はじめに
Astra Control Center

NetApp
November 21, 2023

目次

はじめに

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]
- [アプリケーションクラスタの要件]
- [アプリケーション管理の要件]
- [レプリケーションの前提条件]
- [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

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。

- Google Anthos 1.10または1.11
- Kubernetes 1.22〜1.24
- Rancher Kubernetes Engine（RKE）：
  - Rancher 2.5.12およびRKE 1.3.3（2.6.3）付きRKE 1.2.16
  - Rancher 2.6.3を使用したRKE 2（v1.23.6 + rke2r2）
- Red Hat OpenShift Container Platform 4.8、4.9、または4.10
- VMware Tanzu Kubernetes Grid 1.4または1.5
- VMware Tanzu Kubernetes Grid Integrated Edition 1.12.2または1.13

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
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

コンポーネント	要件
Astra Trident	Trident 21.10.1以降がインストールされ、SnapMirrorベースのアプリケーションレプリケーション用にAstra 22.07以降がインストールされて設定されている必要があります

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 * ： Astra Control Center では、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、 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 - 経済性

OpenShift 環境でのアプリケーションのクローニングでは、 Astra Control Center が OpenShift でボリュームをマウントし、ファイルの所有権を変更できるようにする必要があります。そのため、これらの処理を許可するには、 ONTAP ボリュームのエクスポートポリシーを設定する必要があります。次のコマンドを使用して実行できます。

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

管理対象のコンピューティングリソースとして 2 つ目の OpenShift 運用環境を追加する場合は、 Astra Trident ボリューム Snapshot 機能を有効にする必要があります。Astra Trident でボリューム Snapshot を有効にしてテストするには、 "Astra Trident の公式ガイドをご覧ください"。

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。
- Astra Control Centerのインストールの一部として、PoDセキュリティポリシー（PSP）の制限された環境で次のリソースが作成されます。
  - ポッドセキュリティポリシー

- RBACロール
- RBACロールRBACロールとRoleBindingリソースがに作成されます `netapp-acc` ネームスペース：

- 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"

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。

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 および FAS システム
- NetApp ONTAP 9.8以降AFF およびFAS 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に使用
- NetApp Cloud Volumes ONTAP の略

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できます。を参照してください "ONTAP ライセンスを管理します"。

アプリケーションクラスタの要件

Astra Control Center には、 Astra Control Center から管理するクラスタに対する次の要件があります。これらの要件は、管理するクラスタが Astra Control Center をホストする運用環境クラスタである場合にも適用されます。

- Kubernetes の最新バージョン "Snapshot コントローラコンポーネント" がインストールされている
- Astra Trident "volumesnapshotclass オブジェクト" は管理者によって定義されています
- クラスタにはデフォルトの Kubernetes ストレージクラスが存在します

- Astra Trident を使用するように少なくとも 1 つのストレージクラスが設定されている

> アプリケーションクラスタにが必要です `kubeconfig.yaml` 1つの_context_elementだけを定義するファイル。の Kubernetes のドキュメントを参照してください "kubeconfig ファイルの作成に関する情報"。

> Rancher環境でアプリケーションクラスタを管理する場合は、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ます `kubeconfig` rancher APIサーバコンテキストの代わりにコントロールプレーンコンテキストを使用するためにrancherによって提供されるファイル。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。

アプリケーション管理の要件

Astra Control には、次のアプリケーション管理要件があります。

- * ライセンス * ： Astra Control Center を使用してアプリケーションを管理するには、 Astra Control Center ライセンスが必要です。
- * 名前空間 * ： Astra Control では、アプリケーションが複数の名前空間にまたがることはありませんが、名前空間には複数のアプリケーションを含めることができます。
- * StorageClass* ： StorageClass が明示的に設定されたアプリケーションをインストールし、そのアプリケーションをクローニングする必要がある場合、クローン処理のターゲットクラスタに最初に指定された StorageClass が必要です。明示的に StorageClass を設定したアプリケーションを、同じストレージクラスを使用しないクラスタにクローニングすると、失敗します。
- * Kubernetes リソース * ： Astra Control で収集されていない Kubernetes リソースを使用するアプリケーションには、アプリケーションのデータ管理機能がフル装備されていない可能性があります。Astra Control では、次の Kubernetes リソースが収集されます。

クラスタロール	ClusterRoleBinding	ConfigMap
cronjob	CustomResourceDefinition の場合	CustomResource の場合
デモンセット（ DemonSet ）	DeploymentConfig	HorizontalPodAutoscaler のように表示されます
入力	MutingWebhook	ネットワークポリシー
PersistentVolumeClaim のように表示され	ポッド	PodDisruptionBudget （予算の廃止）
PodTemplate	ReplicaSet	ロール
RoleBinding です	ルート	秘密
サービス	サービスアカウント	Stateful役立つ セット
検証 Webhook		

レプリケーションの前提条件

Astra Controlアプリケーションのレプリケーションを開始するには、次の前提条件を満たしている必要があります。

- シームレスな災害復旧を実現するために、第3の障害ドメインまたはセカンダリサイトにAstra Control Centerを導入することをお勧めします。
- アプリケーションのホストKubernetesクラスタとデスティネーションKubernetesクラスタが使用可能であり、2つのONTAP クラスタに接続されている必要があります。理想的には別々の障害ドメインまたはサイトに配置できます。
- ONTAP クラスタとホストSVMをペアリングする必要があります。を参照してください "クラスタと SVM のピアリングの概要"。
- ペアリングされているリモートSVMがデスティネーションクラスタのTridentで使用可能である必要があります。
- ソースとデスティネーションの両方のONTAP クラスタにTridentバージョン22.07以降が存在する必要があります。
- ソースとデスティネーションの両方のONTAP クラスタで、データ保護バンドルを使用したONTAP SnapMirror非同期ライセンスが有効になっている必要があります。を参照してください "ONTAP のSnapMirrorライセンスの概要"。
- ONTAP ストレージバックエンドをAstra Control Centerに追加する場合は、「admin」ロールでユーザクレデンシャルを適用します。このロールにはアクセス方法があります `http` および `ontapi` 両方のONTAP クラスタで有効にしてください。を参照してください "ユーザアカウントを管理する" を参照してください。
- ソースとデスティネーションの両方のKubernetesクラスタとONTAP クラスタをAstra Controlで管理する必要があります。

> （別のクラスタまたはサイトで実行されている）別のアプリケーションを逆方向に同時にレプリケートできます。たとえば、アプリケーションA、B、Cはデータセンター1からデータセンター2にレプリケートでき、アプリケーションX、Y、Zはデータセンター2からデータセンター1にレプリケートできます。

方法をご確認ください "SnapMirrorテクノロジを使用してアプリケーションをリモートシステムにレプリケート"。

サポートされているアプリケーションのインストール方法

Astra Control は、次のアプリケーションインストール方法をサポートしています。

- * マニフェストファイル * ： Astra Control は、 kubectl を使用してマニフェストファイルからインストールされたアプリケーションをサポートします。例：

```
kubectl apply -f myapp.yaml
```

- * Helm 3 * ： Helm を使用してアプリケーションをインストールする場合、 Astra Control には Helm バージョン 3 が必要です。Helm 3 （または Helm 2 から Helm 3 にアップグレード）を使用してインストールされたアプリケーションの管理とクローニングが完全にサポートされています。Helm 2 でインストールされたアプリケーションの管理はサポートされていません。
- * オペレータが導入したアプリケーション * ： Astra Control は、名前空間を対象とした演算子を使用してインストールされたアプリケーションをサポートします。このインストールモデルで検証されたアプリケーションには、次のものがあります。

- "Apache K8ssandra"
- "Jenkins CI"
- "Percona XtraDB クラスタ"

> インストールする演算子とアプリケーションは、同じ名前空間を使用する必要があります。このような名前空間を使用するには、演算子の deployment.yaml ファイルを変更する必要があります。

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"。

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"。

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"。

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、設定済みでない場合は、 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。

> 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。

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

このページでは、 Astra Control Center の導入に必要な手順の概要を説明します。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。

ぜひお試しください。Astra Control Center を試す場合は、 90 日間の評価ライセンスを使用できます。を参照してください "ライセンス情報" を参照してください。

1 **Kubernetes** クラスタの要件を確認

- Astra は、 Trident が設定された ONTAP ストレージバックエンドまたは Astra データストアストレージバックエンドを使用して、 Kubernetes クラスタと連携します。
- クラスタが正常な状態で稼働し、少なくとも 3 つのオンラインワーカーノードが必要です。
- クラスタで Kubernetes が実行されている必要があります。

の詳細については、を参照してください "Astra Control Center の要件"。

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

- から Astra Control Center をダウンロードします "NetApp Support Siteの「 Astra Control Center Downloads 」ページ"。
- Astra Control Center をローカル環境にインストールします。

  必要に応じて、 Red Hat OperatorHub を使用して Astra Control Center をインストールします。

  必要に応じて、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。

の詳細を確認してください "Astra Control Centerのインストール"。

3 いくつかの初期セットアップ作業を完了します

- Astra Controlライセンスとサポートする ONTAP ライセンスを追加
- Kubernetes クラスタと Astra Control Center を追加すると、詳細が検出されます。
- ONTAP ストレージバックエンドを追加します。
- 必要に応じて、アプリケーションバックアップを格納するオブジェクトストアバケットを追加します。

の詳細については、を参照してください "初期セットアッププロセス"。

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。

- アプリを管理します。方法の詳細については、こちらをご覧ください "アプリの管理"。

- アプリの保護ポリシーの設定、リモートシステムへのアプリのレプリケーション、アプリのクローニングと移行により、アプリを保護します。方法の詳細については、こちらをご覧ください "アプリを保護します"。
- アカウントの管理（ユーザ、ロール、ユーザ認証用のLDAP、クレデンシャル、リポジトリ接続など）方法の詳細については、こちらをご覧ください "ユーザを管理します"。
- 必要に応じて、 NetApp Cloud Insights に接続し、 Astra Control Center UI 内のシステム、容量、およびスループットの健全性に関する指標を表示します。の詳細を確認してください "Cloud Insights に接続しています"。

5 このクイックスタートから続行します

"Astra Control Center をインストールします"。

詳細については、こちらをご覧ください

- "Astra Control API を使用"

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"

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、次の手順を実行して、 Astra Control Center Operator と Astra Control Center を環境にインストールします。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。

Red Hat OpenShift環境では、を使用できます "代替手順" OpenShift OperatorHub を使用して Astra Control Center をインストールします。

