Astra Control Center 22.04 のドキュメント

Astra Control Center

NetApp
November 21, 2023

目次

Astra Control Center 22.04 のドキュメント

リリースノート

この度、アストラコントロールセンターの 22.04.0 リリースが発表されました。

- "このリリースの Astra Control Center の内容"
- "既知の問題"
- "Astra Data Store およびこの Astra Control Center リリースでの既知の問題"
- "既知の制限"

Twitter で @NetAppDoc をフォローしてください。を作成し、ドキュメントに関するフィードバックを送信します "GitHub の貢献者" または、 doccomments@netapp.com に電子メールを送信します。

このリリースの **Astra Control Center** の新機能

この度、最新の 22.04.0 リリースの Astra Control Center がリリースされました。

2022 年 **4** 月 **26** 日（ **22.04.0** ）

新機能とサポート

- "Astra Control CenterからAstraデータストアを導入"
- "ネームスペースのロールベースアクセス制御（ RBAC ）"
- "Cloud Volumes ONTAP のサポート"
- "Astra Control Center の一般的な入力イネーブルメント"
- "Astra Control からバケットを取り外す"
- "VMware Tanzu ポートフォリオのサポート"

既知の問題および制限事項

- "このリリースの既知の問題"
- "Astra Data Store およびこの Astra Control Center リリースでの既知の問題"
- "このリリースの既知の制限事項は以下のとおりです"

2021 年 **12** 月 **14** 日（ **21.12** ）

新機能とサポート

- "アプリケーションのリストア"
- "実行フック"
- "ネームスペースを対象とした演算子を使用して展開されたアプリケーションのサポート"
- "アップストリーム Kubernetes と Rancher もサポートしています"
- "Astra データストアプレビューバックエンドの管理と監視"
- "Astra Control Center のアップグレード"

- "インストール用の Red Hat OperatorHub オプションです"

解決済みの問題

- "このリリースの解決済みの問題"

既知の問題および制限事項

- "このリリースの既知の問題"
- "Astra データストアのプレビューおよびこの Astra Control Center リリースでの既知の問題"
- "このリリースの既知の制限事項は以下のとおりです"

2021 年 8 月 5 日（ 21.08 ）

Astra Control Center の初回リリース。

- "それは何であるか"
- "アーキテクチャとコンポーネントを理解する"
- "開始には何が必要ですか"
- "をインストールします" および "セットアップ（ Setup ）"
- "管理" および "保護" アプリケーション
- "バケットを管理する" および "ストレージバックエンド"
- "アカウントを管理"
- "API による自動化"

詳細については、こちらをご覧ください

- "このリリースの既知の問題"
- "このリリースの既知の制限事項は以下のとおりです"
- "Astra データストアのドキュメント"
- "以前のバージョンの Astra Control Center ドキュメント"

既知の問題

既知の問題は、このリリースの製品を正常に使用できない可能性のある問題を特定します。

現在のリリースに影響する既知の問題は次のとおりです。

アプリケーション

- アプリケーションをリストアすると、 PV のサイズが元の PV よりも大きくなります
- 特定のバージョンの PostgreSQL を使用すると、アプリケーションクローンが失敗します
- サービスアカウントレベルの OCP セキュリティコンテキスト制約（ SCC ）を使用すると、アプリケーションのクローンが失敗する
- [ストレージクラスを設定してアプリケーションを導入すると、アプリケーションのクローンが失敗する]

クラスタ

- デフォルトの kubeconfig ファイルに複数のコンテキストが含まれている場合、 Astra Control Center を使用したクラスタの管理が失敗します

その他の問題

- Astra Trident がオフラインの場合、 Internal Service Error （ 500 ）によりアプリケーションデータ管理処理が失敗する
- Snapshot コントローラバージョン 4.2.0 では、 Snapshot が失敗することがあります

アプリケーションをリストアすると、 PV のサイズが元の PV よりも大きくなります

バックアップの作成後に永続ボリュームのサイズを変更し、そのバックアップからリストアすると、永続ボリュームのサイズはバックアップのサイズではなく PV の新しいサイズと一致します。

特定のバージョンの PostgreSQL を使用すると、アプリケーションクローンが失敗します

Bitnami PostgreSQL 11.5.0 チャートを使用すると、同じクラスタ内のアプリケーションクローンは一貫して失敗します。正常にクローニングするには、以前のバージョンのグラフを使用してください。

サービスアカウントレベルの OCP セキュリティコンテキスト制約（ SCC ）を使用すると、アプリケーションのクローンが失敗する

OpenShift Container Platform クラスタのネームスペース内のサービスアカウントレベルで元のセキュリティコンテキストの制約が設定されていると、アプリケーションクローンが失敗する場合があります。アプリケーションのクローンが失敗すると、 Astra Control Center の管理対象アプリケーション領域にステータス「 Removed 」と表示されます。を参照してください "技術情報アーティクル" を参照してください。

ストレージクラスを設定してアプリケーションを導入すると、アプリケーションのクローンが失敗する

ストレージクラスを明示的に設定してアプリケーションをデプロイした後 (たとえば、「 helm install…-set global.storageClass=NetApp-cvs-perf-extreme 」) 、アプリケーションのクローンを作成しようとすると、ターゲットクラスタに最初に指定されたストレージクラスが必要になります。ストレージクラスを明示的に設定したアプリケーションを、同じストレージクラスを含まないクラスタにクローニングすると、失敗します。このシナリオではリカバリ手順はありません。

デフォルトの kubeconfig ファイルに複数のコンテキストが含まれている場合、 Astra Control Center を使用したクラスタの管理が失敗します

複数のクラスタおよびコンテキストで kubeconfig を使用することはできません。を参照してください "技術情報アーティクル" を参照してください。

Astra Trident がオフラインの場合、 Internal Service Error （ 500 ）によりアプリケーションデータ管理処理が失敗する

アプリケーションクラスタの Astra Trident がオフラインになり（オンラインに戻った）、 500 件の内部サービスエラーが発生した場合に、アプリケーションデータ管理を試みると、アプリケーションクラスタ内のすべての Kubernetes ノードを再起動して機能を復旧します。

Snapshot コントローラバージョン 4.2.0 では、 Snapshot が失敗することがあります

Kubernetes 1.20 または 1.21 で Kubernetes snapshot-controller （別名 external-snapshotter ）バージョン 4.2.0 を使用すると、 Snapshot が失敗することがあります。これを防ぐには、別のを使用してください "サポートされているバージョン" バージョン 4.2.1 などの外部 Snapshot データ。 Kubernetes バージョン 1.20 または 1.21 で使用。

1. POST 呼び出しを実行して更新された kubeconfig ファイルを「 /credentials 」エンドポイントに追加し、応答本文から割り当てられた「 id 」を取得します。
2. 適切なクラスタ ID を使用して '/clusters' エンドポイントから PUT 呼び出しを実行し 'credentialId' を前の手順の id' 値に設定します

これらの手順を完了すると、クラスタに関連付けられたクレデンシャルが更新され、クラスタは再接続して、その状態を「 available 」に更新する必要があります。

詳細については、こちらをご覧ください

- "Astra Data Store の既知の問題と、この Astra Control Center リリース"
- "既知の制限"

Astra Data Store およびこの Astra Control Center リリースでの既知の問題

既知の問題は、このリリースの製品を正常に使用できない可能性のある問題を特定します。

"これらの既知の問題を参照してください" これは、Astra Control Centerの最新リリースでAstraデータストアの管理に影響を与える可能性があります。

詳細については、こちらをご覧ください

- "既知の問題"
- "既知の制限"

既知の制限

既知の制限事項は、このリリースの製品でサポートされていないプラットフォーム、デバイス、機能、または製品と正しく相互運用できない機能を特定します。これらの制限事項を慎重に確認してください

クラスタ管理の制限事項

- 2 つの Astra Control Center インスタンスで同じクラスタを管理することはできません
- Astra Control Center は、同じ名前の 2 つのクラスタを管理できません

Role-Based Access Control （ RBAC ；ロールベースアクセス制御）の制限事項があります

- ネームスペースの RBAC に制約があるユーザは、クラスタの追加と管理解除を行うことができます
- [名前空間の制約を持つメンバは、管理者が名前空間を制約に追加するまで、クローンまたは復元されたアプリケーションにアクセスできません]

アプリケーション管理の制限

- [実行中のアプリケーションのバックアップを停止することはできません]
- [パスバイリファレンス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のクローンが失敗することがあります]
- 証明書マネージャを使用するアプリケーションの In Place リストア処理はサポートされていません
- OLM 対応およびクラスタ対象のオペレータ展開アプリケーションはサポートされていません
- Helm 2 で展開されたアプリケーションはサポートされていません

一般的な制限事項

- Astra Control Center の S3 バケットは、使用可能容量を報告しません
- Astra Control Center は、プロキシサーバー用に入力した詳細を検証しません
- Postgres ポッドへの既存の接続が原因で障害が発生します
- Astra Control Center インスタンスの削除中にバックアップとスナップショットが保持されない場合があります

2 つの Astra Control Center インスタンスで同じクラスタを管理することはできません

別の Astra Control Center インスタンスでクラスタを管理する場合は、最初にを実行する必要があります "クラスタの管理を解除します" 別のインスタンスで管理する前に、管理対象のインスタンスから管理します。管理対象からクラスタを削除したら、次のコマンドを実行してクラスタが管理対象外であることを確認します。

```
oc get pods n -netapp-monitoring
```

そのネームスペースでポッドを実行していないことを確認するか、ネームスペースを存在させないようにします。どちらかが true の場合、クラスタは管理対象外です。

Astra Control Center は、同じ名前の 2 つのクラスタを管理できません

既存のクラスタと同じ名前のクラスタを追加しようとすると、処理に失敗します。この問題は、 Kubernetes 構成ファイルでクラスタ名のデフォルトを変更していない場合、通常は標準の Kubernetes 環境で発生します。

回避策として、次の手順を実行します。

1. kubeadm -config 構成マップを編集します。

   ```
   kubectl edit configmaps -n kube-system kubeadm-config
   ```

2. 「 clusterName 」フィールドの値を「 Kubernetes 」（ Kubernetes のデフォルト名）から一意のカスタム名に変更します。
3. kubeconfig (`.kube/config`) を編集します。
4. クラスタ名を「 Kubernetes 」から一意のカスタム名に更新します（以下の例では「 xyz-cluster 」を使用します）。次の例に示すように 'clusters' および contexts の両方のセクションで更新を行います

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
ExAmPLERb2tCcjZ5K3E2Njk4eQotLExAMpLEORCBDRVJUSUZJQ0FURS0txxxxXX==
    server: https://x.x.x.x:6443
  name: xyz-cluster
contexts:
- context:
    cluster: xyz-cluster
    namespace: default
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
```

ネームスペースの **RBAC** に制約があるユーザは、クラスタの追加と管理解除を行うことができます

ネームスペースの RBAC に制限があるユーザは、クラスタの追加または管理解除を行うことができません。現在の制限により、 Astra は、このようなユーザによるクラスタの管理解除を妨げません。

名前空間の制約を持つメンバは、管理者が名前空間を制約に追加するまで、クローンまたは復元されたアプリケーションにアクセスできません

名前空間の名前 /ID または名前空間ラベルによって RBAC の制約を受けているメンバーユーザーは ' 同じクラスタ上の新しい名前空間 ' または組織のアカウント内の他のクラスタにアプリケーションを複製またはリストアできますただし、同じユーザが、クローニングまたはリストアされたアプリケーションに新しいネームスペースからアクセスすることはできません。クローンまたはリストア操作によって新しい名前空間が作成されると ' アカウントの管理者 / 所有者は ' メンバーのユーザー・アカウントを編集し ' 影響を受けるユーザーの役割制約を更新して ' 新しい名前空間へのアクセスを許可できます

実行中のアプリケーションのバックアップを停止することはできません

実行中のバックアップを停止する方法はありません。バックアップを削除する必要がある場合は、完了するまで待ってから、の手順を実行してください "バックアップを削除します"。失敗したバックアップを削除するには、を使用します "Astra Control API の略"。

パスバイリファレンス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のクローンが失敗することがあります

Astra Control は、名前空間を対象とした演算子で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。これらの演算子は、一般に「パスバイリファレンス」アーキテクチャではなく「パスバイ値」で設計されています。これらのパターンに続くいくつかのオペレータアプリを次に示します。

- "Apache K8ssandra"

K8ssandra では、 In Place リストア処理がサポートされます。新しいネームスペースまたはクラスタにリストアするには、アプリケーションの元のインスタンスを停止する必要があります。これは、ピアグループ情報がインスタンス間通信を行わないようにするためです。アプリケーションのクローニングはサポートされていません。

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

Astra Control では、「パスバイリファレンス」アーキテクチャ（ CockroachDB オペレータなど）で設計されたオペレータをクローニングできない場合があります。クローニング処理では、クローニング処理の一環として独自の新しいシークレットが存在する場合でも、クローニングされたオペレータがソースオペレータから Kubernetes シークレットを参照しようとします。Astra Control がソースオペレータの Kubernetes シークレットを認識しないため、クローニング処理が失敗する場合があります。

証明書マネージャを使用するアプリケーションの **In Place** リストア処理はサポートされていません

このリリースの Astra Control Center では、証明書マネージャを使用したアプリのインプレースリストアはサポートされていません。別のネームスペースへのリストア処理とクローニング処理がサポートされています。

OLM 対応およびクラスタ対象のオペレータ展開アプリケーションはサポートされていません

Astra Control Center は、クラスタを対象としたオペレータによるアプリケーション管理アクティビティをサポートしません。

Helm 2 で展開されたアプリケーションはサポートされていません

Helm を使用してアプリケーションを展開する場合、 Astra Control Center 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）を使用して展開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。詳細については、を参照してください "Astra Control Center の要件"。

Astra Control Center の **S3** バケットは、使用可能容量を報告しません

Astra Control Center で管理されているアプリケーションのバックアップまたはクローニングを行う前に、ONTAP または StorageGRID 管理システムでバケット情報を確認します。

Astra Control Center は、プロキシサーバー用に入力した詳細を検証しません

実行することを確認してください "正しい値を入力します" 接続を確立するとき。

Postgres ポッドへの既存の接続が原因で障害が発生します

Postgres ポッドで操作を実行する場合は、 psql コマンドを使用するためにポッド内で直接接続しないでください。Astra Control では、 psql にアクセスしてデータベースをフリーズし、解凍する必要があります。既存の接続がある場合、スナップショット、バックアップ、またはクローンは失敗します。

Astra Control Center インスタンスの削除中にバックアップとスナップショットが保持されない場合があります

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。

詳細については、こちらをご覧ください

- "既知の問題"
- "Astra Data Store およびこの Astra Control Center リリースでの既知の問題"

概念

Astra Controlの詳細をご確認ください

Astra Control は、 Kubernetes アプリケーションデータライフサイクル管理解決策で、ステートフルアプリケーションの運用を簡易化します。Kubernetes ワークロードの保護、バックアップ、移行を簡易化し、作業用アプリケーションのクローンを瞬時に作成できます。

の機能

Astra Control は、 Kubernetes アプリケーションデータのライフサイクル管理に不可欠な機能を提供

- 永続的ストレージを自動的に管理
- アプリケーション対応のオンデマンドの Snapshot とバックアップを作成
- ポリシーベースのスナップショットおよびバックアップ操作を自動化します
- Kubernetes クラスタ間でアプリケーションとデータを移行
- 本番環境からステージング環境にアプリケーションを簡単にクローニングできます
- アプリケーションの稼働状態と保護状態を視覚化します
- バックアップと移行のワークフローを実装するには、ユーザインターフェイスまたは API を使用してください

Astra Control は、状態の変化を常に監視しているので、新しいアプリケーションを追加していくことを認識しています。

導入モデル

Astra Control には、次の 2 つの導入モデルがあります。

- * Astra Control Service * ： Google Kubernetes Engine （ GKE ）および Azure Kubernetes Service （ AKS ）で Kubernetes クラスタのアプリケーション対応データ管理を提供する、ネットアップが管理するサービス。
- * Astra Control Center * ：オンプレミス環境で実行される Kubernetes クラスタのアプリケーション対応データ管理を提供する、自己管理ソフトウェアです。

	Astra 制御サービス	**Astra Control Center** の略
どのような方法で提供されますか？	ネットアップのフルマネージドクラウドサービス	ソフトウェアとしてダウンロード、インストール、および管理します
ホストされているのはどこですか？	ネットアップが選択したパブリッククラウドで実現	指定した Kubernetes クラスタで実行します
更新方法	管理はネットアップが行います	更新を管理します

	Astra 制御サービス	Astra Control Center の略
アプリケーションデータ管理機能とは何ですか？	ストレージバックエンドと外部サービスを除く両方のプラットフォームで同じ機能を利用できます	ストレージバックエンドと外部サービスを除く両方のプラットフォームで同じ機能を利用できます
ストレージバックエンドでサポートされるもの	ネットアップのクラウドサービス	• NetApp ONTAP AFF および FAS システム • ストレージバックエンドとしての Astra データストア • Cloud Volumes ONTAP ストレージバックエンド

サポートされているアプリケーション

ネットアップでは、 Snapshot とバックアップの安全性と一貫性を確保するために、いくつかのアプリケーションの検証を行っています。

- "Astra Controlの検証済みアプリケーションと標準アプリケーションの違いをご確認ください"。

Astra Control で使用するアプリケーションの種類に関係なく、必ず自分でバックアップとリストアのワークフローをテストして、ディザスタリカバリの要件を満たすことを確認してください。

Astra Control Service の仕組み

Astra Control Service は、常時稼働し、最新の機能で更新される、ネットアップが管理するクラウドサービスです。複数のコンポーネントを利用して、アプリケーションデータのライフサイクル管理を実現します。

Astra Control Service の概要は次のように機能します。

- Astra Control Service の利用を開始するには、クラウドプロバイダをセットアップし、 Astra アカウントに登録します。
  - GKE クラスタでは、 Astra Control Service はを使用します "NetApp Cloud Volumes Service for Google Cloud" または、永続ボリューム用のストレージバックエンドとして Google Persistent Disk を使用します。
  - AKS クラスタの場合、 Astra Control Service はを使用します "Azure NetApp Files の特長" または、永続ボリューム用のストレージバックエンドとして Azure Disk Storage を選択します。
- 最初の Kubernetes コンピューティングを Astra Control サービスに追加します。Astra Control Service は、次の処理を実行します。
  - バックアップコピーが格納されるクラウドプロバイダアカウントにオブジェクトストアを作成します。

    Azure では、 Astra Control Service によって、 BLOB コンテナ用のリソースグループ、ストレージアカウント、およびキーも作成されます。
  - クラスタに新しい admin ロールと Kubernetes サービスアカウントを作成します。
  - 新しい admin ロールを使用してインストールします "Astra Trident" をクリックして、 1 つ以上のスト

レージクラスを作成します。

  - Azure NetApp Files または NetApp Cloud Volumes Service for Google Cloud をストレージバックエンドとして使用している場合、 Astra Control Service は Astra Trident を使用して、アプリケーション用の永続的ボリュームをプロビジョニングします。

- この時点で、アプリケーションをクラスタに追加できます。永続ボリュームは、新しいデフォルトのストレージクラスでプロビジョニングされます。
- 次に、 Astra Control Service を使用してこれらのアプリケーションを管理し、スナップショット、バックアップ、クローンの作成を開始します。

  Astra Control Service は、状態の変化を常に監視しているので、新しいアプリケーションを追加していくことを認識しています。

Astra Control の無料プランを使用すると、最大 10 個のアプリをアカウントで管理できます。10 以上のアプリを管理する場合は、無料プランからプレミアムプランにアップグレードして請求を設定する必要があります。

Astra Control Center の仕組み

Astra Control Center は、お客様のプライベートクラウドでローカルに実行されます。

Astra Control Center は、 OpenShift Kubernetes クラスタを次の機能でサポートします。

- Trident ストレージバックエンドは ONTAP 9.5 以降で構成されています
- Astra データストアストレージバックエンド

クラウド接続環境では、 Cloud Insights を使用して高度なモニタリングとテレメトリを提供します。Cloud Insights 接続がない場合、 Astra Control Center では、限定的な（ 7 日間の指標）監視と計測データを使用できます。また、オープン指標エンドポイントを介して Kubernetes の標準の監視ツール（ Prometheus や Grafana など）にエクスポートすることもできます。

Astra Control Center は、 AutoSupport と Active IQ のエコシステムに完全に統合されており、ユーザとネットアップサポートにトラブルシューティングと使用に関する情報を提供します。

Astra Control Center を試用するには、 90 日間の評価版ライセンスを使用します。評価版は、 E メールとコミュニティ（ Slack チャンネル）のオプションでサポートされています。また、製品内サポートダッシュボードから技術情報アーティクルやドキュメントにアクセスすることもできます。

Astra Control Center をインストールして使用するには、一定の要件を満たす必要があります "要件"。

Astra Control Center の概要は次のように機能します。

- Astra Control Center は、ローカル環境にインストールします。方法の詳細については、こちらをご覧ください "Astra Control Center をインストールします"。
- 次のようなセットアップタスクを実行したとします。
  - ライセンスをセットアップする
  - 最初のクラスタを追加します。
  - クラスタを追加したときに検出されたストレージバックエンドを追加します。
  - 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。

方法の詳細については、こちらをご覧ください "Astra Control Center をセットアップします"。

Astra Control Center は、次のことを行います。

- 管理対象の Kubernetes クラスタに関する詳細を検出します。
- では、管理対象として選択したクラスタに Astra Trident または Astra データストア構成が検出され、ストレージバックエンドを監視できます。
- それらのクラスタ上のアプリケーションを検出し、アプリケーションを管理および保護できます。

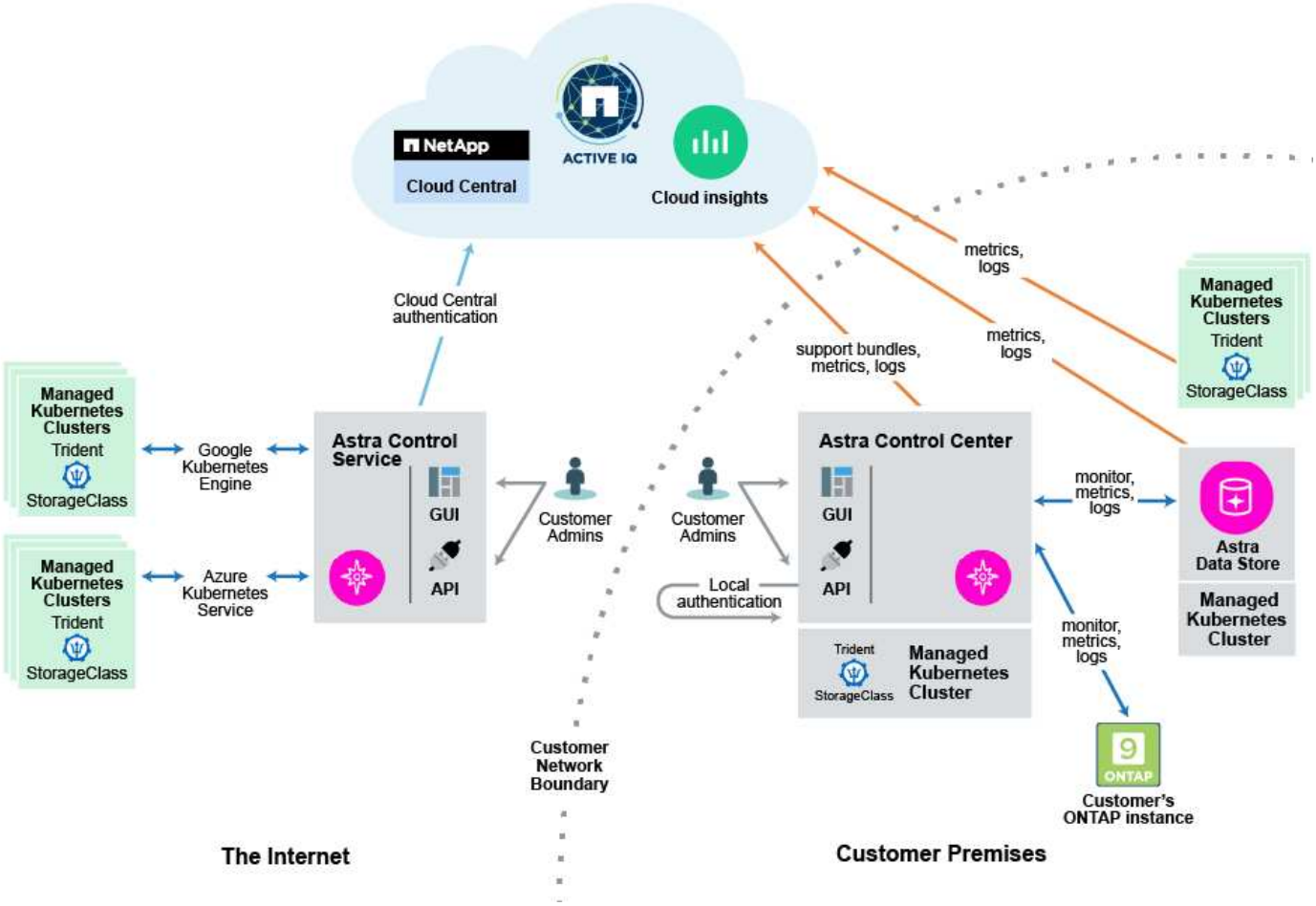
クラスタにアプリケーションを追加できます。また、管理対象のクラスタにすでにアプリケーションがある場合は、 Astra Control Center を使用して検出と管理を行うことができます。次に、 Astra Control Center を使用して、スナップショット、バックアップ、クローンを作成します。

を参照してください。

- "Astra Control Service のマニュアル"
- "Astra Control Center のドキュメント"
- "Astra データストアのドキュメント"
- "Astra Trident のドキュメント"
- "Astra Control API を使用"
- "Cloud Insights のドキュメント"
- "ONTAP のドキュメント"

アーキテクチャとコンポーネント

ここでは、 Astra Control 環境のさまざまなコンポーネントの概要を示します。



Astra Control コンポーネント

- * Kubernetes クラスタ * ： Kubernetes は、コンテナ化されたワークロードとサービスを管理するための、ポータブルで拡張性に優れたオープンソースプラットフォームであり、宣言型の設定と自動化の両方を促進します。Astra は、 Kubernetes クラスタでホストされているアプリケーションに管理サービスを提供します。
- * Trident * ：ネットアップが管理する、完全にサポートされているオープンソースのストレージプロビジョニングおよびオーケストレーションツールである Trident を使用すると、 Docker と Kubernetes で管理するコンテナ化されたアプリケーション用のストレージボリュームを作成できます。Trident を Astra Control Center とともに導入すると、構成済みの ONTAP ストレージバックエンドが含まれ、ストレージバックエンドとして Astra データストアもサポートされます。
- * ストレージバックエンド * ：
  - Astra Control Service で使用される "NetApp Cloud Volumes Service for Google Cloud" GKE クラスタおよびのストレージバックエンドとして "Azure NetApp Files の特長" AKS クラスタのストレージバックエンドとして。
  - Astra Control Service は、 Azure Managed Disks と Google Persistent Disk をバックエンドストレージオプションとしてサポートします。
  - Astra Control Center は、次のストレージバックエンドを使用します。
    - Astra データストアストレージバックエンド
    - ONTAP AFF と FAS のストレージバックエンド。ONTAP は、ストレージソフトウェアおよびハードウェアプラットフォームとして、コアストレージサービス、複数のストレージアクセスプロト

コルのサポート、 Snapshot やミラーリングなどのストレージ管理機能を提供します。

  - Cloud Volumes ONTAP ストレージバックエンド

- * Cloud Insights * ：ネットアップのクラウドインフラ監視ツールである Cloud Insights を使用すると、Astra Control Center で管理された Kubernetes クラスタのパフォーマンスと利用率を監視できます。Cloud Insights ：ストレージ使用率とワークロードの相関関係を示します。Cloud Insights 接続を Astra コントロールセンターで有効にすると、テレメータの情報が Astra コントロールセンターの UI ページに表示されます。

Astra Control インターフェイス

さまざまなインターフェイスを使用してタスクを完了できます。

- * ウェブユーザーインタフェース（ UI ） * ： Astra Control Service と Astra Control Center の両方が、同じ Web ベースの UI を使用して、アプリケーションの管理、移行、保護を行うことができます。また、UI を使用してユーザアカウントと設定を管理することもできます。
- * API* ： Astra Control Service と Astra Control Center は、どちらも同じ Astra Control API を使用します。API を使用するタスクは、 UI を使用するタスクと同じです。

Astra Control Center を使用すると、 VM 環境内で実行される Kubernetes クラスタを管理、移行、保護することもできます。

を参照してください。

- "Astra Control Service のマニュアル"
- "Astra Control Center のドキュメント"
- "Astra Trident のドキュメント"
- "Astra Control API を使用"
- "Cloud Insights のドキュメント"
- "ONTAP のドキュメント"

データ保護

Astra Control Center で使用可能なデータ保護の種類と、それらを使用してアプリケーションを保護する最適な方法について説明します。

Snapshot 、バックアップ、保護のポリシー

a_snapshot_は 、アプリケーションと同じプロビジョニングボリュームに格納されるアプリケーションのポイントインタイムコピーです。通常は高速です。ローカル Snapshot を使用して、アプリケーションを以前の時点にリストアできます。スナップショットは高速クローンに便利です。スナップショットには、構成ファイルを含む、アプリケーションのすべての Kubernetes オブジェクトが含まれます。

a_backup_ は外部のオブジェクトストアに格納されます。ローカル Snapshot と比較すると時間がかかることがあります。アプリケーションのバックアップを同じクラスタにリストアすることも、バックアップを別のクラスタにリストアして移行することもできます。バックアップの保持期間を延長することもできます。バックアップは外部のオブジェクトストアに格納されるため、サーバで障害が発生したりデータが失われたりした場合に備えて、 Snapshot よりも優れた保護機能を提供できます。

a_protection policy__ は、アプリケーション用に定義したスケジュールに従って、スナップショット、バックアップ、またはその両方を自動的に作成することで、アプリケーションを保護する方法です。保護ポリシーでは、スケジュールで保持する Snapshot とバックアップの数も選択できます。バックアップとスナップショットを保護ポリシーを使用して自動化することは、組織のニーズに応じて各アプリケーションを確実に保護する最善の方法です。

> _ 最新のバックアップがあるまで、完全に保護することはできません _ 。これは、永続ボリュームから離れたオブジェクトストアにバックアップが格納されるために重要です。障害または事故によってクラスタとその永続的ストレージが消去された場合は、バックアップをリカバリする必要があります。Snapshot を使用してリカバリすることはできません。

クローン

a_clone_は 、アプリケーション、その構成、永続的ストレージの完全な複製です。クローンは、同じ Kubernetes クラスタまたは別のクラスタに手動で作成できます。アプリケーションとストレージを Kubernetes クラスタ間で移動する必要がある場合は、アプリケーションをクローニングすると便利です。

ライセンス

Astra Control Centerを有効にするには、フルアプリケーションデータ管理機能を有効にするためのライセンスが必要です。Astra Control Center をライセンスなしで導入すると、システム機能が制限されていることを示すバナーが Web UI に表示されます。

次の操作には有効なライセンスが必要です。

- 新しいアプリケーションの管理
- Snapshot またはバックアップを作成します
- Snapshot またはバックアップのスケジュールを設定する保護ポリシーを設定する
- Snapshot またはバックアップからのリストア
- Snapshot または現在の状態からクローニングしています

> クラスタの追加、バケットの追加、 Astra Data Store ストレージバックエンドの管理を、ライセンスなしで実行できます。ただし、 Astra Data Store をストレージバックエンドとして使用するアプリケーションを管理するには、有効な Astra Control Center ライセンスが必要です。

ライセンス消費量の計算方法

新しいクラスタを Astra Control Center に追加しても、クラスター上で実行されているアプリケーションの少なくとも 1 つが Astra Control Center によって管理されるまで、使用済みのライセンスにはカウントされません。また、 Astra Data Store ストレージバックエンドを Astra Control Center に追加しても、ライセンスの消費には影響しません。これにより、ライセンスのない Astra Control Center システムから Astra Data Store バックエンドを管理できます。

クラスタ上でアプリケーションの管理を開始すると、クラスタの CPU ユニットが Astra Control Center ライセンス消費量の計算に含まれます。

詳細については、こちらをご覧ください

- "既存のライセンスを更新する"

検証済みのアプリケーションと標準のアプリケーションの比較

Astra Control には、検証済みと標準の 2 種類のアプリケーションが用意されています。これら 2 つのカテゴリの違いと、プロジェクトと戦略に与える可能性のある影響について説明します。

> この 2 つのカテゴリは、「サポート対象」と「サポート対象外」と考える傾向があります。 しかし、ここでは、 Astra Control の「サポートされていない」アプリケーションなどのものはありません。Astra Control には任意のアプリケーションを追加できますが、検証済みのアプリケーションは、 Astra Control ワークフローを中心に構築された、標準のアプリケーションよりも多くのインフラを備えています。

検証済みのアプリケーション

Astra Control の検証済みアプリケーションには、次のものがあります。

- MySQL 8.0.25
- MariaDB 10.5.9
- PostgreSQL 11.12
- Jenkins 2.277.4 LTS および 2.289.1 LTS

検証済みアプリケーションのリストは、 Astra Control が認識するアプリケーションを表します。Astra Control チームは、これらのアプリケーションを分析して確認し、完全なテストでリストアできることを確認しました。Astra Control は、カスタムワークフローを実行して、アプリケーションレベルでのスナップショットとバックアップの整合性を確保します。

アプリケーションが検証された場合、 Astra Control チームは、アプリケーションと整合性のあるスナップショットを取得するためにスナップショットを取得する前に、アプリケーションを休止するための手順を特定して実装しました。たとえば、 Astra Control が PostgreSQL データベースのバックアップを作成する場合、最初にデータベースを休止します。バックアップが完了すると、 Astra Control はデータベースを通常の動作に復元します。

Astra Control で使用するアプリケーションの種類に関係なく、必ず自分でバックアップとリストアのワークフローをテストして、ディザスタリカバリの要件を満たすようにしてください。

標準アプリ

カスタムプログラムを含むその他のアプリケーションは、標準アプリケーションと見なされます。Astra Control を使用して、標準アプリケーションを追加および管理できます。また、標準アプリケーションの基本的な crash-consistent Snapshot とバックアップを作成することもできます。ただし、これらは、アプリケーションを元の状態に戻すために完全にテストされていません。

> Astra Control 自体は標準のアプリケーションではなく、「システムアプリケーション」です。Astra Control 自体は、管理用にデフォルトでは表示されません。Astra Control 自体は管理しないでください。

ストレージクラスと永続的ボリュームサイズ

Astra Control Center は、 ONTAP または 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します。

概要

Astra Control Center は、次の機能をサポートします。

- *Trident ストレージクラスは Astra Data Store ストレージ * をサポート： 1 つ以上の Astra データストアクラスタを手動でインストールした場合、 Astra Control Center では、これらをインポートしてトポロジ（ノード、ディスク）とさまざまなステータスを取得することができます。

  Astra Control Center には、 Astra Data Store の構成、 Kubernetes クラスタが属するクラウド、 Astra Data Store でプロビジョニングされた永続ボリューム、対応する内部ボリュームの名前、永続ボリュームを使用するアプリケーション、およびアプリケーションを含むクラスタから、基盤となる Kubernetes クラスタが表示されます。

- * ONTAP ストレージ * がサポートする Trident ストレージクラス： ONTAP バックエンドを使用している場合、 Astra Control Center では、 ONTAP バックエンドをインポートしてさまざまな監視情報をレポートすることができます。

> Trident のストレージクラスは、 Astra Control Center の外部で事前に設定しておく必要があります。

ストレージクラス

Astra Control Center にクラスタを追加すると、そのクラスタで以前に設定したストレージクラスをデフォルトのストレージクラスとして選択するように求められます。このストレージクラスは、永続ボリューム要求（ PVC ）でストレージクラスが指定されていない場合に使用されます。デフォルトのストレージクラスは、 Astra Control Center 内でいつでも変更できます。また、 PVC または Helm チャート内のストレージクラスの名前を指定することで、任意のストレージクラスをいつでも使用できます。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。

Astra Control Center を Astra データストアストレージバックエンドと統合して使用する場合、インストール後にストレージクラスは定義されません。Trident のデフォルトストレージクラスを作成し、ストレージバックエンドに適用する必要があります。を参照してください "Astra データストア入門" をクリックして、デフォルトの Astra データストアストレージクラスを作成します。

を参照してください。

- "Astra Trident のドキュメント"

ユーザロールとネームスペース

Astra Control のユーザロールとネームスペースについて説明し、それらを使用して組織内のリソースへのアクセスを制御する方法を説明します。

ユーザロール

ロールを使用して、ユーザが Astra Control のリソースまたは機能にアクセスできるように制御できます。Astra Control のユーザロールは次のとおりです。

- * Viewer * はリソースを表示できます。
- メンバー * には、ビューア・ロールの権限があり、アプリとクラスタの管理、アプリの管理解除、スナップショットとバックアップの削除ができます。
- **Admin** にはメンバーの役割権限があり、 Owner 以外の他のユーザーを追加および削除できます。
- * Owner * には Admin ロールの権限があり、任意のユーザーアカウントを追加および削除できます。

メンバーまたはビューアユーザーに制約を追加して、ユーザーを 1 つ以上に制限できます [ネームスペース]。

ネームスペース

ネームスペースは、 Astra Control によって管理されるクラスタ内の特定のリソースに割り当てることができるスコープです。Astra Control では、 Astra Control にクラスタを追加すると、クラスタのネームスペースが検出されます。検出されたネームスペースは、ユーザに制約として割り当てることができます。そのリソースを使用できるのは、そのネームスペースにアクセスできるメンバーだけです。名前空間を使用すると、組織に適したパラダイム（たとえば、会社内の物理的なリージョンや部門）を使用して、リソースへのアクセスを制御できます。ユーザに制約を追加する場合は、そのユーザにすべてのネームスペースへのアクセス権を設定するか、特定のネームスペースのセットのみを設定できます。ネームスペースラベルを使用して、ネームスペースの制約を割り当てることもできます。

詳細については、こちらをご覧ください

"ロールの管理"

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

Astra Control Center には、次のいずれかのタイプの運用環境が必要です。

- Kubernetes 1.20 〜 1.23
- Rancher 2.5.8 、 2.5.9 、または 2.6 と RKE1
- Red Hat OpenShift Container Platform 4.4.8 、 4.7 、 4.8 、または 4.9
- VMware Tanzu Kubernetes Grid 1.4 の場合
- VMware Tanzu Kubernetes Grid Integrated Edition 1.12.2

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
Astra Trident	• Astra Trident 21.04 以降がインストールおよび設定されている • Astra Trident 21.10.1以降がインストールされ、Astraデータストアがストレージバックエンドとして使用される場合に設定されます

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 」という形式で指定します
> 2. 「 export-policy rule modify -vserver <storage virtual machine name> -policyname <policy name> -ruleindex 1 -anon 65534 」という形式で指定します

> 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、名前空間クラスタ上で「 TanzueutesCluster 」リソースを編集します。
- Astra Control Center がクラスタ内にポッドを作成できるようにするセキュリティポリシーを作成する必要があります。これを行うには、次のコマンドを使用します。

