はじめに
Astra Control Center

NetApp
November 21, 2023

目次

はじめに

= :allow-uri-read:

Astra Control Center の要件

運用環境、アプリケーションクラスタ、アプリケーション、ライセンス、 Web ブラウザの準備ができているかどうかを検証します。

- [運用環境の要件]
- [サポートされるストレージバックエンド]
- [インターネットにアクセスできます]
- [使用許諾]
- オンプレミス Kubernetes クラスタへの入力
- [ネットワーク要件]
- サポートされている Web ブラウザ
- [アプリケーションクラスタのその他の要件]
- Google Anthosクラスタの要件
- VMware Tanzu Kubernetes Grid クラスタの要件

運用環境の要件

Astra Control Centerは、次のタイプの運用環境で検証済みです。

- Kubernetes 1.22を搭載したCisco IKS
- Google Anthos 1.11または1.12（を参照 Google Anthosクラスタの要件)
- Rancher Kubernetes Engine（RKE）：
  - Rancher 2.6.5および2.6.6を備えたRKE 1.3.12
  - Rancher 2.6.8を備えたRKE 1.3.13
  - Rancher 2.6.5および2.6.6を搭載したRKE 2（v1.3.6 + rke2r1）
  - Rancher 2.6.8を搭載したRKE 2（v1.24.x）
- Red Hat OpenShift Container Platform 4.8~4.11
- アップストリームKubernetes 1.23から1.25（Kubernetes 1.25にAstra Trident 22.10以降が必要）
- VMware Tanzu Kubernetes Grid：（を参照してください VMware Tanzu Kubernetes Grid クラスタの要件)
  - VMware Tanzu Kubernetes Grid 1.5
  - VMware Tanzu Kubernetes Grid Integrated Edition 1.13および1.14

Astra Control Center をホストするオペレーティングシステムが、環境の公式ドキュメントに記載されている基本的なリソース要件を満たしていることを確認します。Astra Control Center では、環境のリソース要件に加え、次のリソースが必要です。

コンポーネント	要件
CPU拡張	ホスティング環境のすべてのノードのCPUでAVX拡張機能を有効にする必要があります。
ストレージバックエンドの容量	500GB以上の容量があります
ワーカーノード	少なくとも 3 つのワーカーノードがあり、それぞれ 4 つの CPU コアと 12GB の RAM が搭載されています
FQDN アドレス	Astra Control Center の FQDN アドレス
Astra Trident	Kubernetes 1.25クラスタ用にSnapMirrorベースのアプリケーションレプリケーションAstra 22.07以降がインストールされているAstra Trident 22.01以降がインストールされ、Astra Trident 22.10以降がインストールされている（Kubernetes 1.25にアップグレードする前にAstra Trident 22.10にアップグレードする必要がある）

これらの要件は、運用環境で実行されている唯一のアプリケーションが Astra Control Center であることを前提としています。環境で追加のアプリケーションを実行している場合は、それに応じてこれらの最小要件を調整します。

- * イメージレジストリ * ： Astra Control Center ビルドイメージをプッシュできる、既存のプライベート Docker イメージレジストリが必要です。イメージをアップロードするイメージレジストリの URL を指定する必要があります。
- * Astra Trident / ONTAP 構成*：
  - クラスタにTridentストレージクラスを1つ以上設定する必要があります。デフォルトのストレージクラスが設定されている場合は、そのストレージクラスがデフォルトで指定された唯一のストレージクラスであることを確認します。
  - クラスタ内のワーカーノードで適切なストレージドライバが設定されていることを確認します。これにより、ポッドがバックエンドストレージと通信できるようになります。Astra Control Center は、Astra Trident が提供する次の ONTAP ドライバをサポートしています。
    - ONTAP - NAS
    - ONTAP - SAN
    - ONTAP-SANエコノミー（アプリケーションレプリケーションではサポートされていません）

サポートされるストレージバックエンド

Astra Control Center は、次のストレージバックエンドをサポートします。

- NetApp ONTAP 9.5以降のAFF 、FAS 、ASA システム
- NetApp ONTAP 9.8以降のAFF 、FAS 、ASA システム（SnapMirrorベースのアプリケーションレプリケーション用
- NetApp ONTAP Select 9.5以降
- NetApp ONTAP Select 9.8以降（SnapMirrorベースのアプリケーションレプリケーション用

- NetApp Cloud Volumes ONTAP 9.5以降

Astra Control Centerを使用するには、必要な機能に応じて、次のONTAP ライセンスがあることを確認します。

- FlexClone
- SnapMirror：オプション。SnapMirrorテクノロジを使用してリモートシステムにレプリケートする場合にのみ必要です。を参照してください "SnapMirrorのライセンス情報"。
- S3ライセンス：オプション。ONTAP S3バケットにのみ必要です

ONTAP システムに必要なライセンスがあるかどうかを確認するには、を参照してください "ONTAP ライセンスを管理します"。

インターネットにアクセスできます

インターネットに外部からアクセスできるかどうかを確認する必要があります。この処理を行わないと、NetApp Cloud Insights からの監視データや指標データの受信や、へのサポートバンドルの送信など、一部の機能が制限される可能性があります "NetApp Support Site"。

使用許諾

Astra Control Center の全機能を使用するには、 Astra Control Center ライセンスが必要です。評価用ライセンスまたはフルライセンスをネットアップから取得する。アプリケーションとデータを保護するにはライセンスが必要です。を参照してください "Astra Control Centerの機能" を参照してください。

Astra Control Centerには、評価用ライセンスをお試しいただけます。このライセンスは、Astra Control Centerをダウンロードした日から90日間使用できます。登録すると、無償トライアルに登録できます "こちらをご覧ください"。

ライセンスをセットアップするには、を参照してください "90 日間の評価版ライセンスを使用する"。

ライセンスの機能の詳細については、を参照してください "ライセンス"。

ONTAP ストレージバックエンドに必要なライセンスの詳細については、を参照してください "サポートされるストレージバックエンド"。

オンプレミス **Kubernetes** クラスタへの入力

ネットワーク入力アストラコントロールセンターで使用するタイプを選択できます。デフォルトでは、 Astra Control Center は Astra Control Center ゲートウェイ（サービス / traefik ）をクラスタ全体のリソースとして展開します。また、お客様の環境でサービスロードバランサが許可されている場合は、 Astra Control Center でサービスロードバランサの使用もサポートされます。サービスロードバランサを使用する必要があり、設定していない場合は、MetalLBロードバランサを使用して外部IPアドレスを自動的にサービスに割り当てることができます。内部 DNS サーバ構成では、 Astra Control Center に選択した DNS 名を、負荷分散 IP アドレスに指定する必要があります。

> ロードバランサは、Astra Control CenterワーカーノードのIPアドレスと同じサブネットにあるIPアドレスを使用する必要があります。

> Tanzu Kubernetes GridクラスタでAstra Control Centerをホストしている場合は、を使用します `kubectl get nsxlbmonitors -A` 入力トラフィックを受け入れるように設定されたサービスモニタがすでにあるかどうかを確認するコマンド。MetalLB が存在する場合は、既存のサービスモニタが新しいロードバランサ設定を上書きするため、 MetalLB をインストールしないでください。

詳細については、を参照してください "ロードバランシング用の入力を設定します"。

ネットワーク要件

Astra Control Center をホストする運用環境は、次の TCP ポートを使用して通信します。これらのポートがファイアウォールを通過できることを確認し、 Astra ネットワークからの HTTPS 出力トラフィックを許可するようにファイアウォールを設定する必要があります。一部のポートでは、 Astra Control Center をホストする環境と各管理対象クラスタ（該当する場合はメモ）の両方の接続方法が必要です。

> Astra Control Center はデュアルスタック Kubernetes クラスタに導入でき、 Astra Control Center はデュアルスタック操作用に構成されたアプリケーションとストレージバックエンドを管理できます。デュアルスタッククラスタの要件の詳細については、を参照してください "Kubernetes のドキュメント"。

ソース	宛先	ポート	プロトコル	目的
クライアント PC	Astra Control Center の略	443	HTTPS	UI / API アクセス - Astra Control Center をホストしているクラスタと各管理対象クラスタの間で、このポートが双方向に開いていることを確認します
指標利用者	Astra Control Center ワーカーノード	9090	HTTPS	メトリックデータ通信 - 各管理対象クラスタが、アストラコントロールセンターをホストしているクラスタ上のこのポートにアクセスできることを確認します （双方向通信が必要）
Astra Control Center の略	Hosted Cloud Insights サービスの略	443	HTTPS	Cloud Insights 通信
Astra Control Center の略	Amazon S3 ストレージバケットプロバイダ	443	HTTPS	Amazon S3 ストレージ通信
Astra Control Center の略	NetApp AutoSupport	443	HTTPS	NetApp AutoSupport 通信

サポートされている Web ブラウザ

Astra Control Center は、最新バージョンの Firefox 、 Safari 、 Chrome をサポートし、解像度は 1280 x 720 以上です。

アプリケーションクラスタのその他の要件

次のAstra Control Center機能を使用する場合は、次の要件に注意してください。

- アプリケーションクラスタの要件： "クラスタ管理の要件"
  - アプリケーション要件の管理： "アプリケーション管理の要件"
  - アプリケーション・レプリケーションの追加要件： "レプリケーションの前提条件"

Google Anthosクラスタの要件

Google AnthosクラスタでAstra Control Centerをホストする場合、Google AnthosにはデフォルトでMetalLBロードバランサとIstio入力ゲートウェイサービスが含まれているため、インストール時にAstra Control Centerの一般的な入力機能を使用するだけで済みます。を参照してください "Astra Control Center を設定します" を参照してください。

VMware Tanzu Kubernetes Grid クラスタの要件

VMware Tanzu Kubernetes Grid （ TKG ）または Tanzu Kubernetes Grid Integrated Edition （ TKGi ）クラスタで Astra Control Center をホストする場合、次の点に注意してください。

- TKG または TKGi のデフォルト・ストレージ・クラス・エンフォースメントは、 Astra Control によって管理されるすべてのアプリケーション・クラスタで無効にします。これを行うには、を編集します `TanzuKubernetesCluster` ネームスペースクラスタ上のリソース。
- TKG または TKGi 環境に Astra Control Center を導入する際には、 Astra Trident の特定の要件に注意してください。詳細については、を参照してください "Astra Trident のドキュメント"。

> デフォルトの VMware TKG および TKGi 設定ファイルトークンの有効期限は、展開後 10 時間です。Tanzu ポートフォリオ製品を使用する場合は、 Astra Control Center と管理対象アプリケーションクラスタ間の接続の問題を回避するために、期限切れにならないトークンを含む Tanzu Kubernetes Cluster 構成ファイルを生成する必要があります。手順については、を参照してください "VMware NSX -T Data Center 製品ドキュメント"

次のステップ

を表示します "クイックスタート" 概要（ Overview ）：

Astra Control Center のクイックスタート

ここでは、Astra Control Centerの導入に必要な手順の概要を示します。各ステップ内のリンクから、詳細が記載されたページに移動できます。

1 **Kubernetes** クラスタの要件を確認

環境がこれらの要件を満たしていることを確認します。

- Kubernetesクラスタ*
- "環境が運用環境の要件を満たしていることを確認"
- "オンプレミスKubernetesクラスタでロードバランシングを行うための入力を設定する"

ストレージ統合

- "サポートされているAstra Tridentバージョンが環境に含まれていることを確認します"
- "ワーカーノードを準備します"
- "Astra Tridentストレージバックエンドを設定"
- "Astra Tridentストレージクラスを設定する"
- "Astra Tridentボリュームスナップショットコントローラをインストール"
- "ボリュームSnapshotクラスを作成します"
- ONTAP クレデンシャル*
- "ONTAP クレデンシャルを設定する"

2 **Astra Control Center**をダウンロードしてインストールします

次のインストールタスクを実行します。

- "NetApp Support Site 評価ダウンロードページからAstraコントロールセンターをダウンロードします"
- ネットアップライセンスファイルを入手します。
  - "Astra Control Centerを評価している場合は、評価用ライセンスファイルをダウンロードします"
  - "Astra Control Centerをすでに購入している場合は、ライセンスファイルを生成します"
- "Astra Control Center をインストールします"
- "追加のオプション設定手順を実行します"

3 いくつかの初期セットアップ作業を完了します

いくつかの基本的な作業を完了して開始します。

- "ライセンスを追加します"
- "クラスタ管理のための環境を準備します"
- "クラスタを追加"
- "ストレージバックエンドを追加します"
- "バケットを追加します"

4 Astra Control Center を使用

Astra Control Centerのセットアップが完了したら、次の手順を実行します。Astra Controlのユーザインターフェイス（UI）またはを使用できます "Astra Control API の略"。

- "アプリの管理"
- "アプリを保護します"：保護ポリシーを構成し、アプリケーションのレプリケーション、クローニング、移行を行います。
- "アカウントを管理"ユーザ、ロール、LDAP、クレデンシャルなど
- "必要に応じて、Cloud Insights に接続します"：システムの健全性に関する指標を表示します。

を参照してください。

- "Astra Control API の略"
- "Astra Control Center をアップグレードします"
- "Astra Controlのヘルプ"

インストールの概要

次の Astra Control Center のインストール手順のいずれかを選択して実行します。

- "標準の手順で Astra Control Center をインストールします"
- "（ Red Hat OpenShift を使用する場合） OpenShift OperatorHub を使用して Astra Control Center をインストールします"
- "Cloud Volumes ONTAP ストレージバックエンドに Astra Control Center をインストールします"

環境によっては、Astra Control Centerのインストール後に追加の設定が必要になる場合があります。

- "インストール後にAstra Control Centerを設定します"

標準の手順で **Astra Control Center** をインストールします

Astra Control Centerをインストールするには、NetApp Support Site からインストールバンドルをダウンロードし、次の手順を実行します。この手順を使用して、インターネット接続環境またはエアギャップ環境に Astra コントロールセンターをインストールできます。

その他のインストール手順

- * RedHat OpenShift OperatorHub *でのインストール：これを使用してください "代替手順" OperatorHubを使用してOpenShiftにAstra Control Centerをインストールするには、次の手順を実行します。
- * Cloud Volumes ONTAP バックエンドを使用してパブリッククラウドにインストール*：ユース "これらの手順に従います" Amazon Web Services（AWS）、Google Cloud Platform（GCP）、またはCloud Volumes ONTAP ストレージバックエンドを使用するMicrosoft AzureにAstra Control Centerをインストールするには、次の手順を実行します。

Astra Control Centerのインストールプロセスのデモについては、を参照してください "このビデオでは"。

必要なもの

- "インストールを開始する前に、 Astra Control Center の導入環境を準備します"。
- 使用環境でポッドセキュリティポリシーを設定または設定したい場合は、ポッドセキュリティポリシーと、それらがAstra Control Centerのインストールに与える影響について理解しておいてください。を参照してください "ポッドのセキュリティポリシーの制限事項を理解します"。
- すべての API サービスが正常な状態であり、使用可能であることを確認します。

```
kubectl get apiservices
```

- 使用するネットアップFQDNがこのクラスタにルーティング可能であることを確認します。つまり、内部 DNS サーバに DNS エントリがあるか、すでに登録されているコア URL ルートを使用しています。
- クラスタに証明書マネージャがすでに存在する場合は、いくつかの手順を実行する必要があります "事前に必要な手順" そのため、Astra Control Centerは独自の証明書マネージャのインストールを試みません。デフォルトでは、Astra Control Centerはインストール時に独自の証明書マネージャをインストールします。

このタスクについて

Astra Control Centerのインストールプロセスでは、次の作業を行うことができます。

- にAstraコンポーネントを取り付けます `netapp-acc` （またはカスタム名）ネームスペース。
- デフォルトのAstra Control Owner管理者アカウントを作成します。
- 管理ユーザのEメールアドレスとデフォルトの初期セットアップパスワードを設定します。このユーザには、UIへの初回ログインに必要なオーナーロールが割り当てられます。
- Astra Control Centerのすべてのポッドが稼働していることを確認します。
- Astra Control Center UIをインストールします。

> Astra Control Centerオペレータ（たとえば、 `kubectl delete -f astra_control_center_operator_deploy.yaml`）Astra Control Centerのインストール中または操作中はいつでも、ポッドを削除しないようにします。