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"。
- すべてのクラスタオペレータ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 clusteroperators
```

- 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cert-manager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。

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、次のことが実行されます。

- Astraコンポーネントをにインストールします `netapp-acc` （またはカスタム名）ネームスペース。
- デフォルトアカウントを作成します。
- デフォルトの管理ユーザのEメールアドレスとデフォルトのワンタイム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システムの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実行されていることを確認するのに役立ちます。
- Astra の UI をインストールします。

> （環境 the Astra Data Store Early Access Program（EAP）リリースのみ）Astra Control Center を使用してAstraデータストアを管理し、VMwareワークフローを有効にする場合は、Astra Control Centerのみをに導入します `pcloud` ではなく、ネームスペースで指定します `netapp-acc` この手順 の手順で説明するネームスペースまたはカスタムネームスペース。

> すべてのAstra Control Centerポッドが削除されないようにするため、インストールプロセス全体で次のコマンドを実行しないでください。 `kubectl delete -f astra_control_center_operator_deploy.yaml`

> Docker Engine の代わりに Red Hat の Podman を使用している場合は、 Docker コマンドの代わりに Podman コマンドを使用できます。

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バンドルをダウンロードして開梱します
- 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バンドルをダウンロードして開梱します

1. Astra Control Center バンドルをダウンロードします (`astra-control-center-[version].tar.gz`) をクリックします "NetApp Support Site"。
2. から Astra Control Center 証明書とキーの zip をダウンロードします "NetApp Support Site"。
3. （任意）次のコマンドを使用して、バンドルのシグニチャを確認します。

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

4. 画像を抽出します。

```
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

ネットアップアストラ `kubectl` コマンドラインプラグインにより、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時間を節約できます。

必要なもの

ネットアップでは、プラグイン用のバイナリを提供しており、CPUアーキテクチャやオペレーティングシステムが異なる場合はそのプラグインを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。LinuxおよびMacオペレーティングシステムでは、を使用できます `uname -a` この情報を収集するためのコマンドです。

手順

1. 使用可能なネットアップAstraを選択します `kubectl` プラグインバイナリ。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。

```
ls kubectl-astra/
```

2. ファイルを標準と同じ場所にコピーします `kubectl` ユーティリティ。この例では、を使用しています `kubectl` ユーティリティはにあります `/usr/local/bin` ディレクトリ。交換してください `<binary-name>` 必要なファイル名を使用して、次の操作を実行します。

```
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. Astraディレクトリに移動します。

```
cd acc
```

2. [[[[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1></Z1>_image_local_registry_push]]]]]]]]]</Z2>アストラControl Centerイメージディレクトリ内の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。</Z3>コマンドを実行する前に、次の置き換えを行ってください。
   - bundle_fileをAstra Controlバンドルファイルの名前に置き換えます（例： `acc.manifest.yaml` ）。
   - my_registryをDockerリポジトリのURLに置き換えます。
   - my_registry_userをユーザー名に置き換えます。
   - 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_REGISTRY
-u MY_REGISTRY_USER -p MY_REGISTRY_TOKEN
```

ポドマン

1. レジストリにログインします。

```
podman login [your_registry_path]
```

2. 次のスクリプトを実行して、コメントに記載されているように<your _registry>を置き換えます。

```
# You need to be at the root of the tarball.
# You should see these files to confirm correct location:
#   acc.manifest.yaml
#   acc/

# Replace <YOUR_REGISTRY> with your own registry (e.g
registry.customer.com or registry.customer.com/testing, etc..)
export REGISTRY=<YOUR_REGISTRY>
export PACKAGENAME=acc
export PACKAGEVERSION=22.08.1-26
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
  # Load to local cache
  astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //')

  # Remove path and keep imageName.
  astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')

  # Tag with local image repo.
  podman tag ${astraImage} ${REGISTRY}/netapp/astra/${PACKAGENAME}
/${PACKAGEVERSION}/${astraImageNoPath}

  # Push to the local repo.
  podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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]
```

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/22.08.1-26` ）。

```
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する場合は、ネームスペースが再作成されたあとにネームスペースのシークレットを再生成する必要があります。

c. を作成します `netapp-acc` （またはカスタムの名前付き）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` （またはカスタムの名前付き）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
```

a. [[[[sup_kubeconfig _secret]] （オプション）インストール後に Astra Control Center でクラスタを自動的に管理する場合は、このコマンドを使用して展開する Astra Control Center ネームスペース内のシークレットとして kubeconfig を指定する必要があります。

```
kubectl create secret generic [acc-kubeconfig-cred or custom secret
name] --from-file=<path-to-your-kubeconfig> -n [netapp-acc or custom
namespace]
```

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。

d. （Astra データストアプレビューを使用するインストールの場合）この問題に関する既知の情報を参照してください "ストレージクラスのプロビジョニングと YAML に対する追加の変更"。

```
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
  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
        command:
        - /manager
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        image: [your_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
      imagePullSecrets: []
```

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_min.yaml`)アカウント、AutoSupport 、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。

> `astra_control_center_min.yaml` はデフォルトのCRで、ほとんどのインストールに適しています。すべてのことをよく理解してください "CRオプションとその可能性のある値" お客様の環境にAstra Control Centerを正しく導入できるようにするため。環境で追加のカスタマイズが必要な場合は、を使用できます `astra_control_center.yaml` 代替CRとして。

```
vim astra_control_center_min.yaml
```

> 許可が不要なレジストリを使用している場合は、を削除する必要があります `secret` ラインの内側 `imageRegistry` または、インストールが失敗します。

a. 変更 `[your_registry_path]` 前の手順でイメージをプッシュしたレジストリパスに移動します。

b. を変更します `accountName` stringには、アカウントに関連付ける名前を指定します。

c. を変更します `astraAddress` ブラウザからAstraにアクセスする際に使用するFQDNを文字列で指定します。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。

d. 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。

e. 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。

f. 外部証明書マネージャを使用する場合は、に次の行を追加します `spec`：

```
spec:
  crds:
    externalCertManager: true
```

g. （オプション）名を追加します `firstName` 姓を入力します `lastName` アカウントに関連付けられているユーザのこの手順は、 UI ですぐに実行することもあとで実行することもできます。

h. （オプション）を変更します `storageClass` インストールに必要な場合、別のTridentストレージクラスリソースへの値。

i. （オプション）インストール後に Astra Control Center でクラスタを自動的に管理する場合は このクラスタの kubeconfig を含むシークレットを作成しましたをクリックし、という名前のYAMLファイルに新しいフィールドを追加して、シークレットの名前を指定します `astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"`

j. 次のいずれかの手順を実行します。

- * その他の入力コントローラ（ ingressType: Generic ） * ：これはアストラコントロールセンターでのデフォルトのアクションです。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。

  デフォルトのAstra Control Centerインストールでは、ゲートウェイがセットアップされます（`service/traefik`）を入力します `ClusterIP`。このデフォルトのインストールでは、トラフィックをルーティングするために Kubernetes IngressController/Ingress を追加で設定する必要があります。入力を使用する場合は、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"。

- サービスロードバランサ（**ingressType: AccTrafik**）: IngressControllerをインストールしない場合、または入力リソースを作成しない場合は、を設定します `ingressType` 終了：`AccTraefik`。

  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。

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"。

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"
  ingressType: "Generic"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
```

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_min.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

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。

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-6b44d68d94-d8m55              1/1     Running   0
13m
activity-78f99ddf8-hltct                    1/1     Running   0
10m
api-token-authentication-457nl              1/1     Running   0
9m28s
api-token-authentication-dgwsz              1/1     Running   0
9m28s
api-token-authentication-hmqqc              1/1     Running   0
9m28s
asup-75fd554dc6-m6qzh                       1/1     Running   0
9m38s
authentication-6779b4c85d-92gds             1/1     Running   0
8m11s
bucketservice-7cc767f8f8-lqwr8              1/1     Running   0
9m31s
certificates-549fd5d6cb-5kmd6               1/1     Running   0
9m56s
certificates-549fd5d6cb-bkjh9               1/1     Running   0
9m56s
cloud-extension-7bcb7948b-hn8h2             1/1     Running   0
10m
cloud-insights-service-56ccf86647-fgg69     1/1     Running   0
9m46s
composite-compute-677685b9bb-7vgsf          1/1     Running   0
10m
composite-volume-657d6c5585-dnq79           1/1     Running   0
9m49s
credentials-755fd867c8-vrlmt                1/1     Running   0
11m
entitlement-86495cdf5b-nwhh2                1/1     Running   2
10m
features-5684fb8b56-8d6s8                   1/1     Running   0
10m
fluent-bit-ds-rhx7v                         1/1     Running   0
7m48s
fluent-bit-ds-rjms4                         1/1     Running   0
7m48s
fluent-bit-ds-zf5ph                         1/1     Running   0
7m48s
graphql-server-66d895f544-w6hjd             1/1     Running   0
```

```
3m29s
identity-744df448d5-rlcmm                1/1     Running   0
10m
influxdb2-0                              1/1     Running   0
13m
keycloak-operator-75c965cc54-z7csw       1/1     Running   0
8m16s
krakend-798d6df96f-9z2sk                 1/1     Running   0
3m26s
license-5fb7d75765-f8mjg                 1/1     Running   0
9m50s
login-ui-7d5b7df85d-l2s7s                1/1     Running   0
3m20s
loki-0                                   1/1     Running   0
13m
metrics-facade-599b9d7fcc-gtmgl          1/1     Running   0
9m40s
monitoring-operator-67cc74f844-cdplp     2/2     Running   0
8m11s
nats-0                                   1/1     Running   0
13m
nats-1                                   1/1     Running   0
13m
nats-2                                   1/1     Running   0
12m
nautilus-769f5b74cd-k5jxm                1/1     Running   0
9m42s
nautilus-769f5b74cd-kd9gd                1/1     Running   0
8m59s
openapi-84f6ccd8ff-76kvp                 1/1     Running   0
9m34s
packages-6f59fc67dc-4g2f5                1/1     Running   0
9m52s
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   0
8m7s
polaris-keycloak-1                       1/1     Running   0
5m49s
polaris-keycloak-2                       1/1     Running   0
5m15s
polaris-keycloak-db-0                    1/1     Running   0
```

```
8m6s
polaris-keycloak-db-1                     1/1     Running   0
5m49s
polaris-keycloak-db-2                     1/1     Running   0
4m57s
polaris-mongodb-0                         2/2     Running   0
13m
polaris-mongodb-1                         2/2     Running   0
12m
polaris-mongodb-2                         2/2     Running   0
12m
polaris-ui-565f56bf7b-zwr8b               1/1     Running   0
3m19s
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-6d86d66444-2wbzl           1/1     Running   0
9m30s
storage-backend-metrics-77c5d98dcd-dbhg5 1/1     Running   0
9m44s
storage-provider-78c885f57c-6zcv4         1/1     Running   0
9m36s
telegraf-ds-2l2m9                         1/1     Running   0
7m48s
telegraf-ds-qfzgh                         1/1     Running   0
7m48s
telegraf-ds-shrms                         1/1     Running   0
7m48s
telegraf-rs-bjpkt                         1/1     Running   0
7m48s
telemetry-service-6684696c64-qzfdf        1/1     Running   0
10m
tenancy-6596b6c54d-vmpsm                  1/1     Running   0
10m
traefik-7489dc59f9-6mnst                  1/1     Running   0
3m19s
traefik-7489dc59f9-xrkgg                  1/1     Running   0
3m4s
trident-svc-6c8dc458f5-jswcl              1/1     Running   0
10m
vault-controller-6b954f9b76-gz9nm         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示された場合は、クラスタ追加ワークフローを通じて再度登録を試行できます "UI で" 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にログインする際に使用するワンタイムパスワードを取得します。