```
kubectl config use-context <context-of-workload-cluster>
kubectl create clusterrolebinding default-tkg-admin-privileged-binding
--clusterrole=psp:vmware-system-privileged --group=system:authenticated
```

- 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"

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。

- Astra データストア

- NetApp ONTAP 9.5 以降の AFF および FAS システム
- NetApp Cloud Volumes ONTAP の略

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は 'oneconconfig.yaml' ファイルが 1 つの _context_element だけを定義する必要がありますの 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、 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するように、 rancher から提供される「 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。

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		

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 "ネットアップサポートサイト"。

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。ライセンスがないと、次のことができません。

- カスタムアプリケーションを定義します
- 既存のアプリケーションのスナップショットまたはクローンを作成します

- データ保護ポリシーを設定

Astra Control Center をお試しになりたい場合は "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"。

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、「 kubectl get nsxlbmonitors -a` コマンドを使用して、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します。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

ソース	宛先	ポート	プロトコル	目的
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。

"Astra Control Center の要件の詳細をご覧ください"。

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 "ネットアップサポートサイトの「 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。

"Astra Control Center のインストールの詳細"。

3 いくつかの初期セットアップ作業を完了します

- ライセンスを追加します
- 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 またはを追加します "Astra データストア" ストレージバックエンド：
- 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。

"初期セットアッププロセスの詳細については、こちらをご覧ください"。

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。 "アプリの管理方法については、こちらをご覧ください"。
- 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。 "Cloud Insights への接続の詳細については、こちらをご覧ください"。

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、ネットアップサポートサイト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、を使用することもできます "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"。
- 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。

  OpenShift の例：

  ```
  oc 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。

  OpenShift の例：

  ```
  oc 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必要があります。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。

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 NetApp-acc' （またはカスタム名前の）名前空間にインストールします
- デフォルトアカウントを作成します。
- Astra Control Center のこのインスタンスに対して、デフォルトの管理ユーザの電子メールアドレスとデフォルトのワンタイムパスワード「 ACC-<UUID_OF_INSTALLIVE> 」を設定します。このユーザーには、システムのオーナーロールが割り当てられ、 UI への初回ログイン時に必要になります。
- 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 Astra Control Centerは'pcloud'ネームスペースにのみ導入し'この手順 の手順で説明したNetApp-acc'ネームスペースまたはカスタムネームスペースには導入しないでください

> 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_deployment.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. から Astra Control Center バンドル（「 Astra - control-ccenter-[version].tar.gz` ）をダウンロードします "ネットアップサポートサイト"。
2. から Astra Control Center 証明書とキーの zip をダウンロードします "ネットアップサポートサイト"。
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 astra-control-center[version].pub
-signature <astra-control-center[version].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

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、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、「uname -a」コマ

ンドを使用してこの情報を収集できます。

手順

1. 使用可能なNetApp Astra 'kubectl'プラグイン・バイナリを列挙し'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名を書き留めます

```
ls kubectl-astra/
```

2. ファイルを標準のkubectl'ユーティリティと同じ場所にコピーしますこの例では'kubectl'ユーティリティは'/usr/local/bin'ディレクトリにあります「<binary-name>」を必要なファイル名に置き換えます。

```
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. Astraディレクトリに移動します。

```
cd acc
```

2. Astra Control Center イメージディレクトリ内のファイルをローカルレジストリに追加します。

> 以下の画像の自動ロードについては、サンプルスクリプトを参照してください。

   a. レジストリにログインします。

      Docker ：

```
docker login [your_registry_path]
```

      Podman ：

```
podman login [your_registry_path]
```

   b. 適切なスクリプトを使用して、イメージのロード、イメージのタグ付け、[[[[[</Z1>[</Z1>[</Z1>_image_local_registry_push]] ローカルレジストリにイメージをプッシュします。 </Z2>

      Docker ：

```
export REGISTRY=[Docker_registry_path]
for astraImageFile in $(ls images/*.tar) ; do
  # Load to local cache. And store the name of the loaded image
trimming the 'Loaded images: '
  astraImage=$(docker load --input ${astraImageFile} | sed 's/Loaded
image: //')
  astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
  # Tag with local image repo.
  docker tag ${astraImage} ${REGISTRY}/${astraImage}
  # Push to the local repo.
  docker push ${REGISTRY}/${astraImage}
done
```

Podman ：

```
export REGISTRY=[Registry_path]
for astraImageFile in $(ls images/*.tar) ; do
  # Load to local cache. And store the name of the loaded image trimming
the 'Loaded images: '
  astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //')
  astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
  # Tag with local image repo.
  podman tag ${astraImage} ${REGISTRY}/${astraImage}
  # Push to the local repo.
  podman push ${REGISTRY}/${astraImage}
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. 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. NetApp-acc-operator という名前空間を作成します。

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

   対応：

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

   b. NetApp-acc-operator ネームスペースのシークレットを作成します。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-operator --docker-server=[your_registry_path] --docker
-username=[username] --docker-password=[token]
```

回答例：

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

c. NetApp-acc`（またはカスタムの名前を付けた）ネームスペースを作成します。

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

回答例：

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

d. NetApp-acc`（またはカスタムの名前を付けた）ネームスペースのシークレットを作成します。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. Astra Control Center オペレータの配備 YAML (`Astra_control_center_deployment.yaml ') を編集して、ローカルのレジストリと秘密を参照します。

```
vim astra_control_center_operator_deploy.yaml
```

a. 認証が必要なレジストリを使用する場合は、デフォルト行の「 imagePullSecret:[] 」を次のように置き換えます。

```
imagePullSecrets:
- name: <name_of_secret_with_creds_to_local_registry>
```

b. 「 kube-rbac プロキシ」イメージの「 [Your_registry_path] 」を、でイメージをプッシュしたレジストリパスに変更します 前の手順。

c. 「 acc-operator-controller-manager 」イメージの「 [Your_registry_path] 」を、でイメージをプッシュしたレジストリパスに変更します 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: []
```

2. 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
```

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（「 Astra _control_center_min YAML 」）を編集して、アカウント、 AutoSupport 、レジストリ、およびその他の必要な構成を作成します。

> ⓘ 環境に追加のカスタマイズが必要な場合は 'Astra_control_center.yaml を代替 CR として使用できます「 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。

```
vim astra_control_center_min.yaml
```

> ⓘ CR によって設定されたプロパティは、最初の Astra Control Center の導入後に変更することはできません。

> ⓘ 認証を必要としないレジストリを使用している場合は、「 imageRegistry 」内の「秘密」行を削除する必要があります。削除しないとインストールが失敗します。

a. 前の手順でイメージをプッシュしたレジストリパスに '[Your_registry_path]' を変更します

b. 「 accountName 」文字列を、アカウントに関連付ける名前に変更します。

c. 「 astraトラ アドレス」文字列をブラウザで使用する FQDN に変更して、 Astra にアクセスします。アドレスには 'http://' または https:// を使用しないでくださいこの FQDN をコピーしてで使用します 後の手順。

d. 「 email 」の文字列をデフォルトの初期管理者アドレスに変更します。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. インターネットに接続されていないサイトの場合は AutoSupport の「 enrolled 」を「 false 」に変更し、接続されているサイトの場合は「 true 」を保持します。

f. （オプション）アカウントに関連付けられたユーザの姓「 firstName 」と名「 lastName 」を追加します。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。

g. （任意）インストールで必要に応じて、「 torageClass 」の値を別の Trident ストレージクラスリソースに変更します。

h. （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使用して、シークレットの名前を指定します。この YAML ファイルに「 astraeKubeConfigSecret ： "acc-kubeconfig -cred or custom secret name" 」という名前の新しいフィールドを追加します

i. 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 (service/traefik`) が ClusterIP` タイプに設定されます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: AccTraefik) *: 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、「 ingressType 」を「 AccTraefik 」に設定します。

  これにより 'Astra Control Center traefik' ゲートウェイが Kubernetes LoadBalancer タイプのサービスとして導入されます

  Astra Control Center は、 Astra Control Center ネームスペースの "LoadBalancer (svc/traefik`) タイプのサービスを使用し、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 NetApp-acc` （またはカスタム）ネームスペースを作成していない場合は、次のようにします。

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

回答例：

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

2. Astra Control Center を NetApp-acc' （またはカスタムの）名前空間にインストールします

```
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は「 Running 」になります。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-5f75c5f564-bzqmt              1/1     Running   0
11m
activity-6b8f7cccb9-mlrn4                   1/1     Running   0
9m2s
api-token-authentication-6hznt              1/1     Running   0
8m50s
api-token-authentication-qpfgb              1/1     Running   0
8m50s
api-token-authentication-sqnb7              1/1     Running   0
8m50s
asup-5578bbdd57-dxkbp                       1/1     Running   0
9m3s
authentication-56bff4f95d-mspmq             1/1     Running   0
7m31s
bucketservice-6f7968b95d-9rrrl              1/1     Running   0
8m36s
cert-manager-5f6cf4bc4b-82khn               1/1     Running   0
6m19s
cert-manager-cainjector-76cf976458-sdrbc    1/1     Running   0
6m19s
cert-manager-webhook-5b7896bfd8-2n45j       1/1     Running   0
6m19s
cloud-extension-749d9f684c-8bdhq            1/1     Running   0
9m6s
cloud-insights-service-7d58687d9-h5tzw      1/1     Running   2
8m56s
composite-compute-968c79cb5-nv7l4           1/1     Running   0
9m11s
composite-volume-7687569985-jg9gg           1/1     Running   0
8m33s
```

```
credentials-5c9b75f4d6-nx9cz                1/1     Running   0
8m42s
entitlement-6c96fd8b78-zt7f8                1/1     Running   0
8m28s
features-5f7bfc9f68-gsjnl                   1/1     Running   0
8m57s
fluent-bit-ds-h88p7                         1/1     Running   0
7m22s
fluent-bit-ds-krhnj                         1/1     Running   0
7m23s
fluent-bit-ds-l5bjj                         1/1     Running   0
7m22s
fluent-bit-ds-lrclb                         1/1     Running   0
7m23s
fluent-bit-ds-s5t4n                         1/1     Running   0
7m23s
fluent-bit-ds-zpr6v                         1/1     Running   0
7m22s
graphql-server-5f5976f4bd-vbb4z             1/1     Running   0
7m13s
identity-56f78b8f9f-8h9p9                   1/1     Running   0
8m29s
influxdb2-0                                 1/1     Running   0
11m
krakend-6f8d995b4d-5khkl                    1/1     Running   0
7m7s
license-5b5db87c97-jmxzc                    1/1     Running   0
9m
login-ui-57b57c74b8-6xtv7                   1/1     Running   0
7m10s
loki-0                                      1/1     Running   0
11m
monitoring-operator-9dbc9c76d-8znck         2/2     Running   0
7m33s
nats-0                                      1/1     Running   0
11m
nats-1                                      1/1     Running   0
10m
nats-2                                      1/1     Running   0
10m
nautilus-6b9d88bc86-h8kfb                   1/1     Running   0
8m6s
nautilus-6b9d88bc86-vn68r                   1/1     Running   0
8m35s
openapi-b87d77dd8-5dz9h                     1/1     Running   0
9m7s
```

```
polaris-consul-consul-5ljfb                1/1     Running   0
11m
polaris-consul-consul-s5d5z                1/1     Running   0
11m
polaris-consul-consul-server-0             1/1     Running   0
11m
polaris-consul-consul-server-1             1/1     Running   0
11m
polaris-consul-consul-server-2             1/1     Running   0
11m
polaris-consul-consul-twmpq                1/1     Running   0
11m
polaris-mongodb-0                          2/2     Running   0
11m
polaris-mongodb-1                          2/2     Running   0
10m
polaris-mongodb-2                          2/2     Running   0
10m
polaris-ui-84dc87847f-zrg8w                1/1     Running   0
7m12s
polaris-vault-0                            1/1     Running   0
11m
polaris-vault-1                            1/1     Running   0
11m
polaris-vault-2                            1/1     Running   0
11m
public-metrics-657698b66f-67pgt            1/1     Running   0
8m47s
storage-backend-metrics-6848b9fd87-w7x8r   1/1     Running   0
8m39s
storage-provider-5ff5868cd5-r9hj7          1/1     Running   0
8m45s
telegraf-ds-dw4hg                          1/1     Running   0
7m23s
telegraf-ds-k92gn                          1/1     Running   0
7m23s
telegraf-ds-mmxjl                          1/1     Running   0
7m23s
telegraf-ds-nhs8s                          1/1     Running   0
7m23s
telegraf-ds-rj7lw                          1/1     Running   0
7m23s
telegraf-ds-tqrkb                          1/1     Running   0
7m23s
telegraf-rs-9mwgj                          1/1     Running   0
7m23s
```

```
telemetry-service-56c49d689b-ffrzx         1/1     Running   0
8m42s
tenancy-767c77fb9d-g9ctv                   1/1     Running   0
8m52s
traefik-5857d87f85-7pmx8                   1/1     Running   0
6m49s
traefik-5857d87f85-cpxgv                   1/1     Running   0
5m34s
traefik-5857d87f85-lvmlb                   1/1     Running   0
4m33s
traefik-5857d87f85-t2xlk                   1/1     Running   0
4m33s
traefik-5857d87f85-v9wpf                   1/1     Running   0
7m3s
trident-svc-595f84dd78-zb816               1/1     Running   0
8m54s
vault-controller-86c94fbf4f-krttq          1/1     Running   0
9m24s
```

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、次のコマンドを使用して「 acc-operator' 」ログを監視します。

```
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操作の 1 つであり、失敗した場合、原因 の配備に失敗することはありません。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. すべてのポッドが動作している場合は、 Astra Control Center Operator によってインストールされた「 Astrad ControlCenter 」インスタンスを取得して、インストールが正常に完了したことを確認します。

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

4. YAML で、「 Deployed 」値に対する応答の「 status.deploymentState` 」フィールドを確認します。導入に失敗した場合は、代わりにエラーメッセージが表示されます。
5. Astra Control Center にログインするときに使用するワンタイムパスワードを取得するには、「 status.uuid 」値をコピーします。パスワードは「 ACC-` 」の後に UUID 値（「 ACC-[UUID] 」）、またはこの例では「 ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f 」）です。

YAML の詳細のサンプル

```
name: astra
   namespace: netapp-acc
   resourceVersion: "104424560"
   selfLink: /apis/astra.netapp.io/v1/namespaces/netapp-
acc/astracontrolcenters/astra
   uid: 9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f
 spec:
   accountName: Example
   astraAddress: astra.example.com
   astraVersion: 21.12.60
   autoSupport:
     enrolled: true
     url: https://support.netapp.com/asupprod/post/1.0/postAsup
   crds: {}
   email: admin@example.com
   firstName: SRE
   imageRegistry:
     name: registry_name/astra
     secret: astra-registry-cred
   lastName: Admin
 status:
   accConditionHistory:
     items:
     - astraVersion: 21.12.60
       condition:
         lastTransitionTime: "2021-11-23T02:23:59Z"
         message: Deploying is currently in progress.
         reason: InProgress
         status: "False"
         type: Ready
       generation: 2
       observedSpec:
         accountName: Example
         astraAddress: astra.example.com
         astraVersion: 21.12.60
         autoSupport:
           enrolled: true
           url: https://support.netapp.com/asupprod/post/1.0/postAsup
         crds: {}
         email: admin@example.com
         firstName: SRE
         imageRegistry:
           name: registry_name/astra
```

```
          secret: astra-registry-cred
        lastName: Admin
      timestamp: "2021-11-23T02:23:59Z"
    - astraVersion: 21.12.60
      condition:
        lastTransitionTime: "2021-11-23T02:23:59Z"
        message: Deploying is currently in progress.
        reason: InProgress
        status: "True"
        type: Deploying
      generation: 2
      observedSpec:
        accountName: Example
        astraAddress: astra.example.com
        astraVersion: 21.12.60
        autoSupport:
          enrolled: true
          url: https://support.netapp.com/asupprod/post/1.0/postAsup
        crds: {}
        email: admin@example.com
        firstName: SRE
        imageRegistry:
          name: registry_name/astra
          secret: astra-registry-cred
        lastName: Admin
      timestamp: "2021-11-23T02:23:59Z"
    - astraVersion: 21.12.60
      condition:
        lastTransitionTime: "2021-11-23T02:29:41Z"
        message: Post Install was successful
        observedGeneration: 2
        reason: Complete
        status: "True"
        type: PostInstallComplete
      generation: 2
      observedSpec:
        accountName: Example
        astraAddress: astra.example.com
        astraVersion: 21.12.60
        autoSupport:
          enrolled: true
          url: https://support.netapp.com/asupprod/post/1.0/postAsup
        crds: {}
        email: admin@example.com
        firstName: SRE
        imageRegistry:
```

```
          name: registry_name/astra
          secret: astra-registry-cred
        lastName: Admin
      timestamp: "2021-11-23T02:29:41Z"
    - astraVersion: 21.12.60
      condition:
        lastTransitionTime: "2021-11-23T02:29:41Z"
        message: Deploying succeeded.
        reason: Complete
        status: "False"
        type: Deploying
      generation: 2
      observedGeneration: 2
      observedSpec:
        accountName: Example
        astraAddress: astra.example.com
        astraVersion: 21.12.60
        autoSupport:
          enrolled: true
          url: https://support.netapp.com/asupprod/post/1.0/postAsup
        crds: {}
        email: admin@example.com
        firstName: SRE
        imageRegistry:
          name: registry_name/astra
          secret: astra-registry-cred
        lastName: Admin
      observedVersion: 21.12.60
      timestamp: "2021-11-23T02:29:41Z"
    - astraVersion: 21.12.60
      condition:
        lastTransitionTime: "2021-11-23T02:29:41Z"
        message: Astra is deployed
        reason: Complete
        status: "True"
        type: Deployed
      generation: 2
      observedGeneration: 2
      observedSpec:
        accountName: Example
        astraAddress: astra.example.com
        astraVersion: 21.12.60
        autoSupport:
          enrolled: true
          url: https://support.netapp.com/asupprod/post/1.0/postAsup
        crds: {}
```