手順

Astra Control Center をインストールするには、次の手順に従います。

- Astra Control Centerをダウンロードして展開します
- ネットアップAstra kubectlプラグインをインストール
- [イメージをローカルレジストリに追加します]
- [認証要件を持つレジストリのネームスペースとシークレットを設定します]
- Astra Control Center オペレータを設置します
- Astra Control Center を設定します
- Astra Control Center とオペレータのインストールを完了します
- [システムステータスを確認します]
- [ロードバランシング用の入力を設定します]
- Astra Control Center UI にログインします

Astra Control Centerをダウンロードして展開します

1. にアクセスします "Astra Control Center評価ダウンロードページ" をクリックしますNetApp Support Site。
2. Astra Control Centerを含むバンドルをダウンロードします (`astra-control-center-[version].tar.gz`)。
3. （推奨ですがオプション）Astra Control Centerの証明書と署名のバンドルをダウンロードします (`astra-control-center-certs-[version].tar.gz`)バンドルの署名を確認するには、次の手順を実行します。

   ```
   tar -vxzf astra-control-center-certs-[version].tar.gz
   ```

   ```
   openssl dgst -sha256 -verify certs/AstraControlCenter-public.pub
   -signature certs/astra-control-center-[version].tar.gz.sig astra-
   control-center-[version].tar.gz
   ```

   出力にはと表示されます `Verified OK` 検証が成功したあとに、

4. Astra Control Centerバンドルからイメージを抽出します。

   ```
   tar -vxzf astra-control-center-[version].tar.gz
   ```

ネットアップ**Astra kubectl**プラグインをインストール

NetApp Astra kubectlコマンドラインプラグインは、Astra Control Centerの導入とアップグレードに関連する一般的なタスクを実行する際に時間を節約します。

必要なもの

ネットアップでは、CPUアーキテクチャやオペレーティングシステム別にプラグインのバイナリを提供しています。このタスクを実行する前に、使用しているCPUとオペレーティングシステムを把握しておく必要が

あります。

手順

1. 使用可能なNetApp Astra kubectlプラグインのバイナリを表示し、オペレーティングシステムとCPUアーキテクチャに必要なファイルの名前をメモします。

> kubectlプラグインライブラリはtarバンドルの一部であり、フォルダに解凍されます `kubectl-astra`。

```
ls kubectl-astra/
```

2. 正しいバイナリを現在のパスに移動し、名前をに変更します `kubectl-astra`：

```
cp kubectl-astra/<binary-name> /usr/local/bin/kubectl-astra
```

イメージをローカルレジストリに追加します

1. コンテナエンジンに応じた手順を実行します。

Docker です

1. tarballのルートディレクトリに移動します。次のファイルとディレクトリが表示されます。

```
acc.manifest.bundle.yaml
acc/
```

2. Astra Control Centerのイメージディレクトリにあるパッケージイメージをローカルレジストリにプッシュします。を実行する前に、次の置換を行ってください `push-images` コマンドを実行します
   - <BUNDLE_FILE> をAstra Controlバンドルファイルの名前に置き換えます（`acc.manifest.bundle.yaml`）。
   - <MY_FULL_REGISTRY_PATH> をDockerリポジトリのURLに置き換えます。次に例を示します。 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
   - <MY_REGISTRY_USER> をユーザ名に置き換えます。
   - <MY_REGISTRY_TOKEN> をレジストリの認証済みトークンに置き換えます。

```
kubectl astra packages push-images -m <BUNDLE_FILE> -r
<MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
<MY_REGISTRY_TOKEN>
```

ポドマン

1. tarballのルートディレクトリに移動します。次のファイルとディレクトリが表示されます。

```
acc.manifest.bundle.yaml
acc/
```

2. レジストリにログインします。

```
podman login <YOUR_REGISTRY>
```

3. 使用するPodmanのバージョンに合わせてカスタマイズされた次のいずれかのスクリプトを準備して実行します。<MY_FULL_REGISTRY_PATH> を’サブディレクトリを含むリポジトリのURLに置き換えます

```
<strong>Podman 4</strong>
```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

```
<strong>Podman 3</strong>
```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

レジストリ設定に応じて、スクリプトが作成するイメージパスは次のようになります。 https://netappdownloads.jfrog.io/docker-astra-control-prod/netapp/astra/acc/22.11.0-82/image:version

認証要件を持つレジストリのネームスペースとシークレットを設定します

1. Astra Control Centerホストクラスタ用のKUBECONFIGをエクスポートします。

```
export KUBECONFIG=[file path]
```

インストールを完了する前に、KUBECONFIGがAstra Control Centerをインストールするクラスタを指していることを確認してください。KUBECONFIGには、1つのコンテキストのみを含めることができます。

2. 認証が必要なレジストリを使用する場合は、次の手順を実行する必要があります。

   a. を作成します `netapp-acc-operator` ネームスペース：

   ```
   kubectl create ns netapp-acc-operator
   ```

   対応：

   ```
   namespace/netapp-acc-operator created
   ```

   b. のシークレットを作成します `netapp-acc-operator` ネームスペース：Docker 情報を追加して次のコマンドを実行します。

   プレースホルダ `your_registry_path` 以前にアップロードした画像の場所と一致する必要があります（例： `[Registry_URL]/netapp/astra/astracc/22.11.0-82` ）。

   ```
   kubectl create secret docker-registry astra-registry-cred -n netapp-
   acc-operator --docker-server=[your_registry_path] --docker
   -username=[username] --docker-password=[token]
   ```

   回答例：

   ```
   secret/astra-registry-cred created
   ```

   シークレットの生成後にネームスペースを削除した場合は、ネームスペースを再作成し、ネームスペースのシークレットを再生成します。

   c. を作成します `netapp-acc` （またはカスタム名）ネームスペース。

   ```
   kubectl create ns [netapp-acc or custom namespace]
   ```

   回答例：

   ```
   namespace/netapp-acc created
   ```

d. のシークレットを作成します `netapp-acc` （またはカスタム名）ネームスペース。Docker 情報を追加して次のコマンドを実行します。

```
kubectl create secret docker-registry astra-registry-cred -n [netapp-acc or custom namespace] --docker-server=[your_registry_path] --docker-username=[username] --docker-password=[token]
```

応答

```
secret/astra-registry-cred created
```

Astra Control Center オペレータを設置します

1. ディレクトリを変更します。

```
cd manifests
```

2. Astra Control Centerオペレータ配置YAMLを編集します (`astra_control_center_operator_deploy.yaml`)を参照して、ローカルレジストリとシークレットを参照してください。

```
vim astra_control_center_operator_deploy.yaml
```

> 注釈付きサンプルYAMLは以下の手順に従います。

a. 認証が必要なレジストリを使用する場合は、のデフォルト行を置き換えます `imagePullSecrets: []` 次の条件を満たす場合：

```
imagePullSecrets:
- name: astra-registry-cred
```

b. 変更 `[your_registry_path]` をクリックします `kube-rbac-proxy` でイメージをプッシュしたレジストリパスへのイメージ 前の手順。

c. 変更 `[your_registry_path]` をクリックします `acc-operator-controller-manager` でイメージをプッシュしたレジストリパスへのイメージ 前の手順。

```
<strong>astra_control_center_operator_deploy.yaml</strong>
```

```
apiVersion: apps/v1
```

```
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: [your_registry_path]/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_INSTALLTIMEOUT
          value: 5m
        image: [your_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8081
          initialDelaySeconds: 15
          periodSeconds: 20
```

```
        name: manager
        readinessProbe:
          httpGet:
            path: /readyz
            port: 8081
          initialDelaySeconds: 5
          periodSeconds: 10
        resources:
          limits:
            cpu: 300m
            memory: 750Mi
          requests:
            cpu: 100m
            memory: 75Mi
        securityContext:
          allowPrivilegeEscalation: false
      imagePullSecrets: []
      securityContext:
        runAsUser: 65532
      terminationGracePeriodSeconds: 10
```

3. Astra Control Center オペレータをインストールします。

```
kubectl apply -f astra_control_center_operator_deploy.yaml
```

回答例：

```
namespace/netapp-acc-operator created
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.astra.
netapp.io created
role.rbac.authorization.k8s.io/acc-operator-leader-election-role created
clusterrole.rbac.authorization.k8s.io/acc-operator-manager-role created
clusterrole.rbac.authorization.k8s.io/acc-operator-metrics-reader
created
clusterrole.rbac.authorization.k8s.io/acc-operator-proxy-role created
rolebinding.rbac.authorization.k8s.io/acc-operator-leader-election-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-manager-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-proxy-
rolebinding created
configmap/acc-operator-manager-config created
service/acc-operator-controller-manager-metrics-service created
deployment.apps/acc-operator-controller-manager created
```

4. ポッドが実行中であることを確認します

```
kubectl get pods -n netapp-acc-operator
```

Astra Control Center を設定します

1. Astra Control Centerカスタムリソース（CR）ファイルを編集します (`astra_control_center.yaml`) アカウント、サポート、レジストリ、およびその他の必要な設定を行うには、次の手順を実行します。

```
vim astra_control_center.yaml
```

> 注釈付きサンプルYAMLは以下の手順に従います。

2. 次の設定を変更または確認します。

<code>accountName</code>

設定	ガイダンス（**Guidance**）	を入力します	例
`accountName`	を変更します `accountName` stringには、Astra Control Centerアカウントに関連付ける名前を指定します。アカウント名は1つだけです。	文字列	`Example`

<code>astraVersion</code>

設定	ガイダンス（**Guidance**）	を入力します	例
`astraVersion`	導入するAstra Control Centerのバージョン。この設定には値があらかじめ入力されているため、対処は不要です。	文字列	`22.11.0-82`

<code>astraAddress</code>

設定	ガイダンス（**Guidance**）	を入力します	例
`astraAddress`	を変更します `astraAddress` ブラウザで使用するFQDN（推奨）またはIPアドレスを指定して、Astra Control Centerにアクセスします。このアドレスは、データセンターでAstra Control Centerがどのように検出されるかを定義します。このアドレスは、完了時にロードバランサからプロビジョニングしたFQDNまたはIPアドレスと同じです "Astra Control Center の要件"。注：は使用しないでください `http://` または `https://` をクリックします。この FQDN をコピーしてで使用します 後の手順。	文字列	`astra.example.com`

<code>autoSupport</code>

このセクションで選択することで、ネットアップのプロアクティブサポートアプリケーション、NetApp Active IQ 、およびデータの送信先のどちらに参加するかが決まります。インターネット接続が必要です（ポート442）。サポートデータはすべて匿名化されます。

設定	使用	ガイダンス（**Guidance**）	を入力します	例
`autoSupport.enrolled`	または `enrolled` または `url` フィールドを選択する必要があります	変更 `enrolled` を選択しますAutoSupport `false` インターネットに接続されていないか、または保持されているサイト `true` 接続されているサイト用。の設定 `true` 匿名データをネットアップに送信し、サポートを目的として使用できるようにします。デフォルトの選択はです `false` およびは、サポートデータがネットアップに送信されないことを示します。	ブール値	`false` （デフォルト値）
`autoSupport.url`	または `enrolled` または `url` フィールドを選択する必要があります	このURLは匿名データの送信先を決定します。	文字列	`https://support.netapp.com/asupprod/post/1.0/postAsup`

<code>email</code>

設定	ガイダンス（**Guidance**）	を入力します	例
`email`	を変更します `email` デフォルトの初期管理者アドレスを表す文字列。この E メールアドレスをコピーしてで使用します 後の手順。このEメールアドレスは、最初のアカウントがUIにログインする際のユーザ名として使用され、Astra Controlのイベントが通知されます。	文字列	`admin@example.com`

<code>firstName</code>

設定	ガイダンス（**Guidance**）	を入力します	例
`firstName`	アストラアカウントに関連付けられている初期管理者の名前。ここで使用した名前は、初回ログイン後にUIの見出しに表示されます。	文字列	`SRE`

<code>LastName</code>

設定	ガイダンス（**Guidance**）	を入力します	例
`lastName`	アストラアカウントに関連付けられている初期管理者の姓です。ここで使用した名前は、初回ログイン後にUIの見出しに表示されます。	文字列	`Admin`

<code>imageRegistry</code>

このセクションで選択すると、Astraアプリケーションイメージ、Astra Control Center Operator、Astra Control Center Helmリポジトリをホストするコンテナイメージレジストリが定義されます。

設定	使用	ガイダンス（**Guidance**）	を入力します	例
`imageRegistry.name`	必須	でイメージをプッシュしたイメージレジストリの名前前の手順。使用しないでください `http://` または `https://` をレジストリ名に追加します。	文字列	`example.registry.com/astra`
`imageRegistry.secret`	に入力した文字列の場合は必須です `imageRegistry.name' requires a secret.` `IMPORTANT: If you are using a registry that does not require authorization, you must delete this` `secret` ラインの内側 `imageRegistry` または、インストールが失敗します。	イメージレジストリでの認証に使用するKubernetesシークレットの名前。	文字列	`astra-registry-cred`

<code>storageClass</code>

設定	ガイダンス（**Guidance**）	を入力します	例
`storageClass`	を変更します `storageClass` からの値 `ontap-gold` インストールに必要な別のTridentストレージクラスのリソースに移動します。コマンドを実行します `kubectl get sc` をクリックして、設定済みの既存のストレージクラスを確認します。Tridentベースのストレージクラスの1つをマニフェストファイルに入力する必要があります (`astra-control-center-<version>.manifest`）とをAstra PVSに使用します。設定されていない場合は、デフォルトのストレージクラスが使用されます。メモ：デフォルトのストレージクラスが設定されている場合は、デフォルトのアノテーションが設定されている唯一のストレージクラスであることを確認してください。	文字列	`ontap-gold`

<code>volumeReclaimPolicy</code>

設定	ガイダンス（**Guidance**）	を入力します	オプション（**Options**）
`volumeReclaimPolicy`	これにより、AstraのPVSの再利用ポリシーが設定されます。このポリシーをに設定しています `Retain` Astraが削除されたあとに永続的なボリュームを保持このポリシーをに設定しています `Delete` Astraが削除されたあとに永続的ボリュームを削除する。この値が設定されていない場合、PVSは保持されます。	文字列	• `Retain` （デフォルト値） • `Delete`

<code>ingressType</code>

設定	ガイダンス（**Guidance**）	を入力します	オプション（**Options**）
`ingressType`	次の入力タイプのいずれかを使用します。**`Generic`** (`ingressType: "Generic"`）（デフォルト）別の入力コントローラを使用している場合、または独自の入力コントローラを使用する場合は、このオプションを使用します。Astra Control Centerを導入したら、を設定する必要があります "入力コントローラ" URLを使用してAstra Control Centerを公開します。**`AccTraefik`** (`ingressType: "AccTraefik"`）入力コントローラを設定しない場合は、このオプションを使用します。これにより、Astra Control Centerが導入されます `traefik` Gateway as a Kubernetes LoadBalancer type serviceの略。Astra Control Centerは、タイプ「LoadBalancer」のサービスを使用します。（`svc/traefik` Astra Control Centerの名前空間）で、アクセス可能な外部IPアドレスが割り当てられている必要があります。お使いの環境でロードバランサが許可されていて、設定されていない場合は、MetalLBまたは別の外部サービスロードバランサを使用して外部IPアドレスをサービスに割り当てることができます。内部 DNS サーバ構成では、 Astra Control Center に選択した DNS 名を、負荷分散 IP アドレスに指定する必要があります。注：	文字列	• `Generic` （デフォルト値） • `AccTraefik`

<code>astraResourcesScaler</code>

設定	ガイダンス（**Guidance**）	を入力します	オプション（**Options**）
`astraResourcesScaler`	AstraeControlCenterリソース制限のスケーリングオプションデフォルトでは、Astra Control CenterはAstra内のほとんどのコンポーネントに対してリソース要求を設定して展開します。この構成により、アプリケーションの負荷と拡張性が高い環境では、Astra Control Centerソフトウェアスタックのパフォーマンスが向上します。ただし、小規模な開発またはテストクラスタを使用するシナリオでは、CRフィールドを使用します `astraResourcesScalar` に設定できます `Off`。これにより、リソース要求が無効になり、小規模なクラスタへの導入が可能になります。	文字列	• `Default` （デフォルト値） • `Off`