```
kubectl get AstraControlCenter -n netapp-acc
```

対応：

```
NAME    UUID                                        VERSION     ADDRESS
READY
astra   ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f  22.08.1-26
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

Kubernetes 入力コントローラをセットアップして、クラスタ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できます。

この手順 では、入力コントローラの設定方法について説明します (`ingressType:Generic`）。これは、Astra Control Center でのデフォルトのアクションです。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。

> 入力コントローラを設定しない場合は、を設定できます `ingressType:AccTraefik`)。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"。

この手順は、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。

- 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。
- V1.19 と v1.22 の間で Kubernetes のバージョンを使用している。

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. 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ド）ネームスペースで、v1beta1（Kubernetesバージョンで1.22未満で廃止）またはv1リソースタイプを使用して、非推奨または新しいスキーマのいずれかに対応します。

   出力：

```
apiVersion: networking.k8s.io/v1beta1
kind: IngressClass
metadata:
  name: istio
spec:
  controller: istio.io/ingress-controller
---
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
  namespace: istio-system
spec:
  ingressClassName: istio
  tls:
  - hosts:
    - <ACC addess>
    secretName: [tls secret name]
  rules:
  - host: [ACC addess]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          serviceName: traefik
          servicePort: 80
```

v1の新しいスキーマについては、次の例を参照してください。

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

出力：

```
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: istio-system
spec:
  ingressClassName: istio
  tls:
  - hosts:
    - <ACC addess>
    secretName: [tls secret name]
  rules:
  - host: [ACC addess]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: traefik
            port:
              number: 80
```

5. 通常どおりAstra Control Centerを導入します。
6. 入力のステータスを確認します。

```
kubectl get ingress -n netapp-acc
```

対応：

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

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`]をクリックします `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` （またはカスタムの名前付き）ネームスペースのいずれかを使用します `v1beta1` （Kubernetesのバージョンが1.22より前の場合は廃止）または `v1` 非推奨または新しいスキーマのリソースタイプ：

   a. に設定します `v1beta1` 廃止されたスキーマの例を次に示します。

```
apiVersion: extensions/v1beta1
Kind: IngressClass
metadata:
  name: ingress-acc
  namespace: [netapp-acc or custom namespace]
  annotations:
    kubernetes.io/ingress.class: [class name for nginx controller]
spec:
  tls:
  - hosts:
    - <ACC address>
    secretName: [tls secret name]
  rules:
  - host: [ACC address]
    http:
      paths:
      - backend:
        serviceName: traefik
        servicePort: 80
        pathType: ImplementationSpecific
```

   b. をクリックします `v1` 新しいスキーマの例を次に示します。

```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 addess>
    http:
      paths:
        - path:
          backend:
            service:
              name: traefik
              port:
                number: 80
          pathType: ImplementationSpecific
```

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、で使用したFQDNを入力します `astraAddress` を参照してください `astra_control_center_min.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、自己署名証明書を受け入れます。

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_min.yaml` CR When（時間） Astra Control Center をインストールしたをクリックし、続けてワンタイム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。

   初めてログインする際にパスワードを忘れた場合、他の管理ユーザアカウントがまだ作成されていないときは、ネットアップのサポートに問い合わせて、パスワードのリカバリに関するサポートを依頼してください。

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 --follow -n netapp-acc-operator $(kubectl get pods -n
netapp-acc-operator -o name) -c manager
```

次のステップ

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:

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

Astra Control Centerは、PoDセキュリティポリシー（PSP）による特権制限をサポートします。ポッドセキュリティポリシーを使用すると、ユーザまたはグループがコンテナを実行できる対象や、それらのコンテナに付与できる権限を制限できます。

RKE2などの一部のKubernetesディストリビューションには、デフォルトのポッドセキュリティポリシーが用意されていますが、制限が厳しく、Astra Control Centerのインストール時に問題が発生します。

ここに記載されている情報と例を使用して、Astra Control Centerによって作成されるポッドセキュリティポリシーを理解し、Astra Control Centerの機能を妨げずに必要な保護を提供するポッドセキュリティポリシーを設定できます。

Astra Control Centerによってインストールされた**PSP**

Astra Control Centerは、インストール中にポッドのセキュリティポリシーをいくつか作成します。これらの中には永続的なものもあれば、一部のものは特定の処理中に作成され、処理が完了すると削除されます。

インストール中に作成された**PSP**

Astra Control Centerのインストール中に、Astra Control Centerオペレータは、Astra Control Centerネームスペースへの Astra Control Centerサービスの展開をサポートするために、カスタムのPoDセキュリティポリシー、Roleオブジェクト、およびRoleBindingオブジェクトをインストールします。

新しいポリシーとオブジェクトには次の属性があります。

```
kubectl get psp

NAME                            PRIV    CAPS          SELINUX    RUNASUSER
FSGROUP     SUPGROUP    READONLYROOTFS   VOLUMES
avp-psp                         false                 RunAsAny   RunAsAny
RunAsAny    RunAsAny    false            *
netapp-astra-deployment-psp     false                 RunAsAny   RunAsAny
RunAsAny    RunAsAny    false            *

kubectl get role

NAME                                     CREATED AT
netapp-astra-deployment-role             2022-06-27T19:34:58Z

kubectl get rolebinding

NAME                                     ROLE
AGE
netapp-astra-deployment-rb               Role/netapp-astra-deployment-role
32m
```

バックアップ処理中に作成された**PSP**

バックアップ処理中に、Astra Control Centerは、動的なPODセキュリティポリシー、ClusterRoleオブジェクト、およびRoleBindingオブジェクトを作成します。これらの機能により、別のネームスペースで実行されるバックアッププロセスがサポートされます。

新しいポリシーとオブジェクトには次の属性があります。

```
kubectl get psp

NAME                            PRIV    CAPS
SELINUX    RUNASUSER          FSGROUP     SUPGROUP    READONLYROOTFS
VOLUMES
netapp-astra-backup             false   DAC_READ_SEARCH
RunAsAny   RunAsAny           RunAsAny    RunAsAny    false            *

kubectl get role

NAME                   CREATED AT
netapp-astra-backup    2022-07-21T00:00:00Z

kubectl get rolebinding

NAME                   ROLE                        AGE
netapp-astra-backup    Role/netapp-astra-backup    62s
```

クラスタ管理中に作成された**PSP**

クラスターを管理する場合、Astra Control Centerは管理対象クラスタにNetApp Monitoringオペレータをインストールします。この演算子は、PODセキュリティポリシー、ClusterRoleオブジェクト、およびRoleBindingオブジェクトを作成して、Astra Control Center名前空間にテレメトリサービスを展開します。

新しいポリシーとオブジェクトには次の属性があります。

```
kubectl get psp

NAME                            PRIV    CAPS
SELINUX    RUNASUSER          FSGROUP     SUPGROUP    READONLYROOTFS
VOLUMES
netapp-monitoring-psp-nkmo      true    AUDIT_WRITE,NET_ADMIN,NET_RAW
RunAsAny   RunAsAny           RunAsAny    RunAsAny    false            *

kubectl get role

NAME                                            CREATED AT
netapp-monitoring-role-privileged               2022-07-21T00:00:00Z

kubectl get rolebinding

NAME                                                    ROLE
AGE
netapp-monitoring-role-binding-privileged               Role/netapp-
monitoring-role-privileged              2m5s
```

ネームスペース間のネットワーク通信を有効にします

一部の環境では、NetworkPolicy構造体を使用してネームスペース間のトラフィックを制限します。Astra Control Centerオペレータ、Astra Control Center、およびAstra Plugin for VMware vSphereはすべて異なる名前空間に存在します。これらの異なるネームスペース内のサービスは、相互に通信できる必要があります。この通信をイネーブルにするには、次の手順を実行します。

手順

1. Astra Control Center名前空間に存在するNetworkPolicyリソースをすべて削除します。

```
kubectl get networkpolicy -n netapp-acc
```

2. 前のコマンドで返された各NetworkPolicyオブジェクトについて、次のコマンドを使用して削除します。object_name >を、返されたオブジェクトの名前に置き換えます。

```
kubectl delete networkpolicy <OBJECT_NAME> -n netapp-acc
```

3. 次のリソースファイルを適用して、Astra Plugin for VMware vSphereサービスがAstra Control Centerサービスに要求を送信できるように、accネットワーキングポリシーオブジェクトを設定します。角かっこ内の情報を環境内の情報に置き換えます。

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: acc-avp-network-policy
  namespace: <ACC_NAMESPACE_NAME> # REPLACE THIS WITH THE ASTRA CONTROL
CENTER NAMESPACE NAME
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              kubernetes.io/metadata.name: <PLUGIN_NAMESPACE_NAME> #
REPLACE THIS WITH THE ASTRA PLUGIN FOR VMWARE VSPHERE NAMESPACE NAME
```

4. 次のリソースファイルを適用して、Astra Control CenterオペレータがAstra Control Centerサービスと通信できるように、acc-operator-network-policyオブジェクトを設定します。角かっこ内の情報を環境内の情報に置き換えます。

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: acc-operator-network-policy
  namespace: <ACC_NAMESPACE_NAME> # REPLACE THIS WITH THE ASTRA CONTROL
CENTER NAMESPACE NAME
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              kubernetes.io/metadata.name: <NETAPP-ACC-OPERATOR> #
REPLACE THIS WITH THE OPERATOR NAMESPACE NAME
```

リソースの制限を解除します

一部の環境では、ResourceQuotasオブジェクトとLimitRangesオブジェクトを使用して、ネームスペース内のリソースがクラスタ上の使用可能なCPUとメモリをすべて消費しないようにします。Astra Control Centerでは上限が設定されていないため、これらのリソースに準拠していません。Astra Control Centerをインストールするネームスペースから削除する必要があります。

これらのクォータと制限を取得および削除するには、次の手順を実行します。これらの例では、コマンド出力はコマンド出力の直後に表示されます。

手順

1. NetApp-accネームスペース内のリソースクォータを取得します。