```
        email: admin@example.com
        firstName: SRE
        imageRegistry:
          name: registry_name/astra
          secret: astra-registry-cred
        lastName: Admin
      observedVersion: 21.12.60
      timestamp: "2021-11-23T02:29:41Z"
    - astraVersion: 21.12.60
      condition:
        lastTransitionTime: "2021-11-23T02:29:41Z"
        message: Astra is deployed
        reason: Complete
        status: "True"
        type: Ready
      generation: 2
      observedGeneration: 2
      observedSpec:
        accountName: Example
        astraAddress: astra.example.com
        astraVersion: 21.12.60
        autoSupport:
          enrolled: true
          url: https://support.netapp.com/asupprod/post/1.0/postAsup
        crds: {}
        email: admin@example.com
        firstName: SRE
        imageRegistry:
          name: registry_name/astra
          secret: astra-registry-cred
        lastName: Admin
      observedVersion: 21.12.60
      timestamp: "2021-11-23T02:29:41Z"
  certManager: deploy
  cluster:
    type: OCP
    vendorVersion: 4.7.5
    version: v1.20.0+bafe72f
  conditions:
  - lastTransitionTime: "2021-12-08T16:19:55Z"
    message: Astra is deployed
    reason: Complete
    status: "True"
    type: Ready
  - lastTransitionTime: "2021-12-08T16:19:55Z"
    message: Deploying succeeded.
```

```
      reason: Complete
      status: "False"
      type: Deploying
    - lastTransitionTime: "2021-12-08T16:19:53Z"
      message: Post Install was successful
      observedGeneration: 2
      reason: Complete
      status: "True"
      type: PostInstallComplete
    - lastTransitionTime: "2021-12-08T16:19:55Z"
      message: Astra is deployed
      reason: Complete
      status: "True"
      type: Deployed
    deploymentState: Deployed
    observedGeneration: 2
    observedSpec:
      accountName: Example
      astraAddress: astra.example.com
      astraVersion: 21.12.60
      autoSupport:
        enrolled: true
        url: https://support.netapp.com/asupprod/post/1.0/postAsup
      crds: {}
      email: admin@example.com
      firstName: SRE
      imageRegistry:
        name: registry_name/astra
        secret: astra-registry-cred
      lastName: Admin
    observedVersion: 21.12.60
    postInstall: Complete
    uuid: 9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f
  kind: List
  metadata:
   resourceVersion: ""
   selfLink: ""
```

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（「 ingressType: Generic 」）の設定方法について説明します。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、「 ingressType: AccTraefik 」を設定できます。Astra Control Center は、 Astra Control Center ネームスペースの "LoadBalancer (svc/traefik`) タイプのサービスを使用し、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。

- 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。

Nginx Ingress Controller の手順

1. タイプのシークレットを作成します "8a637503539b25b68130b6e8003579d9" に示すように 'NetApp-acc'（またはカスタム名前の）名前空間内の TLS 秘密鍵と証明書の場合 "TLS シークレット"。
2. 非推奨または新しいスキーマのいずれかのv1beta1'（Kubernetesバージョン1.22で非推奨）またはv1'リソースタイプを使用して'NetApp-acc'（またはカスタムネームド）ネームスペースに入力リソースを導入します
   a. v1beta' 非推奨スキーマについては ' 次の例を参照してください

```
apiVersion: extensions/v1beta1
kind: Ingress
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. 「 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。

```
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 'Astra_control_center_min YAML の 'astraitAddress' で使用した FQDN を入力します 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 'eMA_control_center_min YAML で 'email' に使用した値を次のように入力します Astra Control Center をインストールした続き、ワンタイム・パスワード（「ACC-[UUID]」）を入力します。

   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のステータスが「Error」の場合は、ログを確認できます。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"。

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「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（「 available 」は「 true 」）になっていることを確認します。

```
oc get apiservices
```

- データセンターに Astra Control Center の FQDN アドレスを作成しておきます。
- 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と Red Hat OpenShift Container Platform へのアクセスが必要です。

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. から Astra Control Center バンドル（「 Astra - control-ccenter-[version].tar.gz` ）をダウンロードします "ネットアップサポートサイト"。
2. から Astra Control Center 証明書とキーの zip をダウンロードします "ネットアップサポートサイト"。
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 astra-control-center[version].pub
-signature <astra-control-center[version].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

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、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、「uname -a」コマンドを使用してこの情報を収集できます。

手順

1. 使用可能なNetApp Astra 'kubectl'プラグイン・バイナリを列挙し'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名を書き留めます

```
ls kubectl-astra/
```

2. ファイルを標準のkubectl'ユーティリティと同じ場所にコピーしますこの例では'kubectl'ユーティリティは'/usr/local/bin'ディレクトリにあります「<binary-name>」を必要なファイル名に置き換えます。

```
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. Astraディレクトリに移動します。

```
cd acc
```

2. Astra Control Center イメージディレクトリ内のファイルをローカルレジストリに追加します。

> 以下の画像の自動ロードについては、サンプルスクリプトを参照してください。

   a. レジストリにログインします。

      Docker ：

```
docker login [your_registry_path]
```

      Podman ：

```
podman login [your_registry_path]
```

   b. 適切なスクリプトを使用して、イメージのロード、イメージのタグ付け、[[[[[</Z1>[</Z1>[</Z1>_image_local_registry_push]] ローカルレジストリにイメージをプッシュします。 </Z2>

      Docker ：

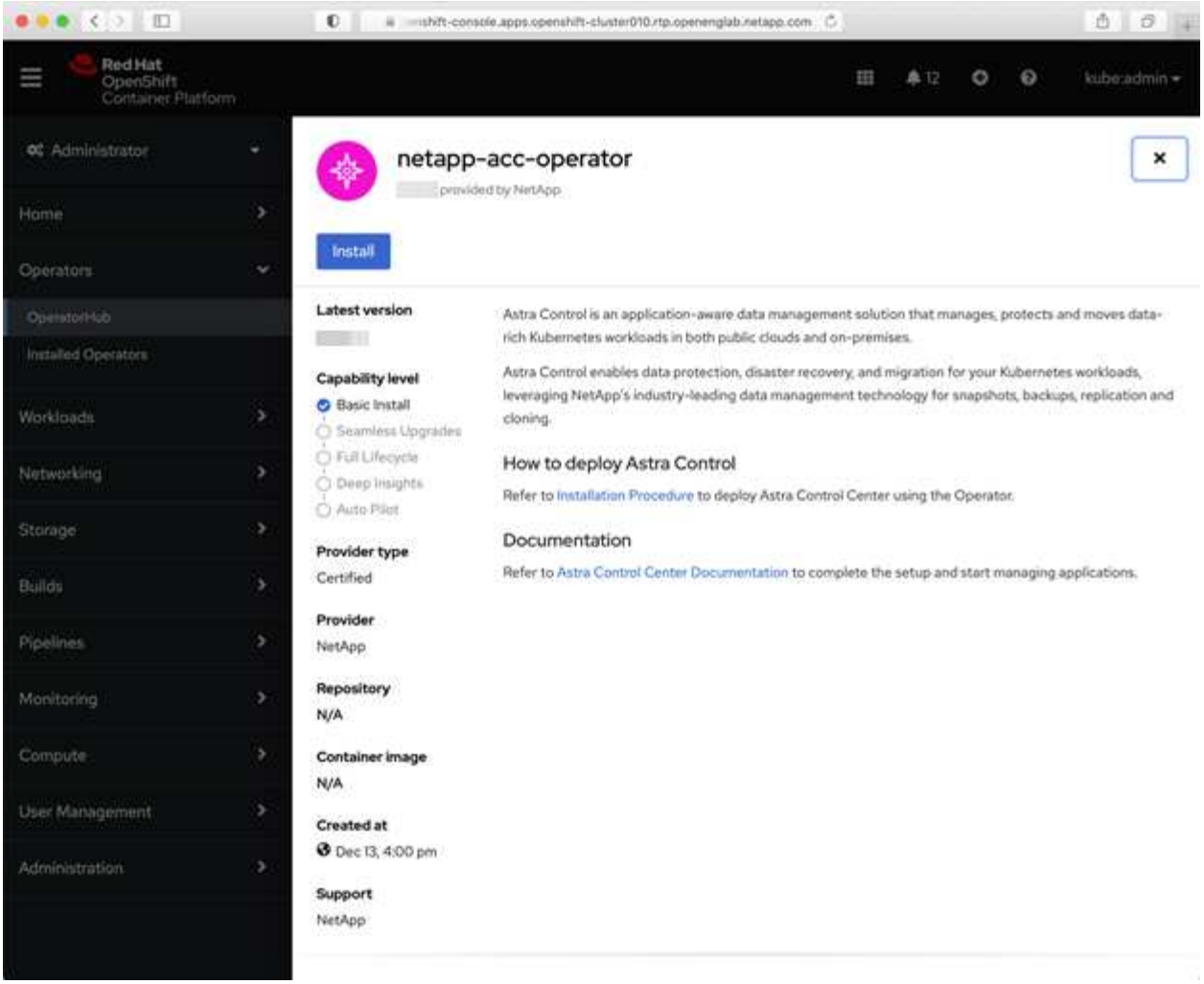
```
export REGISTRY=[Docker_registry_path]
for astraImageFile in $(ls images/*.tar) ; do
  # Load to local cache. And store the name of the loaded image
trimming the 'Loaded images: '
  astraImage=$(docker load --input ${astraImageFile} | sed 's/Loaded
image: //')
  astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
  # Tag with local image repo.
  docker tag ${astraImage} ${REGISTRY}/${astraImage}
  # Push to the local repo.
  docker push ${REGISTRY}/${astraImage}
done
```

Podman ：

```
export REGISTRY=[Registry_path]
for astraImageFile in $(ls images/*.tar) ; do
  # Load to local cache. And store the name of the loaded image trimming
the 'Loaded images: '
  astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //')
  astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
  # Tag with local image repo.
  podman tag ${astraImage} ${REGISTRY}/${astraImage}
  # Push to the local repo.
  podman push ${REGISTRY}/${astraImage}
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



:

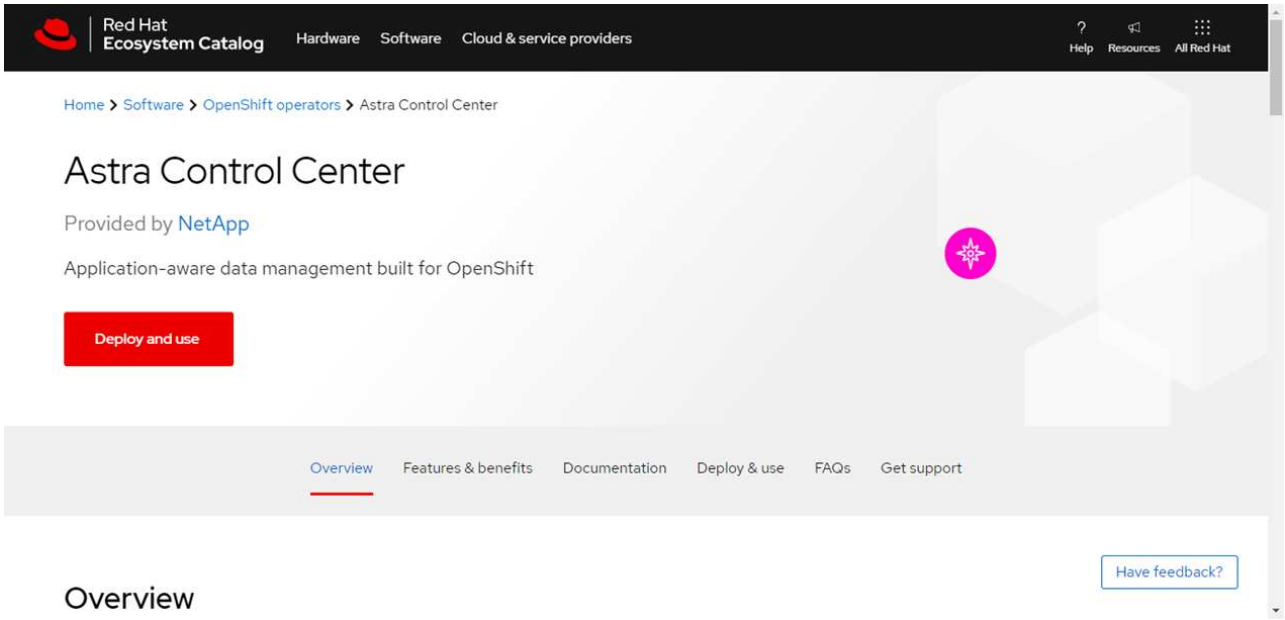
i. OpenShift Container Platform UI にログインします。

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 NetApp-acc-operator' 名前空間を選択すると、オペレータのインストール時に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 オペレータの詳細ビュー内のコンソールから、 [Provided API] セクションの [Create instance] を選択します。

2. Create AstraControlCenter フォーム・フィールドに次のように入力します

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。アドレスには 'http://' または https:// を入力しないでください
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。アドレスには 'http://' または https:// を入力しないでください
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. 「 Create 」を選択します。

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 ）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
- Microsoft Azure に Astra Control Center を導入

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。

Astra Control Center の導入でサポートされるのは、自己管理 OpenShift Container Platform （ OCP ）クラスタのみです。

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
- 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 」 -<>-singsingle-nas csi.trident.netapp.io` • 「 vsaworkingenvironment 」 -<>-singsingle-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"。

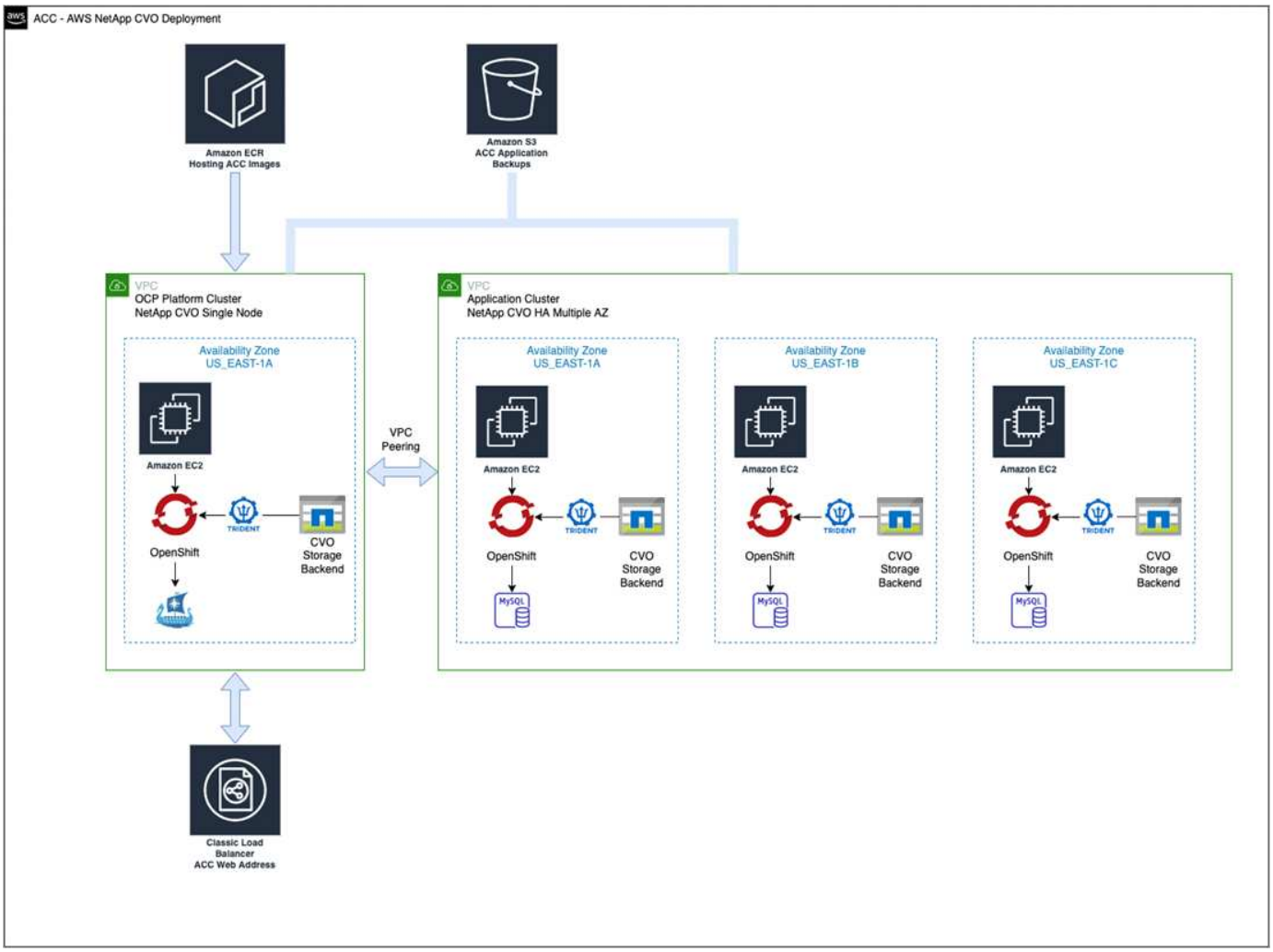
1. AWS仮想ネットワークを作成します。
2. EC2 コンピューティングインスタンスを確認します。AWS ではベアメタルサーバまたは VM を使用できます。
3. インスタンスタイプが、マスターノードとワーカーノードの Astra の最小リソース要件に一致していない場合は、 Astra の要件に合わせて AWS でインスタンスタイプを変更します。 を参照してください "Astra Control Center の要件"。
4. バックアップを格納する AWS S3 バケットを少なくとも 1 つ作成します。
5. すべての ACC イメージをホストする AWS Elastic Container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

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
- Red Hat OpenShift Container Platform （ OCP ） 4.8 の場合
- 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
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 」 -<>-singsingle-nas csi.trident.netapp.io` • 「 vsaworkingenvironment 」 -<>-singsingle-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

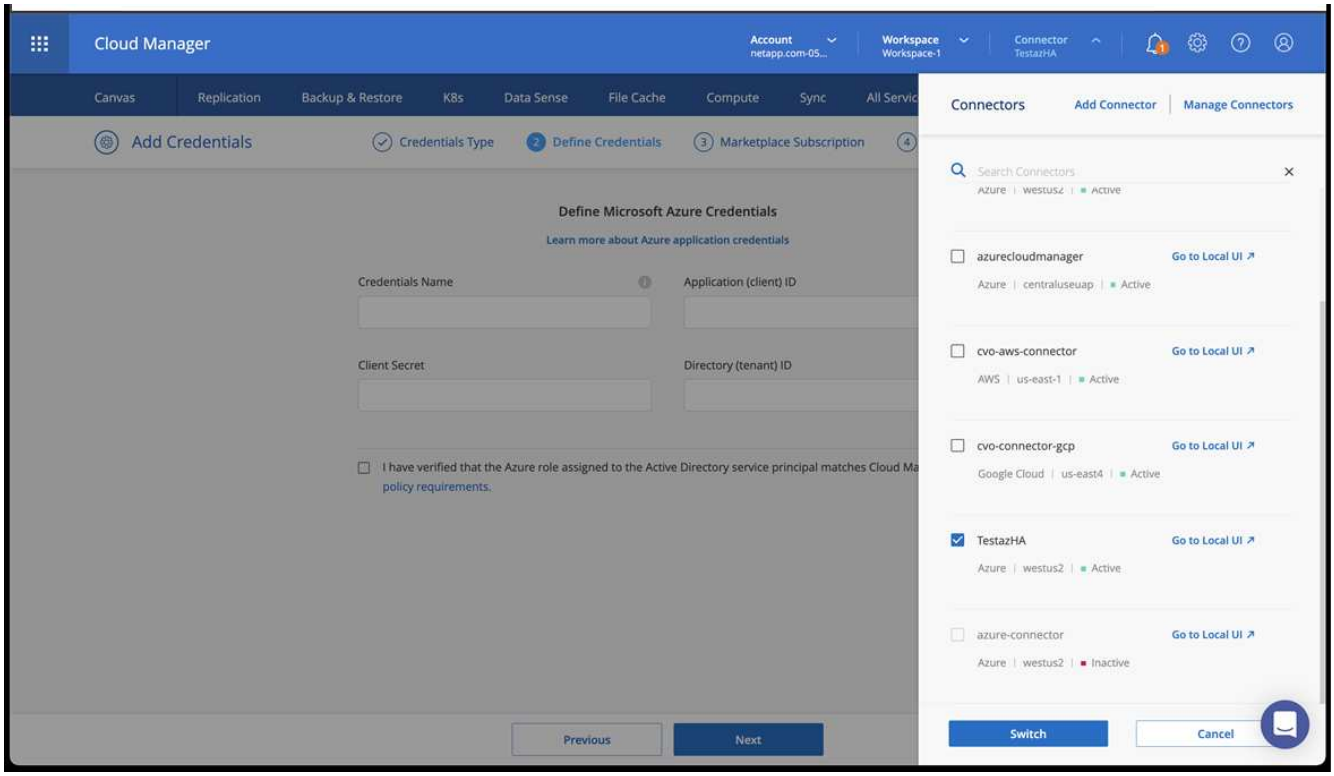
手順

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"。

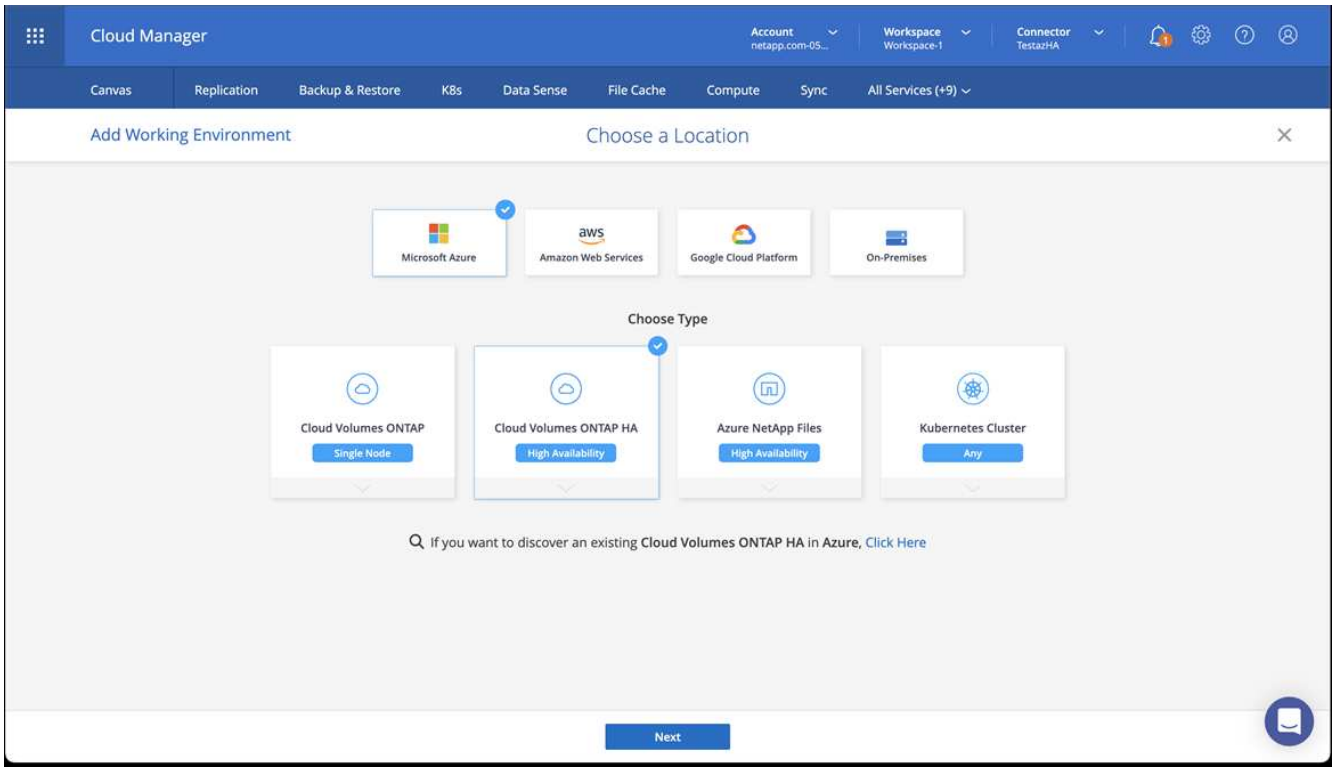
3. コネクタが動作していることを確認し、コネクタに切り替えます。



4. クラウド環境の作業環境を構築

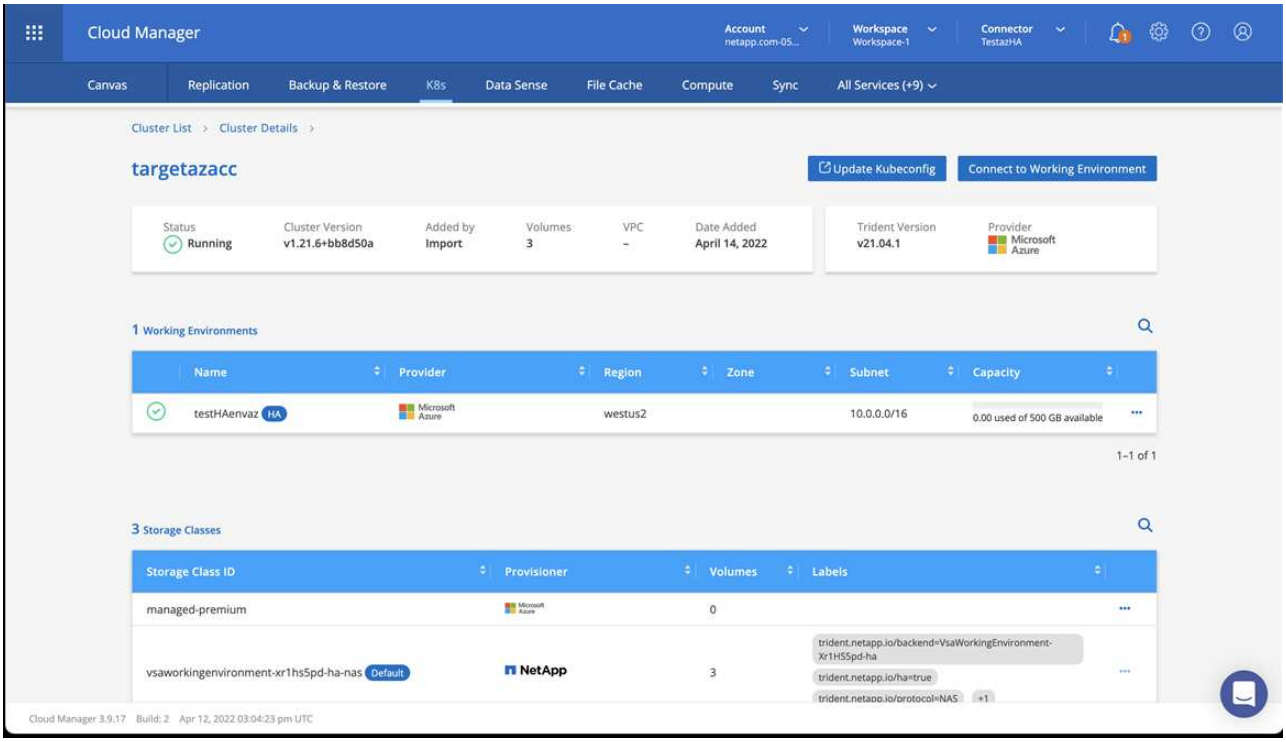
a. 場所：「 Microsoft Azure 」。

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 "ネットアップサポートサイト"。

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 "ネットアップサポートサイト"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。'kubectl'コマンドを使用してストレージ・クラスを変更すると'Astra Controlは変更を元に戻します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のPut / managedClustersエンドポイントを使用し'DefaultStorageClass'パラメータを指定して'別のデフォルトストレージクラスを割り当てます
- 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]。

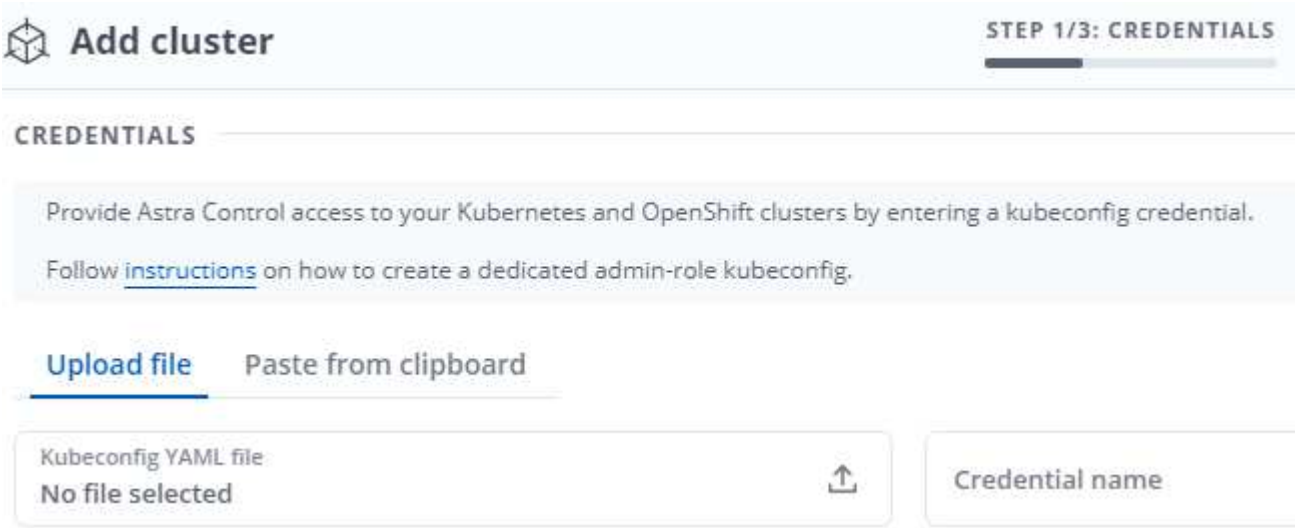
必要なもの

- 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「クラスタを追加」ウィンドウで、「 kubeconfig .yaml 」ファイルをアップロードするか、「 kubeconfig .yaml 」ファイルの内容を貼り付けます。

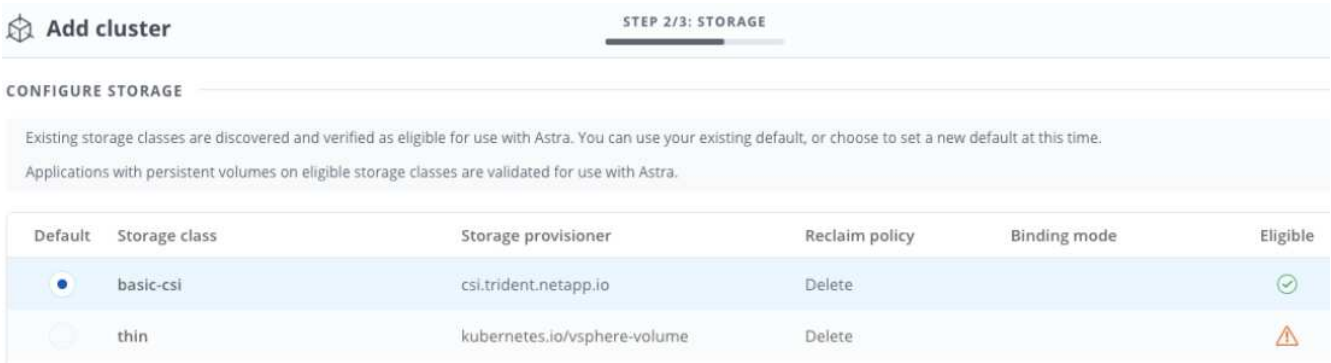
> 「 kubeconfig .yaml 」ファイルには、 1 つのクラスタのクラスタ・クレデンシャルのみを含める必要があります。



> 独自の「 kubeconfig 」ファイルを作成する場合は、その中で * 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" 「 kubeconfig 」ファイルの作成方法については、を参照してください。

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. 「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。数分で問題が解決しない場合は、クラスタに移動し、「 OC get pod -n NetApp-monitoring 」を開始点として実行します。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 Data Store用のKubernetesアプリケーションクラスタを追加し、Astra Controlによって管理されると、クラスタは検出されたバックエンドのリストに「unmanaged」と表示されます。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] メニューを選択し、 [`Manage] を選択して、およびを選択します "クラスタを追加"。「 unmanaged 」のクラスタ状態が Kubernetes クラスタの名前に変わったら、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クラスタの導入テンプレートサイズを選択します。

> 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。各テンプレートオプションは、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、適格なノードの数を提案します。

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astra.datastore.protection.domain`の場合は'astra.datastore.protection.domain=domain1`,`astra.datastore.protection.domain=domain2`およびastra.datastore.protection.domain=domain3`というラベルを作成できます

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、バックエンドはアクティブなパフォーマンス情報とともにバックエンドリストに「Available」状態で表示されます。

> 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。

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 *」を選択します。
     ii. ONTAP の管理者クレデンシャルを入力し、「 * Review * 」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。

結果

バックエンドは ' サマリー情報とともに ' リスト内の [Available （使用可能）] 状態で表示されます

> 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

> 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。

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、「このバケットをこのプライベートクラウドのデフォルトバケットにする」オプションを選択します。

> 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

- 次のいずれかのタイプのクラスタ：
  - OpenShift 4.4.8、4.7、4.8、または4.9を実行しているクラスタ
  - RKE1を使用してRancher 2.5.8、2.5.9、または2.6を実行しているクラスタ
  - Kubernetes 1.20 ~ 1.23を実行するクラスタ
  - VMware Tanzu Kubernetes Grid 1.4を実行しているクラスタ
  - VMware Tanzu Kubernetes Grid Integrated Edition 1.12.2を実行するクラスタ

  クラスタに 1 つ以上のワーカーノードがあり、テレメトリサービスの実行には 1GB 以上の RAM が使用されていることを確認します。

> 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 」
- 管理者によって定義された Astra Trident 'volumesnapshotclass' 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「 csi.trident.netapp.io` 」にする必要があります。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ます
- 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. 「 astracontrol-service-account.yaml 」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。

      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. 「 astracontrol-clusterrolbinding.yaml 」という名前の「 ClusterRoleBinding 」ファイルを作成します。

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. 「 <context> 」をインストールの正しいコンテキストに置き換えて、サービスアカウントのシークレットをリストします。