<code>crds</code>

このセクションで選択した内容によって、Astra Control CenterでのCRDの処理方法が決まります。

設定	ガイダンス（**Guidance**）	を入力します	例
`crds.externalCertManager`	外部証明書マネージャを使用する場合は、変更します `externalCertManager` 終了： `true`。デフォルト `false` Astra Control Centerが、インストール時に独自の証明書マネージャCRDをインストールするようにします。SSDはクラスタ全体のオブジェクトであり、クラスタの他の部分に影響を及ぼす可能性があります。このフラグを使用すると、これらのCRDがAstra Control Centerの外部にあるクラスタ管理者によってインストールおよび管理されることをAstra Control Centerに伝えることができます。	ブール値	`False` （デフォルト値）
`crds.externalTraefik`	デフォルトでは、Astra Control Centerは必要なTraefik CRDをインストールします。SSDはクラスタ全体のオブジェクトであり、クラスタの他の部分に影響を及ぼす可能性があります。このフラグを使用すると、これらのCRDがAstra Control Centerの外部にあるクラスタ管理者によってインストールおよび管理されることをAstra Control Centerに伝えることができます。	ブール値	`False` （デフォルト値）

```
<strong>astra_control_center.yaml</strong>
```

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
  volumeReclaimPolicy: "Retain"
  ingressType: "Generic"
  astraResourcesScaler: "Default"
  additionalValues: {}
  crds:
    externalTraefik: false
    externalCertManager: false
```

Astra Control Center とオペレータのインストールを完了します

1. 前の手順でまだ行っていない場合は、を作成します `netapp-acc` （またはカスタム）ネームスペース：

   ```
   kubectl create ns [netapp-acc or custom namespace]
   ```

   回答例：

   ```
   namespace/netapp-acc created
   ```

2. にAstra Control Centerをインストールします `netapp-acc` （またはカスタムの）ネームスペース：

   ```
   kubectl apply -f astra_control_center.yaml -n [netapp-acc or custom
   namespace]
   ```

   回答例：

```
astracontrolcenter.astra.netapp.io/astra created
```

システムステータスを確認します

kubectlコマンドを使用すると、システムステータスを確認できます。OpenShift を使用する場合は、同等の OC コマンドを検証手順に使用できます。

手順

1. すべてのシステムコンポーネントが正常にインストールされたことを確認します。

   ```
   kubectl get pods -n [netapp-acc or custom namespace]
   ```

   各ポッドのステータスがになっている必要があります `Running`。システムポッドが展開されるまでに数分かかることがあります。

回答例

```
NAME                                          READY   STATUS
RESTARTS        AGE
acc-helm-repo-76d8d845c9-ggds2                1/1     Running   0
14m
activity-6cc67ff9f4-z48mr                     1/1     Running   2
(8m32s ago)   9m
api-token-authentication-7s67v                1/1     Running   0
8m56s
api-token-authentication-bplb4                1/1     Running   0
8m56s
api-token-authentication-p2c9z                1/1     Running   0
8m56s
asup-6cdfbc6795-md8vn                         1/1     Running   0
9m14s
authentication-9477567db-8hnc9                1/1     Running   0
7m4s
bucketservice-f4dbdfcd6-wqzkw                 1/1     Running   0
8m48s
cert-manager-bb756c7c4-wm2cv                  1/1     Running   0
14m
cert-manager-cainjector-c9bb86786-8wrf5       1/1     Running   0
14m
cert-manager-webhook-dd465db99-j2w4x          1/1     Running   0
14m
certificates-68dff9cdd6-kcvml                 1/1     Running   2
(8m43s ago)   9m2s
certificates-68dff9cdd6-rsnsb                 1/1     Running   0
9m2s
cloud-extension-69d48c956c-2s8dt              1/1     Running   3
(8m43s ago)   9m24s
cloud-insights-service-7c4f48b978-7gvlh       1/1     Running   3
(8m50s ago)   9m28s
composite-compute-7d9ff5f68-nxbhl             1/1     Running   0
8m51s
composite-volume-57b4756d64-nl66d             1/1     Running   0
9m13s
credentials-6dbc55f89f-qpzff                  1/1     Running   0
11m
entitlement-67bfb6d7-gl6kp                    1/1     Running   4
(8m33s ago)   9m38s
features-856cc4dccc-mxbdb                     1/1     Running   0
9m20s
fluent-bit-ds-4rtsp                           1/1     Running   0
```

```
6m54s
fluent-bit-ds-9rqll                          1/1     Running   0
6m54s
fluent-bit-ds-w5mp7                          1/1     Running   0
6m54s
graphql-server-7c7cc49776-jz2kn              1/1     Running   0
2m29s
identity-87c59c975-9jpnf                     1/1     Running   0
9m6s
influxdb2-0                                  1/1     Running   0
13m
keycloak-operator-84ff6d59d4-qcnmc           1/1     Running   0
7m1s
krakend-cbf6c7df9-mdtzv                      1/1     Running   0
2m30s
license-5b888b78bf-plj6j                     1/1     Running   0
9m32s
login-ui-846b4664dd-fz8hv                    1/1     Running   0
2m24s
loki-0                                       1/1     Running   0
13m
metrics-facade-779cc9774-n26rw               1/1     Running   0
9m18s
monitoring-operator-974db78f-pkspq           2/2     Running   0
6m58s
nats-0                                       1/1     Running   0
13m
nats-1                                       1/1     Running   0
13m
nats-2                                       1/1     Running   0
13m
nautilus-7bdc7ddc54-49tfn                    1/1     Running   0
7m50s
nautilus-7bdc7ddc54-cwc79                    1/1     Running   0
9m36s
openapi-5584ff9f46-gbrdj                     1/1     Running   0
9m17s
openapi-5584ff9f46-z9mzk                     1/1     Running   0
9m17s
packages-bfc58cc98-lpxq9                     1/1     Running   0
8m58s
polaris-consul-consul-server-0               1/1     Running   0
13m
polaris-consul-consul-server-1               1/1     Running   0
13m
polaris-consul-consul-server-2               1/1     Running   0
```

```
13m
polaris-keycloak-0                          1/1     Running   3
(6m15s ago)   6m56s
polaris-keycloak-1                          1/1     Running   0
4m22s
polaris-keycloak-2                          1/1     Running   0
3m41s
polaris-keycloak-db-0                       1/1     Running   0
6m56s
polaris-keycloak-db-1                       1/1     Running   0
4m23s
polaris-keycloak-db-2                       1/1     Running   0
3m36s
polaris-mongodb-0                           2/2     Running   0
13m
polaris-mongodb-1                           2/2     Running   0
13m
polaris-mongodb-2                           2/2     Running   0
12m
polaris-ui-5ccff47897-8rzgh                 1/1     Running   0
2m33s
polaris-vault-0                             1/1     Running   0
13m
polaris-vault-1                             1/1     Running   0
13m
polaris-vault-2                             1/1     Running   0
13m
public-metrics-6cb7bfc49b-p54xm             1/1     Running   1
(8m29s ago)   9m31s
storage-backend-metrics-5c77994586-kjn48    1/1     Running   0
8m52s
storage-provider-769fdc858c-62w54           1/1     Running   0
8m54s
task-service-9ffc484c5-kx9f4                1/1     Running   3
(8m44s ago)   9m34s
telegraf-ds-bphb9                           1/1     Running   0
6m54s
telegraf-ds-rtsm2                           1/1     Running   0
6m54s
telegraf-ds-s9h5h                           1/1     Running   0
6m54s
telegraf-rs-lbpv7                           1/1     Running   0
6m54s
telemetry-service-57cfb998db-zjx78          1/1     Running   1
(8m40s ago)   9m26s
tenancy-5d5dfbcf9f-vmbxh                    1/1     Running   0
```

```
9m5s
traefik-7b87c4c474-jmgp2                    1/1     Running   0
2m24s
traefik-7b87c4c474-t9k8x                    1/1     Running   0
2m24s
trident-svc-c78f5b6bd-nwdsq                 1/1     Running   0
9m22s
vault-controller-55bbc96668-c6425           1/1     Running   0
11m
vault-controller-55bbc96668-lq9n9           1/1     Running   0
11m
vault-controller-55bbc96668-rfkgg           1/1     Running   0
11m
```

2. （オプション）インストールが完了したことを確認するには、を参照してください `acc-operator` 次のコマンドを使用してログを作成します。

```
kubectl logs deploy/acc-operator-controller-manager -n netapp-acc-
operator -c manager -f
```

> `accHost` クラスタの登録は最後の処理の1つです。登録に失敗しても原因 の導入は失敗しません。ログにクラスタ登録エラーが記録されている場合は、を使用して再度登録を試行できます "UIでクラスタワークフローを追加します" または API 。

3. すべてのポッドが実行中の場合は、インストールが正常に完了したことを確認します (`READY` はです `True`）を使用して、Astra Control Centerにログインするときに使用する初期セットアップパスワードを取得します。

```
kubectl get AstraControlCenter -n [netapp-acc or custom namespace]
```

対応：

```
NAME    UUID                                   VERSION     ADDRESS
READY
astra   9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f  22.11.0-82  10.111.111.111
True
```

> UUIDの値をコピーします。パスワードはです `ACC-` 続けてUUIDの値を指定します (`ACC-[UUID]` または、この例では、 `ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f`）。

ロードバランシング用の入力を設定します

サービスへの外部アクセスを管理するKubernetes入力コントローラを設定できます。これらの手順では、デフォルトのを使用した場合の入力コントローラの設定例を示します `ingressType: "Generic"` Astra Control Centerのカスタムリソース (`astra_control_center.yaml`）。を指定した場合、この手順 を使用する必要はありません `ingressType: "AccTraefik"` Astra Control Centerのカスタムリソース (`astra_control_center.yaml`）。

Astra Control Center を展開したら、 Astra Control Center を URL で公開するように入力コントローラを設定する必要があります。

セットアップ手順は、使用する入力コントローラのタイプによって異なります。Astra Control Centerは、多くの入力コントローラタイプをサポートしています。これらのセットアップ手順では、次の入力コントローラタイプの手順の例を示します。

- Istio入力
- nginx 入力コントローラ
- OpenShift 入力コントローラ

必要なもの

- が必要です "入力コントローラ" すでに導入されている必要があります。
- 。 "入力クラス" 入力コントローラに対応するものがすでに作成されている必要があります。

Istio Ingressの手順

1. Istio Ingressを設定します。

   この手順 では、「デフォルト」の構成プロファイルを使用してIstioが導入されていることを前提としています。

2. 入力ゲートウェイに必要な証明書と秘密鍵ファイルを収集または作成します。

   CA署名証明書または自己署名証明書を使用できます。共通名はAstraアドレス（FQDN）である必要があります。

   コマンド例：

   ```
   openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out
   tls.crt
   ```

3. シークレットを作成します `tls secret name` を入力します `kubernetes.io/tls` でTLS秘密鍵と証明書を使用する場合 `istio-system namespace` TLSシークレットで説明されているように、

   コマンド例：

   ```
   kubectl create secret tls [tls secret name] --key="tls.key"
   --cert="tls.crt" -n istio-system
   ```

シークレットの名前はと一致する必要があります `spec.tls.secretName` で提供されます `istio-ingress.yaml` ファイル。

4. に入力リソースを配置します `netapp-acc` （またはカスタムネームスペース）。スキーマにはv1リソースタイプを使用します (`istio-Ingress.yaml` は次の例で使用されています）。

```
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  name: istio
spec:
  controller: istio.io/ingress-controller
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingress
  namespace: [netapp-acc or custom namespace]
spec:
  ingressClassName: istio
  tls:
  - hosts:
    - <ACC address>
    secretName: [tls secret name]
  rules:
  - host: [ACC address]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: traefik
            port:
              number: 80
```

5. 変更を適用します。

```
kubectl apply -f istio-Ingress.yaml
```

6. 入力のステータスを確認します。

```
kubectl get ingress -n [netapp-acc or custom namespace]
```

対応：

```
NAME    CLASS HOSTS              ADDRESS         PORTS   AGE
ingress istio astra.example.com 172.16.103.248  80, 443 1h
```

7. Astra Control Centerのインストールを完了します。

Nginx Ingress Controller の手順

1. タイプのシークレットを作成します `kubernetes.io/tls` でTLSの秘密鍵と証明書を使用する場合 `netapp-acc` （またはカスタム名前付き）ネームスペース。を参照してください "TLS シークレット"。
2. 入力リソースをに配置します `netapp-acc` （またはカスタムネームスペース）。スキーマにはv1リソースタイプを使用します (`nginx-Ingress.yaml` は次の例で使用されています）。

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: netapp-acc-ingress
  namespace: [netapp-acc or custom namespace]
spec:
  ingressClassName: [class name for nginx controller]
  tls:
  - hosts:
    - <ACC address>
    secretName: [tls secret name]
  rules:
  - host: <ACC address>
    http:
      paths:
        - path:
          backend:
            service:
              name: traefik
              port:
                number: 80
          pathType: ImplementationSpecific
```

3. 変更を適用します。

```
kubectl apply -f nginx-Ingress.yaml
```

ネットアップでは、nginxコントローラをではなく導入環境としてインストールすることを推奨します `daemonSet`。

OpenShift 入力コントローラの手順

1. 証明書を調達し、 OpenShift ルートで使用できるようにキー、証明書、および CA ファイルを取得します。
2. OpenShift ルートを作成します。

```
oc create route edge --service=traefik --port=web -n [netapp-acc or
custom namespace] --insecure-policy=Redirect --hostname=<ACC address>
--cert=cert.pem --key=key.pem
```

Astra Control Center UI にログインします

Astra Control Center をインストールした後、デフォルトの管理者のパスワードを変更し、 Astra Control Center UI ダッシュボードにログインします。

手順

1. ブラウザで、（を含む）FQDNを入力します `https://` プレフィックス）を使用します `astraAddress` を参照してください `astra_control_center.yaml` CR When（時間） Astra Control Center をインストールした。
2. プロンプトが表示されたら、自己署名証明書を承認します。

   カスタム証明書はログイン後に作成できます。

3. Astra Control Centerのログインページで、に使用した値を入力します `email` インチ `astra_control_center.yaml` CR When（時間） Astra Control Center をインストールしたをクリックし、次に初期セットアップパスワードを入力します (`ACC-[UUID]`）。

   誤ったパスワードを 3 回入力すると、管理者アカウントは 15 分間ロックされます。

4. [**Login**] を選択します。
5. プロンプトが表示されたら、パスワードを変更します。

   初めてログインしたときにパスワードを忘れ、他の管理ユーザアカウントがまだ作成されていない場合は、にお問い合わせください "ネットアップサポート" パスワード回復のサポートを受けるには、

6. （オプション）既存の自己署名 TLS 証明書を削除して、に置き換えます "認証局（ CA ）が署名したカスタム TLS 証明書"。

インストールのトラブルシューティングを行います

いずれかのサービスがにある場合 `Error` ステータスを確認すると、ログを調べることができます。400 ~ 500 の範囲の API 応答コードを検索します。これらは障害が発生した場所を示します。

手順

1. Astra Control Center のオペレータログを調べるには、次のように入力します。

```
kubectl logs deploy/acc-operator-controller-manager -n netapp-acc-
operator -c manager -f
```

次のステップ

- （オプション）お使いの環境に応じて、インストール後に実行します "設定手順"。
- を実行して導入を完了します "セットアップのタスク"。

=
:allow-uri-read:

OpenShift OperatorHub を使用して Astra Control Center をインストールします

Red Hat OpenShift を使用する場合は、 Red Hat 認定オペレータを使用して Astra Control Center をインストールできます。この手順を使用して、から Astra Control Center をインストールします "Red Hat エコシステムカタログ" または、 Red Hat OpenShift Container Platform を使用します。

この手順を完了したら、インストール手順に戻ってを実行する必要があります "残りのステップ" インストールが成功したかどうかを確認し、ログオンします。

必要なもの

- 環境前提条件を満たしている： "インストールを開始する前に、 Astra Control Center の導入環境を準備します"。
- 健全なクラスタオペレータと**API**サービス：
  - OpenShiftクラスタから、すべてのクラスタオペレータが正常な状態にあることを確認します。

    ```
    oc get clusteroperators
    ```

  - OpenShiftクラスタから、すべてのAPIサービスが正常な状態であることを確認します。

    ```
    oc get apiservices
    ```

- * FQDN address *：データセンターのAstra Control CenterのFQDNアドレスを取得します。
- * OpenShift Permissions *：説明されているインストール手順を実行するために必要な権限を取得し、Red Hat OpenShift Container Platformにアクセスします。
- **cert manager configure**済み:クラスタにcertマネージャがすでに存在する場合は'いくつかを実行する必要があります "事前に必要な手順" そのため、Astra Control Centerは独自の証明書管理ツールをインストールしません。デフォルトでは、Astra Control Centerはインストール時に独自の証明書マネージャをインストールします。
- * Kubernetes入力コントローラ*：クラスター内のロードバランシングなどのサービスへの外部アクセスを管理するKubernetes入力コントローラがある場合は、Astra Control Centerで使用するようにセットアップ