```
kubectl get quota -n netapp-acc
```

対応：

```
NAME          AGE   REQUEST                                          LIMIT
pods-high     16s   requests.cpu: 0/20, requests.memory: 0/100Gi
limits.cpu: 0/200, limits.memory: 0/1000Gi
pods-low      15s   requests.cpu: 0/1, requests.memory: 0/1Gi
limits.cpu: 0/2, limits.memory: 0/2Gi
pods-medium   16s   requests.cpu: 0/10, requests.memory: 0/20Gi
limits.cpu: 0/20, limits.memory: 0/200Gi
```

2. 名前別にすべてのリソースクォータを削除します。

```
kubectl delete resourcequota  pods-high -n netapp-acc
```

```
kubectl delete resourcequota  pods-low -n netapp-acc
```

```
kubectl delete resourcequota  pods-medium -n netapp-acc
```

3. NetApp-accネームスペース内の制限範囲を取得します。

```
kubectl get limits -n netapp-acc
```

対応：

```
NAME               CREATED AT
cpu-limit-range    2022-06-27T19:01:23Z
```

4. 制限範囲を名前で削除します。

```
kubectl delete limitrange cpu-limit-range -n netapp-acc
```

=
: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

- "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"。
- 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 (`available` はです `true`）：

```
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 (`available` はです

true）：

```
oc get apiservices
```

- データセンターにAstra Control CenterのFQDNアドレスを作成します。
- 説明した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。
- クラスタにcert-manager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。

手順

- Astra Control Centerバンドルをダウンロードして開梱します
- 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バンドルをダウンロードして開梱します

1. Astra Control Center バンドルをダウンロードします (`astra-control-center-[version].tar.gz`) をクリックします "NetApp Support Site"。
2. から Astra Control Center 証明書とキーの zip をダウンロードします "NetApp Support Site"。
3. （任意）次のコマンドを使用して、バンドルのシグニチャを確認します。

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

4. 画像を抽出します。

```
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

ネットアップアストラ `kubectl` コマンドラインプラグインにより、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時間を節約できます。

必要なもの

ネットアップでは、プラグイン用のバイナリを提供しており、CPUアーキテクチャやオペレーティングシステムが異なる場合はそのプラグインを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。LinuxおよびMacオペレーティングシステムでは、を使用できます `uname -a` この情報を収集するためのコマンドです。

手順

1. 使用可能なネットアップAstraを選択します `kubectl` プラグインバイナリ。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。

```
ls kubectl-astra/
```

2. ファイルを標準と同じ場所にコピーします `kubectl` ユーティリティ。この例では、を使用しています `kubectl` ユーティリティはにあります `/usr/local/bin` ディレクトリ。交換してください `<binary-name>` 必要なファイル名を使用して、次の操作を実行します。

```
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. Astraディレクトリに移動します。

```
cd acc
```

2. [[[[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1></Z1>_image_local_registry_push]]]]]]]]]</Z2>アストラControl Centerイメージディレクトリ内の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。</Z3>コマンドを実行する前に、次の置き換えを行ってください。
   - bundle_fileをAstra Controlバンドルファイルの名前に置き換えます（例： `acc.manifest.yaml` ）。
   - my_registryをDockerリポジトリのURLに置き換えます。
   - my_registry_userをユーザー名に置き換えます。
   - 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_REGISTRY
-u MY_REGISTRY_USER -p MY_REGISTRY_TOKEN
```

ポドマン

1. レジストリにログインします。

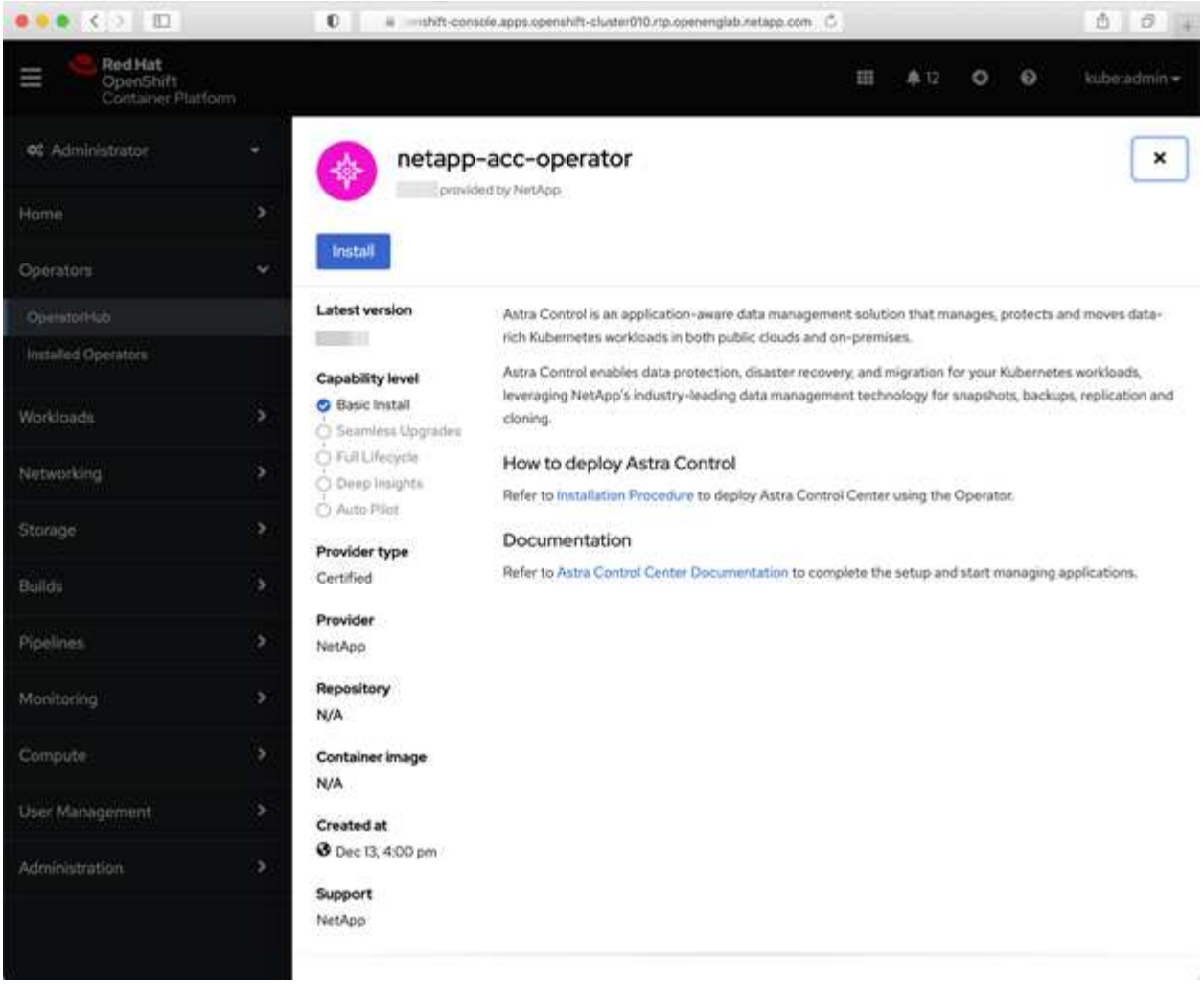
```
podman login [your_registry_path]
```

2. 次のスクリプトを実行して、コメントに記載されているように<your _registry>を置き換えます。

```
# You need to be at the root of the tarball.
# You should see these files to confirm correct location:
#   acc.manifest.yaml
#   acc/

# Replace <YOUR_REGISTRY> with your own registry (e.g
registry.customer.com or registry.customer.com/testing, etc..)
export REGISTRY=<YOUR_REGISTRY>
export PACKAGENAME=acc
export PACKAGEVERSION=22.08.1-26
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
  # Load to local cache
  astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //')

  # Remove path and keep imageName.
  astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')

  # Tag with local image repo.
  podman tag ${astraImage} ${REGISTRY}/netapp/astra/${PACKAGENAME}
/${PACKAGEVERSION}/${astraImageNoPath}

  # Push to the local repo.
  podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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



：

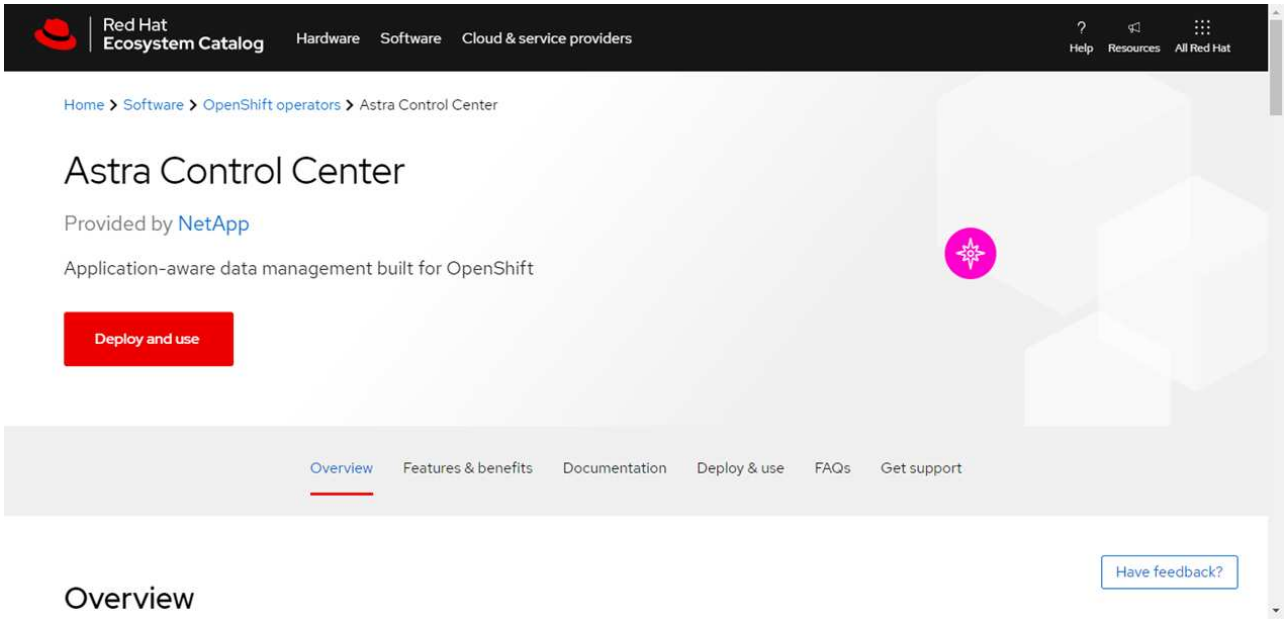
i. OpenShift Container Platform UI にログインします。