```
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"}
]
```

'ecsレット ' 配列内の各要素のインデックスは 0 から始まります上記の例では、「 astracontrol-service-account-dockercfg-vhz87 」のインデックスは 0 になり、「 astracontrol-service-account-token-r59kr 」のインデックスは 1 になります。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. 「 create-kubeconfig .sh` ファイル」を作成します。次のスクリプトの先頭にある「 token_index 」を正しい値に置き換えます。

```
<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
- クラスタ上のコマンド・シェルに管理アクセスして 'kubectl' コマンドを実行します
- 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> 」は、 Astra Control Center 配置名前空間に置き換えてください。

```
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、「 pec.selfSigned` 」の値を「 pec.ca.secretName` 」に変更して、前の手順で作成した TLS シークレットを参照します。

```
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、変更が正しく、クラスタが証明書を検証できる状態にあることを確認します。「 <ACC-deployment-namespace> 」は Astra Control Center 配置名前空間に置き換えてください。

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

```
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。

- 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。kubectl get nodes コマンドを使用して ' 新しいワーカー・ノードの正常性を確認することもできます

- 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。

Astra を使用する

アプリの管理

アプリの管理を開始します

お先にどうぞ "Astra Control 管理にクラスタを追加"では、クラスターにアプリケーションをインストールし（Astra Control の外部）、 Astra Control の [アプリ] ページに移動して、アプリケーションとそのリソースの管理を開始できます。

詳細については、を参照してください "アプリケーション管理の要件"。

サポートされているアプリインストール方法

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。これらの演算子は、一般に「パスバイリファレンス」アーキテクチャではなく「パスバイ値」で設計されています。これらのパターンに続くいくつかのオペレータアプリを次に示します。
  - "Apache K8ssandra"
  - "Jenkins CI"
  - "Percona XtraDB クラスタ"

Astra Control では、「パスバイリファレンス」アーキテクチャ（ CockroachDB オペレータなど）で設計されたオペレータをクローニングできない場合があります。クローニング処理では、クローニング処理の一環として独自の新しいシークレットが存在する場合でも、クローニングされたオペレータがソースオペレータから Kubernetes シークレットを参照しようとします。Astra Control がソースオペレータの Kubernetes シークレットを認識しないため、クローニング処理が失敗する場合があります。

> 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。

クラスタにアプリをインストールします

Astra Control にクラスタを追加したので、クラスタにアプリケーションをインストールしたり、既存のアプリケーションを管理したりできます。名前空間にスコープ指定されているアプリケーションはすべて管理でき

ます。ポッドがオンラインになったら、 Astra Control を使用してアプリケーションを管理できます。

Helm チャートから検証済みのアプリケーションを展開する方法については、次を参照してください。

- "Helm チャートから MariaDB を導入します"
- "Helm チャートから MySQL を導入します"
- "Helm チャートから Postgres を導入します"
- "Helm チャートから Jenkins をデプロイします"

アプリの管理

Astra Control を使用すると、アプリケーションをネームスペースレベルまたは Kubernetes ラベルで管理できます。

> Helm 2 でインストールされたアプリケーションはサポートされていません。

次のアクティビティを実行して、アプリケーションを管理できます。

- アプリの管理
  - [ネームスペースでアプリケーションを管理します]
  - Kubernetes ラベルでアプリケーションを管理
- [アプリケーションを無視します]
- [アプリの管理を解除します]

> Astra Control 自体は標準のアプリケーションではなく、「システムアプリケーション」です。Astra Control 自体は管理しないでください。Astra Control 自体は、管理用にデフォルトでは表示されません。システムアプリを表示するには、「システムアプリを表示」フィルタを使用します。

Astra Control API を使用してアプリケーションを管理する方法については、を参照してください "Astra の自動化と API に関する情報"。

> データ保護処理（クローン、バックアップ、リストア）が完了して永続ボリュームのサイズを変更したあと、新しいボリュームのサイズが UI に表示されるまでに最大 20 分かかります。データ保護処理にかかる時間は数分です。また、ストレージバックエンドの管理ソフトウェアを使用してボリュームサイズの変更を確認できます。

ネームスペースでアプリケーションを管理します

アプリページの * 検出された * セクションには、名前空間と Helm がインストールされたアプリ、またはそれらの名前空間内のカスタムラベル付きアプリが表示されます。各アプリケーションを個別に管理することも、ネームスペースレベルで管理することもできます。データ保護処理に必要な精度のレベルが重要になります。

たとえば、毎週同じ頻度で「 Maria 」のバックアップポリシーを設定したいのに、同じネームスペースにある「 MariaDB 」をバックアップする頻度を高く設定するとします。これらのニーズに基づいて、アプリケーションを個別に管理する必要があり、単一のネームスペースで管理する必要はありません。

Astra Control を使用すると、階層の両方のレベル（名前空間とその名前空間内のアプリケーション）を個別

に管理できますが、いずれか一方を選択することをお勧めします。Astra Control で実行したアクションは、ネームスペースレベルとアプリケーションレベルの両方で同時に実行される場合、失敗する可能性があります。

手順

1. 左側のナビゲーションバーから、「 * アプリケーション * 」を選択します。
2. [* Discovered * （検出されました * ）] フィルタ

3. 検出されたネームスペースのリストを表示します。ネームスペースを展開して、アプリケーションおよび関連するリソースを表示します。

   Astra Control では、 Helm アプリケーションとカスタムラベルの付いたアプリケーションがネームスペースに表示されます。Helm ラベルがある場合は、タグアイコンで指定されます。

4. [**Group**] 列を参照して、アプリケーションが実行している名前空間を確認します (フォルダアイコンで指定されています) 。
5. 各アプリケーションを個別に管理するか、ネームスペースレベルで管理するかを決定します。
6. 階層内の目的のレベルで目的のアプリケーションを検索し、 [アクション *] 列の [オプション] メニューから [* 管理 *] を選択します。
7. アプリを管理しない場合は、 [アクション *] 列の [オプション] メニューから [* 無視 *] を選択します。

   たとえば、「 Maria 」ネームスペースの下にあるすべてのアプリケーションを同じスナップショットポリシーとバックアップポリシーで管理したい場合は、ネームスペースを管理し、ネームスペース内のアプリケーションは無視してください。

8. 管理対象アプリのリストを表示するには、表示フィルターとして「 * 管理対象 * 」を選択します。

> 追加したアプリケーションの保護列に警告アイコンが表示されている場合は、バックアップされておらず、まだバックアップのスケジュールが設定されていないことを示しています。

9. 特定のアプリケーションの詳細を表示するには、アプリケーション名を選択します。

結果

管理対象として選択したアプリは、 [管理対象 *] タブから利用できるようになりました。無視されたアプリは、 * 無視された * タブに移動します。新しいアプリケーションがインストールされると、検出されたタブにはアプリが表示されないため、見つけやすくなり、管理も簡単になります。

Kubernetes ラベルでアプリケーションを管理

Astra Control の [アプリ] ページの上部には、「 * カスタムアプリの定義 * 」という名前のアクションが含まれています。このアクションを使用して、 Kubernetes ラベルで識別されるアプリケーションを管理できます。 "Kubernetes ラベルでカスタムアプリケーションを定義する方法については、こちらをご覧ください"。

手順

1. 左側のナビゲーションバーから、「 * アプリケーション * 」を選択します。
2. [* 定義（ Define ）] を選択します
3. [* カスタムアプリケーションの定義 * （ Define custom application * ）] ダイアログボックスで、アプリケーションを管理するために必要な情報を入力します。
   a. * 新しいアプリ * ：アプリの表示名を入力します。
   b. * クラスタ * ：アプリケーションが存在するクラスタを選択します。
   c. * 名前空間： * アプリケーションの名前空間を選択します。
   d. * ラベル： * ラベルを入力するか、以下のリソースからラベルを選択してください。
   e. * 選択したリソース * ：保護する Kubernetes リソース（ポッド、シークレット、永続ボリュームなど）を表示および管理します。
      - リソースを展開し、ラベル数を選択して、使用可能なラベルを表示します。
      - ラベルを 1 つ選択します。

        ラベルを選択すると、 [**Label**] フィールドにラベルが表示されます。Astra Control は、 [選択されていないリソース *] セクションも更新して、選択したラベルと一致しないリソースを表示します。
   f. * 選択されていないリソース * ：保護する必要がないアプリケーションリソースを確認します。
4. 「 * カスタムアプリケーションの定義 * 」を選択します。

結果

Astra Control を使用すると、アプリケーションを管理できます。これで、 [* 管理対象 * （ * Managed * ）] タブに表示されます。

アプリケーションを無視します

検出されたアプリケーションは、検出されたリストに表示されます。この場合は、新しくインストールされたアプリケーションを簡単に検索できるように、検出されたリストをクリーンアップできます。また、管理しているアプリケーションがあり、後でそれらを管理する必要がなくなる場合もあります。これらのアプリケーションを管理したくない場合は、無視するように指定できます。

また、アプリケーションを 1 つのネームスペースで同時に管理することもできます（ネームスペース管理）。ネームスペースから除外するアプリケーションは無視してかまいません。

手順

1. 左側のナビゲーションバーから、「 * アプリケーション * 」を選択します。
2. フィルタとして * Discovered * を選択します。
3. アプリケーションを選択します。
4. [* アクション * （ * Actions * ）] 列の [オプション（ Options ）] メニューから、 [* 無視 * （ * Ignore * ）] を選択
5. 無視を解除するには、 * 無視解除 * を選択します。

アプリの管理を解除します

アプリケーションのバックアップ、スナップショット、またはクローンを作成する必要がなくなった場合は、管理を停止できます。

> アプリケーションの管理を解除すると、以前に作成したバックアップやスナップショットは失われます。

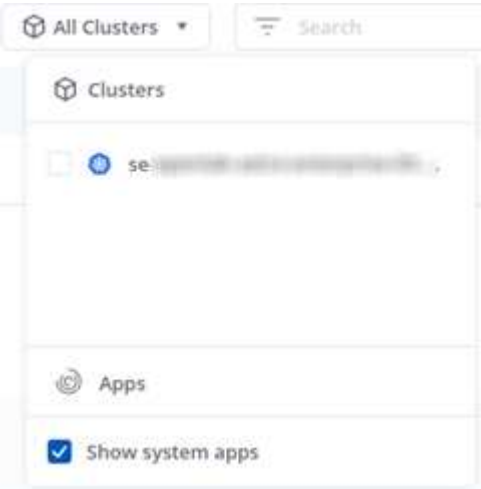
手順

1. 左側のナビゲーションバーから、「 * アプリケーション * 」を選択します。
2. フィルタとして [*Managed] を選択します。
3. アプリケーションを選択します。
4. * アクション * 列のオプションメニューから、 * 管理解除 * を選択します。
5. 情報を確認します。
6. 「 unmanage 」と入力して確定します。
7. [はい、アプリケーションの管理を解除 *] を選択します。

システムアプリケーションについて教えてください。

Astra Control は、 Kubernetes クラスタで実行されているシステムアプリケーションも検出します。これらのシステムアプリは、バックアップが必要になることが稀であるため、デフォルトでは表示されません。

ツールバーのクラスターフィルターの下にあるシステムアプリを表示 * システムアプリを表示 * チェックボックスをオンにすると、アプリケーションページからシステムアプリを表示できます。



> Astra Control 自体は標準のアプリケーションではなく、「システムアプリケーション」です。Astra Control 自体は管理しないでください。Astra Control 自体は、管理用にデフォルトでは表示されません。

詳細については、こちらをご覧ください

- "Astra Control API を使用"

カスタムアプリケーションの例を定義します

カスタムアプリケーションを作成すると、 Kubernetes クラスタの要素を 1 つのアプリケーションにグループ化できます。Kubernetes リソースのこの収集は、ネームスペースとラベルに基づいています。

カスタムアプリケーションを使用すると、 Astra Control 操作に含めるものをより細かく制御できます。次のものが含まれます。

- クローン
- スナップショット
- バックアップ
- 保護ポリシー

ほとんどの場合、アプリケーション全体で Astra Control の機能を使用します。ただし、これらの機能を使用するカスタムアプリケーションを、名前空間内の Kubernetes オブジェクトに割り当てるラベルで作成することもできます。

> カスタムアプリケーションは、単一クラスタの指定したネームスペース内でのみ作成できます。Astra Control では、カスタムアプリケーションを複数のネームスペースまたはクラスタにまたがって使用することはできません。

ラベルは、 Kubernetes オブジェクトに割り当てて識別できるキーと値のペアです。ラベルを使用すると、Kubernetes オブジェクトのソート、整理、検索が簡単になります。Kubernetes のラベルの詳細については、"Kubernetes の公式ドキュメントを参照してください"。

> 同じリソースに対して名前の異なるポリシーが重複していると、原因のデータが競合する可能性があります。リソースのカスタムアプリケーションを作成する場合は、そのアプリケーションが他のポリシーに基づいてクローニングまたはバックアップされていないことを確認してください。

必要なもの

- Astra Control にクラスタを追加

手順

1. [アプリケーション] ページで、 [**+ 定義] を選択します。

   [カスタムアプリケーション] ウィンドウには、カスタムアプリケーションに含まれるリソースまたはカスタムアプリケーションから除外されるリソースが表示されます。これにより、カスタムアプリケーションを定義するための正しい条件を選択できるようになります。

2. ポップアップウィンドウで、アプリケーション名を入力し、 **Cluster** ドロップダウンでクラスタを選択し、 **Namespace** ドロップダウンからアプリケーションの名前空間を選択します。
3. ドロップダウン * ラベル * リストから、アプリと名前空間のラベルを選択します。
4. 1 つの配置に対してカスタムアプリケーションを定義した後、必要に応じて他の配置についても同じ手順を繰り返します。

2 つのカスタムアプリケーションの作成が完了したら、これらのリソースを他の Astra Control アプリケーシ

ョンとして扱うことができます。Kubernetes ラベルに基づいて、リソースグループごとにデータのクローンを作成し、バックアップと Snapshot を作成し、リソースグループごとにカスタムの保護ポリシーを作成できます。

例：リリースごとに保護ポリシーを分ける

この例では、 DevOps チームがカナリアリリースの導入を管理しています。そのクラスタには nginx を実行するポッドが 3 つあります。そのうちの 2 つのポッドは、安定版リリース専用です。3 番目のポッドはカナリアリリース用です。

DevOps チームの Kubernetes 管理者は、安定したリリースポッドに「展開 = 安定」というラベルを追加します。チームは、カナリアリリースポッドに「展開 = カナリア」というラベルを追加します。

チームの安定版リリースには、 1 時間ごとの Snapshot と日次バックアップの要件が含まれています。カナリアリリースは、より一時的なものです。したがって、「配置」 = 「カナリア」というラベルの付いたすべてのものに対して、より積極的で短期的な保護ポリシーを作成したいと考えています。

データの競合を回避するために、管理者は 2 つのカスタムアプリケーションを作成します。 1 つは「カナリア」リリース用、もう 1 つは「 stable 」リリース用です。これにより、 Kubernetes オブジェクトの 2 つのグループに対して、バックアップ、 Snapshot 、およびクローニングの処理が分離されます。

アプリを保護します

保護の概要

Astra Control Center を使用して、アプリケーションのバックアップ、クローン、スナップショット、および保護ポリシーを作成できます。アプリケーションをバックアップすることで、サービスや関連データを可能な限り利用できるようになります。災害時にバックアップからリストアすることで、アプリケーションと関連データを最小限の中断で完全にリカバリできます。バックアップ、クローン、 Snapshot を使用すると、ランサムウェアや偶発的なデータ損失、環境障害などの一般的な脅威からデータを保護できます。 "Astra Control Center で使用可能なデータ保護の種類と、それらを使用するタイミングについて説明します"。

アプリケーション保護のワークフロー

次のワークフロー例を使用して、アプリケーションの保護を開始できます。

[1つ] すべてのアプリケーションをバックアップ

アプリケーションをすぐに保護するには、次の手順を実行します。 "すべてのアプリケーションの手動バックアップを作成する"。

[2 つ] 各アプリケーションの保護ポリシーを設定します

将来のバックアップとスナップショットを自動化するには、 "各アプリケーションの保護ポリシーを設定します"。たとえば、週単位のバックアップと日単位の Snapshot をそれぞれ 1 カ月ずつ保持して開始できます。手動バックアップやスナップショットよりも、保護ポリシーを使用してバックアップとスナップショットを自動化することを強く推奨します。

[3つ] オプション：保護ポリシーを調整します

アプリとその使用パターンが変化したら、必要に応じて保護ポリシーを調整して、最適な保護を実現します。

[4.] 災害が発生した場合は、アプリケーションをリストアします

データ損失が発生した場合は、を使用してリカバリできます "最新のバックアップをリストアしています" まず、各アプリケーションについて説明します。その後、最新の Snapshot をリストアできます（使用可能な場合）。

Snapshot とバックアップでアプリケーションを保護

自動保護ポリシーまたはアドホックベースを使用してスナップショットやバックアップを作成することで、アプリケーションを保護します。Astra の UI またはを使用できます "Astra Control API" アプリを保護します。

> Helm を使用してアプリケーションを展開する場合、 Astra Control Center 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）を使用して展開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 で展開されたアプリケーションはサポートされていません。

> OpenShift クラスタでアプリケーションをホストするプロジェクトを作成すると、プロジェクト（または Kubernetes ネームスペース）に SecurityContext UID が割り当てられます。Astra Control Center でアプリケーションを保護し、 OpenShift でそのアプリケーションを別のクラスタまたはプロジェクトに移動できるようにするには、アプリケーションを任意の UID として実行できるようにポリシーを追加する必要があります。たとえば、次の OpenShift CLI コマンドは、 WordPress アプリケーションに適切なポリシーを付与します。
>
> OC new-project ワードプレス ` OC adm policy add -scc to -group anyuid system: serviceaccounts:wordpress `OC adm policy add -scc to -user Privileged -z default-n wordpress

保護ポリシーを設定します

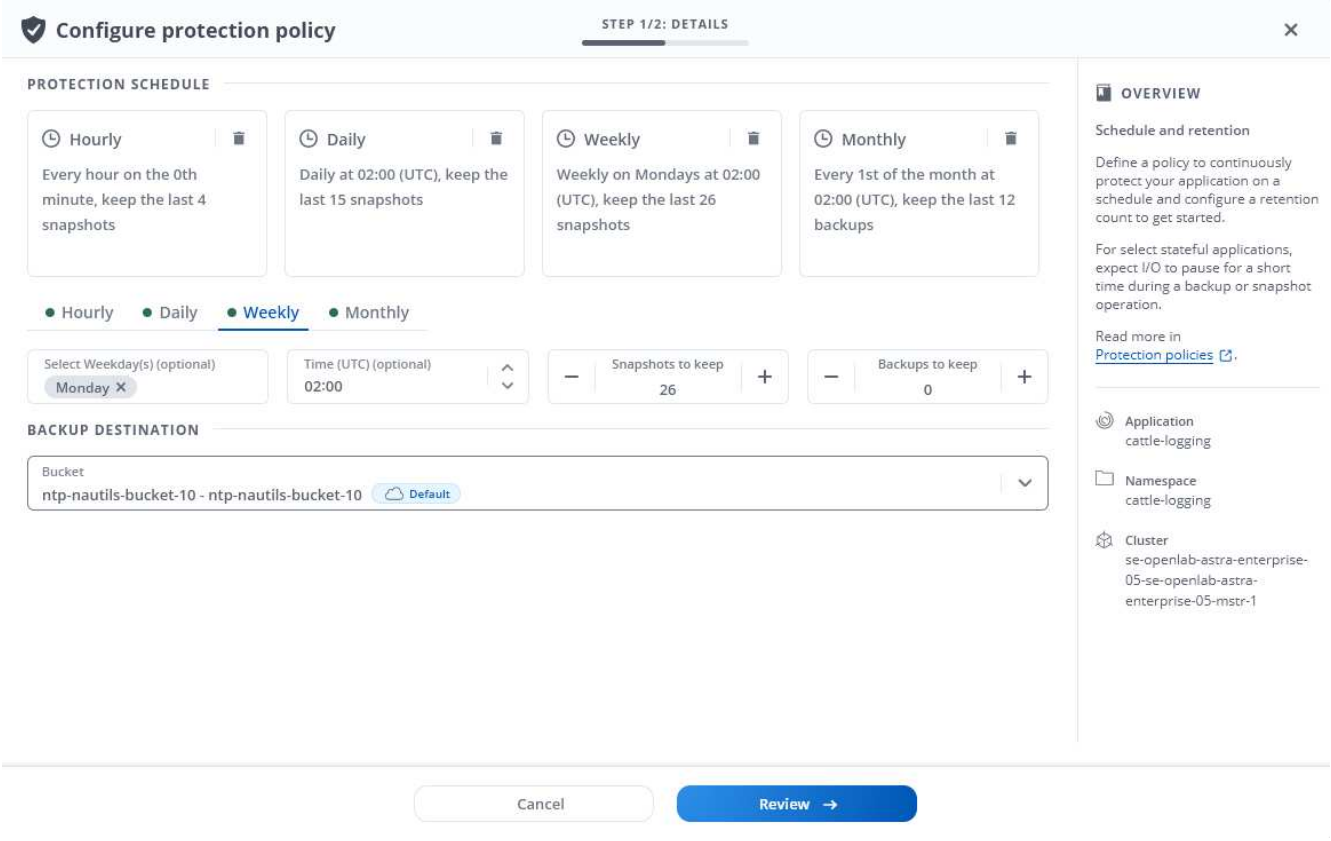
保護ポリシーは、定義されたスケジュールでスナップショット、バックアップ、またはその両方を作成することでアプリケーションを保護します。Snapshot とバックアップを毎時、日次、週次、および月単位で作成し、保持するコピーの数を指定できます。たとえば、保護ポリシーでは、週単位のバックアップと日単位の Snapshot が作成され、 1 カ月間はバックアップと Snapshot が保持されます。スナップショットやバックアップを作成する頻度と、それらを保持する期間は、組織のニーズによって異なります。

手順

1. 「 * アプリケーション」を選択し、アプリケーションの名前を選択します。
2. [* データ保護 *] を選択します。
3. ［ 保護ポリシーの設定 ］ を選択します。
4. 毎時、日次、週次、および月単位で保持する Snapshot とバックアップの数を選択して、保護スケジュールを定義します。

   スケジュールは、毎時、毎日、毎週、および毎月の各スケジュールで同時に定義できます。保持レベルを設定するまで、スケジュールはアクティブになりません。

   次の例では、 Snapshot とバックアップの保護スケジュールとして、毎時、毎日、毎週、毎月の 4 つを設定します。



5. [* Review （レビュー）] を選択します
6. ［ * 保護ポリシーの設定 * ］ を選択します

結果

Astra Control Center は、定義したスケジュールと保持ポリシーを使用して、スナップショットとバックアップを作成し、保持することによって、データ保護ポリシーを実装します。

Snapshot を作成します

オンデマンド Snapshot はいつでも作成できます。

手順

1. 「 * アプリケーション * 」を選択します。
2. 目的のアプリケーションの * アクション * 列のオプションメニューから、 * スナップショット * を選択します。
3. スナップショットの名前をカスタマイズし、 * Review * を選択します。
4. Snapshot の概要を確認し、「 * Snapshot * 」を選択します。

結果

スナップショットプロセスが開始されます。スナップショットは、ステータスが「 * 使用可能 * 」である場合に成功します。この場合、「 * データ保護 * > * スナップショット * 」ページの「 * アクション * 」列に表示されます。

バックアップを作成します

アプリケーションはいつでもバックアップできます。

> Astra Control Center の S3 バケットは、使用可能容量を報告しません。Astra Control Center で管理されているアプリケーションのバックアップまたはクローニングを行う前に、 ONTAP または StorageGRID 管理システムでバケット情報を確認します。

手順

1. 「 * アプリケーション * 」を選択します。
2. 目的のアプリケーションの * アクション * 列のオプションメニューから、 * バックアップ * を選択します。
3. バックアップ名をカスタマイズする。
4. 既存のスナップショットからアプリケーションをバックアップするかどうかを選択します。このオプションを選択すると、既存の Snapshot のリストから選択できます。
5. ストレージバケットのリストから選択して、バックアップのデスティネーションを選択します。
6. [* Review （レビュー）] を選択します
7. バックアップの概要を確認し、「 * Backup * 」を選択します。

結果

Astra Control Center は、アプリケーションのバックアップを作成します。

> ネットワークに障害が発生している場合や、処理速度が異常に遅い場合は、バックアップ処理がタイムアウトする可能性があります。その結果、バックアップは失敗します。

> 実行中のバックアップを停止する方法はありません。バックアップを削除する必要がある場合は、完了するまで待ってから、の手順を実行してください [バックアップを削除します]。失敗したバックアップを削除するには、 "Astra Control API を使用"。

> データ保護処理（クローン、バックアップ、リストア）が完了して永続ボリュームのサイズを変更したあと、新しいボリュームのサイズが UI に表示されるまでに最大 20 分かかります。データ保護処理にかかる時間は数分です。また、ストレージバックエンドの管理ソフトウェアを使用してボリュームサイズの変更を確認できます。

Snapshot とバックアップを表示します

アプリケーションのスナップショットとバックアップは、 [データ保護（ Data Protection ）] タブで表示できます。

手順

1. 「 * アプリケーション」を選択し、アプリケーションの名前を選択します。
2. [* データ保護 *] を選択します。

   デフォルトでは、 Snapshot が表示されます。

3. バックアップのリストを表示するには、「 * Backups * 」を選択します。

Snapshot を削除します

不要になったスケジュール済みまたはオンデマンドの Snapshot を削除します。

手順

1. 「 * アプリケーション」を選択し、アプリケーションの名前を選択します。
2. [* データ保護 *] を選択します。
3. 目的のスナップショットの * アクション * 列のオプションメニューから、 * スナップショットの削除 * を選択します。
4. 削除を確認するために「 delete 」と入力し、「 * はい、 Snapshot を削除します * 」を選択します。

結果

Astra Control Center がスナップショットを削除します。

バックアップを削除します

不要になったスケジュール済みまたはオンデマンドのバックアップを削除します。

> 実行中のバックアップを停止する方法はありません。バックアップを削除する必要がある場合は、完了するまで待ってから、以下の手順を実行してください。失敗したバックアップを削除するには、 "Astra Control API を使用"。

1. 「 * アプリケーション」を選択し、アプリケーションの名前を選択します。
2. [* データ保護 *] を選択します。
3. 「 * Backups * 」を選択します。
4. 目的のバックアップの [* アクション *] 列の [オプション] メニューから、 [* バックアップの削除 *] を選択します。
5. 削除を確認するために「 delete 」と入力し、「 * はい、バックアップを削除 * 」を選択します。

結果

Astra Control Center はバックアップを削除します。

アプリケーションのリストア

Astra Control を使用すると、スナップショットまたはバックアップからアプリケーションをリストアできます。同じクラスタにアプリケーションをリストアする場合、既存の Snapshot からのリストアは高速です。Astra Control UI またはを使用できます "Astra Control API" アプリを復元するには、

このタスクについて

- アプリケーションをリストアする前に、アプリケーションのスナップショットを作成するか、バックアップすることを強くお勧めします。リストアに失敗した場合に、Snapshotまたはバックアップからクローニングできます。
- Helm を使用してアプリケーションを展開する場合、 Astra Control Center 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）を使用して展開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 で展開されたアプリケーションはサポートされていません。

- 別のクラスタにリストアする場合は、同じ永続的ボリュームアクセスモード（ ReadWriteMany など）をクラスタで使用していることを確認してください。デスティネーションの永続ボリュームアクセスモードが異なると、リストア処理は失敗します。
- 名前空間の名前 / ID または名前空間ラベルによる名前空間の制約を持つメンバーユーザーは、同じクラスタ上の新しい名前空間、または組織のアカウント内の他のクラスタに対して、アプリケーションのクローンまたはリストアを実行できます。ただし、同じユーザが、クローニングまたはリストアされたアプリケーションに新しいネームスペースからアクセスすることはできません。クローンまたはリストア処理によって新しいネームスペースが作成されると、アカウントの管理者 / 所有者はメンバーユーザアカウントを編集し、該当するユーザに新しいネームスペースへのアクセスを許可するロールの制限を更新できます。
- OpenShift クラスタでアプリケーションをホストするプロジェクトを作成すると、プロジェクト（または Kubernetes ネームスペース）に SecurityContext UID が割り当てられます。Astra Control Center でアプリケーションを保護し、 OpenShift でそのアプリケーションを別のクラスタまたはプロジェクトに移動できるようにするには、アプリケーションを任意の UID として実行できるようにポリシーを追加する必要があります。たとえば、次の OpenShift CLI コマンドは、 WordPress アプリケーションに適切なポリシーを付与します。

  OC new-project ワードプレス ` OC adm policy add -scc to -group anyuid system: serviceaccounts:wordpress `OC adm policy add -scc to -user Privileged -z default-n wordpress

手順

1. 「 * アプリケーション」を選択し、アプリケーションの名前を選択します。
2. 「 * データ保護 * 」を選択します。
3. Snapshot からリストアする場合は、 * Snapshots * アイコンを選択したままにします。それ以外の場合は、「 * Backups * 」アイコンを選択してバックアップからリストアします。
4. リストア元のスナップショットまたはバックアップの [* アクション *] 列の [オプション] メニューから、[* アプリケーションのリストア *] を選択します。
5. * リストアの詳細 * ：リストアされたアプリの詳細を指定します。デフォルトでは、現在のクラスタとネームスペースが表示されます。これらの値をそのままにしておくと、アプリがインプレースで復元され、アプリは以前のバージョンのに戻ります。別のクラスタまたはネームスペースにリストアする場合は、これらの値を変更してください。
   - アプリケーションの名前と名前空間を入力します。
   - アプリケーションのデスティネーションクラスタを選択します。
   - [* Review （レビュー）] を選択します

   > 以前に削除したネームスペースにリストアすると、同じ名前の新しいネームスペースがリストアプロセスで作成されます。以前に削除したネームスペースでアプリケーションを管理する権限を持つユーザは、新しく作成したネームスペースに手動で権限を復元する必要があります。

6. * リストアの概要 * ：リストア操作の詳細を確認し、「 restore 」と入力して、 * Restore * を選択します。

結果

Astra Control Center は、指定した情報に基づいてアプリケーションを復元します。アプリケーションをインプレースでリストアした場合、既存の永続ボリュームの内容が、リストアしたアプリケーションの永続ボリュ

ームの内容に置き換えられます。

> データ保護処理（クローン、バックアップ、リストア）が完了して永続ボリュームのサイズが変更されたあと、Web UIに新しいボリュームサイズが表示されるまでに最大20分かかります。データ保護処理にかかる時間は数分です。また、ストレージバックエンドの管理ソフトウェアを使用してボリュームサイズの変更を確認できます。

アプリケーションのクローン作成と移行

既存のアプリケーションをクローニングして、同じ Kubernetes クラスタまたは別のクラスタに重複するアプリケーションを作成する。Astra Control Center は、アプリケーションのクローンを作成するときに、アプリケーション構成と永続的ストレージのクローンを作成します。

Kubernetes クラスタ間でアプリケーションとストレージを移動する必要がある場合は、クローニングが役立ちます。たとえば、 CI / CD パイプラインや Kubernetes ネームスペース間でワークロードを移動できます。Astra の UI またはを使用できます "Astra Control API" アプリケーションのクローン作成と移行を実行します。

必要なもの

アプリケーションを別のクラスタにクローニングするには、デフォルトのバケットが必要です。最初のバケットを追加した時点でデフォルトのバケットになります。

このタスクについて

- StorageClass が明示的に設定されたアプリケーションを展開し、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、ターゲットクラスタには元の StorageClass が指定されている必要があります。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。
- オペレータが配置した Jenkins CI のインスタンスをクローニングする場合は、永続データを手動でリストアする必要があります。これは、アプリケーションの展開モデルの制限事項です。
- Astra Control Center の S3 バケットは、使用可能容量を報告しません。Astra Control Center で管理されているアプリケーションのバックアップまたはクローニングを行う前に、 ONTAP または StorageGRID 管理システムでバケット情報を確認します。
- アプリケーションのバックアップやリストア時に、バケット ID を必要に応じて指定することができます。ただし、アプリケーションのクローニング処理では、定義済みのデフォルトバケットが常に使用されます。クローンのバケットを変更するオプションはありません。どのバケットを使用するかを制御する必要がある場合は、どちらかを選択できます "バケットのデフォルト設定を変更する" または、を実行します "バックアップ" その後にを押します "リストア" 個別。
- 名前空間の名前 / ID または名前空間ラベルによる名前空間の制約を持つメンバーユーザーは、同じクラスタ上の新しい名前空間、または組織のアカウント内の他のクラスタに対して、アプリケーションのクローンまたはリストアを実行できます。ただし、同じユーザが、クローニングまたはリストアされたアプリケーションに新しいネームスペースからアクセスすることはできません。クローンまたはリストア処理によって新しいネームスペースが作成されると、アカウントの管理者 / 所有者はメンバーユーザアカウントを編集し、該当するユーザに新しいネームスペースへのアクセスを許可するロールの制限を更新できます。

OpenShift に関する考慮事項

- クラスタ間でアプリケーションをクローニングする場合、ソースクラスタとデスティネーションクラスタは OpenShift の同じディストリビューションである必要があります。たとえば、 OpenShift 4.7 クラスタからアプリケーションをクローニングする場合は、 OpenShift 4.7 でもあるデスティネーションクラスタ

を使用します。

- OpenShift クラスタでアプリケーションをホストするプロジェクトを作成すると、プロジェクト（または Kubernetes ネームスペース）に SecurityContext UID が割り当てられます。Astra Control Center でアプリケーションを保護し、 OpenShift でそのアプリケーションを別のクラスタまたはプロジェクトに移動できるようにするには、アプリケーションを任意の UID として実行できるようにポリシーを追加する必要があります。たとえば、次の OpenShift CLI コマンドは、 WordPress アプリケーションに適切なポリシーを付与します。

  OC new-project ワードプレス ` OC adm policy add -scc to -group anyuid system: serviceaccounts:wordpress `OC adm policy add -scc to -user Privileged -z default-n wordpress

手順

1. 「 * アプリケーション * 」を選択します。
2. 次のいずれかを実行します。
   - 目的のアプリケーションの [* アクション * （ * Actions * ）] 列で [オプション（ Options ）] メニューを選択します。
   - 目的のアプリケーションの名前を選択し、ページの右上にあるステータスドロップダウンリストを選択します。
3. 「 * Clone * 」を選択します。
4. * クローンの詳細 * ：クローンの詳細を指定します。
   - 名前を入力します。
   - クローンのネームスペースを入力します。
   - クローンのデスティネーションクラスタを選択してください。
   - 既存の Snapshot からクローンを作成するかバックアップを作成するかを選択します。このオプションを選択しない場合、 Astra Control Center はアプリケーションの現在の状態からクローンを作成します。
5. * 出典 * ：既存のスナップショットまたはバックアップからクローンを作成する場合は、使用するスナップショットまたはバックアップを選択します。
6. [* Review （レビュー）] を選択します
7. * Clone Summary * ：クローンの詳細を確認し、 * Clone * を選択します。

結果

Astra Control Center では、入力した情報に基づいてアプリケーションのクローンを作成します。新しいアプリケーション・クローンが [*Applications] ページの [Available （使用可能）] 状態になっている場合 ' クローン操作は正常に実行されます

> データ保護処理（クローン、バックアップ、リストア）が完了して永続ボリュームのサイズを変更したあと、新しいボリュームのサイズが UI に表示されるまでに最大 20 分かかります。データ保護処理にかかる時間は数分です。また、ストレージバックエンドの管理ソフトウェアを使用してボリュームサイズの変更を確認できます。

アプリケーション実行フックを管理します

実行フックは、管理対象アプリケーションのスナップショットの前または後に実行でき

るカスタムスクリプトです。たとえば、データベースアプリケーションがある場合、実行フックを使用して、スナップショットの前にすべてのデータベーストランザクションを一時停止し、スナップショットの完了後にトランザクションを再開できます。これにより、アプリケーションと整合性のある Snapshot を作成できます。

デフォルトの実行フックと正規表現

一部のアプリケーションでは、ネットアップが提供するデフォルトの実行フックが Astra Control に付属しており、スナップショットの前後にフリーズや再開の操作を処理します。Astra Control では、正規表現を使用して、アプリケーションのコンテナイメージを次のアプリケーションに照合します。

- MariaDB
  - 正規表現 \bmariadb\b に一致しています
- MySQL
  - 正規表現 \bmysqql\b に一致しています
- PostgreSQL
  - 正規表現 \bpostgresql\b と一致します

一致した場合は、そのアプリケーションのデフォルトの実行フックがアプリケーションのアクティブな実行フックのリストに表示され、そのアプリケーションのスナップショットが作成されると、それらのフックが自動的に実行されます。カスタムアプリケーションの 1 つに、正規表現の 1 つと一致するように表示されるイメージ名が似ている場合（デフォルトの実行フックを使用しない場合）、イメージ名を変更することができます。 または、そのアプリケーションのデフォルト実行フックを無効にして、代わりにカスタムフックを使用します。

デフォルトの実行フックを削除または変更することはできません。

カスタム実行フックに関する重要な注意事項

アプリケーションの実行フックを計画するときは、次の点を考慮してください。

- Astra Control では、実行フックを実行可能なシェルスクリプトの形式で記述する必要があります。
- スクリプトサイズは 128KB に制限されています。
- Astra Control は、実行フックの設定と一致条件を使用して、スナップショットに適用できるフックを決定します。
- 実行フックの障害はすべてソフトな障害です他のフックとスナップショットは ' フックが失敗しても試行されますただし、フックが失敗すると、 * アクティビティ * ページイベントログに警告イベントが記録されます。
- 実行フックを作成、編集、または削除するには、 Owner 、 Admin 、または Member 権限を持つユーザーである必要があります。
- 実行フックの実行に 25 分以上かかる場合 ' フックは失敗し ' 戻りコードが N/A のイベント・ログ・エントリが作成されます該当する Snapshot はタイムアウトして失敗とマークされ、タイムアウトを通知するイベントログエントリが生成されます。

> 実行フックは、実行中のアプリケーションの機能を低下させるか、完全に無効にすることが多いため、カスタム実行フックの実行時間を最小限に抑えるようにしてください。

スナップショットが実行されると、実行フックイベントが次の順序で実行されます。

1. ネットアップが提供するデフォルトの Snapshot 前実行フックは、該当するコンテナで実行されます。
2. 適用可能なカスタムスナップショット前実行フックは、適切なコンテナで実行されます。必要な数のカスタムスナップショット前フックを作成して実行できますが ' スナップショットの実行順序は保証も構成もされていません
3. スナップショットが実行されます。
4. 適用可能なカスタムスナップショット後実行フックは、適切なコンテナで実行されます。必要な数のカスタムスナップショット後フックを作成して実行できますが ' スナップショット後のこれらのフックの実行順序は保証されておらず ' 構成もできません
5. ネットアップが提供するデフォルトのポスト Snapshot 実行フックは、該当するコンテナで実行されます。

> 本番環境で実行スクリプトを有効にする前に、必ず実行フックスクリプトをテストしてください。'kubectl exec' コマンドを使用すると、スクリプトを簡単にテストできます。本番環境で実行フックを有効にしたら、作成されたスナップショットの整合性をテストします。これを行うには、アプリケーションを一時ネームスペースにクローニングし、スナップショットをリストアしてから、アプリケーションをテストします。

既存の実行フックを表示します

既存のカスタム実行フックまたはネットアップが提供するアプリケーションのデフォルト実行フックを表示できます。

手順

1. 「 * アプリケーション」に移動し、管理アプリの名前を選択します。
2. [実行フック *] タブを選択します。

   有効または無効になっているすべての実行フックを結果リストに表示できます。フックのステータス ' ソース ' および実行時（スナップショット前またはスナップショット後）を表示できます実行フックに関連するイベントログを表示するには、左側のナビゲーション領域の * アクティビティ * ページに移動します。

カスタム実行フックを作成します

アプリケーションのカスタム実行フックを作成できます。を参照してください "実行フックの例" フックの例を参照してください。実行フックを作成するには、 Owner 、 Admin 、または Member のいずれかの権限が必要です。

> 実行フックとして使用するカスタムシェルスクリプトを作成する場合は、 Linux コマンドを実行しているか、実行可能ファイルへの完全パスを提供している場合を除き、ファイルの先頭に適切なシェルを指定するようにしてください。

手順

1. 「 * アプリケーション」を選択し、管理アプリの名前を選択します。
2. [実行フック *] タブを選択します。
3. [* 新しいフックを追加 *] を選択します。

4. フックの詳細 * 領域で、フックを実行するタイミングに応じて、「 * Pre-Snapshot * 」または「 * Post-Snapshot * 」を選択します。
5. フックの一意の名前を入力します。
6. （オプション）実行中にフックに渡す引数を入力し、各引数を入力した後で Enter キーを押して、それぞれを記録します。
7. [* Container Images * （コンテナイメージ * ）] 領域で、アプリケーションに含まれるすべてのコンテナイメージに対してフックを実行する必要がある場合は、 [* Apply to all container images * （すべてのコンテナイメージに適用 * ）] チェックボックスを有効にします。代わりに、フックが 1 つ以上の指定されたコンテナイメージに対してのみ機能する場合は、 * Container image names to match * フィールドにコンテナイメージ名を入力します。
8. [* スクリプト * （ * Script * ）] 領域で、次のいずれかを実行します。
   - カスタムスクリプトをアップロードする。
     i. [ファイルのアップロード（ Upload file ）] オプションを選択します。
     ii. ファイルを参照してアップロードします。
     iii. スクリプトに一意の名前を付けます。
     iv. （オプション）他の管理者がスクリプトについて知っておく必要があるメモを入力します。
   - クリップボードからカスタムスクリプトを貼り付けます。
     i. クリップボードから貼り付け * オプションを選択します。
     ii. テキストフィールドを選択し、スクリプトテキストをフィールドに貼り付けます。
     iii. スクリプトに一意の名前を付けます。
     iv. （オプション）他の管理者がスクリプトについて知っておく必要があるメモを入力します。
9. [* フックを追加 *] を選択します。

実行フックを無効にします

アプリケーションのスナップショットの前または後に実行を一時的に禁止する場合は、実行フックを無効にできます。実行フックを無効にするには、 Owner 、 Admin 、または Member のいずれかの権限が必要です。

手順

1. 「 * アプリケーション」を選択し、管理アプリの名前を選択します。
2. [実行フック *] タブを選択します。
3. 無効にするフックの * アクション * 列のオプションメニューを選択します。
4. [**Disable**] を選択します。

実行フックを削除します

不要になった実行フックは完全に削除できます。実行フックを削除するには、 Owner 、 Admin 、または Member のいずれかの権限が必要です。

手順

1. 「 * アプリケーション」を選択し、管理アプリの名前を選択します。
2. [実行フック *] タブを選択します。

3. 削除するフックの * アクション * 列のオプションメニューを選択します。
4. 「 * 削除」を選択します。

実行フックの例

次の例を使用して、実行フックの構造を確認してください。これらのフックは、テンプレートまたはテストスクリプトとして使用できます。

シンプルな成功例

次に、成功し、標準出力および標準エラーにメッセージを書き込む単純フックの例を示します。