する必要があります。

a. operatorネームスペースを作成します。

```
oc create namespace netapp-acc-operator
```

b. "セットアップを完了" 入力コントローラのタイプ。

手順

- Astra Control Centerをダウンロードして展開します
- ネットアップAstra kubectlプラグインをインストール
- [イメージをローカルレジストリに追加します]
- [オペレータインストールページを検索します]
- [オペレータをインストールします]
- Astra Control Center をインストールします

Astra Control Centerをダウンロードして展開します

1. にアクセスします "Astra Control Center評価ダウンロードページ" をクリックしますNetApp Support Site。
2. Astra Control Centerを含むバンドルをダウンロードします (`astra-control-center-[version].tar.gz`）。
3. （推奨ですがオプション）Astra Control Centerの証明書と署名のバンドルをダウンロードします (`astra-control-center-certs-[version].tar.gz`）バンドルの署名を確認するには、次の手順を実行します。

```
tar -vxzf astra-control-center-certs-[version].tar.gz
```

```
openssl dgst -sha256 -verify certs/AstraControlCenter-public.pub
-signature certs/astra-control-center-[version].tar.gz.sig astra-
control-center-[version].tar.gz
```

出力にはと表示されます `Verified OK` 検証が成功したあとに、

4. Astra Control Centerバンドルからイメージを抽出します。

```
tar -vxzf astra-control-center-[version].tar.gz
```

ネットアップ**Astra kubectl**プラグインをインストール

NetApp Astra kubectlコマンドラインプラグインは、Astra Control Centerの導入とアップグレードに関連する一般的なタスクを実行する際に時間を節約します。

必要なもの

ネットアップでは、CPUアーキテクチャやオペレーティングシステム別にプラグインのバイナリを提供しています。このタスクを実行する前に、使用しているCPUとオペレーティングシステムを把握しておく必要があります。

手順

1. 使用可能なNetApp Astra kubectlプラグインのバイナリを表示し、オペレーティングシステムとCPUアーキテクチャに必要なファイルの名前をメモします。

> kubectlプラグインライブラリはtarバンドルの一部であり、フォルダに解凍されます `kubectl-astra`。

```
ls kubectl-astra/
```

2. 正しいバイナリを現在のパスに移動し、名前をに変更します `kubectl-astra`：

```
cp kubectl-astra/<binary-name> /usr/local/bin/kubectl-astra
```

イメージをローカルレジストリに追加します

1. コンテナエンジンに応じた手順を実行します。

Docker です

1. tarballのルートディレクトリに移動します。次のファイルとディレクトリが表示されます。

   ```
   acc.manifest.bundle.yaml
   acc/
   ```

2. Astra Control Centerのイメージディレクトリにあるパッケージイメージをローカルレジストリにプッシュします。を実行する前に、次の置換を行ってください `push-images` コマンドを実行します
   - <BUNDLE_FILE> をAstra Controlバンドルファイルの名前に置き換えます（`acc.manifest.bundle.yaml`）。
   - <MY_FULL_REGISTRY_PATH> をDockerリポジトリのURLに置き換えます。次に例を示します。 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
   - <MY_REGISTRY_USER> をユーザ名に置き換えます。
   - <MY_REGISTRY_TOKEN> をレジストリの認証済みトークンに置き換えます。

     ```
     kubectl astra packages push-images -m <BUNDLE_FILE> -r
     <MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
     <MY_REGISTRY_TOKEN>
     ```

ポドマン

1. tarballのルートディレクトリに移動します。次のファイルとディレクトリが表示されます。

   ```
   acc.manifest.bundle.yaml
   acc/
   ```

2. レジストリにログインします。

   ```
   podman login <YOUR_REGISTRY>
   ```

3. 使用するPodmanのバージョンに合わせてカスタマイズされた次のいずれかのスクリプトを準備して実行します。<MY_FULL_REGISTRY_PATH> を’サブディレクトリを含むリポジトリのURLに置き換えます

   ```
   <strong>Podman 4</strong>
   ```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

```
<strong>Podman 3</strong>
```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

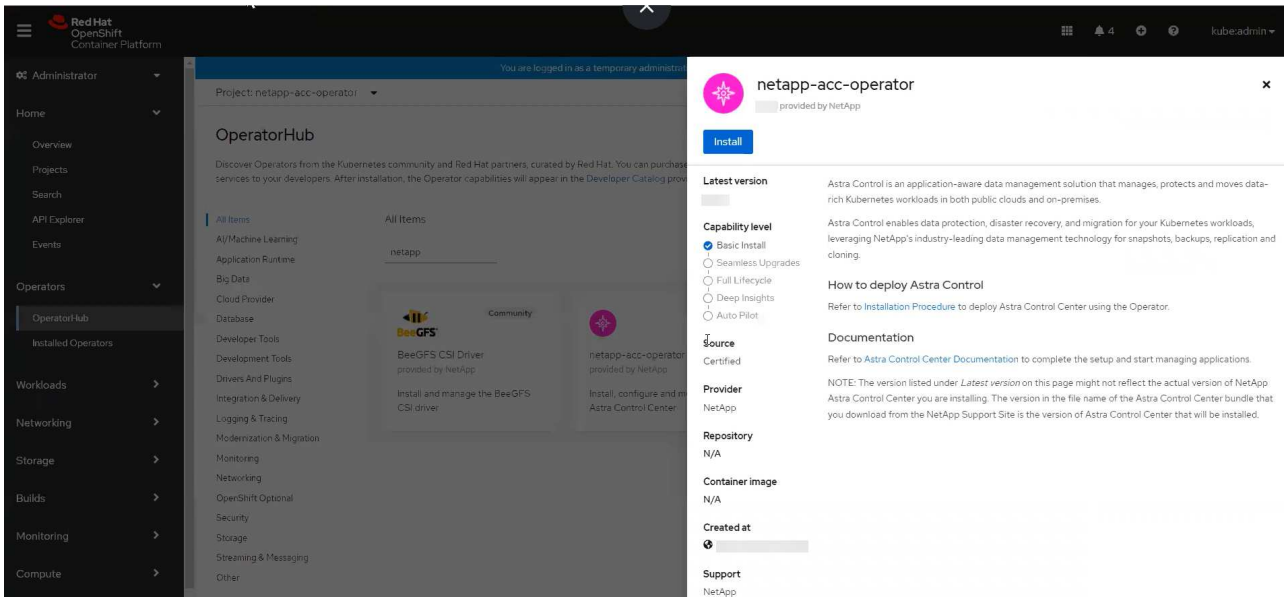
> レジストリ設定に応じて、スクリプトが作成するイメージパスは次のようになります。 https://netappdownloads.jfrog.io/docker-astra-control-prod/netapp/astra/acc/22.11.0-82/image:version

オペレータインストールページを検索します

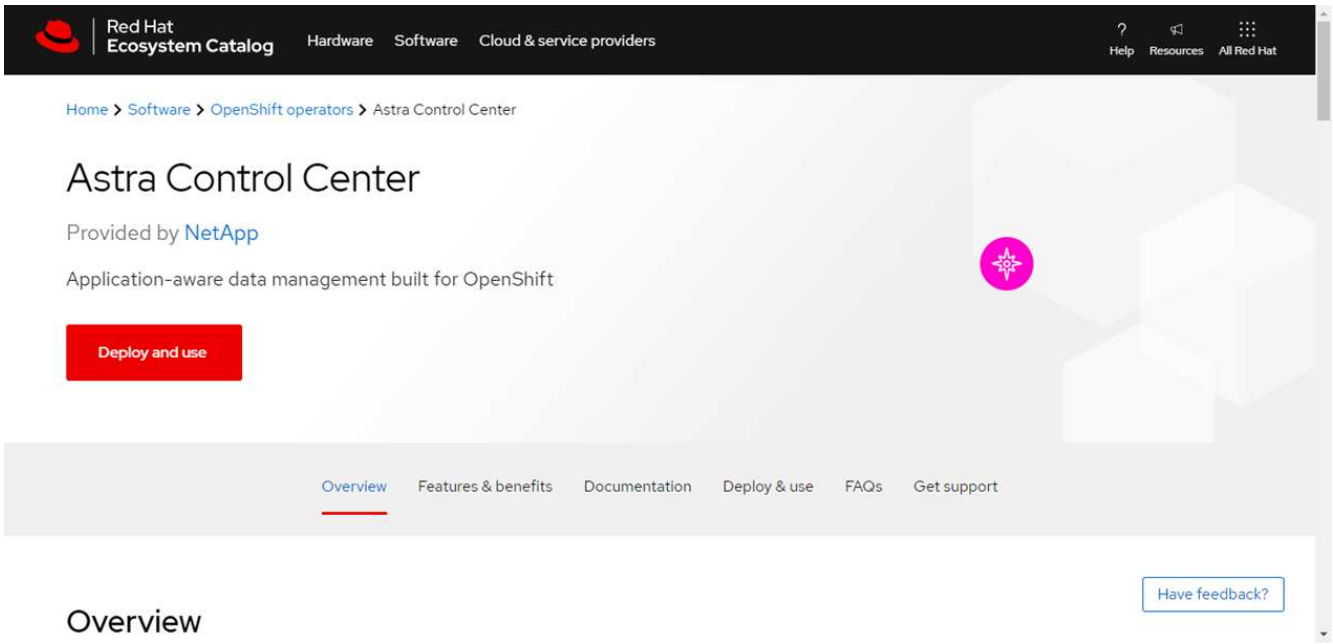
1. 次のいずれかの手順を実行して、オペレータインストールページにアクセスします。
   - Red Hat OpenShift の Web コンソールから：
     i. OpenShift Container Platform UI にログインします。

ii. サイドメニューから、 * 演算子 > OperatorHub * を選択します。

iii. NetApp Astra Control Centerオペレータを検索して選択します。



- Red Hat エコシステムカタログから：
  i. NetApp Astra Control Center を選択します "演算子"。
  ii. [Deploy and Use] を選択します。



オペレータをインストールします

1. 「 * インストールオペレータ * 」ページに必要事項を入力し、オペレータをインストールします。

   オペレータはすべてのクラスタネームスペースで使用できます。

   a. operator名前空間またはを選択します `netapp-acc-operator` オペレータのインストールの一環と

して、名前空間が自動的に作成されます。

b. 手動または自動の承認方法を選択します。

> 手動による承認が推奨されます。1 つのクラスタで実行する演算子インスタンスは 1 つだけです。

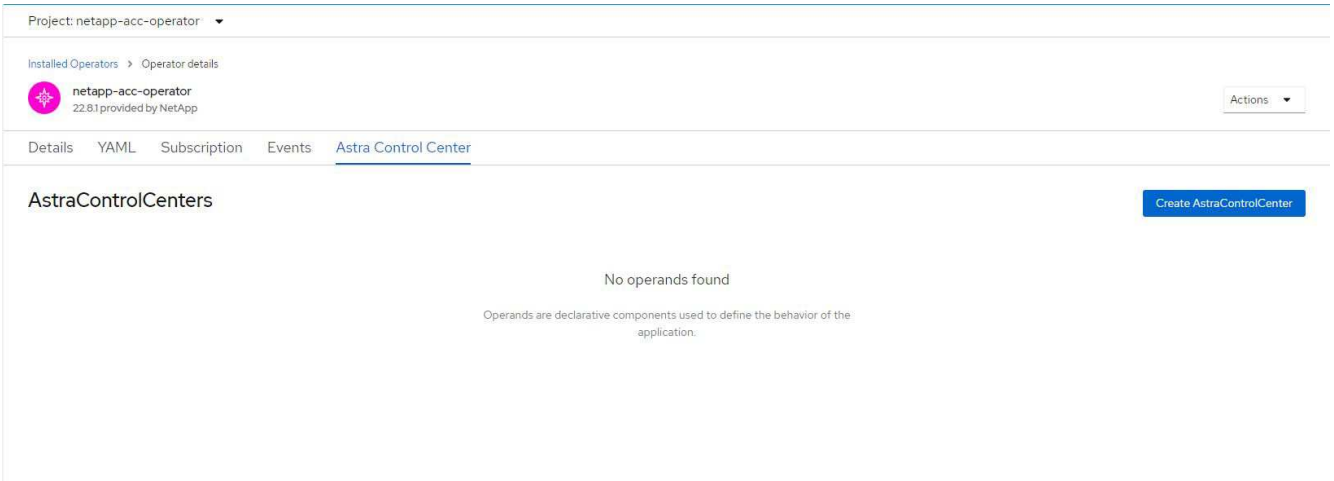
c. 「 * Install * 」を選択します。

> 手動承認方式を選択した場合は、このオペレータの手動インストール計画を承認するように求められます。

2. コンソールで、 OperatorHub メニューに移動して、オペレータが正常にインストールされたことを確認します。

Astra Control Center をインストールします

1. Astra Control Centerオペレータの[**Astra Control Center**]タブ内のコンソールから'[*Create AstraControlCenter *]を選択します



2. を実行します `Create AstraControlCenter` フォームフィールド：

   a. Astra Control Center の名前を保持または調整します。

   b. Astra Control Centerのラベルを追加します。

   c. AutoSupportを有効または無効にします。Auto Support 機能の保持を推奨します。

   d. Astra Control CenterのFQDNまたはIPアドレスを入力します。入らないでください `http://` または `https://` をクリックします。

   e. Astra Control Centerのバージョンを入力します（例：22.04.1）。

   f. アカウント名、 E メールアドレス、および管理者の姓を入力します。

   g. ボリューム再利用ポリシーを選択してください `Retain`、 `Recycle`または `Delete`。デフォルト値はです `Retain`。

   h. 入力タイプを選択します。

      - **`Generic`** (`ingressType: "Generic"`) （デフォルト）

このオプションは、別の入力コントローラを使用している場合、または独自の入力コントローラを使用する場合に使用します。Astra Control Centerを導入したら、を設定する必要があります "入力コントローラ" URLを使用してAstra Control Centerを公開します。

- **AccTraefik** (`ingressType: "AccTraefik"`)

  入力コントローラを設定しない場合は、このオプションを使用します。これにより、Astra Control Centerが導入されます `traefik` ゲートウェイをKubernetesの「LoadBalancer」タイプのサービスとして使用します。

Astra Control Centerは、タイプ「LoadBalancer」のサービスを使用します。（`svc/traefik` Astra Control Centerの名前空間）で、アクセス可能な外部IPアドレスが割り当てられている必要があります。お使いの環境でロードバランサが許可されていて、設定されていない場合は、MetalLBまたは別の外部サービスロードバランサを使用して外部IPアドレスをサービスに割り当てることができます。内部 DNS サーバ構成では、 Astra Control Center に選択した DNS 名を、負荷分散 IP アドレスに指定する必要があります。

> サービスタイプ「 LoadBalancer 」および入力の詳細については、を参照してください "要件"。

a. * Image Registry * に、ローカルコンテナイメージのレジストリパスを入力します。入らないでください `http://` または `https://` をクリックします。

b. 認証が必要なイメージレジストリを使用する場合は、イメージシークレットを入力します。

> 認証が必要なレジストリを使用する場合は、 クラスタでシークレットを作成します。

c. 管理者の名を入力します。

d. リソースの拡張を構成する。

e. デフォルトのストレージクラスを指定します。

> デフォルトのストレージクラスが設定されている場合は、そのストレージクラスがデフォルトのアノテーションを持つ唯一のストレージクラスであることを確認します。

f. CRD 処理の環境設定を定義します。

3. YAMLビューを選択して、選択した設定を確認します。

4. 選択するオプション `Create`。

レジストリシークレットを作成します

認証が必要なレジストリを使用する場合は、OpenShiftクラスタでシークレットを作成し、にシークレット名を入力します `Create AstraControlCenter` フォームフィールド。

1. Astra Control Centerオペレータの名前空間を作成します。

```
oc create ns [netapp-acc-operator or custom namespace]
```

2. この名前空間にシークレットを作成します。

```
oc create secret docker-registry astra-registry-cred n [netapp-acc-
operator or custom namespace] --docker-server=[your_registry_path]
--docker username=[username] --docker-password=[token]
```