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 オペレータを選択します。

iv. 「 * Install * 」を選択します。

- 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. コンソールのAstra Control Centerオペレータの詳細ビューで、を選択します `Create instance` を参照してください。

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. （オプション） 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。

c. Astra Control Center の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。

d. Astra Control Center のバージョンを入力します。たとえば、 21.12.60 と入力します。

e. 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。

f. デフォルトのボリューム再利用ポリシーをそのまま使用します。

g. * 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。

h. 認証が必要なレジストリを使用する場合は、シークレットを入力します。

i. 管理者の名を入力します。

j. リソースの拡張を構成する。

k. デフォルトのストレージクラスは保持します。

l. CRD 処理の環境設定を定義します。

3. 選択するオプション `Create`。

次のステップ

Astra Control Center が正しくインストールされたことを確認し、を完了します "残りのステップ" ログインしてください。さらに、の導入も完了します "セットアップのタスク"。

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 （ NetApp Cloud Manager 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 では、 Kubernetes クラスタを NetApp Cloud Manager にインポートする際に作成される次の ONTAP Kubernetes ストレージクラスがサポートされます。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 Cloud Manager を設定します。

5. Astra Control Center をインストールします。

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 Cloud Manager Connectorをインストールできる十分な数の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"。

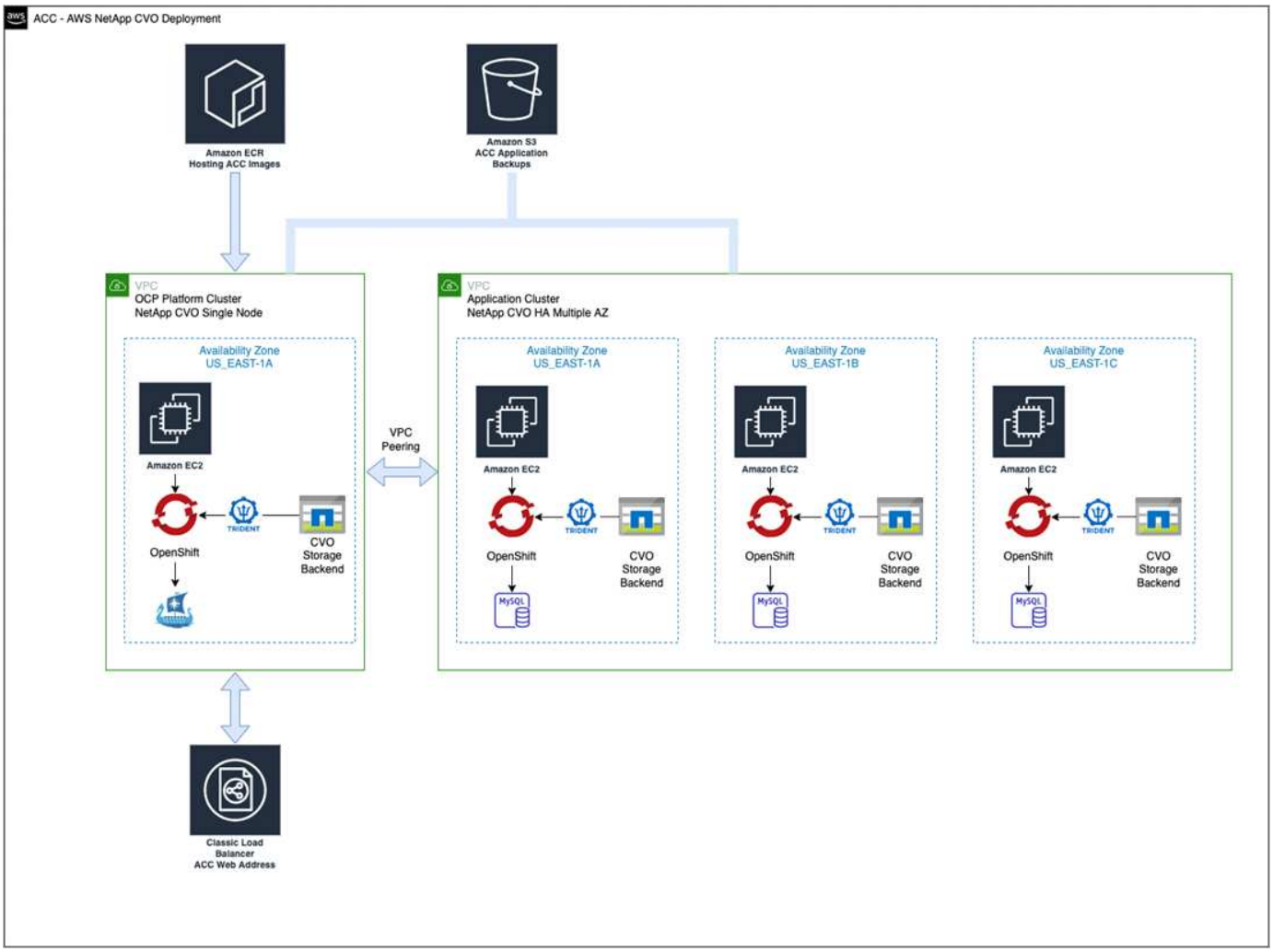
1. AWS仮想ネットワークを作成します。
2. EC2 コンピューティングインスタンスを確認します。AWS ではベアメタルサーバまたは VM を使用できます。
3. インスタンスタイプが、マスターノードとワーカーノードの Astra の最小リソース要件に一致していない場合は、 Astra の要件に合わせて AWS でインスタンスタイプを変更します。 を参照してください "Astra Control Center の要件"。
4. バックアップを格納する AWS S3 バケットを少なくとも 1 つ作成します。
5. すべての ACC イメージをホストする AWS Elastic Container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

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。

Cloud Manager のドキュメントに従って、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"。
- "Cloud Manager を使用して AWS でコネクタを作成します"

手順

1. Cloud Manager にクレデンシャルを追加します。
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 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"。

> 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 （ **NetApp Cloud Manager** 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 では、 Kubernetes クラスタを NetApp Cloud Manager にインポートする際に作成される次の ONTAP Kubernetes ストレージクラスがサポートされます。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 Cloud Manager を設定します。
6. Astra Control Center をインストールして設定します。

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 Cloud Manager Connectorをインストールできる十分な数の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 Cloud Manager を設定します

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

Cloud Manager 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 "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. Cloud Manager にクレデンシャルを追加します。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 "Cloud Manager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 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 Cloud Manager** 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 では、 Kubernetes クラスタを NetApp Cloud Manager にインポートする際に作成される次の ONTAP Kubernetes ストレージクラスがサポートされます。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 Cloud Manager を設定します。
6. Astra Control Center をインストールして設定します。

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、のRedHatのマニュアルを参照してください "AzureにOpenShiftクラスタをインストールしています" および "Azureアカウントをインストールしています"。

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 Cloud Manager Connectorをインストールできる十分な数の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 Cloud Manager を設定します

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。

Cloud Manager 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 で Cloud Manager を使用する準備をしています"。

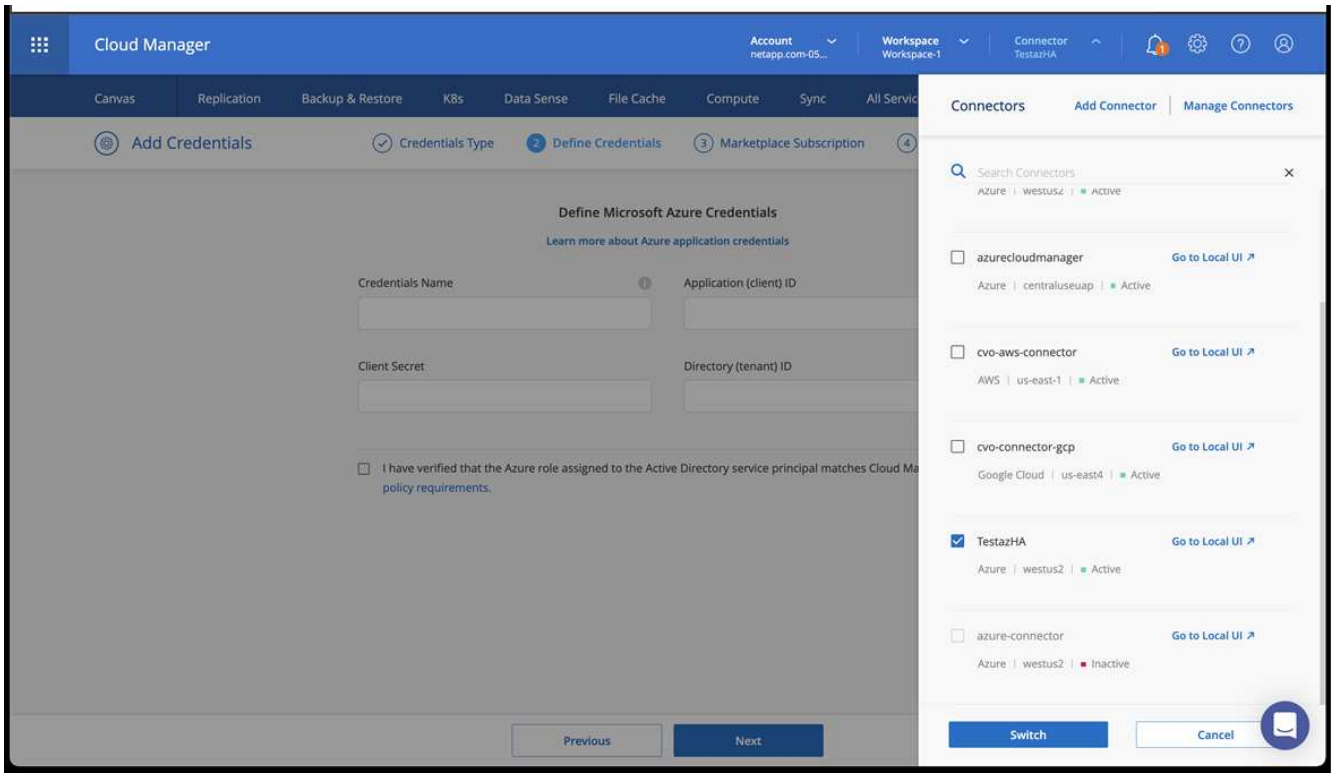
必要なもの

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. Cloud Manager にクレデンシャルを追加します。
2. Azure 用のコネクタを追加します。を参照してください "Cloud Manager のポリシー"。
   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。