```
#!/bin/sh

# success_sample.sh
#
# A simple noop success hook script for testing purposes.
#
# args: None
#



#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}



#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
```

```
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running success_sample.sh"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

シンプルな成功の例（ **bash** バージョン）

次に、 bash 用に書かれた標準出力と標準エラーにメッセージを書き込む単純なフックの例を示します。

```
#!/bin/bash

# success_sample.bash
#
# A simple noop success hook script for testing purposes.
#
# args: None


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
```

```
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}



#
# main
#

# log something to stdout
info "running success_sample.bash"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

単純な成功例（**zsh** バージョン）

これは、成功した単純なフックの例であり、標準出力と標準エラーに Z シェル用に記述されたメッセージを書き込みます。

```
#!/bin/zsh

# success_sample.zsh
#
# A simple noop success hook script for testing purposes.
#
# args: None
#



#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}
```

```
#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}

#
# main
#

# log something to stdout
info "running success_sample.zsh"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

引数を指定した成功の例

次の例は、フックで args を使用する方法を示しています。

```
#!/bin/sh

# success_sample_args.sh
#
# A simple success hook script with args for testing purposes.
#
# args: Up to two optional args that are echoed to stdout

#
# Writes the given message to standard output
#
```

```
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running success_sample_args.sh"


# collect args
arg1=$1
arg2=$2

# output args and arg count to stdout
info "number of args: $#"
info "arg1 ${arg1}"
info "arg2 ${arg2}"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

スナップショット前 / スナップショット後のフックの例

次の例は、 Snapshot 前フックと Snapshot 後フックの両方に同じスクリプトを使用する方法を示しています。

```
#!/bin/sh

# success_sample_pre_post.sh
#
# A simple success hook script example with an arg for testing purposes
# to demonstrate how the same script can be used for both a prehook and
posthook
#
# args: [pre|post]

# unique error codes for every error case
ebase=100
eusage=$((ebase+1))
ebadstage=$((ebase+2))
epre=$((ebase+3))
epost=$((ebase+4))


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
```

```
error() {
    msg "ERROR: $*" 1>&2
}


#
# Would run prehook steps here
#
prehook() {
    info "Running noop prehook"
    return 0
}

#
# Would run posthook steps here
#
posthook() {
    info "Running noop posthook"
    return 0
}


#
# main
#

# check arg
stage=$1
if [ -z "${stage}" ]; then
    echo "Usage: $0 <pre|post>"
    exit ${eusage}
fi

if [ "${stage}" != "pre" ] && [ "${stage}" != "post" ]; then
    echo "Invalid arg: ${stage}"
    exit ${ebadstage}
fi

# log something to stdout
info "running success_sample_pre_post.sh"

if [ "${stage}" = "pre" ]; then
    prehook
    rc=$?
    if [ ${rc} -ne 0 ]; then
        error "Error during prehook"
```

```
    fi
fi

if [ "${stage}" = "post" ]; then
    posthook
    rc=$?
    if [ ${rc} -ne 0 ]; then
        error "Error during posthook"
    fi
fi

exit ${rc}
```

失敗の例

次の例は、フックで障害を処理する方法を示しています。

```
#!/bin/sh

# failure_sample_arg_exit_code.sh
#
# A simple failure hook script for testing purposes.
#
# args: [the exit code to return]
#



#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}



#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}
```

```
#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running failure_sample_arg_exit_code.sh"

argexitcode=$1

# log to stderr
error "script failed, returning exit code ${argexitcode}"

# exit with specified exit code
exit ${argexitcode}
```

詳細なエラーの例

次の例では ' フックの失敗をより詳細なロギングで処理する方法を示します

```
#!/bin/sh

# failure_sample_verbose.sh
#
# A simple failure hook script with args for testing purposes.
#
# args: [The number of lines to output to stdout]


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}
```

```
#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running failure_sample_verbose.sh"


# output arg value to stdout
linecount=$1
info "line count ${linecount}"

# write out a line to stdout based on line count arg
i=1
while [ "$i" -le ${linecount} ]; do
    info "This is line ${i} from failure_sample_verbose.sh"
    i=$(( i + 1 ))
done

error "exiting with error code 8"
exit 8
```

終了コード例を使用した失敗

次の例は、終了コードを使用したフックの失敗を示しています。

```
#!/bin/sh

# failure_sample_arg_exit_code.sh
#
# A simple failure hook script for testing purposes.
#
# args: [the exit code to return]
#


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running failure_sample_arg_exit_code.sh"
```

```
argexitcode=$1

# log to stderr
error "script failed, returning exit code ${argexitcode}"

# exit with specified exit code
exit ${argexitcode}
```

失敗後の成功の例

次の例では、最初の実行時にフックが失敗していますが、 2 回目の実行後に成功しています。

```
#!/bin/sh

# failure_then_success_sample.sh
#
# A hook script that fails on initial run but succeeds on second run for
testing purposes.
#
# Helpful for testing retry logic for post hooks.
#
# args: None
#