Astra Controlは、Dockerレジストリシークレットのみをサポートします。

3. の残りのフィールドに値を入力します Create AstraControlCenterフォーム・フィールド。

次のステップ

を実行します "残りのステップ" Astra Control Centerが正常にインストールされたことを確認するには、入力コントローラ（オプション）をセットアップし、UIにログインします。また、を実行する必要があります "セットアップのタスク" インストールが完了したら、

Cloud Volumes ONTAP ストレージバックエンドに Astra Control Center をインストールします

Astra Control Center を使用すると、 Kubernetes クラスタと Cloud Volumes ONTAP インスタンスを自己管理することで、ハイブリッドクラウド環境でアプリケーションを管理できます。Astra Control Center は、オンプレミスの Kubernetes クラスタ、またはクラウド環境内の自己管理型 Kubernetes クラスタのいずれかに導入できます。

これらのいずれかの環境では、 Cloud Volumes ONTAP をストレージバックエンドとして使用して、アプリケーションデータの管理処理を実行できます。バックアップターゲットとして S3 バケットを設定することもできます。

Amazon Web Services（AWS）、Google Cloud Platform（GCP）、およびCloud Volumes ONTAP ストレージバックエンドを使用するMicrosoft AzureにAstra Control Centerをインストールするには、クラウド環境に応じて次の手順を実行します。

- Amazon Web Services に Astra Control Center を導入
- Astra Control CenterをGoogle Cloud Platformに導入
- Microsoft Azure に Astra Control Center を導入

OpenShift Container Platform（OCP）などの自己管理型Kubernetesクラスタを使用して、ディストリビューション内のアプリケーションを管理できます。Astra Control Centerを導入するために検証されるのは、自己管理型のOCPクラスタのみです。

Amazon Web Services に Astra Control Center を導入

Amazon Web Services （ AWS ）パブリッククラウドでホストされる自己管理型の Kubernetes クラスタに Astra Control Center を導入できます。

AWSに必要なもの

AWS に Astra Control Center を導入する前に、次のものが必要です。

- Astra Control Center ライセンス。を参照してください "Astra Control Center のライセンス要件"。
- "Astra Control Center の要件を満たす"。
- NetApp Cloud Central アカウント
- OCPを使用する場合は、Red Hat OpenShift Container Platform（OCP）権限（ポッドを作成するためのネームスペースレベル）
- バケットとコネクタを作成するための権限を持つ AWS クレデンシャル、アクセス ID 、シークレットキー
- AWS アカウント Elastic Container Registry （ ECR ）アクセスおよびログイン
- AWS がホストするゾーンと Route 53 エントリは、 Astra Control UI にアクセスするために必要です

AWS の運用環境の要件

Astra Control Center を使用するには、 AWS 向けに次の運用環境が必要です。

- Red Hat OpenShift Container Platform 4.8 の場合

> Astra Control Center をホストするオペレーティングシステムが、環境の公式ドキュメントに記載されている基本的なリソース要件を満たしていることを確認します。

Astra Control Center では、環境のリソース要件に加え、次のリソースが必要です。

コンポーネント	要件
バックエンドの **NetApp Cloud Volumes ONTAP** ストレージ容量	300GB 以上のデータがあります
ワーカーノード（ **AWS EC2** の要件）	少なくとも 3 つのワーカーノードが必要です。 vCPU コア 4 基、 RAM はそれぞれ 12GB です
ロードバランサ	動作環境クラスタ内のサービスに送信される入力トラフィックに使用できるサービスタイプ「 LoadBalancer 」
FQDN	Astra Control Center の FQDN をロードバランシング IP アドレスに指定する方法
Astra Trident（以前の**Cloud Manager**で、**Kubernetes**クラスタ検出の一部として**NetApp BlueXP** にインストール）	Trident 21.04 以降がインストールおよび設定され、 NetApp ONTAP バージョン 9.5 以降がストレージバックエンドとしてインストールされている必要があります

コンポーネント	要件
イメージレジストリ	Astra Control Center のビルドイメージをプッシュできる、 AWS Elastic Container Registry などの既存のプライベートレジストリが必要です。イメージをアップロードするイメージレジストリの URL を指定する必要があります。 Restic ベースのイメージを使用してアプリケーションをバックアップおよび復元するには、 Astra Control Center ホストクラスタと管理対象クラスタが同じイメージレジストリにアクセスできる必要があります。
Astra Trident / ONTAP 構成	Astra Control Center を使用するには、ストレージクラスを作成してデフォルトのストレージクラスとして設定する必要があります。Astra Control Centerは、KubernetesクラスタをNetApp BlueXP（旧Cloud Manager）にインポートするときに作成される次のONTAP Kubernetesストレージクラスをサポートします。Astra Trident によって提供される機能は次のとおりです。 • `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io` • `vsaworkingenvironment-<>-single-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-single-san csi.trident.netapp.io`

これらの要件は、運用環境で実行されている唯一のアプリケーションが Astra Control Center であることを前提としています。環境で追加のアプリケーションを実行している場合は、それに応じてこれらの最小要件を調整します。

AWS レジストリトークンは 12 時間で期限切れになり、その後 Docker イメージのレジストリシークレットを更新する必要があります。

AWS の導入の概要を参照してください

Cloud Volumes ONTAP をストレージバックエンドとして使用して Astra Control Center for AWS をインストールするプロセスの概要を以下に示します。

これらの各手順については、以下で詳しく説明します。

1. 十分な IAM 権限があることを確認します。
2. AWS に Red Hat OpenShift クラスタをインストールします。
3. AWS を設定します。
4. NetApp BlueXP for AWSを構成します。

5. Astra Control Center for AWSをインストール。

十分な IAM 権限があることを確認します

Red Hat OpenShiftクラスタとNetApp BlueXP（旧Cloud Manager）コネクタをインストールできる十分なIAMロールと権限があることを確認します。

を参照してください "AWS の初期クレデンシャル"。

AWS に Red Hat OpenShift クラスタをインストールします

AWS に Red Hat OpenShift Container Platform クラスタをインストールします。

インストール手順については、を参照してください "AWS で OpenShift Container Platform にクラスタをインストールします"。

AWS を設定します

次に、仮想ネットワークの作成、EC2コンピューティングインスタンスのセットアップ、AWS S3バケットの作成、Astra Control Centerイメージをホストする Elastic Container Register（ECR）の作成、このレジストリへのイメージのプッシュを行うようにAWSを設定します。

AWS のドキュメントに従って次の手順を実行します。を参照してください "AWS インストールドキュメント"。

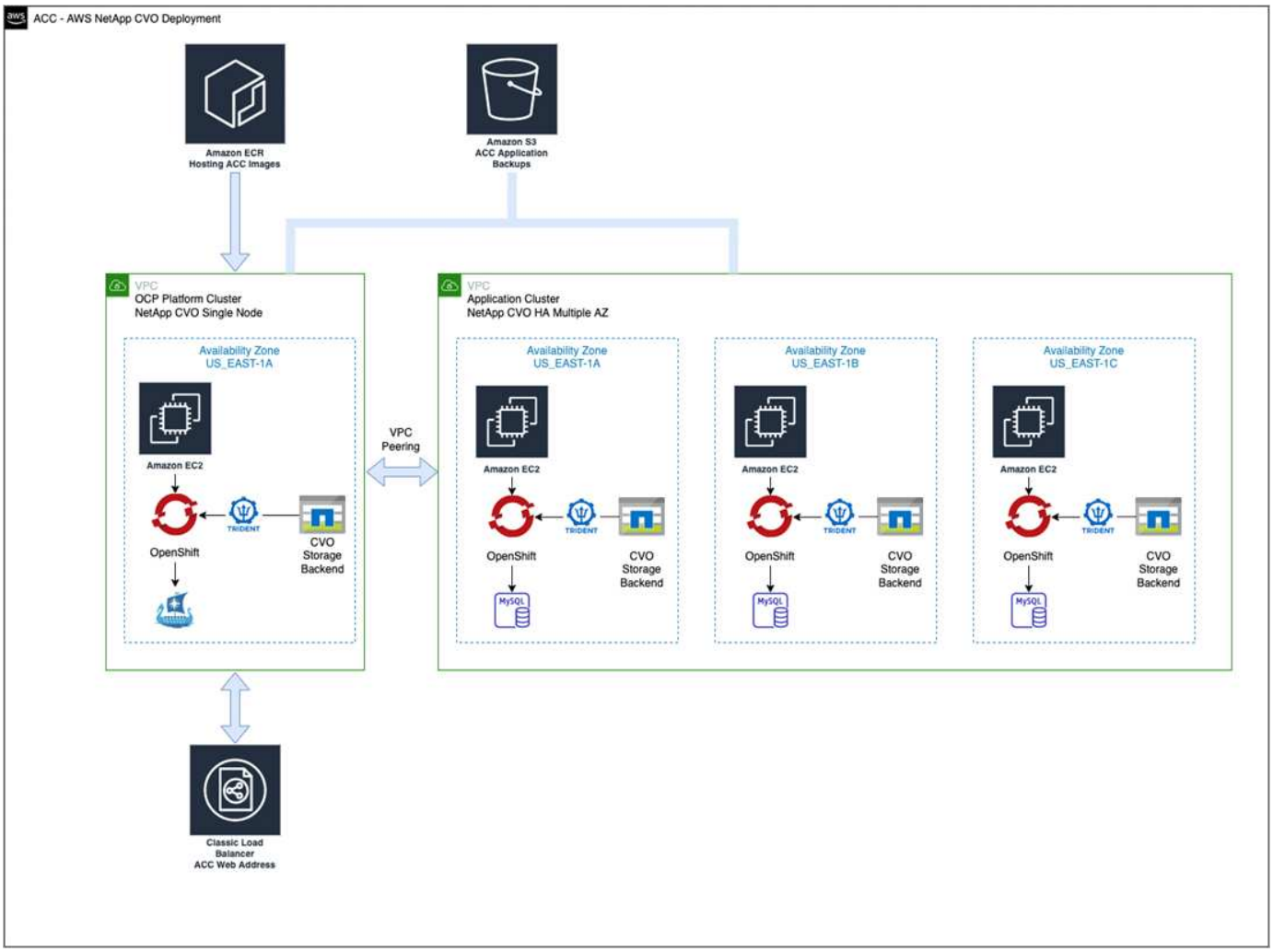
1. AWS仮想ネットワークを作成します。
2. EC2 コンピューティングインスタンスを確認します。AWS ではベアメタルサーバまたは VM を使用できます。
3. インスタンスタイプが、マスターノードとワーカーノードの Astra の最小リソース要件に一致していない場合は、 Astra の要件に合わせて AWS でインスタンスタイプを変更します。を参照してください "Astra Control Center の要件"。
4. バックアップを格納する AWS S3 バケットを少なくとも 1 つ作成します。
5. すべての ACC イメージをホストする AWS Elastic Container Registry （ ECR ）を作成します。

> ECRを作成しないと、Astra Control Centerは、AWSバックエンドを持つCloud Volumes ONTAP を含むクラスタからモニタリングデータにアクセスできません。問題 は、 Astra Control Center を使用して検出および管理しようとしたクラスタに AWS ECR アクセスがない場合に発生します。

6. ACC イメージを定義済みのレジストリにプッシュします。

> AWS Elastic Container Registry （ ECR ）トークンの有効期限は 12 時間です。有効期限が切れたため、クラスタ間のクローニング処理が失敗します。この問題 は、AWS用に設定されたCloud Volumes ONTAP からストレージバックエンドを管理する場合に発生します。この問題 を修正するには、 ECR で再度認証を行い、クローン操作を再開するための新しいシークレットを生成します。

AWS 環境の例を次に示します。



NetApp BlueXP for AWSを構成します

NetApp BlueXP（旧Cloud Manager）を使用して、ワークスペースの作成、AWSへのコネクタの追加、作業環境の作成、クラスタのインポートを行います。

BlueXPのマニュアルに従って'次の手順を実行します以下を参照してください。

- "AWS で Cloud Volumes ONTAP を使用するための準備"。
- "BlueXPを使用してAWSでコネクタを作成します"

手順

1. 資格情報をBlueXPに追加します。
2. ワークスペースを作成します。
3. AWS 用のコネクタを追加します。プロバイダとして AWS を選択します。
4. クラウド環境の作業環境を構築
   a. 場所：「 Amazon Web Services （ AWS ）」
   b. 「 Cloud Volumes ONTAP HA 」と入力します。
5. OpenShift クラスタをインポートします。作成した作業環境にクラスタが接続されます。

a. ネットアップクラスタの詳細を表示するには、 * K8s * > * Cluster list * > * Cluster Details * を選択します。

b. 右上隅に Trident のバージョンが表示されていることを確認します。

c. Cloud Volumes ONTAP クラスタのストレージクラスは、プロビジョニングツールとしてネットアップを使用していることに注目してください。

   これにより、 Red Hat OpenShift クラスタがインポートされ、デフォルトのストレージクラスに割り当てられます。ストレージクラスを選択します。Trident は、インポートと検出のプロセスの一環として自動的にインストールされます。

6. このCloud Volumes ONTAP 環境内のすべての永続ボリュームとボリュームをメモします。

> Cloud Volumes ONTAP は、シングルノードまたはハイアベイラビリティとして動作できます。HA が有効になっている場合は、 AWS で実行されている HA ステータスとノード導入ステータスを確認します。

Astra Control Center for AWSをインストール

標準に従ってください "Astra Control Center のインストール手順"。

> AWSでは汎用のS3バケットタイプが使用されます。

Astra Control CenterをGoogle Cloud Platformに導入

Astra Control Centerは、Google Cloud Platform（GCP）パブリッククラウドでホストされる自己管理型のKubernetesクラスタに導入できます。

GCPに必要なもの

GCPでAstra Control Centerを導入する前に、次の項目が必要です。

- Astra Control Center ライセンス。を参照してください "Astra Control Center のライセンス要件"。
- "Astra Control Center の要件を満たす"。
- NetApp Cloud Central アカウント
- OCPを使用している場合は、Red Hat OpenShift Container Platform（OCP）4.10
- OCPを使用する場合は、Red Hat OpenShift Container Platform（OCP）権限（ポッドを作成するためのネームスペースレベル）
- バケットとコネクタの作成を可能にする権限を持つGCPサービスアカウント

GCPの運用環境の要件

> Astra Control Center をホストするオペレーティングシステムが、環境の公式ドキュメントに記載されている基本的なリソース要件を満たしていることを確認します。

Astra Control Center では、環境のリソース要件に加え、次のリソースが必要です。

コンポーネント	要件
バックエンドの **NetApp Cloud Volumes ONTAP** ストレージ容量	300GB 以上のデータがあります
ワーカーノード（**GCP**コンピューティング要件）	少なくとも 3 つのワーカーノードが必要です。 vCPU コア 4 基、 RAM はそれぞれ 12GB です
ロードバランサ	動作環境クラスタ内のサービスに送信される入力トラフィックに使用できるサービスタイプ「 LoadBalancer 」
FQDN（**GCP DNS**ゾーン）	Astra Control Center の FQDN をロードバランシング IP アドレスに指定する方法
Astra Trident（以前の**Cloud Manager**で、**Kubernetes**クラスタ検出の一部として**NetApp BlueXP** にインストール）	Trident 21.04 以降がインストールおよび設定され、 NetApp ONTAP バージョン 9.5 以降がストレージバックエンドとしてインストールされている必要があります
イメージレジストリ	Astra Control Centerビルドイメージをプッシュできる、Google Container Registryなどの既存のプライベートレジストリが必要です。イメージをアップロードするイメージレジストリの URL を指定する必要があります。 バックアップ用にリストアイメージを取得するには、匿名アクセスを有効にする必要があります。
Astra Trident / ONTAP 構成	Astra Control Center を使用するには、ストレージクラスを作成してデフォルトのストレージクラスとして設定する必要があります。Astra Control Centerは、KubernetesクラスタをNetApp BlueXPにインポートするときに作成される次のONTAP Kubernetesストレージクラスをサポートします。Astra Trident によって提供される機能は次のとおりです。 • `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io` • `vsaworkingenvironment-<>-single-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-single-san csi.trident.netapp.io`

これらの要件は、運用環境で実行されている唯一のアプリケーションが Astra Control Center であることを前提としています。環境で追加のアプリケーションを実行している場合は、それに応じてこれらの最小要件を調整します。

GCPの導入の概要

ここでは、Cloud Volumes ONTAP をストレージバックエンドとして使用して、GCP内の自己管理型OCPクラスタにAstra Control Centerをインストールするプロセスの概要を示します。