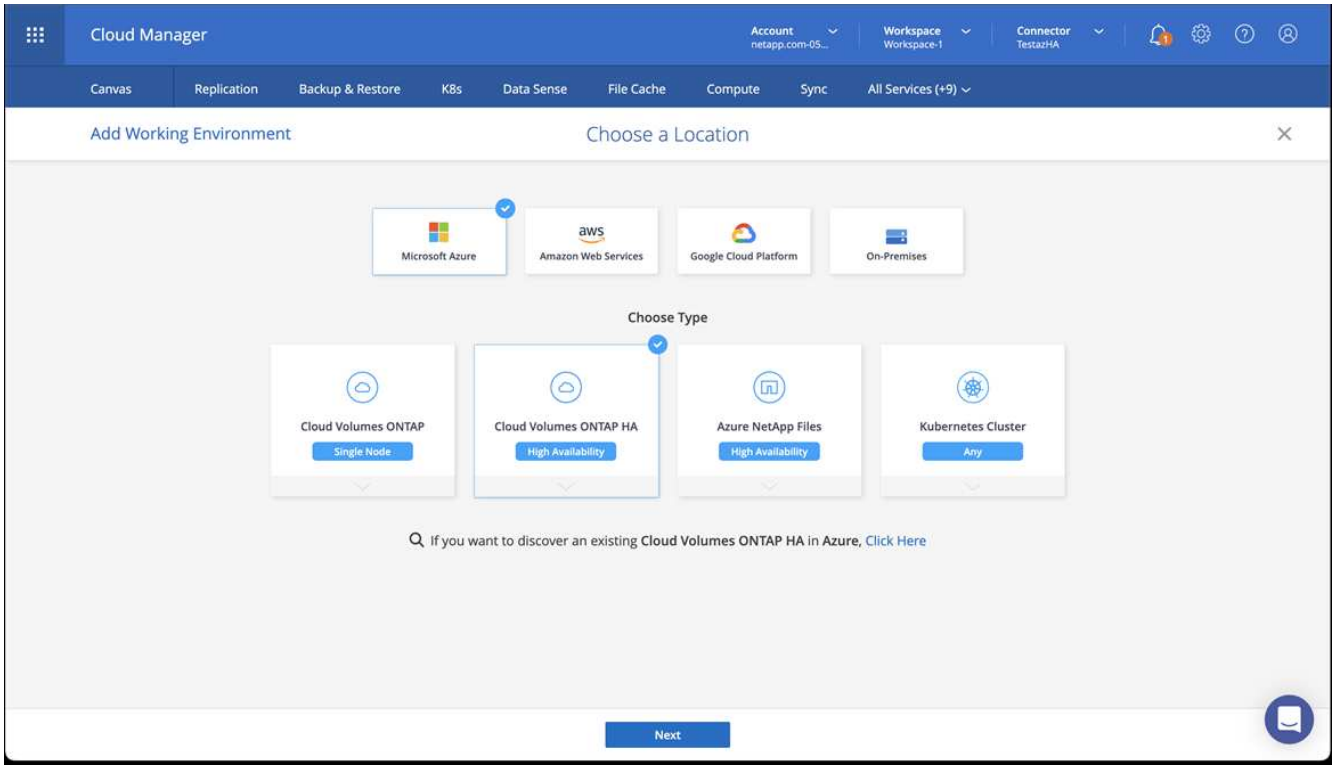
      を参照してください "Cloud Manager から Azure にコネクタを作成する"。
3. コネクタが動作していることを確認し、コネクタに切り替えます。



4. クラウド環境の作業環境を構築

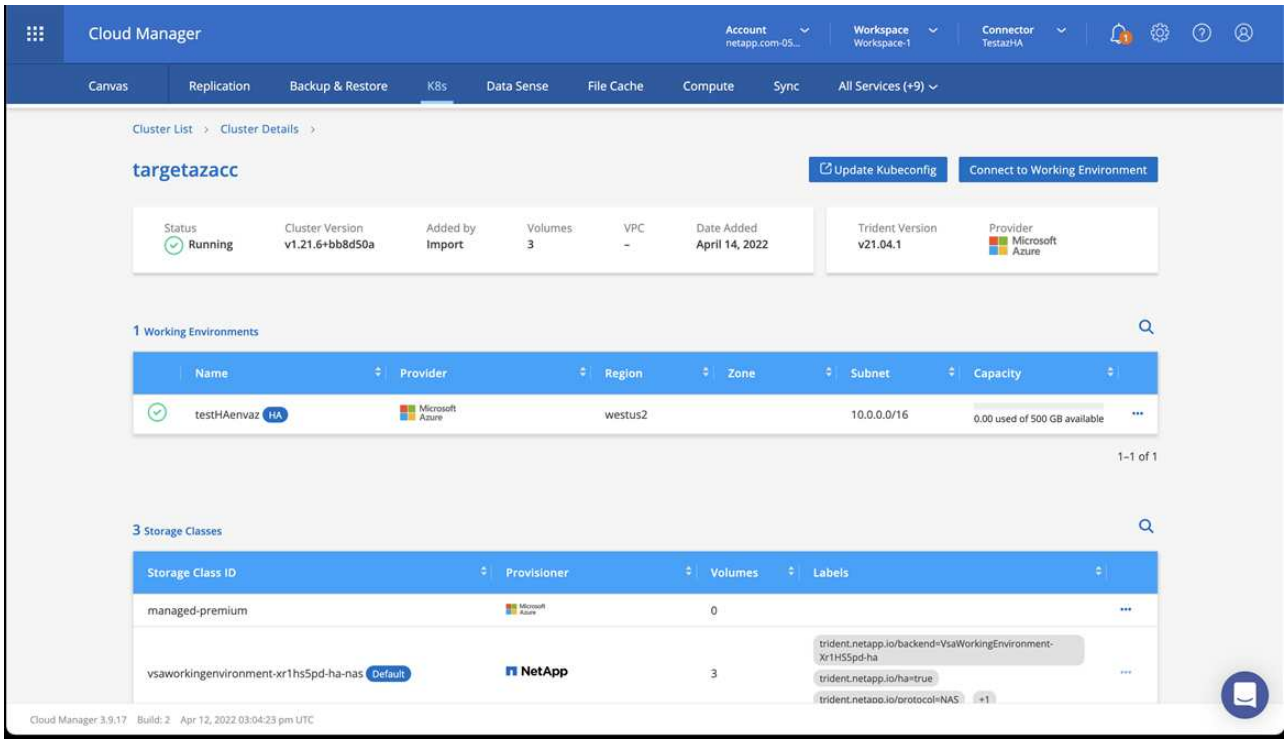
   a. 場所：「 Microsoft Azure 」。

   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。

Astra Control Center をインストールして設定します

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"。

Astra Control Center をセットアップします

Astra Control Center は、 ONTAP と Astra データストアをストレージバックエンドとしてサポートおよび監視します。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
- [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

UI またはを使用して新しいライセンスを追加できます "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。

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"。

> 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"。

Astra Control Center ライセンスは、 Kubernetes CPU ユニットを使用して CPU リソースを測定します。ライセンスには、管理対象のすべての Kubernetes クラスタのワーカーノードに割り当てられた CPU リソースが含まれている必要があります。ライセンスを追加する前に、からライセンスファイル（ NLF ）を取得する必要があります "NetApp Support Site"。

また、 Astra Control Center に評価ライセンスをお試しいただくこともできます。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"。

> 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。

必要なもの

から Astra Control Center をダウンロードした場合 "NetApp Support Site"には、 NetApp License File （ NLF ）もダウンロードします。このライセンスファイルにアクセスできることを確認してください。

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の場合は、 Astra Control Center に障害が発生したときに ASUP を送信していない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。

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。Astra データストアの場合は、 Astra データストアによってプロビジョニングされたボリュームを使用するアプリケーションを含む Kubernetes アプリケーションクラスタを追加します。

> 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。Add Cluster * 機能を使用して、 Astra Control Center でクラスタを管理できます。

> Astra Controlは、クラスタを管理する際に、クラスタのデフォルトストレージクラスを追跡します。を使用してストレージクラスを変更する場合は、を使用します `kubectl` コマンドを実行すると、Astra Controlによって変更が元に戻されます。Astra Controlで管理されるクラスタのデフォルトのストレージクラスを変更するには、次のいずれかの方法を使用します。
>
> - Astra Control API を使用 `PUT /managedClusters` エンドポイントを追加し、で別のデフォルトストレージクラスを割り当てます `DefaultStorageClass` パラメータ
> - Astra Control Web UIを使用して、別のデフォルトストレージクラスを割り当てます。を参照してください [デフォルトのストレージクラスを変更する]。

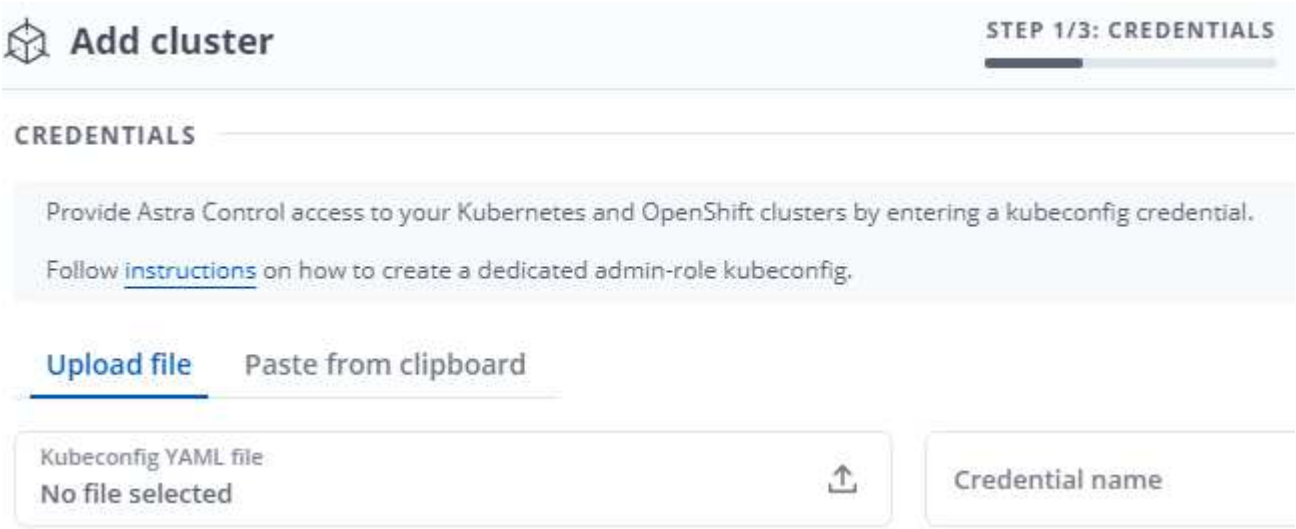
必要なもの

- 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. Astra Control Center UI の * ダッシュボード * から、クラスタセクションで * 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。