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
```

```
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}



#
# main
#

# log something to stdout
info "running failure_success sample.sh"


if [ -e /tmp/hook-test.junk ] ; then
    info "File does exist.  Removing /tmp/hook-test.junk"
    rm /tmp/hook-test.junk
    info "Second run so returning exit code 0"
    exit 0
else
    info "File does not exist.  Creating /tmp/hook-test.junk"
    echo "test" > /tmp/hook-test.junk
    error "Failed first run, returning exit code 5"
    exit 5
fi
```

アプリケーションとクラスタの健常性を表示します

アプリケーションとクラスタの健常性の概要を表示します

ダッシュボード * を選択すると、アプリ、クラスター、ストレージバックエンド、それらのヘルスの概要が表示されます。

これらは静的な数値やステータスだけでなく、それぞれからドリルダウンすることもできます。たとえば、アプリが完全に保護されていない場合は、アイコンの上にカーソルを置くと、完全に保護されていないアプリを特定できます。その理由が含まれます。

アプリケーションタイル

「 * アプリケーション * 」タイルは、次の項目を識別するのに役立ちます。

- Astra で現在管理しているアプリケーションの数。
- それらの管理アプリが正常であるかどうか。
- アプリケーションが完全に保護されているかどうか（最新のバックアップがある場合は保護されます）。

- 検出されたものの、まだ管理されていないアプリケーションの数。

  アプリケーションが検出された後で管理または無視するため、この数はゼロになるのが理想的です。さらに、ダッシュボードで検出されたアプリケーションの数を監視して、開発者がクラスタに新しいアプリケーションを追加するタイミングを特定します。

クラスタタイル

クラスタタイルには、 Astra Control Center を使用して管理しているクラスタの健常性に関する同様の詳細が表示され、ドリルダウンしてアプリと同様に詳細を確認できます。

ストレージバックエンドはタイル張りです

「ストレージバックエンド * 」タイルは、ストレージバックエンドの健全性を特定するための情報を提供します。これには次のものが含まれます。

- 管理対象のストレージバックエンドの数
- これらの管理バックエンドが正常であるかどうか
- バックエンドが完全に保護されているかどうか
- 検出されたがまだ管理されていないバックエンドの数。

クラスタの健常性と詳細を表示します

Astra Control Center で管理するクラスタを追加すると、その場所、ワーカーノード、永続ボリューム、ストレージクラスなど、クラスタに関する詳細を表示できます。

手順

1. Astra Control Center UI で、 [* Clusters] を選択します。
2. [* Clusters] ページで、詳細を表示するクラスタを選択します。

> クラスタの構成 `removed` クラスタとネットワークの接続が正常であると表示される（Kubernetes APIを使用してクラスタに外部からアクセスしようとすると成功する）場合は、Astra Controlに指定したkubeconfigが無効になる可能性があります。クラスタでの証明書のローテーションまたは有効期限が原因の可能性があります。この問題を修正するには、を使用して、 Astra Control のクラスタに関連付けられたクレデンシャルを更新します "Astra Control API の略"。

3. [Overview （概要）] 、 [* Storage （ * ストレージ）] 、 [* Activity * （アクティビティ * ）] タブの情報を表示して、必要な情報を検索します。
   - * 概要 * ：状態を含むワーカーノードの詳細。
   - * ストレージ * ：ストレージクラスと状態を含む、コンピューティングに関連付けられた永続的ボリューム。
   - * アクティビティ * ：クラスタに関連するアクティビティを表示します。

Astra Control Center * Dashboard * から始まるクラスタ情報を表示することもできます。[* クラスタ *] タブの [* リソースサマリ *] で、管理対象クラスタを選択して [* クラスタ *] ページに移動できます。[* Clusters] ページが表示されたら、上記の手順を実行します。

アプリの状態と詳細を表示します

アプリケーションの管理を開始すると、アプリケーションのステータス（正常かどうか）、保護ステータス（障害発生時に完全に保護されているかどうか）、ポッド、永続的ストレージなどを識別できる詳細が Astra から提供されます。

手順

1. Astra Control Center UI で、 * アプリケーション * を選択し、アプリの名前を選択します。
2. お探しの情報を検索してください。

アプリステータス

Kubernetes でのアプリケーションの状態を反映するステータスを提供します。たとえば、ポッドと永続ボリュームはオンラインか？アプリケーションが正常な状態でない場合は、 Kubernetes のログでクラスタの問題を調べてトラブルシューティングする必要があります。Astra は、壊れたアプリケーションの修正に役立つ情報を提供していません。

アプリ保護ステータス

アプリが適切に保護されているかどうかのステータスを表示します。

- * 完全に保護されている * ：アプリにはアクティブなバックアップスケジュールがあり、 1 週間も経過していない正常なバックアップがあります
- * 部分的に保護 * ：アプリケーションには、アクティブなバックアップスケジュール、アクティブなスナップショットスケジュール、または正常なバックアップまたはスナップショットがあります
- * 保護されていない * ：完全に保護されていない、または部分的に保護されていないアプリ

  _ 最新のバックアップがあるまで、完全に保護することはできません _ 。これは、永続ボリュームから離れたオブジェクトストアにバックアップが格納されるために重要です。障害や事故によってクラスタと永続的ストレージが消去された場合は、バックアップをリカバリする必要があります。スナップショットを使用してリカバリすることはできません。

概要

アプリケーションに関連付けられているポッドの状態に関する情報。

データ保護

データ保護ポリシーを設定し、既存の Snapshot とバックアップを表示できます。

ストレージ

アプリケーションレベルの永続ボリュームが表示されます。永続ボリュームの状態は、 Kubernetes クラスタから見たものです。

リソース

バックアップおよび管理対象のリソースを確認できます。

アクティビティ

アプリケーションに関連するアクティビティを表示します。

> Astra Control Center * Dashboard * から始まるアプリ情報を表示することもできます。[* アプリケーション *] タブの [リソースの概要 *] で、管理アプリを選択して [* アプリケーション *] ページに移動できます。[**Applications**] ページが表示されたら、上記の手順に従います。

アカウントを管理します

ユーザを管理します

Astra Control UI を使用して、 Astra Control Center インストールのユーザーを招待、追加、削除、および編集できます。Astra Control UI またはを使用できます "Astra Control API" ユーザを管理するには、を実行

ユーザーを招待します

アカウント所有者と管理者は、 Astra Control Center に新しいユーザを招待できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [**Users**] タブを選択します。
3. [* ユーザーの招待 *] を選択します。
4. ユーザの名前と E メールアドレスを入力します。
5. 適切なシステム権限を持つユーザロールを選択します。

   各ロールには次の権限があります。

   - * Viewer * はリソースを表示できます。
   - メンバー * には、ビューア・ロールの権限があり、アプリとクラスタの管理、アプリの管理解除、スナップショットとバックアップの削除ができます。
   - **Admin** にはメンバーの役割権限があり、 Owner 以外の他のユーザーを追加および削除できます。
   - * Owner * には Admin ロールの権限があり、任意のユーザーアカウントを追加および削除できます。
6. メンバーロールまたはビューアロールを持つユーザーに制約を追加するには、 * 制約へのロールの制限 * チェックボックスをオンにします。

   制約の追加の詳細については、を参照してください "ロールの管理"。

7. [* ユーザーを招待する *] を選択します。

   ユーザーは、 Astra Control Center に招待されたことを通知する電子メールを受信します。このメールには一時的なパスワードが含まれています。このパスワードは初回ログイン時に変更する必要があります。

ユーザを追加します

アカウント所有者と管理者は、 Astra Control Center のインストールにさらにユーザーを追加できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [**Users**] タブを選択します。
3. [ユーザーの追加] を選択します。
4. ユーザ名、 E メールアドレス、および一時パスワードを入力します。

   ユーザは初回ログイン時にパスワードを変更する必要があります。

5. 適切なシステム権限を持つユーザロールを選択します。

   各ロールには次の権限があります。

   - * Viewer * はリソースを表示できます。
   - メンバー * には、ビューア・ロールの権限があり、アプリとクラスタの管理、アプリの管理解除、スナップショットとバックアップの削除ができます。
   - **Admin** にはメンバーの役割権限があり、 Owner 以外の他のユーザーを追加および削除できます。
   - * Owner * には Admin ロールの権限があり、任意のユーザーアカウントを追加および削除できます。

6. メンバーロールまたはビューアロールを持つユーザーに制約を追加するには、 * 制約へのロールの制限 * チェックボックスをオンにします。

   制約の追加の詳細については、を参照してください "ロールの管理"。

7. 「 * 追加」を選択します。

パスワードを管理します

Astra Control Center では、ユーザーアカウントのパスワードを管理できます。

パスワードを変更します

ユーザアカウントのパスワードはいつでも変更できます。

手順

1. 画面の右上にあるユーザアイコンを選択します。
2. * プロファイル * を選択します。
3. [* アクション * （ * Actions * ）] 列の [オプション（ Options ）] メニューから、 [* パスワードの変更 * （ * Change Password ）] を選択します
4. パスワードの要件に準拠するパスワードを入力します。
5. 確認のためパスワードをもう一度入力します。
6. 「 * パスワードの変更 * 」を選択します。

別のユーザのパスワードをリセットします

アカウントに Admin ロールまたは Owner ロールの権限がある場合は、自分だけでなく他のユーザアカウントのパスワードもリセットできます。パスワードをリセットする場合は、ログイン時にユーザが変更しなければならない一時パスワードを割り当てます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [* アクション * （ * Actions * ）] ドロップダウンリストを選択します。
3. 「 * パスワードのリセット * 」を選択します。
4. パスワードの要件に適合する一時パスワードを入力します。
5. 確認のためパスワードをもう一度入力します。

> 次回ユーザがログインするときに、パスワードの変更を求めるプロンプトが表示されます。

6. 「 * パスワードのリセット * 」を選択します。

ユーザのロールを変更します

Owner ロールのユーザはすべてのユーザのロールを変更できますが、 Admin ロールのユーザは Admin 、 Member 、 Viewer のロールを持つユーザのロールを変更できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [* アクション * （ * Actions * ）] ドロップダウンリストを選択します。
3. [役割の編集] を選択します。
4. 新しいロールを選択します。
5. ロールに制約を適用するには、 * 制約へのロールの制限 * チェックボックスを有効にして、リストから制約を選択します。

   拘束がない場合は、拘束を追加できます。詳細については、を参照してください "ロールの管理"。

6. [* 確認 *] を選択します。

結果

Astra Control Center は、選択した新しいロールに基づいてユーザーの権限を更新します。

ユーザを削除します

所有者ロールまたは管理者ロールを持つユーザは、いつでもそのアカウントから他のユーザを削除できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [* ユーザー *] タブで、削除する各ユーザーの行にあるチェックボックスをオンにします。
3. [* アクション * （ * Actions * ）] 列の [オプション（ Options ）] メニューから、 [* ユーザー / 秒を削除（ * Remove user/s * ）] を選択する
4. プロンプトが表示されたら、「 remove 」という単語を入力して削除を確認し、「 * Yes 、 Remove User * 」を選択します。

結果

Astra Control Center は、アカウントからユーザーを削除します。

ロールの管理

ロールを管理するには、ネームスペースの制約を追加し、ユーザロールをその制約に制限します。これにより、組織内のリソースへのアクセスを制御できます。Astra Control UI またはを使用できます "Astra Control API" をクリックしてください。

ロールに名前空間制約を追加します

Admin または Owner ユーザは、ネームスペースの制約を追加できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [**Users**] タブを選択します。
3. [* アクション * （ * Actions * ）] 列で、メンバーまたはビューアーの役割を持つユーザーのメニューボタンを選択します。
4. [役割の編集] を選択します。
5. [ロールを制約に制限する *] チェックボックスをオンにします。

   このチェックボックスは、メンバーロールまたはビューアロールでのみ使用できます。[*Role] ドロップダウン・リストから別のロールを選択できます

6. [* 制約の追加 *] を選択します。

   使用可能な制約の一覧は、ネームスペースまたはネームスペースラベルで確認できます。

7. [制約タイプ * （ Constraint type * ）] ドロップダウンリストで、ネームスペースの構成方法に応じて、 [* Kubernetes namespace] * または [* Kubernetes namespace label*] を選択します。
8. リストから 1 つ以上の名前空間またはラベルを選択して、それらの名前空間にロールを制限する制約を構成します。
9. [* 確認 *] を選択します。

   [役割の編集 *] ページには、この役割に選択した拘束のリストが表示されます。

10. [* 確認 *] を選択します。

    [**Account**] ページでは、 [*Role] 列のメンバまたはビューアの役割の制約を表示できます。

> 制約を追加せずに役割の制約を有効にし、 * 確認 * を選択すると、役割には完全な制限があると見なされます（役割は、名前空間に割り当てられているリソースへのアクセスを拒否されます）。

ロールから名前空間制約を削除します

管理者または所有者ユーザーは、役割から名前空間の制約を削除できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [**Users**] タブを選択します。
3. [* アクション * （ * Actions * ）] 列で、アクティブな拘束を持つメンバーまたはビューアーの役割を持つユーザーのメニューボタンを選択する。
4. [役割の編集] を選択します。
   - 役割の編集 * （ Edit role * ）ダイアログには、役割のアクティブな拘束が表示されます。
5. 削除する拘束の右側にある * X * を選択します。
6. [* 確認 *] を選択します。

を参照してください。

- "ユーザロールとネームスペース"

通知を表示および管理します

アクションが完了または失敗すると、 Astra から通知が表示されます。たとえば、アプリケーションのバックアップが正常に完了した場合に通知が表示されます。

これらの通知は、インターフェイスの右上から管理できます。

手順

1. 右上の未読通知の数を選択します。
2. 通知を確認し、 [* 既読としてマークする *] または [すべての通知を表示する *] を選択します。

   [すべての通知を表示する *] を選択した場合は、 [通知] ページがロードされます。

3. [* 通知 *] ページで、通知を表示し、既読としてマークする通知を選択し、 [* アクション *] を選択して、 [* 既読としてマークする *] を選択します。

クレデンシャルを追加および削除します

ONTAP S3 、 OpenShift で管理される Kubernetes クラスタ、未管理の Kubernetes クラスタなどのローカルプライベートクラウドプロバイダのクレデンシャルを、お客様のアカウントにいつでも追加、削除できます。Astra Control Center は、これらのクレデンシャルを使用して、クラスタ上の Kubernetes クラスタとアプリケーションを検出し、ユーザに代わってリソースをプロビジョニングします。

Astra Control Center のすべてのユーザーが同じ資格情報セットを共有することに注意してください。

クレデンシャルを追加する

クラスターの管理時に、 Astra Control Center に資格情報を追加できます。新しいクラスタを追加してクレデンシャルを追加する手順については、を参照してください "Kubernetes クラスタを追加"。

独自の「 kubeconfig 」ファイルを作成する場合は、その中で * 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" 「 kubeconfig 」ファイルの作成方法については、を参照してください。

クレデンシャルを削除する

アカウントからのクレデンシャルの削除はいつでも実行できます。クレデンシャルは、のあとに削除してください "関連するすべてのクラスタの管理を解除します"。

Astra Control Center は、 Astra Control Center の認証情報を使用してバックアップバケットに認証するため、 Astra Control Center に追加する最初の資格情報セットは常に使用されています。これらのクレデンシャルは削除しないことを推奨します。

手順

1. 「 * アカウント * 」を選択します。
2. [*Credentials] タブを選択します。
3. 削除するクレデンシャルの [状態 *] 列で [オプション] メニューを選択します。
4. 「 * 削除」を選択します。
5. 削除を確認するために「削除」と入力し、「はい」、「認証情報を削除」を選択します。

結果

Astra Control Center は、アカウントから資格情報を削除します。

アカウントのアクティビティを監視

Astra Control アカウントのアクティビティの詳細を表示できます。たとえば、新しいユーザを招待したとき、クラスタが追加されたとき、 Snapshot が作成されたときなどです。アカウントアクティビティを CSV ファイルにエクスポートすることもできます。

Astra Control のアカウントアクティビティをすべて表示

1. 「 * Activity * 」を選択します。
2. フィルタを使用してアクティビティのリストを絞り込むか、検索ボックスを使用して探しているものを正確に検索します。
3. アカウントアクティビティを CSV ファイルにダウンロードするには、「 * CSV にエクスポート」を選択します。

特定のアプリケーションのアカウントアクティビティを表示します

1. 「 * アプリケーション」を選択し、アプリケーションの名前を選択します。
2. 「 * Activity * 」を選択します。

クラスタのアカウントアクティビティを表示します

1. 「 * クラスタ」を選択し、クラスタの名前を選択します。
2. 「 * Activity * 」を選択します。

対応が必要なイベントを解決するための操作を実行します

1. 「 * Activity * 」を選択します。
2. 注意が必要なイベントを選択してください。
3. [**Take action**] ドロップダウンオプションを選択します。

   このリストから、実行できる対処方法のほか、問題 に関するドキュメントを参照したり、問題 の解決に役立つサポートを受けたりできます。

既存のライセンスを更新する

評価用ライセンスをフルライセンスに変換したり、既存の評価用ライセンスまたはフルライセンスを新しいライセンスで更新したりできます。フルライセンスがない場合は、ネットアップの営業担当者に連絡して、ライセンスとシリアル番号の全文を入手してください。Astra の UI またはを使用できます "Astra Control API" 既存のライセンスを更新します。

手順

1. にログインします "ネットアップサポートサイト"。
2. Astra Control Center のダウンロードページにアクセスし、シリアル番号を入力して、ネットアップライセンスファイル（ NLF ）をダウンロードする。
3. Astra Control Center UI にログインします。
4. 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。
5. [**Account**>*License*] ページで、既存のライセンスのステータスドロップダウンメニューを選択し、[**Replace**] を選択します。
6. ダウンロードしたライセンスファイルを参照します。
7. 「 * 追加」を選択します。

[**Account**>*Licenses*] 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。

を参照してください。

- "Astra Control Center のライセンス"

リポジトリ接続を管理します

ソフトウェアパッケージのインストールイメージやアーティファクトの参照として使用するリポジトリをAstra Controlに接続できます。ソフトウェアパッケージをインポートすると、Astra Controlは、イメージリポジトリ内のインストールイメージと、アーティファクトリポジトリ内のバイナリおよびその他のアーティファクトを参照します。

必要なもの

- Astra Control Center をインストールした Kubernetes クラスタ
- アクセス可能な稼働中のDockerリポジトリ
- アクセス可能なアーティファクトリポジトリ（Artifactoryなど）が実行されている必要があります

Dockerイメージリポジトリを接続する

Dockerイメージリポジトリを接続して、Astraデータストアなどのパッケージインストールイメージを保持できます。パッケージをインストールすると、Astra Controlはイメージリポジトリからパッケージイメージファイルをインポートします。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [接続（Connections *）]タブを選択します。
3. 「* Docker Image Repository *」セクションで、右上のメニューを選択します。
4. 「 * 接続」を選択します。
5. リポジトリのURLとポートを追加します。
6. リポジトリのクレデンシャルを入力します。
7. 「 * 接続」を選択します。

結果

リポジトリが接続されました。「* Docker Image Repository *」セクションに、リポジトリのステータスが「Connected」になっていることを確認します。

Dockerイメージリポジトリの接続を解除する

不要になったDockerイメージリポジトリへの接続を削除できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [接続（Connections *）]タブを選択します。
3. 「* Docker Image Repository *」セクションで、右上のメニューを選択します。
4. 「切断」を選択します。
5. 「* Yes、disconnect Docker image repository *」を選択します。

結果

リポジトリが切断されました。「* Docker Image Repository *」セクションには、リポジトリのステータスが「Disconnected」になっているはずです。

アーティファクトリポジトリを接続します

アーティファクトリポジトリをソフトウェアパッケージのバイナリなどのホストアーティファクトに接続できます。パッケージをインストールすると、Astra Controlによって、ソフトウェアパッケージのアーティファクトがイメージリポジトリからインポートされます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [接続（Connections *）]タブを選択します。
3. [アーティファクトリポジトリ*]セクションで’右上のメニューを選択します

4. 「 * 接続」を選択します。
5. リポジトリのURLとポートを追加します。
6. 認証が必要な場合は、*認証を使用*チェックボックスを有効にして、リポジトリのクレデンシャルを入力します。
7. 「 * 接続」を選択します。

結果

リポジトリが接続されました。[アーティファクトリポジトリ*]セクションで'リポジトリに[接続済み]ステータスが表示されるはずです

アーティファクトリポジトリの接続を解除します

不要になったアーティファクトリポジトリへの接続を削除できます

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [接続（Connections *）]タブを選択します。
3. [アーティファクトリポジトリ*]セクションで'右上のメニューを選択します
4. 「切断」を選択します。
5. [はい]を選択し'アーティファクト・リポジトリを切断します*

結果

リポジトリが切断されました。[アーティファクトリポジトリ*]セクションで'リポジトリに[接続済み]ステータスが表示されるはずです

詳細については、こちらをご覧ください

- "ソフトウェアパッケージを管理します"

ソフトウェアパッケージを管理します

ネットアップでは、ネットアップサポートサイトからダウンロード可能なソフトウェアパッケージを使用して、Astra Control Center向けの機能を追加しています。Dockerとアーティファクトのリポジトリを接続したら、パッケージをアップロードしてインポートし、この機能をAstra Control Centerに追加できます。CLIまたはAstra Control CenterのWeb UIを使用して、ソフトウェアパッケージを管理できます。

必要なもの

- Astra Control Center をインストールした Kubernetes クラスタ
- ソフトウェアパッケージイメージを格納するために接続されたDockerイメージリポジトリ。詳細については、を参照してください "リポジトリ接続を管理します"。
- ソフトウェアパッケージのバイナリやアーティファクトを保持するための、接続されたアーティファクトリポジトリ。詳細については、を参照してください "リポジトリ接続を管理します"。
- ネットアップサポートサイトから提供されるソフトウェアパッケージ

ソフトウェアパッケージのイメージをリポジトリにアップロードします

Astra Control Centerは、接続されたリポジトリ内のパッケージイメージとアーティファクトを参照します。CLIを使用して、リポジトリにイメージとアーティファクトをアップロードできます。

手順

1. ネットアップサポートサイトからソフトウェアパッケージをダウンロードし、「kubectl」ユーティリティがインストールされているマシンに保存します。
2. 圧縮されたパッケージファイルを展開し、ディレクトリをAstra Controlバンドルファイルの場所に変更します（例：「acc.manifest .bundle.yaml」）。
3. パッケージイメージをDockerリポジトリにプッシュします。次の置換を行います。
   - bundle_fileをAstra Controlバンドルファイルの名前に置き換えます。
   - my_registryをDockerリポジトリのURLに置き換えます。
   - my_registry_userとmy_registry_passwordをリポジトリの資格情報に置き換えます

   ```
   kubectl astra packages push-images -m BUNDLE_FILE -r MY_REGISTRY -u
   MY_REGISTRY_USER -p MY_REGISTRY_PASSWORD
   ```

4. パッケージにアーティファクトがある場合は’アーティファクトをアーティファクトリポジトリにコピーしますbundle_fileをAstra Controlバンドルファイルの名前に置き換え’network_locationをネットワークロケーションに置き換えて’アーティファクトファイルを次の場所にコピーします

   ```
   kubectl astra packages copy-artifacts -m BUNDLE_FILE -n NETWORK_LOCATION
   ```

ソフトウェアパッケージを追加します

Astra Control Centerバンドルファイルを使用して、ソフトウェアパッケージをインポートできます。これにより、パッケージがインストールされ、Astra Control Centerで使用できるようになります。

Astra Control Web UIを使用してソフトウェアパッケージを追加

Astra Control Center Web UIを使用して、接続されたリポジトリにアップロードされたソフトウェアパッケージを追加できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。
2. [パッケージ]タブを選択します。
3. [*追加（Add *）]ボタンを選択します。
4. ファイル選択ダイアログで、アップロードアイコンを選択します。
5. アップロードするAstra Controlバンドルファイルを「.yaml」形式で選択します。
6. 「 * 追加」を選択します。

結果

バンドルファイルが有効で、パッケージイメージとアーティファクトが接続されているリポジトリにある場合、パッケージはAstra Control Centerに追加されます。[ステータス*]列のステータスが[使用可能*]に変わったら、パッケージを使用できます。パッケージのステータスにカーソルを合わせると、詳細を確認できます。

> リポジトリ内にパッケージのイメージまたはアーティファクトが1つでも見つからない場合は’そのパッケージのエラーメッセージが表示されます

CLIを使用してソフトウェアパッケージを追加します

CLIを使用して、接続されたリポジトリにアップロードしたソフトウェアパッケージをインポートできます。そのためには、最初にAstra Control CenterのアカウントIDとAPIトークンを記録する必要があります。

手順

1. Webブラウザを使用して、Astra Control Center Web UIにログインします。
2. ダッシュボードの右上にあるユーザアイコンを選択します。
3. [API access*]を選択します。
4. 画面上部のアカウントIDをメモします。
5. [APIトークンの生成]を選択します。
6. 表示されたダイアログで、* APIトークンの生成*を選択します。
7. 生成されたトークンをメモし、*閉じる*を選択します。CLIで、展開されたパッケージの内容の中で、ディレクトリを「.yaml」バンドルファイルの場所に変更します。
8. バンドルファイルを使用してパッケージをインポートし、次のように置き換えます。
   - bundle_fileをAstra Controlバンドルファイルの名前に置き換えます。
   - serverをAstra ControlインスタンスのDNS名に置き換えます。
   - account_IDとtokenは、以前に記録したアカウントIDとAPIトークンに置き換えてください。

