
clusterrolebinding.rbac.authorization.k8s.io/csi-nodeplugin created
service/csi-provisioner-hxcsi created
statefulset.apps/csi-provisioner-hxcsi created
serviceaccount/csi-provisioner created
clusterrole.rbac.authorization.k8s.io/external-provisioner-runner created
clusterrolebinding.rbac.authorization.k8s.io/csi-provisioner-role created
deployment.apps/csi-resizer-hxcsi created
serviceaccount/csi-resizer created
clusterrole.rbac.authorization.k8s.io/external-resizer-runner created
clusterrolebinding.rbac.authorization.k8s.io/csi-resizer-role created
role.rbac.authorization.k8s.io/external-resizer-cfg created
```

```
rolebinding.rbac.authorization.k8s.io/csi-resizer-role-cfg created
secret/hxcsitoken created
configmap/hxcsi-config created
```

**ステップ 2** 管理者ホストで `kubectl get pods` コマンドを使用して、HXCSI コンポーネントが展開され、ステータスが `[実行中（Running）]` であることを確認します。

**例**：

（注） 各 Kubernetes ワーカー ノードに対して、`csi-attacher-hxcsi` ポッドの 2 つのインスタンス、`csi-provisioner-hxcsi` ポッドの2つのインスタンス、`csi-resizer-hxcsi` ポッドの 2 つのインスタンス、および `csi-nodeplugin-hxcsi` ポッドの 1 つのインスタンスが必要です。したがって、合計2つのKubernetesワーカーノードがある場合は、次の例に示すように、`csi-nodeplugin-hxcsi` ポッドの 2 つのインスタンスが表示されます。

```
administrator-host:hxcsi$ kubectl get pods
NAME                                READY    STATUS    RESTARTS     AGE
csi-attacher-hxcsi-0                 2/2     Running       0         37h
csi-attacher-hxcsi-1                 2/2     Running       0         37h
csi-nodeplugin-hxcsi-2nsfq           2/2     Running       2         37h
csi-nodeplugin-hxcsi-qjh9n           2/2     Running       2         37h
csi-provisioner-hxcsi-0              2/2     Running       0         37h
csi-provisioner-hxcsi-1              2/2     Running       0         37h
csi-resizer-hxcsi-0                  2/2     Running       0         37h
csi-resizer-hxcsi-1                  2/2     Running       0         37h
```

### 次のタスク

Cisco HyperFlex CSI ストレージ クラスを作成します。

# HXCSI サンプルポッド

HXCSI パッケージには、ポッドを作成するためのいくつかの例が含まれています。

**表 *2 : HXCSI* パッケージのサンプル**

| # | ディレクトリ名 | 説明 |
|---|---|---|
| 1 | `sample-hxcsi` | `nginx` を実行しているポッド。基本的な例では、「`hxpvclaim-default`」という名前のPVCを作成します。 |
| 2 | `sample-hxcsi-csi-clone` | 「`sample-hxcsi`」からクローンを作成するポッド。「`dataSource`」は「`hxpvclaim-default`」として表示されます。 |
| 3 | `sample-hxcsi-ds` | 名前付きデータストア「`test-ds`」を使用するポッド |
| 4 | `sample-hxcsi-fs` | デフォルトのデータストアおよびファイル システム型「`xfs`」を使用するポッド |

| # | ディレクトリ名 | 説明 |
|---|---|---|
| 5 | sample-hxcsi-no-ds | デフォルトのデータストアとデフォルトのファイル システムを使用するポッド。 |
| 6 | sample-hxcsi-no-ds-clone | データストア名が指定されたサンプル「sample-hxcsi-no-ds」から複製します。 |
| 7 | sample-resize-block | ブロックボリュームのサイズを変更し、属性として allowVolumeExpansion: true を使用します。 |
| 8 | sample-resize-fs | デフォルト ファイルシステム「ext4」をサイズ変更します。<br>allowVolumeExpansion：true |
| 9 | sample-resize-clone | hxpvclaim-default-resize 「sample-resize-fs」サンプル ポッドからのクローン。 |

（注） ボリュームのサイズを変更するには、次の手順を実行します。

1. PVC の yaml ファイルのボリュームのサイズを変更します。
2. kubectl apply -f <pvc.yaml>コマンドを実行して、新しいサイズ設定を適用します。

# Cisco HyperFlex CSI ストレージ クラスの作成

コンポーネントが稼働したら、Cisco HyperFlex CSI インテグレーションを通じて開発者がストレージを使用できるストレージ クラスを作成する必要があります。

### 始める前に

Cisco HyperFlex CSI コンポーネントの展開

**ステップ 1** 管理者ホストで、「hxcsi-storage-class.yaml」という名前のファイルを次の内容で作成します。

**例**：

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
    name: csi-hxcsi-default
provisioner: csi-hxcsi
parameters:
    datastore: default-ds
    datastoreSize:"20000000000000”
```