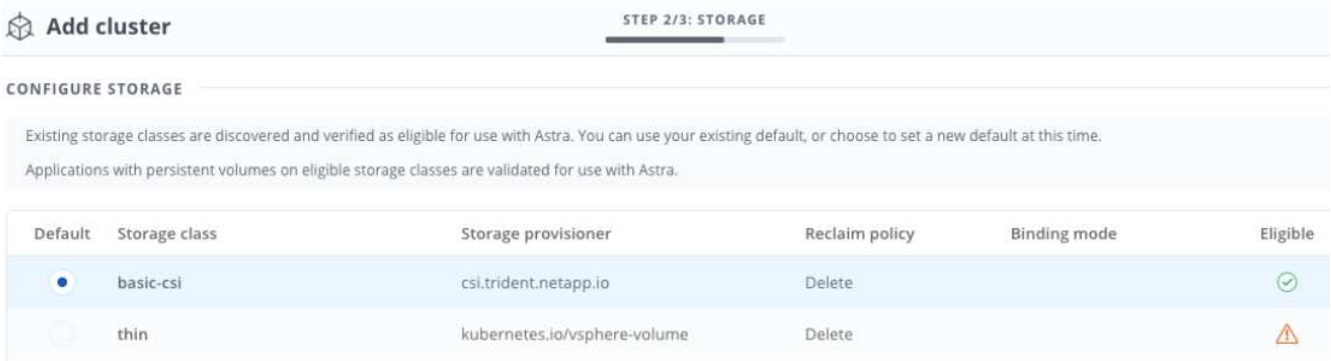
> 。 `kubeconfig.yaml` ファイルには、1つのクラスタのクラスタクレデンシャルのみを含める必要があります*。



> 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。

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. 「ストレージの設定」を選択します。
5. この Kubernetes クラスタに使用するストレージクラスを選択し、 * Review * を選択します。

> ONTAP ストレージまたは Astra データストアからバックアップされた 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、問題がない場合は「 * クラスタの追加 * 」を選択します。

結果

クラスタが「 Discovering * 」ステータスになり、「 Running 」に変わります。Kubernetes クラスタが正常に追加され、 Astra Control Center で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

ストレージバックエンドを追加して、 Astra Control がリソースを管理できるようにすることができます。管理対象クラスタにストレージバックエンドを導入するか、既存のストレージバックエンドを使用できます。

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。

既存の**Astra**データストアの導入に必要なもの

- Kubernetesアプリケーションクラスタと基盤となるコンピューティングクラスタを追加しておきます。

Astraデータストア用のKubernetesアプリケーションクラスタを追加し、Astra Controlで管理したあと、クラスタはのように表示されます `unmanaged` 検出されたバックエンドのリスト。次に、 Astra データストアを含むコンピューティングクラスタを追加し、Kubernetes アプリケーションクラスタの基盤を構築する必要があります。これは、 UI の * Backends * から実行できます。クラスタのActions（操作）メニューを選択し、を選択します `Manage`および "クラスタを追加"。をクラスタの状態のあとに続けて追加します `unmanaged` Kubernetesクラスタの名前を変更した場合は、バックエンドの追加に進むことができます。

新しい**Astra**データストアの導入に必要なもの

- これで完了です "導入するインストールバンドルのバージョンをアップロードしました" Astra Controlからアクセス可能な場所への移動。
- 導入に使用するKubernetesクラスタを追加しておきます。
- をアップロードしました Astraデータストアライセンス Astra Controlからアクセス可能な場所への導入をサポートします。

オプション（ **Options** ）

- [ストレージリソースを導入]
- [既存のストレージバックエンドを使用する]

ストレージリソースを導入

新しいAstraデータストアを導入して、関連するストレージバックエンドを管理できます。

手順

1. ダッシュボードまたはバックエンドメニューから移動します。
   - ダッシュボードから*：リソースサマリからストレージバックエンドペインからリンクを選択し、バックエンドセクションから*追加*を選択します。
   - バックエンドから * ：
     i. 左側のナビゲーション領域で、 * Backends * を選択します。
     ii. 「 * 追加」を選択します。
2. Deploy タブで Astra Data Store *導入オプションを選択します。
3. 導入するAstraデータストアパッケージを選択：
   a. Astraデータストアアプリケーションの名前を入力します。
   b. 導入するAstraデータストアのバージョンを選択します。

   展開するバージョンをまだアップロードしていない場合は、*パッケージの追加*オプションを使用するか、ウィザードを終了してを使用できます "パッケージ管理" インストールバンドルをアップロードします。

4. 以前にアップロードしたAstraデータストアライセンスを選択するか、*ライセンスの追加*オプションを使用して、アプリケーションで使用するライセンスをアップロードします。

> 完全な権限を持つAstra Data StoreライセンスはKubernetesクラスタに関連付けられており、この関連クラスタは自動的に表示されるはずです。管理対象クラスタがない場合は、*クラスタの追加*オプションを選択してAstra Control管理に追加できます。Astra Data Storeライセンスの場合、ライセンスとクラスタの間に関連付けが行われていない場合は、ウィザードの次のページでこの関連付けを定義できます。

5. KubernetesクラスタをAstra Control管理に追加していない場合は、* Kubernetes cluster *ページから追加する必要があります。リストから既存のクラスタを選択するか、「*基盤となるクラスタを追加」を選択してAstra Control管理用にクラスタを追加します。
6. Astraデータストアにリソースを提供するKubernetesクラスタのテンプレートサイズを選択します。次のいずれかを選択できます。
   - をクリックします `Recommended Kubernetes worker node requirements`をクリックし、ライセンスで許可されている内容に基づいて、テンプレートを大規模から小に選択します。
   - をクリックします `Custom Kubernetes worker node requirements`をクリックし、各クラスタノードに必要なコア数と総メモリを選択します。また、コアとメモリの選択基準を満たす、クラスタ内の対応するノード数も表示できます。

> テンプレートを選択する際は、大規模なワークロードにはメモリとコアが多く、小規模なワークロードにはノード数が多い大規模なノードを選択します。ライセンスで許可されている内容に基づいてテンプレートを選択する必要があります。推奨されるテンプレートオプションごとに、各ノードのメモリとコアおよび容量のテンプレートパターンを満たす、適格なノードの数が提示されます。

7. ノードを設定します。
   a. ノードラベルを追加して、このAstraデータストアクラスタをサポートするワーカーノードのプールを特定します。

   > このラベルは、Astraデータストアの導入に使用するクラスタ内の各ノードに追加してからでないと、導入や導入が失敗します。

   b. ノードあたりの容量（GiB）を手動で設定するか、許容される最大ノード容量を選択します。
   c. クラスタで許可される最大ノード数を設定するか、クラスタで許容される最大ノード数を設定します。
8. （Astraデータストアフルライセンスのみ）保護ドメインに使用するラベルのキーを入力します。

> 各ノードのキーに対して、少なくとも3つの一意のラベルを作成します。たとえば、キーがの場合などです `astra.datastore.protection.domain`次のラベルを作成できます。`astra.datastore.protection.domain=domain1,astra.datastore.protection.domain=domain2`および `astra.datastore.protection.domain=domain3`。

9. 管理ネットワークを設定します。
   a. Astraデータストアの内部管理用の管理IPアドレスを入力します。このIPアドレスは、ワーカーノードのIPアドレスと同じサブネットにあります。
   b. 管理ネットワークとデータネットワークで同じNICを使用するか、または個別に設定します。
   c. データネットワークのIPアドレスプール、サブネットマスク、ストレージアクセス用のゲートウェイを入力してください。

10. 設定を確認し、「* Deploy *」を選択してインストールを開始します。

結果

インストールが正常に完了すると、バックエンドがに表示されます `available` バックエンドにアクティブなパフォーマンス情報とともに表示

> 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。

既存のストレージバックエンドを使用する

検出されたONTAP またはAstraデータストアのストレージバックエンドをAstra Control Center管理に組み込むことができます。

手順

1. ダッシュボードまたはバックエンドメニューから移動します。
   - ダッシュボードから*：リソースサマリからストレージバックエンドペインからリンクを選択し、バックエンドセクションから*追加*を選択します。
   - バックエンドから * ：
     i. 左側のナビゲーション領域で、 * Backends * を選択します。
     ii. 管理対象クラスタから検出されたバックエンドで* Manage を選択するか、 Add *を選択して追加の既存バックエンドを管理します。
2. [既存の使用（ Use Existing ）] * タブを選択します。
3. バックエンドの種類に応じて、次のいずれかの操作を行います。
   - * Astra データストア * ：
     i. 「* Astra Data Store *」を選択します。
     ii. 管理対象のコンピューティングクラスタを選択し、 * Next * を選択します。
     iii. バックエンドの詳細を確認し、「Add storage backend *」を選択します。
   - * ONTAP * ：
     i. 「* ONTAP 」を選択し、「 Next *」を選択します。
     ii. 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。

     > 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、アクセス方法を有効にします `ontapi` および `http` 両方のONTAP クラスタ上のユーザに対して設定します。を参照してください "ユーザアカウントを管理する" を参照してください。

     iii. [* Review （レビュー）] を選択します
     iv. バックエンドの詳細を確認し、「Add storage backend *」を選択します。

結果

バックエンドがに表示されます `available` 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

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。

バケットを追加すると、 Astra Control によって、 1 つのバケットがデフォルトのバケットインジケータとしてマークされます。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。

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、バケットは必要ありません。

次のいずれかのバケットタイプを使用します。

- NetApp ONTAP S3
- NetApp StorageGRID S3 の略
- 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。

- Microsoft Azure

  Astra Control Center は Amazon S3 を汎用 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 サポートを要求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。

- Microsoft Azure

Astra Control API を使用してバケットを追加する手順については、を参照してください "Astra の自動化と API に関する情報"。

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。
   a. 「 * 追加」を選択します。
   b. 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。

   c. 新しいバケット名を作成するか、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、バックアップを作成するときに後で選択できるバックアップの場所として表示されます。この名前は、保護ポリシーの設定時にも表示されます。

d. 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。

e. このバケットをすべてのバックアップのデフォルトバケットにする場合は、を確認します `Make this bucket the default bucket for this private cloud` オプション