```
kubectl astra packages import -m BUNDLE_FILE -u SERVER -a ACCOUNT_ID
-k TOKEN
```

結果

バンドルファイルが有効で、パッケージイメージとアーティファクトが接続されているリポジトリにある場合、パッケージはAstra Control Centerに追加されます。

> リポジトリ内にパッケージのイメージまたはアーティファクトが1つでも見つからない場合は’そのパッケージのエラーメッセージが表示されます

ソフトウェアパッケージを削除します

Astra Control Center Web UIを使用して、Astra Control Centerに以前にインポートしたソフトウェアパッケージを削除できます。

手順

1. 「アカウントの管理」ナビゲーション領域で、「 * アカウント * 」を選択します。

2. [パッケージ]タブを選択します。

   このページには、インストールされているパッケージとそのステータスのリストが表示されます。

3. パッケージの*アクション*列で、アクションメニューを開きます。
4. 「 * 削除」を選択します。

結果

パッケージはAstra Control Centerから削除されますが、パッケージのイメージとアーティファクトはリポジトリに残ります。

詳細については、こちらをご覧ください

- "リポジトリ接続を管理します"

バケットを管理する

アプリケーションや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をクローニングする場合は、オブジェクトストアバケットプロバイダが不可欠です。Astra Control Center を使用して、オフクラスタのバックアップ先として、アプリケーションのオブジェクトストアプロバイダを追加します。

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、バケットは必要ありません。

次の Amazon Simple Storage Service （ S3 ）バケットプロバイダのいずれかを使用します。

- NetApp ONTAP S3
- NetApp StorageGRID S3 の略
- 汎用 S3
- 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。

バケットの状態は次のいずれかになります。

- Pending ：バケットの検出がスケジュールされています。
- Available ：バケットは使用可能です。
- Removed ：バケットには現在アクセスできません。

Astra Control API を使用してバケットを管理する方法については、を参照してください "Astra の自動化と API に関する情報"。

バケットの管理に関連して次のタスクを実行できます。

- "バケットを追加します"

- [バケットを編集する]
- [バケットのクレデンシャルをローテーションするか、削除する]
- [バケットを削除する]

> Astra Control Center の S3 バケットは、使用可能容量を報告しません。Astra Control Center で管理されているアプリケーションのバックアップまたはクローニングを行う前に、 ONTAP または StorageGRID 管理システムでバケット情報を確認します。

バケットを編集する

バケットのアクセスクレデンシャル情報を変更したり、選択したバケットがデフォルトバケットかどうかを変更したりできます。

> 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。を参照してください "リリースノート"。

手順

1. 左側のナビゲーションから、*バケット*を選択します。
2. [* アクション * （ * Actions * ）] 列の [オプション（ Options ）] メニューから、 [* 編集（ * Edit ）] を選択する。
3. バケットタイプ以外の情報を変更します。

   > バケットタイプは変更できません。

4. 「 * Update * 」を選択します。

バケットのクレデンシャルをローテーションするか、削除する

Astra Controlは、バケットのクレデンシャルを使用してS3バケットにアクセスし、シークレットキーを提供することで、Astra Control Centerがバケットと通信できるようにします。

バケットのクレデンシャルをローテーションする

クレデンシャルのローテーションを行う場合は、バックアップが進行中でないとき（スケジュール設定またはオンデマンド）に、ローテーションを継続して実行してください。

クレデンシャルの編集やローテーションを行う手順

1. 左側のナビゲーションから、*バケット*を選択します。
2. [* アクション * （ * Actions * ）] 列の [オプション（ Options ）] メニューから、 [* 編集（ * Edit ）] を選択する。
3. 新しいクレデンシャルを作成します。
4. 「 * Update * 」を選択します。

バケットのクレデンシャルを削除する

バケットのクレデンシャルを削除するのは、新しいクレデンシャルがバケットに適用されている場合やバケットがアクティブに使用されなくなった場合だけにしてください。

> Astra Control に追加する最初のクレデンシャルセットは、 Astra Control がバックアップバケットの認証にクレデンシャルを使用するため、常に使用されています。バケットがアクティブな状態で使用されている場合は、これらのクレデンシャルを削除しないでください。削除すると、バックアップが失敗してバックアップが使用できなくなります。

> アクティブなバケットクレデンシャルを削除する場合は、を参照してください "バケットのクレデンシャル削除のトラブルシューティング"。

Astra Control APIを使用してS3クレデンシャルを削除する方法については、を参照してください "Astra の自動化と API に関する情報"。

バケットを削除する

使用されなくなったバケットや正常でないバケットを削除することができます。これは、オブジェクトストアの設定をシンプルかつ最新の状態に保つために役立ちます。

> デフォルトバケットを削除することはできません。そのバケットを削除する場合は、最初に別のバケットをデフォルトとして選択します。

必要なもの

- 開始する前に、このバケットの実行中または完了済みのバックアップがないことを確認してください。
- アクティブな保護ポリシーでバケットが使用されていないことを確認する必要があります。

ある場合は、続行できません。

手順

1. 左ナビゲーションから、 * バケット * を選択します。
2. [アクション * （ Actions * ）] メニューから、 [* 削除（ Remove ）] を選択します。

> Astra Control を使用すると、最初にバケットを使用してバックアップを実行するスケジュールポリシーが存在せず、削除しようとしているバケットにアクティブなバックアップが存在しないようにすることができます。

3. 「 remove 」と入力して操作を確認します。
4. 「 * Yes 、 remove bucket * 」を選択します。

詳細については、こちらをご覧ください

- "Astra Control API を使用"

ストレージバックエンドを管理します

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。ストレージ容量と健全性の詳細を監視できます。 Astra Control Center が Cloud Insights に接続されている場合のパフォーマンスも監視できます。

Astra Control API を使用してストレージバックエンドを管理する方法については、を参照してください "Astra の自動化と API に関する情報"。

ストレージバックエンドの管理に関連して、次のタスクを実行できます。

- "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の詳細を表示します]
- [ストレージバックエンドの管理を解除します]
- [ストレージバックエンドライセンスを更新する]
- [ストレージバックエンドクラスタにノードを追加します]
- [ストレージバックエンドを削除します]

ストレージバックエンドの詳細を表示します

ストレージバックエンドの情報は、ダッシュボードまたはバックエンドオプションで確認できます。

Storage Backend DetailsページにあるAstraデータストアの場合は、次の情報が表示されます。

- Astraデータストアクラスタ
  - スループット、IOPS、およびレイテンシ
  - 使用済み容量と総容量の比較
- 各Astraデータストアクラスタボリューム
  - 使用済み容量と総容量の比較
  - スループット

ダッシュボードでストレージバックエンドの詳細を確認します

手順

1. 左側のナビゲーションから、 * ダッシュボード * を選択します。
2. 状態を示す Storage backend セクションを確認します。
   - * 正常でない * ：ストレージが最適な状態ではありません。これは、レイテンシの問題やコンテナの問題が原因でアプリケーションがデグレードした場合などに発生します。
   - * すべて正常 * ：ストレージは管理されており、最適な状態です。
   - * 検出 * ：ストレージは検出されましたが、 Astra Control では管理されていません。

バックエンドからストレージバックエンドの詳細を表示するオプションを選択します

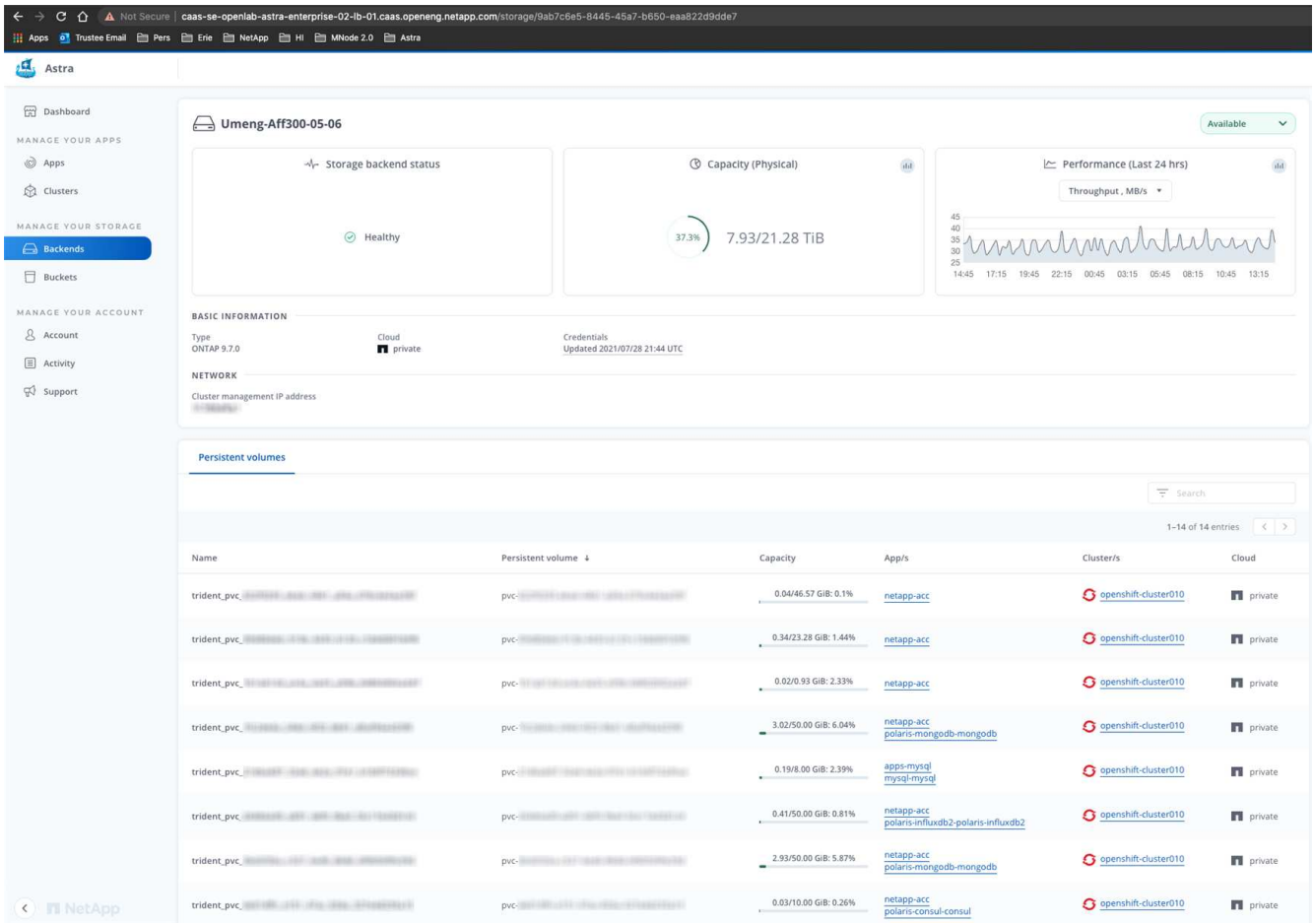
バックエンドの健常性、容量、パフォーマンス（ IOPS スループット、レイテンシ）に関する情報を表示します。

Cloud Insights に接続すると、 Kubernetes アプリケーションが使用しているボリュームが表示されます。このボリュームは、選択したストレージバックエンドに格納されます。

手順

1. 左側のナビゲーション領域で、 * Backends * を選択します。
2. ストレージバックエンドを選択します。

> NetApp Cloud Insights に接続した場合、 Cloud Insights からの抜粋がバックエンドのページに表示されます。



3. Cloud Insights に直接移動するには、指標画像の横にある * Cloud Insights * アイコンを選択します。

ストレージバックエンドの管理を解除します

バックエンドの管理を解除できます。

手順

1. 左のナビゲーションから、 * Backends * を選択します。

2. ストレージバックエンドを選択します。
3. * アクション * 列のオプションメニューから、 * 管理解除 * を選択します。
4. 「 unmanage 」と入力して操作を確定します。
5. 「 * Yes 、 unmanage storage backend * 」を選択します。

ストレージバックエンドを削除します

使用されなくなったストレージバックエンドを削除できます。これは、設定をシンプルかつ最新の状態に保つために役立ちます。

> Astra データストアバックエンドを削除する場合、 vCenter で作成されていないことが必要です。

必要なもの

- ストレージバックエンドが管理対象外であることを確認します。
- ストレージバックエンドに Astra データストアクラスタに関連付けられたボリュームがないことを確認します。

手順

1. 左ナビゲーションから、 * Backends * を選択します。
2. バックエンドが管理されている場合は、管理を解除します。
   a. [*Managed] を選択します。
   b. ストレージバックエンドを選択します。
   c. [* アクション * （ * Actions * ）] オプションから、 [* アンマネージ * （ * Unmanage * ）] を
   d. 「 unmanage 」と入力して操作を確定します。
   e. 「 * Yes 、 unmanage storage backend * 」を選択します。
3. [* Discovered （検出済み）] を選択
   a. ストレージバックエンドを選択します。
   b. [* アクション * （ * Actions * ）] オプションから、 [* 削除（ * Remove ）] を選択する。
   c. 「 remove 」と入力して操作を確認します。
   d. 「 * Yes 、 remove storage backend * 」を選択します。

ストレージバックエンドライセンスを更新する

より大規模な導入や拡張機能をサポートするために、 Astra データストアストレージバックエンドのライセンスを更新できます。

必要なもの

- 導入および管理された Astra データストアストレージバックエンド
- Astra データストアライセンスファイル（ネットアップの営業担当者に連絡して Astra データストアライセンスを購入）

手順

1. 左のナビゲーションから、 * Backends * を選択します。
2. ストレージバックエンドの名前を選択します。
3. [基本情報]では、インストールされているライセンスのタイプを確認できます。

   ライセンス情報にカーソルを合わせると、有効期限や使用権の情報などの詳細情報を示すポップアップが表示されます。

4. [* License] で、ライセンス名の横にある編集アイコンを選択します。
5. [ライセンスの更新*]ページで、次のいずれかを実行します。

ライセンスステータス	アクション
Astraデータストアに少なくとも1つのライセンスが追加されている。	リストからライセンスを選択します。
Astraデータストアにライセンスが追加されていない。	a. [*追加（Add *）]ボタンを選択します。 b. アップロードするライセンスファイルを選択してください。 c. 「*追加」を選択して、ライセンスファイルをアップロードします。

6. 「 * Update * 」を選択します。

ストレージバックエンドクラスタにノードを追加します

Astra Data Store クラスタにノードを追加できます。このノードは、 Astra Data Store 用にインストールされたライセンスのタイプでサポートされるノード数まで追加できます。

必要なもの

- 導入済みでライセンス供与されている Astra データストアストレージバックエンド
- Astra Data Store ソフトウェアパッケージを Astra Control Center に追加しておきます
- クラスタに追加する 1 つ以上の新しいノード

手順

1. 左のナビゲーションから、 * Backends * を選択します。
2. ストレージバックエンドの名前を選択します。
3. 基本情報では、このストレージバックエンドクラスタ内のノード数を確認できます。
4. [ノード数（ * Nodes ）] で、ノード数の横にある編集アイコンを選択します。
5. [ノードの追加 *] ページで、新しいノードに関する情報を入力します。
   a. 各ノードにノードラベルを割り当てます。
   b. 次のいずれかを実行します。
      - Astra データストアでライセンスに基づいて常に使用可能な最大数のノードを使用する場合は、「常に最大数のノードを使用する」チェックボックスを有効にします。

- Astra データストアで常に使用可能なノードの最大数を使用しない場合は、使用するノードの合計数を必要な数だけ選択します。

c. 保護ドメインを有効にした状態で Astra データストアを導入した場合は、新しいノードを保護ドメインに割り当てます。

6. 「 * 次へ * 」を選択します。
7. 新しい各ノードの IP アドレスとネットワーク情報を入力します。1 つの新しいノードに 1 つの IP アドレスを入力するか、複数の新しいノードに 1 つの IP アドレスプールを入力します。

   Astra データストアで導入時に設定した IP アドレスを使用できる場合は、 IP アドレス情報を入力する必要はありません。

8. 「 * 次へ * 」を選択します。
9. 新しいノードの設定を確認します。
10. [ノードの追加] を選択します。

詳細については、こちらをご覧ください

- "Astra Control API を使用"

インフラを監視、保護

複数のオプション設定を構成して、 Astra Control Center の操作性を高めることができます。Astra Control Center を実行しているネットワークで、インターネットに接続するためのプロキシが必要な場合（サポートバンドルをネットアップサポートサイトにアップロードする場合、または Cloud Insights への接続を確立する場合）は、 Astra Control Center でプロキシサーバを設定する必要があります。インフラ全体を監視して詳細を把握するには、 NetApp Cloud Insights への接続を確立します。Astra Control Center によって監視されているシステムから Kubernetes イベントを収集するには、 Fluentd 接続を追加します。

プロキシサーバを追加します

Astra Control Center を実行しているネットワークで、インターネットに接続するためのプロキシが必要な場合（サポートバンドルをネットアップサポートサイトにアップロードする場合、または Cloud Insights への接続を確立する場合）は、 Astra Control Center でプロキシサーバを設定する必要があります。

> Astra Control Center は、プロキシサーバー用に入力した詳細を検証しません。必ず正しい値を入力してください。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. ドロップダウンリストから [**Connect**] を選択して、プロキシサーバを追加します。

4. プロキシサーバの名前または IP アドレスとプロキシポート番号を入力します。
5. プロキシサーバで認証が必要な場合は、このチェックボックスをオンにしてユーザ名とパスワードを入力します。
6. 「 * 接続」を選択します。

結果

入力したプロキシ情報が保存されている場合は、 **Account**>*Connections* ページの **HTTP Proxy** セクションに、接続されていることが示され、サーバー名が表示されます。

プロキシサーバの設定を編集します

プロキシサーバの設定を編集できます。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. ドロップダウンリストから * Edit * を選択して、接続を編集します。
4. サーバの詳細と認証情報を編集します。
5. [保存（ Save ）] を選択します。

プロキシサーバ接続を無効にします

プロキシサーバ接続を無効にすることができます。他の接続が中断される可能性があることを無効にする前に警告が表示されます。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. 接続を無効にするには、ドロップダウンリストから * 切断 * を選択します。

4. 表示されたダイアログボックスで、処理を確認します。

Cloud Insights に接続します

NetApp Cloud Insights を Astra Control Center インスタンスに接続すると、インフラ全体を監視して詳細に把握できます。Cloud Insights は、 Astra Control Center ライセンスに含まれています。

Cloud Insights には、 Astra Control Center が使用するネットワークから、またはプロキシサーバー経由で間接的にアクセスできる必要があります。

Cloud Insights にアストラコントロールセンターを接続すると、 Acquisition Unit ポッドが作成されます。このポッドは、 Astra Control Center で管理されているストレージバックエンドからデータを収集し、 Cloud Insights にプッシュします。このポッドには、 8GB の RAM と 2 つの CPU コアが必要です。

> Cloud Insights 接続を有効にすると、スループット情報をバックエンド * ページで確認できるほか、ストレージバックエンドを選択したあとにここから Cloud Insights に接続できます。ダッシュボード * の情報はクラスタセクションでも確認できます。また、そこから Cloud Insights に接続できます。

必要なもの

- admin * / * owner * 権限を持つ Astra Control Center アカウント。
- 有効な Astra Control Center ライセンス。
- Astra Control Center を実行しているネットワークで、インターネットに接続するためにプロキシが必要な場合は、プロキシサーバーです。

> Cloud Insights を初めて使用する場合は、の機能について理解しておいてください。を参照してください "Cloud Insights のドキュメント"。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. 接続を追加するには、ドロップダウンリストで * 切断されている * と表示されている * 接続 * を選択します。

オプションを表示して、 Cloud Insights 接続を有効にします。"]

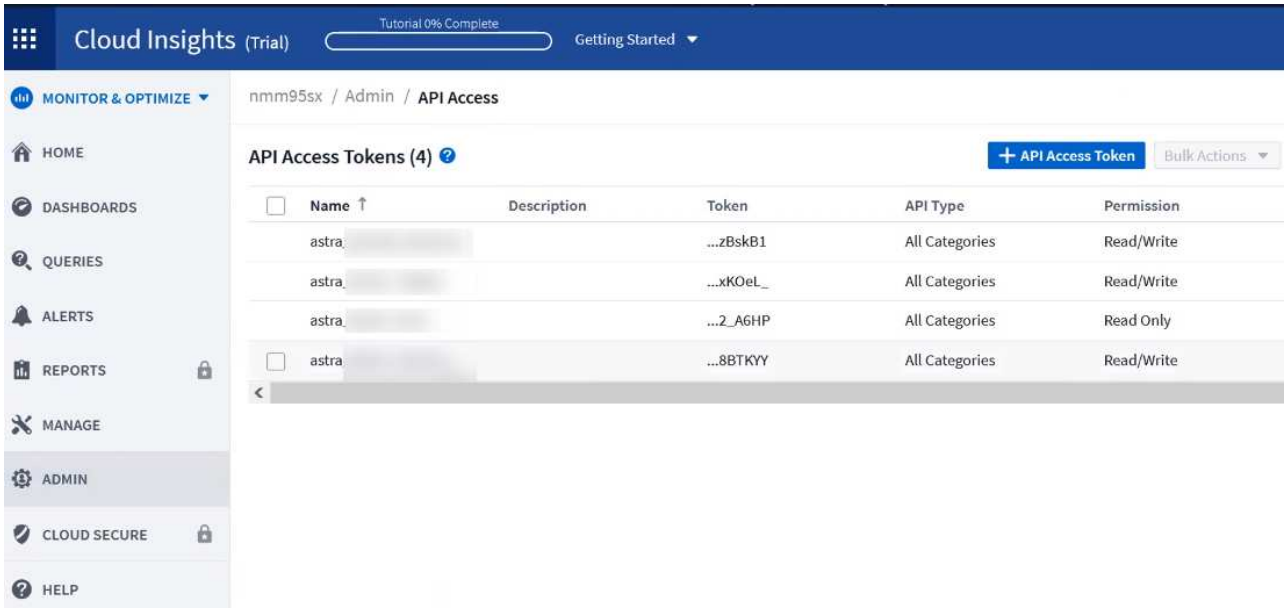
4. Cloud Insights API トークンとテナント URL を入力します。テナント URL の形式は次のようになります。

```
https://<environment-name>.c01.cloudinsights.netapp.com/
```

テナント URL は、 Cloud Insights ライセンスを取得すると取得されます。テナント URL がない場合は、を参照してください "Cloud Insights のドキュメント"。

a. をダウンロードしてください "API トークン"をクリックし、 Cloud Insights テナントの URL にログインします。

b. Cloud Insights で、 * Admin*>* API Access* をクリックして、 * Read/Write * と * Read Only* API Access トークンの両方を生成します。



c. 「 * Read Only * 」キーをコピーします。Cloud Insights 接続を有効にするには、 [Astra Control Center] ウィンドウに貼り付ける必要があります。Read API Access Token Key 権限で、 Assets 、 Alerts 、 Acquisition Unit 、 and Data Collection を選択します。

d. 「 * Read/Write 」キーをコピーします。Astra Control Center * Connect Cloud Insights * ウィンドウに貼り付ける必要があります。Read/Write API Access Token キー権限で、 Assets 、 Data Ingestion 、 Log Ingestion 、 Acquisition Unit を選択します。 および Data Collection 。

> * 読み取り専用 * キーと * 読み取り / 書き込み * キーを生成することを推奨します。両方の目的で同じキーを使用することは推奨しません。デフォルトでは、トークンの有効期限は 1 年に設定されています。トークンが期限切れになるまでの最大期間を指定するために、デフォルトの選択を維持することをお勧めします。トークンの有効期限が切れると、テレメトリが停止します。

e. Cloud Insights からコピーしたキーを Astra コントロールセンターに貼り付けます。

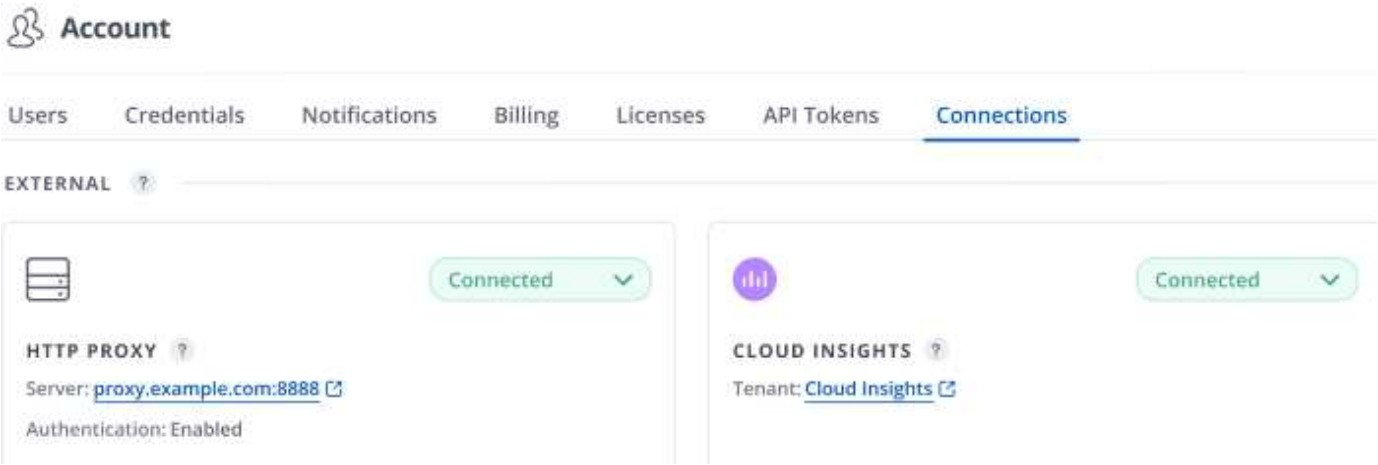
5. 「 * 接続」を選択します。

> [* 接続] を選択すると、 [* アカウント * > * 接続 *] ページの [* Cloud Insights *] セクションで、接続の状態が [* 保留中] に変わります。接続が有効になり、ステータスが * 接続済み * に変わるまで数分かかることがあります。

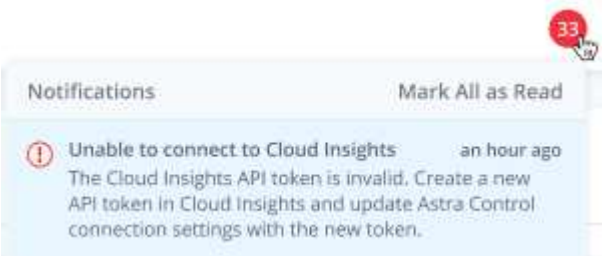
> Astra Control Center と Cloud Insights UI の間を簡単に行き来するには、両方にログインしていることを確認します。

Cloud Insights でデータを表示します

接続に成功した場合は、「 * アカウント * > * 接続 * 」ページの「 * Cloud Insights * 」セクションに接続されていることが示され、テナントの URL が表示されます。Cloud Insights にアクセスして、データが正常に受信されて表示されることを確認できます。

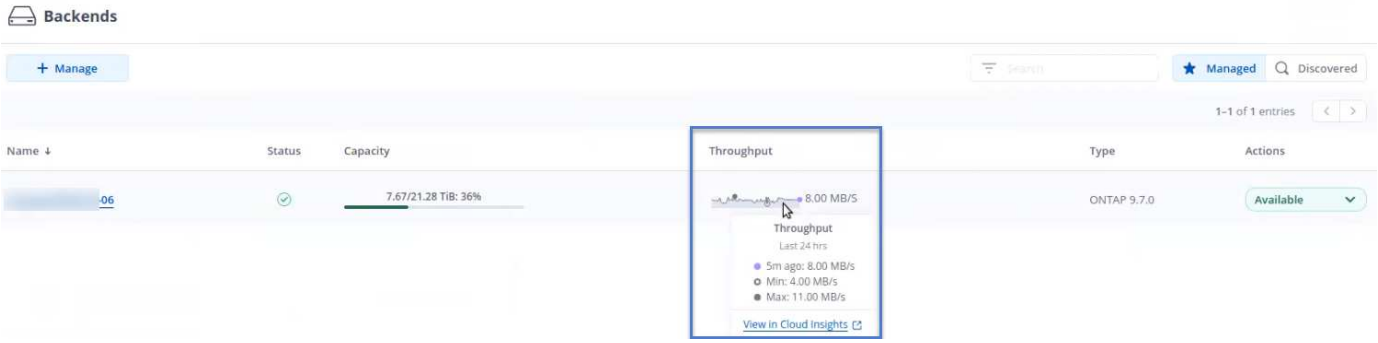


何らかの理由で接続に失敗した場合、ステータスは「 * 失敗 * 」と表示されます。失敗の理由は、 UI の右上にある * Notifications * で確認できます。



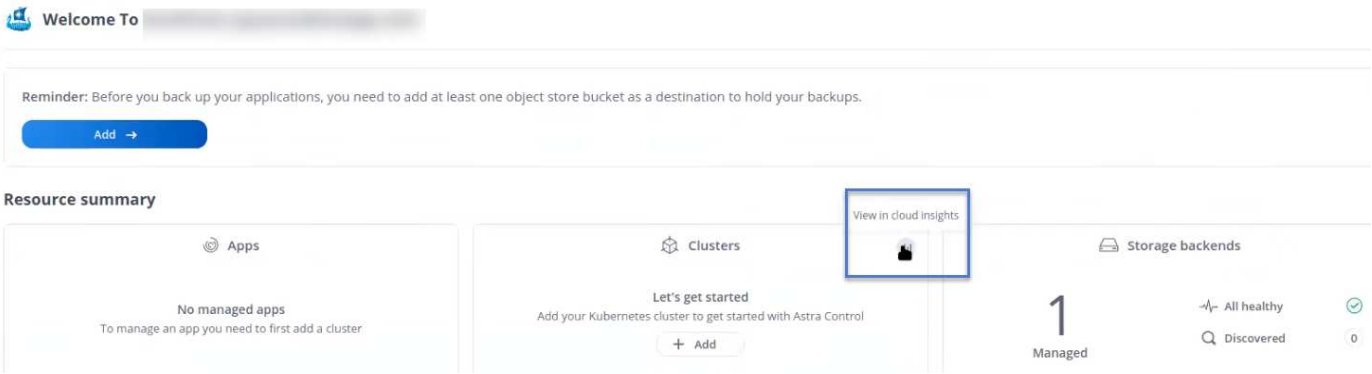
同じ情報は、「 * アカウント * > * 通知 * 」にも記載されています。

Astra Control Center では、スループット情報をバックエンド * ページで表示したり、ストレージバックエンドを選択した後にここから Cloud Insights に接続したりできます。



Cloud Insights に直接移動するには、指標画像の横にある * Cloud Insights * アイコンを選択します。

また、情報は * ダッシュボード * でも確認できます。



Cloud Insights 接続を有効にした後、 Astra Control Center に追加したバックエンドを削除すると、バックエンドは Cloud Insights へのレポートを停止します。

Cloud Insights 接続を編集します

Cloud Insights 接続を編集できます。

編集できるのは API キーのみです。Cloud Insights テナント URL を変更するには、 Cloud Insights 接続を切断して新しい URL に接続することを推奨します。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. ドロップダウンリストから * Edit * を選択して、接続を編集します。
4. Cloud Insights 接続設定を編集します。
5. [保存（ Save ）] を選択します。

Cloud Insights 接続を無効にします

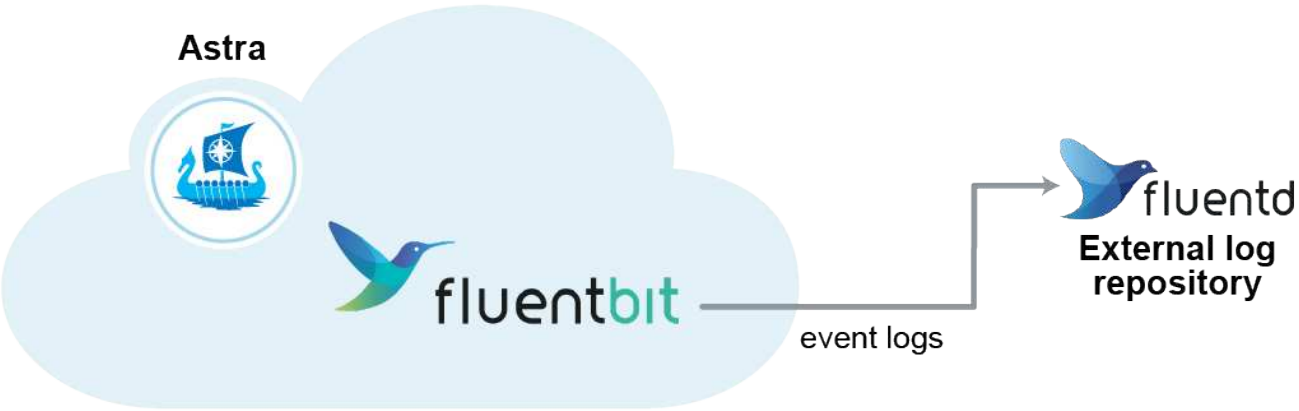
Cloud Insights 接続は、 Astra Control Center で管理されている Kubernetes クラスタに対して無効にすることができます。Cloud Insights 接続を無効にしても、すでに Cloud Insights にアップロードされている計測データは削除されません。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. 接続を無効にするには、ドロップダウンリストから * 切断 * を選択します。
4. 表示されたダイアログボックスで、処理を確認します。操作を確定すると、 [**Account**>*Connections*] ページで、 Cloud Insights のステータスが [*Pending （保留中）] に変わります。ステータスが * 切断された * に変わるまで数分かかります。

Fluentd に接続します

Astra Control Center から Fluentd エンドポイントにログ（ Kubernetes イベント）を送信できます。Fluentd 接続はデフォルトで無効になっています。



管理対象クラスタのイベントログのみが Fluentd に転送されます。

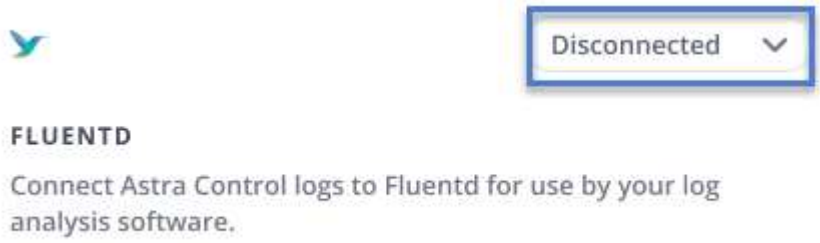
必要なもの

- admin * / * owner * 権限を持つ Astra Control Center アカウント。
- Kubernetes クラスタに Astra Control Center をインストールして実行

Astra Control Center では、 Fluentd サーバーに入力した詳細は検証されません。必ず正しい値を入力してください。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. 接続を追加するには、ドロップダウンリストから [* 接続（ * Connect * ）] を選択します。



4. Fluentd サーバーのホスト IP アドレス、ポート番号、および共有キーを入力します。
5. 「 * 接続」を選択します。

結果

Fluentd サーバーに入力した詳細が保存されている場合は、 * アカウント * > * 接続 * ページの * Fluentd * セクションに接続されていることが示されます。これで、接続した Fluentd サーバーにアクセスし、イベントログを表示できます。

何らかの理由で接続に失敗した場合、ステータスは「 * 失敗 * 」と表示されます。失敗の理由は、 UI の右上にある * Notifications * で確認できます。

同じ情報は、「 * アカウント * > * 通知 * 」にも記載されています。

> ログ収集に問題がある場合は ' ワーカー・ノードにログインし ' ログが /var/log/container/ で使用可能であることを確認してください

Fluentd 接続を編集します

Fluentd 接続を Astra Control Center インスタンスに編集できます。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. ドロップダウンリストから * Edit * を選択して、接続を編集します。
4. Fluentd エンドポイントの設定を変更します。
5. [保存（ Save ）] を選択します。

Fluentd 接続を無効にします

Astra Control Center インスタンスへの Fluentd 接続を無効にできます。

手順

1. * admin * / * owner * 権限を持つアカウントを使用して Astra Control Center にログインします。
2. [**Account**>*Connections*] を選択します。
3. 接続を無効にするには、ドロップダウンリストから * 切断 * を選択します。
4. 表示されたダイアログボックスで、処理を確認します。

アプリケーションとクラスタの管理を解除します

管理する必要がなくなったアプリケーションやクラスタを Astra Control Center から削除します。

アプリの管理を解除します

バックアップ、スナップショット、またはクローンを作成する必要がなくなったアプリケーションの管理を Astra Control Center から停止します。

- 既存のバックアップと 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。

手順

1. 左側のナビゲーションバーから、「 * アプリケーション * 」を選択します。
2. 管理する必要がなくなったアプリのチェックボックスをオンにします。
3. [アクション * （ Action * ）] メニューから、 [* アンマネージ * （ * Unmanage * ）] を選択し
4. 「 unmanage 」と入力して確定します。

5. アプリの管理を解除することを確認し、 [はい、アプリケーションの管理を解除します *] を選択します。

結果

Astra Control Center がアプリケーションの管理を停止。

クラスタの管理を解除します

管理する必要がなくなったクラスタの管理を Astra Control Center から解除します。

- この処理を実行すると、 Astra Control Center でクラスタが管理されなくなります。クラスタの構成は変更されず、クラスタも削除されません。
- Trident はクラスタからアンインストールされません。 "Trident のアンインストール方法をご確認ください"。

> クラスタの管理を解除する前に、クラスタに関連付けられているアプリケーションの管理を解除する必要があります。

手順

1. 左側のナビゲーションバーから、 * クラスタ * を選択します。
2. Astra Control Center で管理する必要がなくなったクラスタのチェックボックスを選択します。
3. * アクション * 列のオプションメニューから、 * 管理解除 * を選択します。
4. クラスタの管理を解除することを確認し、「 * Yes 、 unmanage cluster * 」を選択します。

結果

クラスタのステータスが「クラスタ」ページから「削除中」に変わり、クラスタが「クラスタ」ページから削除され、 Astra Control Center によって管理されなくなります。

> * Astra Control Center と Cloud Insights が接続されていない場合 * 、クラスタの管理を解除すると、テレメトリ・データの送信用にインストールされたすべてのリソースが削除されます。*Astra Control Center と Cloud Insights が接続されている場合 * ' クラスタの管理を解除すると 'fluentbit' および 'event-exporter ポッドのみが削除されます

Astra Control Center をアップグレードします

Astra Control Center をアップグレードするには、ネットアップサポートサイトからインストールバンドルをダウンロードし、以下の手順を実行して、環境内の Astra Control Center コンポーネントをアップグレードします。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の Astra コントロールセンターをアップグレードできます。

必要なもの

- "アップグレードを開始する前に、環境が Astra Control Center 導入の最小要件を満たしていることを確認します"。
- 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。

  OpenShift の例：