これらの各手順については、以下で詳しく説明します。

1. GCPにRed Hat OpenShiftクラスタをインストールします。
2. GCPプロジェクトとVirtual Private Cloudを作成します。
3. 十分な IAM 権限があることを確認します。
4. GCPを設定します。
5. NetApp BlueXP for GCPを構成します。
6. Astra Control Center for GCPをインストールします。

GCPにRed Hat OpenShiftクラスタをインストールします

まず、GCPにRedHat OpenShiftクラスタをインストールします。

インストール手順については、次を参照してください。

- "GCPにOpenShiftクラスタをインストールする"
- "GCPサービスアカウントの作成"

GCPプロジェクトとVirtual Private Cloudを作成します

少なくとも1つのGCPプロジェクトとVirtual Private Cloud（VPC）を作成します。

> OpenShift では、独自のリソースグループを作成できます。さらに、GCP VPCも定義する必要があります。OpenShift のドキュメントを参照してください。

プラットフォームクラスタリソースグループおよびターゲットアプリケーション OpenShift クラスタリソースグループを作成できます。

十分な IAM 権限があることを確認します

Red Hat OpenShiftクラスタとNetApp BlueXP（旧Cloud Manager）コネクタをインストールできる十分なIAMロールと権限があることを確認します。

を参照してください "GCPの初期資格情報と権限"。

GCPを設定します

次に、VPCの作成、コンピューティングインスタンスのセットアップ、Google Cloud Object Storageの作成、Astra Control Centerイメージのホストに Google Container Registerの作成、このレジストリへのイメージのプッシュを行うようにGCPを設定します。

GCPのドキュメントに従って、次の手順を実行します。「GCPへのOpenShiftクラスタのインストール」を参照してください。

1. GCPでGCPプロジェクトとVPCを作成します。GCPでは、CVOバックエンドでOCPクラスタ用にを使用する予定です。
2. コンピューティングインスタンスを確認します。GCP内のベアメタルサーバまたはVMです。
3. インスタンスタイプが、マスターノードとワーカーノードのAstra最小リソース要件と一致していない場合は、GCPでインスタンスタイプを変更してAstraの要件を満たします。を参照してください "Astra Control Center の要件"。
4. バックアップを保存するGCP Cloud Storageバケットを少なくとも1つ作成します。
5. バケットへのアクセスに必要なシークレットを作成します。
6. すべてのAstra Control Centerイメージをホストする Google Container Registryを作成します。
7. すべてのAstra Control Centerイメージに対して、Dockerプッシュ/プル用のGoogle Container Registryアクセスを設定します。

   例：次のスクリプトを入力すると、ACCイメージをこのレジストリにプッシュできます。

   ```
   gcloud auth activate-service-account <service account email address>
   --key-file=<GCP Service Account JSON file>
   ```

   このスクリプトには、Astra Control Centerマニフェストファイルと Google Image Registryの場所が必要です。

   例

   ```
   manifestfile=astra-control-center-<version>.manifest
   GCP_CR_REGISTRY=<target image repository>
   ASTRA_REGISTRY=<source ACC image repository>

   while IFS= read -r image; do
       echo "image: $ASTRA_REGISTRY/$image $GCP_CR_REGISTRY/$image"
       root_image=${image%:*}
       echo $root_image
       docker pull $ASTRA_REGISTRY/$image
       docker tag $ASTRA_REGISTRY/$image $GCP_CR_REGISTRY/$image
       docker push $GCP_CR_REGISTRY/$image
   done < astra-control-center-22.04.41.manifest
   ```

8. DNS ゾーンを設定します。

NetApp BlueXP for GCPを構成します

NetApp BlueXP（旧Cloud Manager）を使用して、ワークスペースの作成、GCPへのコネクタの追加、作業環境の作成、クラスタのインポートを行います。

BlueXPのマニュアルに従って'次の手順を実行しますを参照してください "GCPでCloud Volumes ONTAP の使用を開始する"。

必要なもの

- 必要なIAM権限と役割を持つGCPサービスアカウントにアクセスします

手順

1. 資格情報をBlueXPに追加します。を参照してください "GCP アカウントの追加"。
2. GCPのコネクターを追加します。
   a. プロバイダーとして[GCP]を選択します。
   b. GCP資格情報を入力します。を参照してください "BlueXPからGCPでコネクタを作成する"。
   c. コネクタが動作していることを確認し、コネクタに切り替えます。
3. クラウド環境の作業環境を構築
   a. 場所："GCP"
   b. 「 Cloud Volumes ONTAP HA 」と入力します。
4. OpenShift クラスタをインポートします。作成した作業環境にクラスタが接続されます。
   a. ネットアップクラスタの詳細を表示するには、 * K8s * > * Cluster list * > * Cluster Details * を選択します。
   b. 右上隅に Trident のバージョンが表示されていることを確認します。
   c. Cloud Volumes ONTAP クラスタのストレージクラスは、プロビジョニングツールとして「ネットアップ」を使用していることに注目してください。

      これにより、 Red Hat OpenShift クラスタがインポートされ、デフォルトのストレージクラスに割り当てられます。ストレージクラスを選択します。Trident は、インポートと検出のプロセスの一環として自動的にインストールされます。
5. このCloud Volumes ONTAP 環境内のすべての永続ボリュームとボリュームをメモします。

> Cloud Volumes ONTAP は、シングルノードまたはハイアベイラビリティ（HA）で動作します。HAが有効になっている場合は、GCPで実行されているHAステータスとノード導入ステータスを確認します。

Astra Control Center for GCPをインストールします

標準に従ってください "Astra Control Center のインストール手順"。

> GCPでは汎用S3バケットタイプが使用されます。

1. Astra Control Centerインストール用のイメージをプルするDocker Secretを生成します。

```
kubectl create secret docker-registry <secret name> --docker
-server=<Registry location> --docker-username=_json_key --docker
-password="$(cat <GCP Service Account JSON file>)" --namespace=pcloud
```

Microsoft Azure に Astra Control Center を導入

Microsoft Azure パブリッククラウドでホストされる自己管理型の Kubernetes クラスタに Astra Control Center を導入できます。

Azureに必要なもの

Azure に Astra Control Center を導入する前に、次のものが必要です。

- Astra Control Center ライセンス。を参照してください "Astra Control Center のライセンス要件"。
- "Astra Control Center の要件を満たす"。
- NetApp Cloud Central アカウント
- OCPを使用する場合、Red Hat OpenShift Container Platform（OCP）4.8
- OCPを使用する場合は、Red Hat OpenShift Container Platform（OCP）権限（ポッドを作成するためのネームスペースレベル）
- バケットとコネクタの作成を可能にする権限を持つ Azure クレデンシャル

Azure の運用環境の要件

Astra Control Center をホストするオペレーティングシステムが、環境の公式ドキュメントに記載されている基本的なリソース要件を満たしていることを確認します。

Astra Control Center では、環境のリソース要件に加え、次のリソースが必要です。

を参照してください "Astra Control Center の運用環境要件"。

コンポーネント	要件
バックエンドの **NetApp Cloud Volumes ONTAP** ストレージ容量	300GB 以上のデータがあります
ワーカーノード（**Azure** コンピューティング要件）	少なくとも 3 つのワーカーノードが必要です。 vCPU コア 4 基、 RAM はそれぞれ 12GB です
ロードバランサ	動作環境クラスタ内のサービスに送信される入力トラフィックに使用できるサービスタイプ「 LoadBalancer 」
FQDN （ **Azure DNS** ゾーン）	Astra Control Center の FQDN をロードバランシング IP アドレスに指定する方法
Astra Trident（**NetApp BlueXP** の**Kubernetes**クラスタ検出の一部としてインストール）	Trident 21.04 以降がインストールおよび設定され、 NetApp ONTAP バージョン 9.5 以降がストレージバックエンドとして使用されます

コンポーネント	要件
イメージレジストリ	Astra Control Center ビルドイメージをプッシュできる、 Azure Container Registry （ACR ）などの既存のプライベートレジストリが必要です。イメージをアップロードするイメージレジストリの URL を指定する必要があります。 バックアップ用にリストアイメージを取得するには、匿名アクセスを有効にする必要があります。
Astra Trident / ONTAP 構成	Astra Control Center を使用するには、ストレージクラスを作成してデフォルトのストレージクラスとして設定する必要があります。Astra Control Centerは、KubernetesクラスタをNetApp BlueXPにインポートするときに作成される次のONTAP Kubernetesストレージクラスをサポートします。Astra Trident によって提供される機能は次のとおりです。 • `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io` • `vsaworkingenvironment-<>-single-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-single-san csi.trident.netapp.io`

これらの要件は、運用環境で実行されている唯一のアプリケーションが Astra Control Center であることを前提としています。環境で追加のアプリケーションを実行している場合は、それに応じてこれらの最小要件を調整します。

Azure の導入の概要

ここでは、 Astra Control Center for Azure のインストールプロセスの概要を示します。

これらの各手順については、以下で詳しく説明します。

1. Azure に Red Hat OpenShift クラスタをインストールします。
2. Azure リソースグループを作成する。
3. 十分な IAM 権限があることを確認します。
4. Azure を設定。
5. NetApp BlueXP（旧Cloud Manager）をAzure向けに設定します。
6. Azure向けAstra Control Centerのインストールと設定。

Azure に Red Hat OpenShift クラスタをインストールします

まず、 Azure に Red Hat OpenShift クラスタをインストールします。

インストール手順については、次を参照してください。

- "Azure への OpenShift クラスタのインストール"。
- "Azure アカウントをインストールする"。

Azure リソースグループを作成する

Azure リソースグループを少なくとも 1 つ作成します。

> OpenShift では、独自のリソースグループを作成できます。さらに、 Azure リソースグループも定義する必要があります。OpenShift のドキュメントを参照してください。

プラットフォームクラスタリソースグループおよびターゲットアプリケーション OpenShift クラスタリソースグループを作成できます。

十分な IAM 権限があることを確認します

Red Hat OpenShiftクラスタとNetApp BlueXP Connectorをインストールできる十分なIAMロールと権限があることを確認します。

を参照してください "Azure のクレデンシャルと権限"。

Azure を設定

次に、仮想ネットワークの作成、コンピューティングインスタンスのセットアップ、Azure Blobコンテナの作成、Astra Control Centerイメージをホストするazure Container Register（ACR）の作成、このレジストリへのイメージのプッシュを行うようにAzureを設定します。

Azure のドキュメントに従って、次の手順を実行します。を参照してください "Azure への OpenShift クラスタのインストール"。

1. Azure Virtual Networkの作成
2. コンピューティングインスタンスを確認します。Azure の場合、ベアメタルサーバまたは VM を使用できます。
3. インスタンスタイプがまだマスターノードとワーカーノードの Astra 最小リソース要件に一致していない場合は、 Azure でインスタンスタイプを変更して Astra の要件を満たします。を参照してください "Astra Control Center の要件"。
4. バックアップを格納するAzure BLOBコンテナを少なくとも1つ作成します。
5. ストレージアカウントを作成します。Astra Control Center でバケットとして使用するコンテナを作成するには、ストレージアカウントが必要です。
6. バケットへのアクセスに必要なシークレットを作成します。
7. Azure Container Registry （ ACR ）を作成して、すべての Astra Control Center イメージをホストします。
8. ACR アクセスを設定して Docker プッシュ / プルをすべての Astra Control Center イメージに適用します。
9. 次のスクリプトを入力して、 ACC イメージをこのレジストリにプッシュします。

```
az acr login -n <AZ ACR URL/Location>
This script requires ACC manifest file and your Azure ACR location.
```

- 例 * ：

```
manifestfile=astra-control-center-<version>.manifest
AZ_ACR_REGISTRY=<target image repository>
ASTRA_REGISTRY=<source ACC image repository>

while IFS= read -r image; do
    echo "image: $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRY/$image"
    root_image=${image%:*}
    echo $root_image
    docker pull $ASTRA_REGISTRY/$image
    docker tag $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRYY/$image
    docker push $AZ_ACR_REGISTRY/$image
done < astra-control-center-22.04.41.manifest
```

10. DNS ゾーンを設定します。

NetApp BlueXP（旧Cloud Manager）をAzure向けに設定します

BlueXP（旧Cloud Manager）を使用して、ワークスペースの作成、Azureへのコネクタの追加、作業環境の作成、クラスタのインポートを行います。

BlueXPのマニュアルに従って'次の手順を実行しますを参照してください "BlueXPの使用を開始しました"。

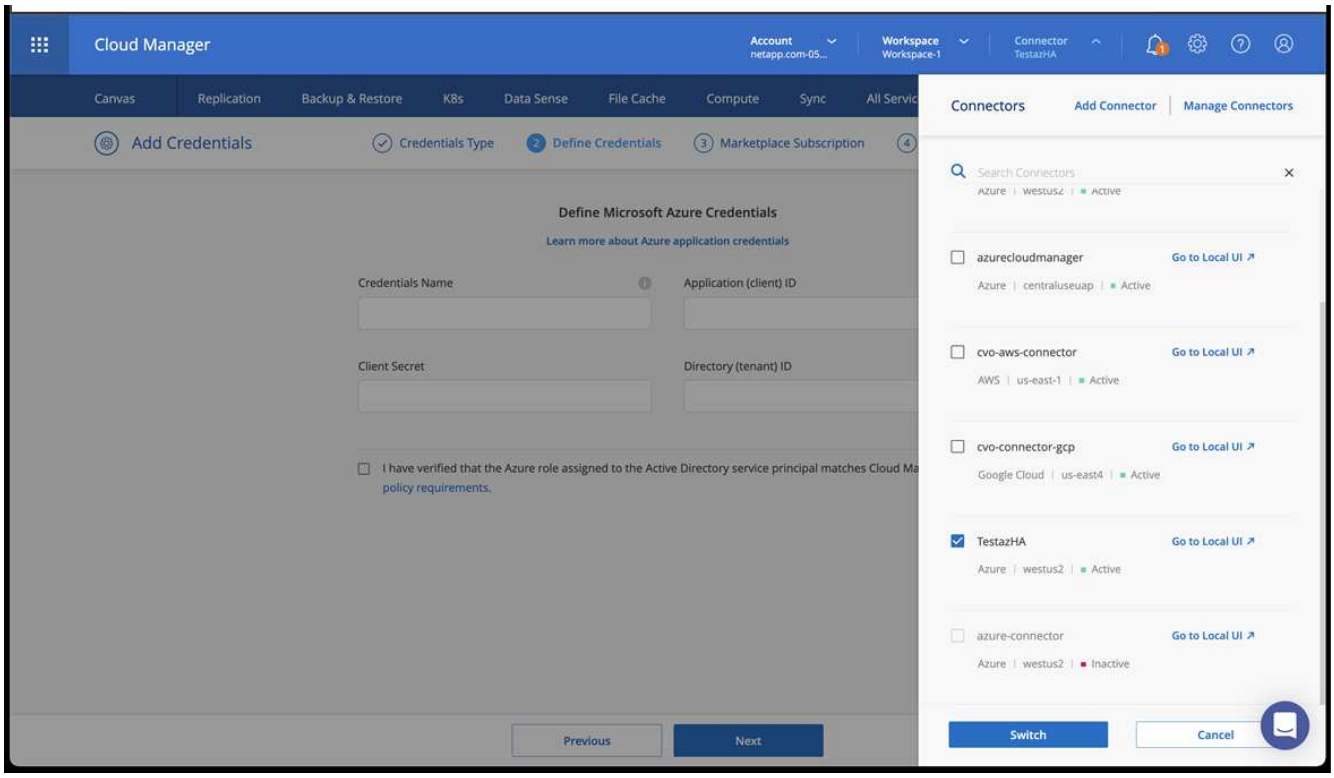
必要なもの

必要な IAM 権限とロールを持つ Azure アカウントにアクセスします

手順

1. 資格情報をBlueXPに追加します。
2. Azure 用のコネクタを追加します。を参照してください "BlueXPポリシー"。
   a. プロバイダとして「 * Azure * 」を選択します。
   b. アプリケーション ID 、クライアントシークレット、ディレクトリ（テナント） ID など、 Azure クレデンシャルを入力します。