> このオプションは、最初に作成したバケットに対しては表示されません。

f. 追加して続行します クレデンシャル情報。

S3 アクセスクレデンシャルを追加します

S3 アクセスクレデンシャルはいつでも追加できます。

手順

1. バケット（ Buckets ）ダイアログで、 * 追加（ Add ） * または * 既存の * を使用（ Use Existing * ）タブのいずれかを選択します。
   a. 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。
   b. 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。

デフォルトのストレージクラスを変更する

クラスタのデフォルトのストレージクラスは変更できます。

手順

1. Astra Control Center Web UIで、[* Clusters]を選択します。
2. [* Clusters]ページで、変更するクラスタを選択します。
3. [* ストレージ *] タブを選択します。
4. 「ストレージクラス」カテゴリを選択します。
5. デフォルトとして設定するストレージクラスの* Actions *メニューを選択します。
6. 「デフォルトに設定」を選択します。

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。

- "ユーザを管理します"
- "アプリの管理を開始します"
- "アプリを保護します"
- "アプリケーションをクローニング"
- "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"

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"

クラスタを追加するための前提条件

クラスタを追加する前に、前提条件が満たされ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、アストラコントロールセンターにクラスタを追加する準備ができていることを確認する必要があります。

クラスタを追加する前に必要な作業

クラスタがに記載された要件を満たしていることを確認します "アプリケーションクラスタの要件"。

> 管理対象のコンピューティングリソースとして 2 つ目の OpenShift 4.6 、 4.7 、または 4.8 クラスタを追加する場合は、 Astra Trident ボリュームスナップショット機能が有効になっていることを確認する必要があります。ネットアップの公式 Astra Trident をご覧ください "手順 " Trident を使用して、ボリューム Snapshot を有効にしてテストしてください。

- で構成されたAstra Trident StorageClasses "サポートされるストレージバックエンド" （すべてのタイプのクラスタに必要）
- 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復元するためにバックアップ ONTAP システムで設定されたスーパーユーザーおよびユーザー ID 。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 sysm --anon 65534`
- Astra Trident `volumesnapshotclass` 管理者によって定義されたオブジェクト。Astra Trident の詳細をご確認ください "手順" Trident を使用して、ボリューム Snapshot を有効にしてテストしてください。
- Kubernetes クラスタにデフォルトのストレージクラスが 1 つだけ定義されていることを確認します。

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    21.04.0
```

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. 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
NAME                    PROVISIONER                    RECLAIMPOLICY
VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
ontap-gold (default)    csi.trident.netapp.io          Delete
Immediate           true                   5d23h
thin                    kubernetes.io/vsphere-volume   Delete
Immediate           false                  6d
```

admin ロールの kubeconfig を作成します

手順を実行する前に、マシンに次のものがあることを確認してください。

- `kubectl` V1.19以降がインストールされている
- アクティブなコンテキストのクラスタ管理者権限があるアクティブな kubeconfig です

手順

1. 次の手順で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. （オプション）権限付きポッドの作成を許可しない制限付きポッドセキュリティポリシーをクラスタで使用している場合、またはポッドコンテナ内のプロセスをルートユーザとして実行できるようにしていない場合は、 Astra Control でポッドを作成および管理できるように、クラスタ用のカスタムポッドセキュリティポリシーを作成します。手順については、を参照してください "カスタムのポッドセキュリティポリシーを作成します"。
3. 次のようにクラスタ管理者権限を付与します。

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: cluster-admin
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ベ econfig の名前を変更します。クラスタのクレデンシャルを保護します。

```
chmod 700 create-kubeconfig.sh
mv kubeconfig-sa.txt 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"。

詳細については、こちらをご覧ください

- "Trident のドキュメント"
- "Astra Control API を使用"

カスタム TLS 証明書を追加します

既存の自己署名 TLS 証明書を削除して、認証局（ CA ）が署名した TLS 証明書に置き換えることができます。

必要なもの

- Astra Control Center をインストールした Kubernetes クラスタ
- 実行するクラスタ上のコマンドシェルへの管理アクセス `kubectl` コマンド
- CA の秘密鍵ファイルと証明書ファイル

自己署名証明書を削除します

既存の自己署名 TLS 証明書を削除します。

1. SSH を使用して、 Astra Control Center をホストする Kubernetes クラスタに管理ユーザとしてログインします。
2. 次のコマンドを使用して、現在の証明書に関連付けられているTLSシークレットを検索します `<ACC-deployment-namespace>` Astra Control Center導入ネームスペースを使用して、次の作業を行います。

```
kubectl get certificate -n <ACC-deployment-namespace>
```

3. 次のコマンドを使用して、現在インストールされているシークレットと証明書を削除します。

```
kubectl delete cert cert-manager-certificates -n <ACC-deployment-
namespace>
kubectl delete secret secure-testing-cert -n <ACC-deployment-namespace>
```

新しい証明書を追加します

CA によって署名された新しい TLS 証明書を追加します。

1. 次のコマンドを使用して、 CA の秘密鍵ファイルと証明書ファイルを使用して新しい TLS シークレットを作成し、括弧 <> の引数を適切な情報に置き換えます。

```
kubectl create secret tls <secret-name> --key <private-key-filename>
--cert <certificate-filename> -n <ACC-deployment-namespace>
```

2. 次のコマンドと例を使用して、クラスタカスタムリソース定義（CRD）ファイルを編集し、を変更します `spec.selfSigned` の値 `spec.ca.secretName` 以前に作成したTLSシークレットを参照するには、次の手順を実行します

```
kubectl edit clusterissuers.cert-manager.io/cert-manager-certificates -n
<ACC-deployment-namespace>
....

#spec:
#  selfSigned: {}

spec:
  ca:
    secretName: <secret-name>
```

3. 次のコマンドと出力例を使用して、変更が正しいこと、および交換する証明書をクラスタで検証する準備ができていることを確認します `<ACC-deployment-namespace>` Astra Control Center導入ネームスペースを使用して、次の作業を行います。

```
kubectl describe clusterissuers.cert-manager.io/cert-manager-
certificates -n <ACC-deployment-namespace>
....

Status:
  Conditions:
    Last Transition Time:  2021-07-01T23:50:27Z
    Message:               Signing CA verified
    Reason:                KeyPairVerified
    Status:                True
    Type:                  Ready
Events:                    <none>
```

4. を作成します `certificate.yaml` 次の例を使用してファイルを作成し、括弧<>のプレースホルダ値を適切な情報に置き換えます。

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: <certificate-name>
  namespace: <ACC-deployment-namespace>
spec:
  secretName: <certificate-secret-name>
  duration: 2160h # 90d
  renewBefore: 360h # 15d
  dnsNames:
  - <astra.dnsname.example.com> #Replace with the correct Astra Control
Center DNS address
  issuerRef:
    kind: ClusterIssuer
    name: cert-manager-certificates
```

5. 次のコマンドを使用して証明書を作成します。

```
kubectl apply -f certificate.yaml
```

6. 次のコマンドと出力例を使用して、証明書が正しく作成されていること、および作成時に指定した引数（名前、期間、更新期限、 DNS 名など）を使用していることを確認します。

```
kubectl describe certificate -n <ACC-deployment-namespace>
....

Spec:
  Dns Names:
    astra.example.com
  Duration:  125h0m0s
  Issuer Ref:
    Kind:        ClusterIssuer
    Name:        cert-manager-certificates
  Renew Before:  61h0m0s
  Secret Name:   <certificate-secret-name>
Status:
  Conditions:
    Last Transition Time:  2021-07-02T00:45:41Z
    Message:               Certificate is up to date and has not expired
    Reason:                Ready
    Status:                True
    Type:                  Ready
  Not After:               2021-07-07T05:45:41Z
  Not Before:              2021-07-02T00:45:41Z
  Renewal Time:            2021-07-04T16:45:41Z
  Revision:                1
Events:                    <none>
```

7. 次のコマンドおよび例を使用して、入力 CRD TLS オプションを編集し、新しい証明書シークレットを指定します。括弧 <> のプレースホルダ値を適切な情報に置き換えます。

```
kubectl edit ingressroutes.traefik.containo.us -n <ACC-deployment-
namespace>
....

# tls:
#    options:
#      name: default
#    secretName: secure-testing-cert
#    store:
#      name: default

 tls:
    options:
      name: default
    secretName: <certificate-secret-name>
    store:
      name: default
```

8. Web ブラウザを使用して、 Astra Control Center の導入 IP アドレスにアクセスします。
9. 証明書の詳細がインストールした証明書の詳細と一致していることを確認します。
10. 証明書をエクスポートし、結果を Web ブラウザの証明書マネージャにインポートします。

カスタムのポッドセキュリティポリシーを作成します

Astra Control では、管理対象のクラスタに Kubernetes ポッドを作成して管理する必要があります。クラスタで、特権ポッドの作成を許可しない制限付きポッドセキュリティポリシーを使用している場合、またはポッドコンテナ内のプロセスをルートユーザとして実行できるように許可していない場合は、制限の少ないポッドセキュリティポリシーを作成して、 Astra Control がこれらのポッドを作成および管理できるようにする必要があります。

手順

1. デフォルトよりも制限の緩いクラスタの PoD セキュリティポリシーを作成してファイルに保存します。
   例：

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: astracontrol
  annotations:
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
spec:
  privileged: true
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  volumes:
  - '*'
  hostNetwork: true
  hostPorts:
  - min: 0
    max: 65535
  hostIPC: true
  hostPID: true
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
```

2. ポッドセキュリティポリシーの新しいロールを作成します。

```
kubectl-admin create role psp:astracontrol \
    --verb=use \
    --resource=podsecuritypolicy \
    --resource-name=astracontrol
```

3. 新しいロールをサービスアカウントにバインドします。

```
kubectl-admin create rolebinding default:psp:astracontrol \
    --role=psp:astracontrol \
    --serviceaccount=astracontrol-service-account:default
```

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_min YAML カスタムリソース定義（ CRD ）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。電子メールは、 Astra_control_center_min YAML CRD 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 *
- 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。
- 「 * ライセンスの追加 * 」を選択します。
- 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

- 作成していないスナップショットがアカウントにあります。彼らはどこから来たのですか。 *

一部の状況では、バックアップ、クローン、またはリストアのプロセスの一環として、 Astra Control によってスナップショットが自動的に作成されます。

- アプリケーションは複数の PVS を使用しています。Astra Control は、これらすべての PVC 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。