上記のように、パラメータ セクションでデータストアの名前とサイズを指定できます。オプションで、これをデフォルトのストレージクラスにすることもできます。つまり、使用する他のストレージクラスを指定しない永続ボリューム クレームに対しては、Cisco HyperFlex CSI ストレージ インテグレーションがデフォルトで使用されます。Cisco HyperFlex CSI ストレージ クラスをデフォルトのストレージ クラスにする場合は、「hxcsi-storage-class.yaml」ファイルに次の内容が含まれている必要があります。

**例**：

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
    name: csi-hxcsi-default
    annotations:
        storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi-hxcsi
parameters:
```

（注） データストアがまだ存在しない場合は、新しいデータストアが作成されます。データストア名を指定しない場合、「iscsiDs」という名前のデフォルトのデータストアが作成されます。

（注） 作成するボリュームよりも大きいデータストアを常に作成します。

**ステップ 2** 管理者ホストで kubectl create -f コマンドを使用して、Cisco HyperFlex CSI ストレージ クラスを作成します。

**例**：

```
root@administrator-host:hxcsi$ kubectl create -f ./hxcsi-storage-class.yaml

storageclass.storage.k8s.io/csi-hxcsi-default created
```

### ボリューム サイズ変更のストレージ クラスの例

**例**：

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
    name: csi-hxcsi-default-resize
provisioner: csi-hxcsi
parameters:
          datastore: default-ds
          datastoreSize:"20000000000000"
allowVolumeExpansion: true
```

ボリュームのサイズ変更では、このストレージクラスに対してプロビジョニングされたボリュームのみがサイズ変更をサポートすることに注意してください。ボリュームの実際のサイズを変更するには、PVC 仕様を編集して新しいサイズに変更する必要があります。たとえば、PVC YAMLファイルを編集し、kubectl apply -f<pvc-yaml> を実行します。

### ファイル システムのサンプル ストレージ クラス

**例**：

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
    name: csi-hxcsi-default-fs
provisioner: csi-hxcsi
parameters:
    fsType: xfs
```

注：デフォルトのファイル システムは「ext4」です。

### 次のタスク

Cisco HyperFlex CSI ストレージ クラスの作成を確認します。

# Verifying Cisco HyperFlex CSI ストレージ クラスの作成

ストレージ クラスの作成を確認するには、次の手順を実行します。

（注） Cisco HyperFlex CSIストレージクラスをデフォルトとして設定する場合は、ストレージ クラス名の横に「（default）」が表示されていることを確認します。

### 始める前に

Cisco HyperFlex CSI ストレージ クラスを作成します。

管理者ホストで kubectl get sc コマンドを使用して、Cisco HyperFlex CSI ストレージ クラスが作成されたことを確認します。

例：

```
root@administrator-host:hxcsi$ kubectl get sc
NAME    PROVISIONER  AGE
csi-hxcsi (default) csi-hxcsi  67s
```

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。