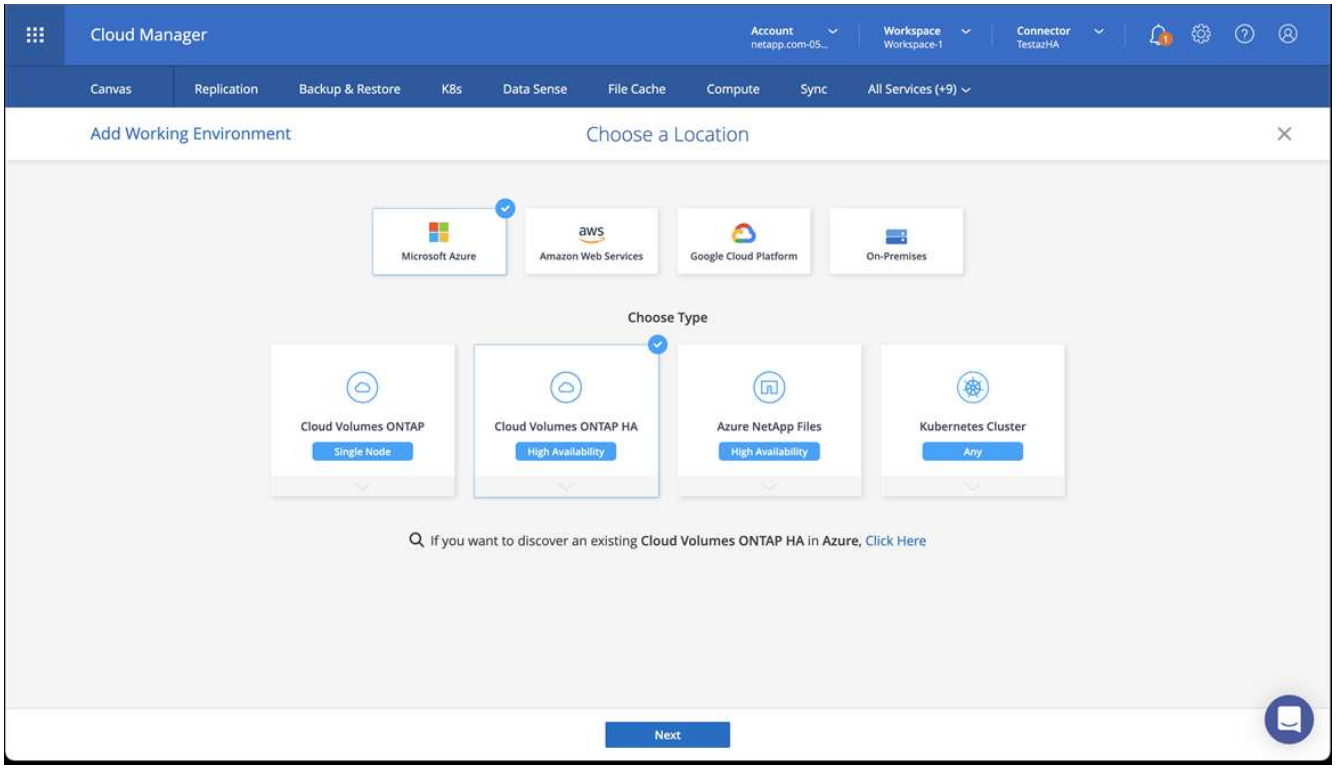
      を参照してください "BlueXPrからAzureでコネクタを作成しています"。
3. コネクタが動作していることを確認し、コネクタに切り替えます。



4. クラウド環境の作業環境を構築

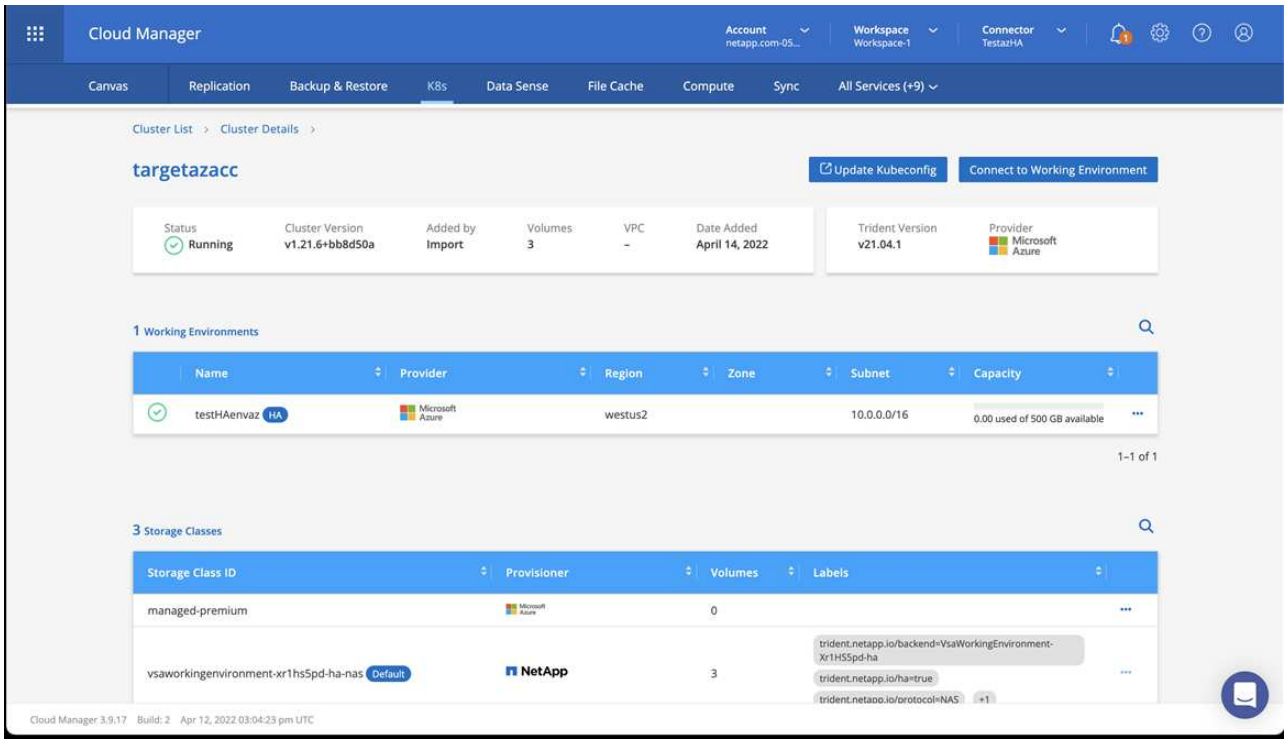
   a. 場所：「 Microsoft Azure 」。

   b. 「 Cloud Volumes ONTAP HA 」と入力します。



5. OpenShift クラスタをインポートします。作成した作業環境にクラスタが接続されます。

a. ネットアップクラスタの詳細を表示するには、 * K8s * > * Cluster list * > * Cluster Details * を選択します。



b. 右上隅に Trident のバージョンが表示されていることを確認します。

c. Cloud Volumes ONTAP クラスタのストレージクラスは、プロビジョニングツールとしてネットアップを使用していることに注目してください。

これにより、 Red Hat OpenShift クラスタがインポートされ、デフォルトのストレージクラスが割り当てられます。ストレージクラスを選択します。Trident は、インポートと検出のプロセスの一環として自動的にインストールされます。

6. このCloud Volumes ONTAP 環境内のすべての永続ボリュームとボリュームをメモします。

7. Cloud Volumes ONTAP は、シングルノードまたはハイアベイラビリティとして動作できます。HA が有効になっている場合は、 Azure で実行されている HA ステータスとノード導入ステータスを確認します。

Azure向け Astra Control Center のインストールと設定

Astra Control Center を標準でインストールします "インストール手順"。

Astra Control Center を使用して、 Azure バケットを追加する。を参照してください "Astra Control Center をセットアップし、バケットを追加する"。

=
:allow-uri-read:

Astra Control Center をセットアップします

Astra Control Center をインストールして UI にログインし、パスワードを変更したら、ライセンスの設定、クラスタの追加、ストレージの管理、バケットの追加を行います。

タスク

- Astra Control Center のライセンスを追加します
- Astra Controlを使用して、クラスタ管理のための環境を準備する
- [クラスタを追加]
- [ストレージバックエンドを追加します]
- [バケットを追加します]

Astra Control Center のライセンスを追加します

新しいライセンスは、Astra Control UIまたはを使用して追加できます "API" Astra Control Center の全機能を利用できます。ライセンスがないと、 Astra Control Center の使用は、ユーザの管理と新しいクラスタの追加に限定されます。

Astra Control Centerライセンスは、Kubernetes CPUユニットを使用してCPUリソースを測定し、すべての管理対象Kubernetesクラスタのワーカーノードに割り当てられたCPUリソースを考慮します。ライセンスはvCPUの使用量に基づいています。ライセンスの計算方法の詳細については、を参照してください "ライセンス"。

> インストールがライセンス数を超えると、 Astra Control Center は新しいアプリケーションを管理できなくなります。容量を超えるとアラートが表示されます。

> 既存の評価版またはフルライセンスを更新するには、を参照してください "既存のライセンスを更新する"。

必要なもの

- 新しくインストールしたAstra Control Centerインスタンスへのアクセス。
- 管理者ロールの権限。
- A "ネットアップライセンスファイル" （NLF）。

手順

1. Astra Control Center UI にログインします。
2. 「 * アカウント * > * ライセンス * 」を選択します。
3. 「 * ライセンスの追加 * 」を選択します。
4. ダウンロードしたライセンスファイル（ NLF ）を参照します。
5. 「 * ライセンスの追加 * 」を選択します。

Account>*License* ページには、ライセンス情報、有効期限、ライセンスシリアル番号、アカウント ID 、および使用されている CPU ユニットが表示されます。

> 評価用ライセンスをお持ちで、AutoSupport にデータを送信していない場合は、Astra Control Centerに障害が発生したときにデータが失われないように、アカウントIDを必ず保存してください。

Astra Controlを使用して、クラスタ管理のための環境を準備する

クラスタを追加する前に、次の前提条件を満たしていることを確認する必要があります。また、資格チェックを実行して、クラスタをAstra Control Centerに追加し、クラスタ管理の役割を作成する準備ができていることを確認する必要もあります。

必要なもの

- クラスタ内のワーカーノードで適切なストレージドライバが設定されていることを確認します。これにより、ポッドがバックエンドストレージと通信できるようになります。
- が環境に合っている "運用環境の要件" Astra TridentとAstra Control Centerに最適。
- Astra Tridentの一バージョン "Astra Control Centerによってサポートされます" がインストールされている：

  > 可能です "Astra Tridentを導入" Tridentオペレータ（手動またはHelmチャートを使用）またはを使用します `tridentctl`。Astra Tridentのインストールまたはアップグレードを行う前に、を参照してください "サポートされるフロントエンド、バックエンド、およびホスト構成"。

  - * Tridentストレージバックエンドが設定されています*：少なくとも1つのAstra Tridentストレージバックエンドが必要です "を設定します" クラスタのポリシーを確認してください。
  - * Tridentストレージクラスを設定*：少なくとも1つのAstra Tridentストレージクラスが必要です "を設定します" クラスタのポリシーを確認してください。デフォルトのストレージクラスが設定されている場合は、そのストレージクラスがデフォルトのアノテーションを持つ唯一のストレージクラスであることを確認します。
  - * Astra Tridentボリュームスナップショットコントローラとボリュームスナップショットクラスがインストールおよび設定されている*：ボリュームスナップショットコントローラがである必要があります "インストール済み" Astra Controlでスナップショットを作成できるようにします。Astra Tridentが少なくとも1つ `VolumeSnapshotClass` はい "セットアップ" 管理者による。
- **Kubeconfig**にアクセス可能:にアクセスできます "クラスタkubeconfig" コンテキスト要素が1つだけ含まれます。
- * ONTAP クレデンシャル*：Astra Control Centerを使用してアプリケーションをバックアップおよびリストアするには、バックアップONTAP システムでONTAP クレデンシャルとスーパーユーザーIDを設定する必要があります。

  ONTAP コマンドラインで次のコマンドを実行します。

  ```
  export-policy rule modify -vserver <storage virtual machine name>
  -policyname <policy name> -ruleindex 1 -superuser sys
  export-policy rule modify -vserver <storage virtual machine name>
  -policyname <policy name> -ruleindex 1 -anon 65534
  ```

- **rancher**のみ: Rancher環境でアプリケーションクラスタを管理する場合、rancherから提供されたkubeconfigファイルでアプリケーションクラスタのデフォルトコンテキストを変更して、rancher APIサーバコンテキストではなくコントロールプレーンコンテキストを使用します。これにより、 Rancher APIサーバの負荷が軽減され、パフォーマンスが向上します。

資格チェックを実行します

次の資格チェックを実行して、 Astra Control Center にクラスタを追加する準備ができていることを確認します。

手順

1. Trident のバージョンを確認

```
kubectl get tridentversions -n trident
```

Trident が存在する場合は、次のような出力が表示されます。

```
NAME      VERSION
trident   22.10.0
```

Trident が存在しない場合は、次のような出力が表示されます。

```
error: the server doesn't have a resource type "tridentversions"
```

> Trident がインストールされていない場合や、インストールされているバージョンが最新でない場合は、次に進む前に最新バージョンの Trident をインストールする必要があります。を参照してください "Trident のドキュメント" 手順については、を参照し

2. ポッドが実行されていることを確認します。

```
kubectl get pods -n trident
```

3. サポートされているTridentドライバをストレージクラスが使用しているかどうかを確認します。プロビジョニング担当者の名前はとします `csi.trident.netapp.io`。次の例を参照してください。

```
kubectl get sc
```

回答例：

```
NAME                    PROVISIONER             RECLAIMPOLICY
VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
ontap-gold (default)   csi.trident.netapp.io   Delete          Immediate
true                    5d23h
```

制限されたクラスタロール**kubeconfig**を作成します

必要に応じて、Astra Control Centerの限定管理者ロールを作成できます。これは、Astra Control Centerのセットアップに必要な手順 ではありません。この手順 を使用すると、管理対象のクラスタのAstra Control権限を制限する別のkubeconfigを作成できます。

必要なもの

手順 の手順を実行する前に、管理するクラスタに次の情報があることを確認してください。

- kubectl v1.23以降がインストールされている
- Astra Control Centerを使用して追加および管理するクラスタへのアクセス

> この手順 では、Astra Control Centerを実行しているクラスタにkubectlでアクセスする必要はありません。

- アクティブなコンテキストのクラスタ管理者の権限で管理するクラスタのアクティブなkubeconfigです

手順

1. サービスアカウントを作成します。

   a. という名前のサービスアカウントファイルを作成します `astracontrol-service-account.yaml`。

      名前と名前空間を必要に応じて調整します。ここで変更を行った場合は、以降の手順でも同じ変更を適用する必要があります。

```
<strong>astracontrol-service-account.yaml</strong>
```

+

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: astracontrol-service-account
  namespace: default
```

   a. サービスアカウントを適用します。

```
kubectl apply -f astracontrol-service-account.yaml
```

2. Astra Controlでクラスタを管理するために必要な最小限の権限を持つ、制限付きのクラスタロールを作成します。

   a. を作成します `ClusterRole` という名前のファイルです `astra-admin-account.yaml`。

      名前と名前空間を必要に応じて調整します。ここで変更を行った場合は、以降の手順でも同じ変更を適用する必要があります。

```
<strong>astra-admin-account.yaml</strong>
```

+

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: astra-admin-account
rules:

# Get, List, Create, and Update all resources
```

```yaml
# Necessary to backup and restore all resources in an app
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - get
  - list
  - create
  - patch

# Delete Resources
# Necessary for in-place restore and AppMirror failover
- apiGroups:
  - ""
  - apps
  - autoscaling
  - batch
  - crd.projectcalico.org
  - extensions
  - networking.k8s.io
  - policy
  - rbac.authorization.k8s.io
  - snapshot.storage.k8s.io
  - trident.netapp.io
  resources:
  - configmaps
  - cronjobs
  - daemonsets
  - deployments
  - horizontalpodautoscalers
  - ingresses
  - jobs
  - namespaces
  - networkpolicies
  - persistentvolumeclaims
  - poddisruptionbudgets
  - pods
  - podtemplates
  - podsecuritypolicies
  - replicasets
  - replicationcontrollers
  - replicationcontrollers/scale
  - rolebindings
  - roles
  - secrets
```

```
  - serviceaccounts
  - services
  - statefulsets
  - tridentmirrorrelationships
  - tridentsnapshotinfos
  - volumesnapshots
  - volumesnapshotcontents
  verbs:
  - delete

# Watch resources
# Necessary to monitor progress
- apiGroups:
  - ""
  resources:
  - pods
  - replicationcontrollers
  - replicationcontrollers/scale
  verbs:
  - watch

# Update resources
- apiGroups:
  - ""
  - build.openshift.io
  - image.openshift.io
  resources:
  - builds/details
  - replicationcontrollers
  - replicationcontrollers/scale
  - imagestreams/layers
  - imagestreamtags
  - imagetags
  verbs:
  - update

# Use PodSecurityPolicies
- apiGroups:
  - extensions
  - policy
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

a. クラスタロールを適用します。

```
kubectl apply -f astra-admin-account.yaml
```

3. サービスアカウントへのクラスタロールバインド用に、クラスタロールを作成します。

   a. を作成します `ClusterRoleBinding` という名前のファイルです `astracontrol-clusterrolebinding.yaml`。

      必要に応じて、サービスアカウントの作成時に変更した名前と名前空間を調整します。

```
<strong>astracontrol-clusterrolebinding.yaml</strong>
```

+

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: astracontrol-admin
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: astra-admin-account
subjects:
- kind: ServiceAccount
  name: astracontrol-service-account
  namespace: default
```

   a. クラスタロールバインドを適用します。

```
kubectl apply -f astracontrol-clusterrolebinding.yaml
```

4. サービスアカウントのシークレットを一覧表示します（置き換えます） `<context>` インストールに適したコンテキストを使用して、次の操作を行います。

```
kubectl get serviceaccount astracontrol-service-account --context
<context> --namespace default -o json
```

出力の末尾は次のようになります。

```
"secrets": [
{ "name": "astracontrol-service-account-dockercfg-vhz87"},
{ "name": "astracontrol-service-account-token-r59kr"}
]
```

内の各要素のインデックス `secrets` アレイは0から始まります。上記の例では、のインデックスです `astracontrol-service-account-dockercfg-vhz87` は0、のインデックスです `astracontrol-service-account-token-r59kr` は1です。出力で、 "token" という単語が含まれるサービスアカウント名のインデックスをメモしてください。

5. 次のように kubeconfig を生成します。

   a. を作成します `create-kubeconfig.sh` ファイル。交換してください `TOKEN_INDEX` 次のスクリプトの先頭に正しい値を入力します。

```
<strong>create-kubeconfig.sh</strong>
```

```
# Update these to match your environment.
# Replace TOKEN_INDEX with the correct value
# from the output in the previous step. If you
# didn't change anything else above, don't change
# anything else here.

SERVICE_ACCOUNT_NAME=astracontrol-service-account
NAMESPACE=default
NEW_CONTEXT=astracontrol
KUBECONFIG_FILE='kubeconfig-sa'

CONTEXT=$(kubectl config current-context)

SECRET_NAME=$(kubectl get serviceaccount ${SERVICE_ACCOUNT_NAME}
\
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  -o jsonpath='{.secrets[TOKEN_INDEX].name}')
TOKEN_DATA=$(kubectl get secret ${SECRET_NAME} \
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  -o jsonpath='{.data.token}')

TOKEN=$(echo ${TOKEN_DATA} | base64 -d)

# Create dedicated kubeconfig
# Create a full copy
kubectl config view --raw > ${KUBECONFIG_FILE}.full.tmp
```

```
# Switch working context to correct context
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp config use-
context ${CONTEXT}

# Minify
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp \
  config view --flatten --minify > ${KUBECONFIG_FILE}.tmp

# Rename context
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  rename-context ${CONTEXT} ${NEW_CONTEXT}

# Create token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-credentials ${CONTEXT}-${NAMESPACE}-token-user \
  --token ${TOKEN}

# Set context to use token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --user ${CONTEXT}-${NAMESPACE}-token
-user

# Set context to correct namespace
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --namespace ${NAMESPACE}

# Flatten/minify kubeconfig
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  view --flatten --minify > ${KUBECONFIG_FILE}

# Remove tmp
rm ${KUBECONFIG_FILE}.full.tmp
rm ${KUBECONFIG_FILE}.tmp
```

b. コマンドをソースにし、 Kubernetes クラスタに適用します。

```
source create-kubeconfig.sh
```

6. （オプション）クラスタにわかりやすい名前にコバーベキューの名前を変更します。

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

次の手順

前提条件が満たされていることを確認したら、次は準備ができています クラスタを追加。

クラスタを追加

アプリケーションの管理を開始するには、 Kubernetes クラスタを追加し、コンピューティングリソースとして管理します。Kubernetes アプリケーションを検出するには、 Astra Control Center のクラスタを追加する必要があります。

> 他のクラスタを Astra Control Center に追加して管理する前に、 Astra Control Center が最初に導入したクラスタを管理することをお勧めします。指標およびトラブルシューティング用の Kubemetrics データとクラスタ関連データを送信するには、最初のクラスタを管理下に配置する必要があります。

必要なもの

- クラスタを追加する前に、必要なを確認し、実行しておきます 前提条件となるタスク。

手順

1. ダッシュボードまたはクラスタメニューのいずれかから移動します。
   - リソースサマリの*ダッシュボード*で、クラスタペインから*追加*を選択します。
   - 左側のナビゲーション領域で、*クラスタ*を選択し、クラスタページから*クラスタの追加*を選択します。
2. 表示された*クラスタの追加*ウィンドウで、をアップロードします `kubeconfig.yaml` の内容をファイルまたは貼り付けます `kubeconfig.yaml` ファイル。

   > 。 `kubeconfig.yaml` ファイルには、1つのクラスタのクラスタクレデンシャルのみを含める必要があります*。

   > 自分で作成する場合は `kubeconfig` ファイルには、* 1つの*コンテキストエレメントのみを定義する必要があります。を参照してください "Kubernetes のドキュメント" を参照してください `kubeconfig` ファイル。を使用して、制限されたクラスタロールのkubeconfigを作成した場合 上記のプロセスこの手順では、kubeconfigをアップロードまたは貼り付けてください。

3. クレデンシャル名を指定します。デフォルトでは、クレデンシャル名がクラスタの名前として自動的に入力されます。
4. 「 * 次へ * 」を選択します。
5. このKubernetesクラスタに使用するデフォルトのストレージクラスを選択し、* Next *を選択します。

   > ONTAP ストレージをベースとする Trident ストレージクラスを選択する必要があります。

6. 情報を確認し、すべてが良好な場合は、「*追加」を選択します。

結果

クラスタが「* discovering *」状態になり、「Healthy *」に変わります。これで、Astra Control Centerを使用してクラスタを管理できるようになりました。

> Astra Control Center で管理するクラスタを追加したあと、監視オペレータの配置に数分かかる場合があります。それまでは、通知アイコンが赤に変わり、 * モニタリングエージェントステータスチェック失敗 * イベントが記録されます。この問題は無視してかまいません。問題は、Astra Control Center が正しいステータスを取得したときに解決します。数分経っても問題 が解決しない場合は、クラスタに移動してを実行します `oc get pods -n netapp-monitoring` を開始点として指定します。問題をデバッグするには、監視オペレータのログを調べる必要があります。

ストレージバックエンドを追加します

既存のONTAP ストレージバックエンドをAstra Control Centerに追加して、そのリソースを管理できます。

ストレージバックエンドとして Astra Control のストレージクラスタを管理することで、永続ボリューム（PVS ）とストレージバックエンドの間のリンケージを取得できるだけでなく、追加のストレージ指標も取得できます。

手順

1. 左側のナビゲーション領域のダッシュボードで、* Backends *を選択します。
2. 次のいずれかを実行します。
   - 新しいバックエンド：「追加」を選択して既存のバックエンドを管理し、「 ONTAP 」を選択して、「*次へ」を選択します。
   - 検出されたバックエンド：Actionsメニューから、管理対象クラスタから検出されたバックエンドで* Manage *を選択します。
3. ONTAP クラスタ管理IPアドレスと管理者クレデンシャルを入力します。クレデンシャルはクラスタ全体のクレデンシャルである必要があります。

   > ここで入力するクレデンシャルのユーザは、を持っている必要があります `ontapi` ONTAP クラスタのONTAP System Managerで有効になっているユーザログインアクセス方法。SnapMirrorレプリケーションを使用する場合は、アクセス方法が指定された「admin」ロールのユーザクレデンシャルを適用します `ontapi` および `http`、ソースとデスティネーションの両方のONTAP クラスタ。を参照してください "ONTAP ドキュメントの「ユーザーアカウントの管理」を参照してください" を参照してください。

4. 「 * 次へ * 」を選択します。
5. バックエンドの詳細を確認し、 * Manage * を選択します。

結果

バックエンドがに表示されます `Healthy` リストに概要情報を表示します。

> バックエンドが表示されるようにページを更新する必要がある場合があります。

バケットを追加します

バケットは、Astra Control UIまたはを使用して追加できます "API"。アプリケーションと永続的ストレージをバックアップする場合や、クラスタ間でアプリケーションのクローニングを行う場合は、オブジェクトストアバケットプロバイダの追加が不可欠です。Astra Control は、これらのバックアップまたはクローンを、定義したオブジェクトストアバケットに格納します。

アプリケーション構成と永続的ストレージを同じクラスタにクローニングする場合、Astra Controlにバケットを作成する必要はありません。アプリケーションのSnapshot機能にはバケットは必要ありません。

必要なもの

- Astra Control Centerで管理しているクラスタから到達できるバケット。
- バケットのクレデンシャル。
- 次のタイプのバケット
  - NetApp ONTAP S3
  - NetApp StorageGRID S3 の略
  - Microsoft Azure
  - 汎用 S3

> Amazon Web Services（AWS）とGoogle Cloud Platform（GCP）では、汎用のS3バケットタイプを使用します。

> Astra Control CenterはAmazon S3を汎用のS3バケットプロバイダとしてサポートしていますが、Astra Control Centerは、AmazonのS3をサポートしていると主張するすべてのオブジェクトストアベンダーをサポートしているわけではありません。

手順

1. 左側のナビゲーション領域で、 * バケット * を選択します。
2. 「 * 追加」を選択します。
3. バケットタイプを選択します。

> バケットを追加するときは、正しいバケットプロバイダを選択し、そのプロバイダに適したクレデンシャルを指定します。たとえば、タイプとして NetApp ONTAP S3 が許可され、 StorageGRID クレデンシャルが受け入れられますが、このバケットを使用して原因の以降のアプリケーションのバックアップとリストアはすべて失敗します。

4. 既存のバケット名とオプションの概要 を入力します。

> バケット名と概要 はバックアップ先として表示されるため、あとでバックアップを作成する際に選択できます。この名前は、保護ポリシーの設定時にも表示されます。

5. S3 エンドポイントの名前または IP アドレスを入力します。
6. [資格情報の選択*]で、[追加]または[*既存の*を使用]タブのいずれかを選択します。
   - 「*追加」を選択した場合：
     i. Astra Control の他のクレデンシャルと区別するクレデンシャルの名前を入力します。
     ii. クリップボードからコンテンツを貼り付けて、アクセス ID とシークレットキーを入力します。
   - [既存の使用*]を選択した場合：
     i. バケットで使用する既存のクレデンシャルを選択します。
7. 選択するオプション `Add`。

バケットを追加すると、デフォルトのバケットインジケータで1つのバケットがAstra Controlによってマークされます。最初に作成したバケットがデフォルトバケットになります。バケットを追加する際、あとでを選択できます "別のデフォルトバケットを設定する"。

次の手順

Astra Control Centerにログインしてクラスタを追加したので、Astra Control Centerのアプリケーションデータ管理機能を使い始めることができます。

- "ローカルユーザとロールを管理します"
- "アプリの管理を開始します"
- "アプリを保護します"
- "通知を管理します"
- "Cloud Insights に接続します"
- "カスタム TLS 証明書を追加します"
- "デフォルトのストレージクラスを変更する"

詳細については、こちらをご覧ください

- "Astra Control API を使用"
- "既知の問題"

Astra Control Center に関するよくある質問

この FAQ は、質問に対する簡単な回答を探している場合に役立ちます。

概要

次のセクションでは、 Astra Control Center を使用しているときに発生する可能性のあるその他の質問に対する回答を示します。詳しい説明については、 astra.feedback@netapp.com までお問い合わせください

Astra Control Center へのアクセス

- Astra Control の URL は何であるか。 *

Astra Control Center は、ローカル認証と各環境に固有の URL を使用します。

URLには、ブラウザで、Astra Control Centerをインストールしたときに、Astra_control_center.yamlカスタムリソース（CR）ファイルのspec.astraatAddressフィールドに設定した完全修飾ドメイン名（FQDN）を入力します。emailは、Astra_control_center.yaml CRのspec.emailフィールドで設定した値です。

ライセンス

- 評価ライセンスを使用しています。フルライセンスに変更する方法を教えてください。 *

ネットアップライセンスファイル（ NLF ）を取得することで、フルライセンスに簡単に変更できます。

- 手順 *
  1. 左側のナビゲーションから、 * アカウント * > * ライセンス * を選択します。
  2. 「 * ライセンスの追加 * 」を選択します。
  3. ダウンロードしたライセンスファイルを参照し、 * 追加 * を選択します。
- 評価ライセンスを使用しています。アプリを管理できますか？ *

はい、評価ライセンスを使用して、管理アプリケーション機能をテストできます。

Kubernetes クラスタを登録しています

- Astra Control に追加したワーカーノードを Kubernetes クラスタに追加する必要があります。どうすればよいですか？ *

新しいワーカーノードを既存のプールに追加できます。これらは Astra Control によって自動的に検出されます。新しいノードが Astra Control に表示されない場合は、新しいワーカーノードでサポートされているイメージタイプが実行されているかどうかを確認します。を使用して、新しいワーカーノードの健全性を確認することもできます `kubectl get nodes` コマンドを実行します

- クラスタの管理を適切に解除するにはどうすればよいですか *
  1. "Astra Control からアプリケーションの管理を解除"。
  2. "Astra Control からクラスタの管理を解除"。
- Kubernetes クラスタを Astra Control から削除した後、アプリケーションとデータはどうなりますか。 *

Astra Control からクラスタを削除しても、クラスタの構成（アプリケーションと永続的ストレージ）は変更されません。このクラスタで作成されたアプリケーションの Snapshot やバックアップを Astra Control で復元することはできません。Astra Control で作成した永続的ストレージのバックアップは Astra Control に残っていますが、リストアには使用できません。

> 他の方法でクラスタを削除する場合は、必ず事前に Astra Control からクラスタを削除してください。Astra Control で管理している間に別のツールを使用してクラスタを削除した場合、原因で Astra Control アカウントに問題が発生する可能性があります。

- NetApp Trident は、管理を解除すると自動的にクラスタからアンインストールされますか？ * Astra Control Center からクラスタを管理を解除しても、 Trident は自動的にクラスタからアンインストールされることはありません。Trident をアンインストールするには、が必要です "Trident のドキュメントでは、次の手順を実行します"。

アプリケーションの管理

- Astra Control はアプリケーションを導入できますか。 *

Astra Control はアプリケーションを導入しない。アプリケーションは Astra Control の外部に導入する必要があります。

- アプリケーションを Astra Control から管理しなくなった後、どうなりますか。 *

既存のバックアップまたは Snapshot がすべて削除されます。アプリケーションとデータは引き続き使用できます。管理対象外のアプリケーション、またはそのアプリケーションに属するバックアップや Snapshot では、データ管理操作を実行できません。

- ネットアップ以外のストレージにあるアプリケーションは Astra Control で管理できますか。 *

いいえネットアップ以外のストレージを使用しているアプリケーションは Astra Control で検出できますが、ネットアップ以外のストレージを使用しているアプリケーションは管理できません。

- Astra Control 自体を管理すべきですか？ * いいえ、 Astra Control 自体は「システムアプリケーション」であるため、管理すべきではありません。
- 正常でないポッドはアプリケーション管理に影響しますか？ * 管理対象アプリケーションにポッドが正常な状態でない場合、 Astra Control は新しいバックアップとクローンを作成できません。

データ管理の操作

- アプリケーションは複数の PVS を使用しています。Astra ControlはこれらのPVSのスナップショットとバックアップを作成しますか*

はい。Astra Controlによるアプリケーションのスナップショット操作には、アプリケーションのPVCにバインドされているすべてのPVSのスナップショットが含まれます。

- Astra Control で取得したスナップショットを、別のインターフェイスやオブジェクトストレージから直接管理できますか。 *

いいえAstra Control で作成したスナップショットとバックアップは、 Astra Control でのみ管理できます。