```
oc 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。

  OpenShift の例：

```
oc get apiservices
```

- Astra Control Center からログアウトします。

このタスクについて

Astra Control Center のアップグレードプロセスでは、次の手順を実行できます。

- Astra Control Center バンドルをダウンロードします
- [バンドルを開梱し、ディレクトリを変更します]
- [イメージをローカルレジストリに追加します]
- 更新された Astra Control Center オペレータをインストールします
- Astra Control Center をアップグレードします
- [サードパーティサービスのアップグレード（オプション）]
- [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]

すべての Astra Control Center ポッドが削除されないようにするため、アップグレード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_deployment.yaml

スケジュール、バックアップ、 Snapshot が実行されていないときは、メンテナンス時間内にアップグレードを実行します。

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。

Astra Control Center バンドルをダウンロードします

1. から Astra Control Center アップグレードバンドル (`Astra - control-ccenter-[version].tar.gz') をダウンロードします "ネットアップサポートサイト"。
2. （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 astra-control-center[version].pub
-signature <astra-control-center[version].sig astra-control-
center[version].tar.gz
```

バンドルを開梱し、ディレクトリを変更します

1. 画像を抽出します。

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

2. Astra ディレクトリに移動します。

```
cd astra-control-center-[version]
```

イメージをローカルレジストリに追加します

1. Astra Control Center イメージディレクトリ内のファイルをローカルレジストリに追加します。

   以下の画像の自動ロードについては、サンプルスクリプトを参照してください。

   a. Docker レジストリにログインします。

   ```
   docker login [your_registry_path]
   ```

   b. Docker にイメージをロードする。

   c. 画像にタグを付けます。

   d. [[[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1>[</Z1></Z1></Z1>_image_local_registry_push]]]]]]]]</Z2> ローカルレジストリにイメージをプッシュ

   ```
   export REGISTRY=[your_registry_path]
   for astraImageFile in $(ls images/*.tar)
     # Load to local cache. And store the name of the loaded image
   trimming the 'Loaded images: '
     do astraImage=$(docker load --input ${astraImageFile} | sed
   's/Loaded image: //')
     astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
     # Tag with local image repo.
     docker tag ${astraImage} ${REGISTRY}/${astraImage}
     # Push to the local repo.
     docker push ${REGISTRY}/${astraImage}
   done
   ```

更新された **Astra Control Center** オペレータをインストールします

1. Astra Control Center オペレータの配備 YAML (`Astra_control_center_deployment.yaml ') を編集して、ローカルのレジストリと秘密を参照します。

```
vim astra_control_center_operator_deploy.yaml
```

   a. 認証が必要なレジストリを使用する場合は、デフォルト行の「 imagePullSecret:[] 」を次のように置き換えます。

```
imagePullSecrets:
- name: <name_of_secret_with_creds_to_local_registry>
```

   b. 「 kube-rbac プロキシ」イメージの「 [Your_registry_path] 」を、でイメージをプッシュしたレジストリパスに変更します 前の手順。

   c. 「 acc-operator-controller-manager 」イメージの「 [Your_registry_path] 」を、でイメージをプッシュしたレジストリパスに変更します 前の手順。

   d. 「 env 」セクションに次の値を追加します。

```
- name: ACCOP_HELM_UPGRADETIMEOUT
  value: 300m
```

```
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"
        - name: ACCOP_HELM_UPGRADETIMEOUT
          value: 300m
        image: [your_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
      imagePullSecrets: []
```

2. 更新された 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 unchanged
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.astra.
netapp.io configured
role.rbac.authorization.k8s.io/acc-operator-leader-election-role
unchanged
clusterrole.rbac.authorization.k8s.io/acc-operator-manager-role
configured
clusterrole.rbac.authorization.k8s.io/acc-operator-metrics-reader
unchanged
clusterrole.rbac.authorization.k8s.io/acc-operator-proxy-role unchanged
rolebinding.rbac.authorization.k8s.io/acc-operator-leader-election-
rolebinding unchanged
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-manager-
rolebinding configured
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-proxy-
rolebinding unchanged
configmap/acc-operator-manager-config unchanged
service/acc-operator-controller-manager-metrics-service unchanged
deployment.apps/acc-operator-controller-manager configured
```

Astra Control Center をアップグレードします

1. Astra Control Center カスタムリソース（ CR ）（ 'Astra_control_center_min.yaml ）を編集し、 Astra バージョン（ 'Spec' の中の 'astrave) の番号を最新のものに変更します。

```
kubectl edit acc -n [netapp-acc or custom namespace]
```

> レジストリパスは、のイメージをプッシュしたレジストリパスと一致する必要があります前の手順。

2. Astra Control Center CR の 'Spec' の中にある 'additionalValues' 内に次の行を追加します

```
additionalValues:
    nautilus:
      startupProbe:
        periodSeconds: 30
        failureThreshold: 600
```

3. 次のいずれかを実行します。

   a. 独自の IngressController または入力がなく、トラフィックゲートウェイをロードバランサタイプサービスとして使用していて、そのセットアップを続行する場合は、別のフィールド「 ingressType 」を指定し（まだ存在しない場合）、それを「 AccTraefik 」に設定します。

   ```
   ingressType: AccTraefik
   ```

   b. デフォルトの Astra Control Center の一般的な入力配置に切り替える場合は、独自の IngressController/Ingress セットアップ（ TLS 終端など）を指定し、 Astra Control Center へのルートを開き、「 ingressType 」を「 Generic 」に設定します。

   ```
   ingressType: Generic
   ```

   フィールドを省略すると、プロセスは汎用的な配置になります。汎用的な導入が不要な場合は、必ずフィールドを追加してください。

4. （オプション）ポッドが終了し、再び使用可能になったことを確認します。

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

5. Astra のステータス状態がアップグレードが完了し、準備ができたことを示すまで待ちます。

```
kubectl get -o yaml -n [netapp-acc or custom namespace]
astracontrolcenters.astra.netapp.io astra
```

対応：

```
conditions:
  - lastTransitionTime: "2021-10-25T18:49:26Z"
    message: Astra is deployed
    reason: Complete
    status: "True"
    type: Ready
  - lastTransitionTime: "2021-10-25T18:49:26Z"
    message: Upgrading succeeded.
    reason: Complete
    status: "False"
    type: Upgrading
```

6. ログインし直して、すべての管理対象クラスタとアプリケーションが引き続き存在し、保護されていることを確認します。

7. オペレータが Cert-manager を更新しなかった場合は、次の手順でサードパーティのサービスをアップグレードします。

サードパーティサービスのアップグレード（オプション）

以前のアップグレード手順では、サードパーティサービス Traefik および Cert-manager はアップグレードされません。オプションで、ここで説明する手順を使用してアップグレードしたり、システムに必要な既存のサービスバージョンを保持したりできます。

- * Traefik* ：デフォルトでは、 Astra Control Center が Traefik 導入のライフサイクルを管理します。「 externalTraefik 」を「 false 」（デフォルト）に設定すると、外部 Traefik がシステムに存在せず、 Astra Control Center によってインストールおよび管理されていることを示します。この場合、「 externalTraefik 」は「 false 」に設定されます。

  一方、 Traefik を独自に導入している場合は、「 externalTraefik 」を「 true 」に設定します。この場合 ' 配置を維持して 'Astra Control Center は 'shouldUpgrade' が true' に設定されていない限り 'CRD をアップグレードしません

- **Cert-managor**: デフォルトでは 'externalCertManager` を TRUE に設定しない限り 'Astra Control Center は cert-manager (および CRD) をインストールします’shouldUpgrade' を 'true' に設定すると 'Astra Control Center が CRD をアップグレードします

次のいずれかの条件に該当する場合は、 Traefik がアップグレードされます。

- externalTraefik ： false または
- externalTraefik: true と shouldUpgrade: true 。

手順

1. 「 acc`cr: 」を編集します。

```
kubectl edit acc -n [netapp-acc or custom namespace]
```

2. 「 externalTraefik 」フィールドと「 shouldUpgrade 」フィールドを必要に応じて「 true 」または「 false 」に変更します。

```
crds:
    externalTraefik: false
    externalCertManager: false
    shouldUpgrade: false
```

システムステータスを確認します

1. Astra Control Center にログインします。
2. すべての管理対象クラスタとアプリケーションが引き続き存在し、保護されていることを確認します。

ロードバランシング用の入力を設定します

Kubernetes 入力オブジェクトを設定して、クラスタ内で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。

- デフォルトアップグレードでは、一般的な入力配置が使用されます。この場合は、入力コントローラまたは入力リソースも設定する必要があります。
- 入力コントローラが不要で、すでに持っているものを保持したい場合は、「 ingressType 」を「 AccTraefik 」に設定します。

> 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。

- nginx 入力コントローラ
- OpenShift 入力コントローラ

必要なもの

- CR 仕様で、
  - 「 CRD .externalTraefik 」が存在する場合は、「 false 」またはに設定する必要があります
  - 「 CRD.externalTraefik 」が「真」の場合、「 CRD.shoulldUpgrade 」も「真」でなければなりません。
- 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 v1.21 の間で Kubernetes のバージョンを使用している。

Nginx Ingress Controller の手順

1. 既存のシークレット「 secure-testing-cert 」を使用するか、タイプのシークレットを作成します "8a637503539b25b68130b6e8003579d9" に示すように 'NetApp-acc' （またはカスタム名前の）名前空間内の TLS 秘密鍵と証明書の場合 "TLS シークレット"。
2. 非推奨または新しいスキーマのいずれかの入力リソースを NetApp-acc` （またはカスタム名前付き）ネームスペースに配置します。

   a. 廃止されたスキーマについては、次の例を参照してください。

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

b. 新しいスキーマについては、次の例を参照してください。

```
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
```

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
```

入力セットアップを確認します

入力セットアップを確認してから、続行できます。

1. Loadbalancer から Traefik が clusterIP に変更されていることを確認します

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

2. Traefik でルートを確認します。

```
Kubectl get ingressroute ingressroutetls -n [netapp-acc or custom
namespace]
-o yaml | grep "Host("
```

結果は空である必要があります。

Astra Control Center をアンインストールします

試用版からフルバージョンの製品にアップグレードする場合は、 Astra Control Center コンポーネントの削除が必要になることがあります。Astra Control Center と Astra Control Center Operator を削除するには、この手順で説明されているコマンドを順に実行します。

アンインストールに問題がある場合は、を参照してください [アンインストールに関する問題のトラブルシューティング]。

必要なもの

- Astra Control Center UI を使用して、すべての管理を解除します "クラスタ"。

手順

1. Astra Control Center を削除します。次のコマンド例は、デフォルトのインストールに基づいています。カスタム構成を作成した場合は、コマンドを変更します。

```
kubectl delete -f astra_control_center_min.yaml -n netapp-acc
```

結果

```
astracontrolcenter.astra.netapp.io "astra" deleted
```

2. 次のコマンドを使用して 'NetApp-acc' ネームスペースを削除します

```
kubectl delete ns netapp-acc
```

結果

```
namespace "netapp-acc" deleted
```

3. Astra Control Center オペレータシステムコンポーネントを削除するには、次のコマンドを使用します。

```
kubectl delete -f astra_control_center_operator_deploy.yaml
```

結果

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

アンインストールに関する問題のトラブルシューティング

Astra Control Center のアンインストールで発生した問題に対処するには、次の回避策を実行します。

Astra Control Center をアンインストールしても、管理対象クラスタで監視オペレータポッドがクリーンアップされない

Astra Control Center をアンインストールする前にクラスタの管理を解除していない場合は、次のコマンドを使用して、ネットアップ監視ネームスペースとネームスペース内のポッドを手動で削除できます。

手順

1. 「 acc-monitoring 」エージェントを削除します。

```
kubectl delete agents acc-monitoring -n netapp-monitoring
```

結果

```
agent.monitoring.netapp.com "acc-monitoring" deleted
```

2. ネームスペースを削除します。

```
kubectl delete ns netapp-monitoring
```

結果

```
namespace "netapp-monitoring" deleted
```

3. リソースの削除を確認します。

```
kubectl get pods -n netapp-monitoring
```

結果

```
No resources found in netapp-monitoring namespace.
```

4. 監視エージェントが削除されたことを確認：

```
kubectl get crd|grep agent
```

サンプル結果：

```
agents.monitoring.netapp.com                    2021-07-21T06:08:13Z
```

5. カスタムリソース定義（ CRD ）情報の削除：

```
kubectl delete crds agents.monitoring.netapp.com
```

結果

```
customresourcedefinition.apiextensions.k8s.io
"agents.monitoring.netapp.com" deleted
```

Astra Control Center をアンインストールしても、 **Traefik CRD** をクリーンアップできない

Traefik CRD を手動で削除できます。CRD はグローバルリソースであり、削除するとクラスタ上の他のアプリケーションに影響を与える可能性があります。

手順

1. クラスタにインストールされている Traefik CRD を表示します。

```
kubectl get crds |grep -E 'traefik'
```

応答

```
ingressroutes.traefik.containo.us             2021-06-23T23:29:11Z
ingressroutetcps.traefik.containo.us          2021-06-23T23:29:11Z
ingressrouteudps.traefik.containo.us          2021-06-23T23:29:12Z
middlewares.traefik.containo.us               2021-06-23T23:29:12Z
middlewaretcps.traefik.containo.us            2021-06-23T23:29:12Z
serverstransports.traefik.containo.us         2021-06-23T23:29:13Z
tlsoptions.traefik.containo.us                2021-06-23T23:29:13Z
tlsstores.traefik.containo.us                 2021-06-23T23:29:14Z
traefikservices.traefik.containo.us           2021-06-23T23:29:15Z
```

2. CRD を削除します。

```
kubectl delete crd ingressroutes.traefik.containo.us
ingressroutetcps.traefik.containo.us
ingressrouteudps.traefik.containo.us middlewares.traefik.containo.us
serverstransports.traefik.containo.us tlsoptions.traefik.containo.us
tlsstores.traefik.containo.us traefikservices.traefik.containo.us
middlewaretcps.traefik.containo.us
```

詳細については、こちらをご覧ください

- "アンインストールに関する既知の問題"

REST API で自動化

Astra Control REST API による自動化

Astra Control には REST API が搭載されており、 Curl などのプログラミング言語やユーティリティを使用して Astra Control 機能に直接アクセスできます。また、 Ansible などの自動化テクノロジを使用して Astra Control 環境を管理することもできます。

Kubernetes アプリケーションをセットアップおよび管理するには、 Astra UI または Astra Control API を使用できます。

詳細については、を参照してください "Astra 自動ドキュメント"。

アプリケーションの導入

Helm チャートから Jenkins をデプロイします

で Jenkins を導入する方法について説明します "Bitnami Helm チャート"。クラスタに Jenkins を導入したら、アプリケーションを Astra Control に登録できます。

Jenkins は Astra Control 用の検証済みアプリケーションです。

- "Astra Controlの検証済みアプリケーションと標準アプリケーションの違いをご確認ください"。

以下の手順は、 Astra Control Service と Astra Control Center の両方に適用されます。

> Google Marketplace から導入されたアプリケーションは検証されていません。一部のユーザから、 Postgres 、 MariaDB 、 MySQL を導入した Google Marketplace での検出やバックアップで問題が報告されることがあります。

要件

- Astra Control に追加されたクラスタ。

> Astra Control Center では、最初にクラスタを Astra Control Center に追加するか、最初にアプリケーションを追加できます。

- クラスタの適切な kubeconfig を使用してローカルマシンにインストールされている Helm （バージョン 3.2+ ）および Kubectl のバージョンを更新しました

Astra Control は、現在のところをサポートしていません "Jenkins 用の Kubernetes プラグイン"。プラグインを使用せずに Kubernetes クラスタで Jenkins を実行できます。プラグインは Jenkins クラスタに拡張性を提供します。

Jenkins をインストールします

このプロセスに関する 2 つの重要な注意事項：

- アプリケーションは、クラスタを Astra Control Service に追加したあとに、以前には導入しないでください。Astra Control Center は、 Astra Control Center にクラスタを追加する前後のアプリケーションを受け入れます。
- Helm チャートは、デフォルト以外の名前空間に展開する必要があります。

手順

1. Bitnami チャート repo を追加します。

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

2. 「 Jenkins 」ネームスペースを作成し、次のコマンドを使用して Jenkins を展開します。

```
helm install <name> bitnami/jenkins --namespace <namespace> --create
-namespace
--set global.storageClass=<storage_class_name>
```

> ボリュームサイズを変更する場合は、 Kibibyte （ KI ）、 Meibyte （ MI ）、または Gibyte （ Gi ）単位を使用します。

ストレージクラスを定義する必要があるのは、次の場合のみです。

- Astra Control サービスを使用していて、デフォルトのストレージクラスを使用しない場合。
- Astra Control Center を使用していて、まだ Astra Control Center にクラスタをインポートしていない。あるいは、クラスタをインポートしたあとに、デフォルトのストレージクラスを使用する必要がなくなった場合。

結果

これにより、次の処理が行われます。

- ネームスペースを作成します。
- 正しいストレージクラスを設定します。

ポッドがオンラインになったら、 Astra Control を使用してアプリケーションを管理できます。Astra Control を使用すると、アプリケーションをネームスペースレベルで管理したり、 Helm ラベルを使用して管理したりできます。

Helm チャートから MariaDB を導入します

から MariaDB を導入する方法について説明します "Bitnami Helm チャート"。クラスタに MariaDB を導入すると、 Astra Control を使用してアプリケーションを管理できます。

MariaDB は、 Astra 向けの検証済みアプリケーションです。

- "Astra Controlの検証済みアプリケーションと標準アプリケーションの違いをご確認ください"。

以下の手順は、 Astra Control Service と Astra Control Center の両方に適用されます。

> Google Marketplace から導入されたアプリケーションは検証されていません。一部のユーザから、 Postgres 、 MariaDB 、 MySQL を導入した Google Marketplace での検出やバックアップで問題が報告されることがあります。

要件

- Astra Control に追加されたクラスタ。

> Astra Control Center では、最初にクラスタを Astra Control Center に追加するか、最初にアプリケーションを追加できます。

- クラスタの適切な kubeconfig を使用してローカルマシンにインストールされている Helm （バージョン 3.2+ ）および Kubectl のバージョンを更新しました

MariaDB をインストールします

このプロセスに関する 2 つの重要な注意事項：

- アプリケーションは、クラスタを Astra Control Service に追加したあとに、以前には導入しないでください。Astra Control Center は、 Astra Control Center にクラスタを追加する前後のアプリケーションを受け入れます。
- Helm チャートは、デフォルト以外の名前空間に展開する必要があります。

手順

1. Bitnami チャート repo を追加します。

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

2. コマンドを使用して、 MariaDB を導入します。

```
helm install <name> bitnami/MariaDB --namespace <namespace> --create
-namespace
--set global.storageClass=<storage_class_name>
```

> ボリュームサイズを変更する場合は、 Kibibyte （ KI ）、 Meibyte （ MI ）、または Gibyte （ Gi ）単位を使用します。

ストレージクラスを定義する必要があるのは、次の場合のみです。

- Astra Control サービスを使用していて、デフォルトのストレージクラスを使用しない場合。
- Astra Control Center を使用していて、まだ Astra Control Center にクラスタをインポートしていない。あるいは、クラスタをインポートしたあとに、デフォルトのストレージクラスを使用する必要がなくなった場合。

結果

これにより、次の処理が行われます。

- ネームスペースを作成します。
- ネームスペースに MariaDB を導入します。
- データベースを作成します。

> 導入時にパスワードを設定するこの方法は安全ではありません。本番環境にはお勧めしません。

ポッドがオンラインになったら、 Astra Control を使用してアプリケーションを管理できます。Astra Control を使用すると、アプリケーションをネームスペースレベルで管理したり、 Helm ラベルを使用して管理したり

できます。

Helm チャートから MySQL を導入します

から MySQL を導入する方法について説明します "Bitnami Helm チャート"。Kubernetes クラスタに MySQL を導入したら、 Astra Control を使用してアプリケーションを管理できます。

MySQL は Astra Control の検証済みアプリケーションです。

- "Astra Controlの検証済みアプリケーションと標準アプリケーションの違いをご確認ください"。

以下の手順は、 Astra Control Service と Astra Control Center の両方に適用されます。

> Google Marketplace から導入されたアプリケーションは検証されていません。一部のユーザから、 Postgres 、 MariaDB 、 MySQL を導入した Google Marketplace での検出やバックアップで問題が報告されることがあります。

要件

- Astra Control に追加されたクラスタ。

  > Astra Control Center では、最初にクラスタを Astra Control Center に追加するか、最初にアプリケーションを追加できます。

- クラスタの適切な kubeconfig を使用してローカルマシンにインストールされている Helm （バージョン 3.2+ ）および Kubectl のバージョンを更新しました

MySQL をインストール

このプロセスに関する 2 つの重要な注意事項：

- アプリケーションは、クラスタを Astra Control Service に追加したあとに、以前には導入しないでください。Astra Control Center は、 Astra Control Center にクラスタを追加する前後のアプリケーションを受け入れます。
- Helm チャートをデフォルト以外の名前空間に展開することをお勧めします。

手順

1. Bitnami チャート repo を追加します。

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

2. 次のコマンドを使用して MySQL を導入します。

```
helm install <name> bitnami/mysql --namespace <namespace> --create
-namespace
--set global.storageClass=<storage_class_name>
```

> ボリュームサイズを変更する場合は、 Kibibyte （ KI ）、 Meibyte （ MI ）、または Gibyte （ Gi ）単位を使用します。

ストレージクラスを定義する必要があるのは、次の場合のみです。

- Astra Control サービスを使用していて、デフォルトのストレージクラスを使用しない場合。
- Astra Control Center を使用していて、まだ Astra Control Center にクラスタをインポートしていない。あるいは、クラスタをインポートしたあとに、デフォルトのストレージクラスを使用する必要がなくなった場合。

結果

これにより、次の処理が行われます。

- ネームスペースを作成します。
- ネームスペースに MySQL を導入します。

ポッドがオンラインになったら、 Astra Control を使用してアプリケーションを管理できます。Astra Control を使用すると、名前を付けてアプリケーションを管理したり、ネームスペースレベルでアプリケーションを管理したり、 Helm ラベルを使用したりできます。

Helm チャートから Postgres を導入します

で Postgres を導入する方法について説明します "Bitnami Helm チャート"。クラスタに Postgres を導入したら、 Astra Control にアプリケーションを登録できます。

Postgres は Astra 向けの検証済みアプリケーションです。

- "Astra Controlの検証済みアプリケーションと標準アプリケーションの違いをご確認ください"。

以下の手順は、 Astra Control Service と Astra Control Center の両方に適用されます。

> Google Marketplace から導入されたアプリケーションは検証されていません。一部のユーザから、 Postgres 、 MariaDB 、 MySQL を導入した Google Marketplace での検出やバックアップで問題が報告されることがあります。

要件

- Astra Control に追加されたクラスタ。

> Astra Control Center では、最初にクラスタを Astra Control Center に追加するか、最初にアプリケーションを追加できます。

- クラスタの適切な kubeconfig を使用してローカルマシンにインストールされている Helm （バージョン 3.2+ ）および Kubectl のバージョンを更新しました

Postgres をインストールします

このプロセスに関する 2 つの重要な注意事項：

- アプリケーションは、クラスタを Astra Control Service に追加したあとに、以前には導入しないでください。Astra Control Center は、 Astra Control Center にクラスタを追加する前後のアプリケーションを受け入れます。
- Helm チャートは、デフォルト以外の名前空間に展開する必要があります。

手順

1. Bitnami チャート repo を追加します。

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

2. 次のコマンドを使用して Postgres を導入します。

```
helm install <name> bitnami/postgresql --namespace <namespace> --create
-namespace
--set global.storageClass=<storage_class_name>
```

> ボリュームサイズを変更する場合は、 Kibibyte （ KI ）、 Meibyte （ MI ）、または Gibyte （ Gi ）単位を使用します。

ストレージクラスを定義する必要があるのは、次の場合のみです。

- Astra Control サービスを使用していて、デフォルトのストレージクラスを使用しない場合。
- Astra Control Center を使用していて、まだ Astra Control Center にクラスタをインポートしていない。あるいは、クラスタをインポートしたあとに、デフォルトのストレージクラスを使用する必要がなくなった場合。

結果

これにより、次の処理が行われます。

- ネームスペースを作成します。
- 名前空間に Postgres を導入します。

ポッドがオンラインになったら、 Astra Control を使用してアプリケーションを管理できます。Astra Control を使用すると、アプリケーションをネームスペースレベルで管理したり、 Helm ラベルを使用して管理したりできます。

知識とサポート

トラブルシューティング

発生する可能性のある一般的な問題を回避する方法について説明します。

https://kb.netapp.com/Advice_and_Troubleshooting/Cloud_Services/Astra

詳細については、こちらをご覧ください

- "ネットアップにファイルをアップロードする方法（ログインが必要）"
- "ネットアップにファイルを手動でアップロードする方法（ログインが必要）"

ヘルプを表示します

ネットアップでは、 Astra Control をさまざまな方法でサポートしています。ナレッジベース（ KB ）記事や Slack チャンネルなど、幅広いセルフサポートオプションを 24 時間 365 日ご利用いただけます。Astra Control アカウントには、 Web チケット発行によるリモートテクニカルサポートが含まれています。

> Astra Control Center の評価用ライセンスをお持ちの場合は、テクニカルサポートを受けることができます。ただし、ネットアップサポートサイト（ NSS ）でケースを作成することはできません。フィードバックオプションを通じてサポートに連絡するか、 Slack チャンネルを利用してセルフサービスで連絡できます。

最初に実行する必要があります "ネットアップのシリアル番号のサポートを有効にします" これらの非セルフサービスサポートオプションを使用するには、次の手順に従います。チャットや Web でのチケット発行、ケース管理には、ネットアップサポートサイト（ NSS ）の SSO アカウントが必要です。

セルフサポートオプション

メインメニューから * Support * タブを選択すると、 Astra Control Center UI からサポートオプションにアクセスできます。

次のオプションは、 24 時間 365 日無料で利用できます。

- "* ナレッジベース * （ログインが必要）"： Astra Control に関する記事、 FAQ 、またはトラブルシューティング情報を検索します。
- * ドキュメントセンター * ：現在表示しているドキュメントサイトです。
- "* Slack * でヘルプを入手してください"： thePub ワークスペースのコンテナチャネルに移動して、ピアやエキスパートと交流しましょう。
- * サポートケースの作成 * ：トラブルシューティングのためにネットアップサポートに提供するサポートバンドルを生成
- * Astra Control についてのフィードバック * ： astra.feedback@netapp.com に電子メールを送信して、あなたの考え、アイデア、懸念を知らせてください。

ネットアップサポートへのサポートバンドルの日次アップロードを有効にします

Astra Control Center のインストール時に、 Astra Control Center Custom Resource Definition （ CRD ）ファイル（「 Astra_control_center_min.yaml 」）で「 AutoSupport 」に「 enroled:true 」を指定すると、毎日のサポートバンドルがに自動的にアップロードされます "ネットアップサポートサイト"。

ネットアップサポートに提供するサポートバンドルを生成する

Astra Control Center を使用すると、管理者ユーザはバンドルを生成できます。バンドルには、ネットアップサポートに役立つログ、 Astra 環境のすべてのコンポーネントに対するイベント、管理対象のクラスタとアプリケーションに関する指標、トポロジ情報などが含まれます。インターネットに接続している場合は、ネットアップサポートサイト（ NSS ）に Astra Control Center の UI から直接サポートバンドルをアップロードすることができます。

> Astra Control Center がバンドルを生成するのにかかる時間は、 Astra Control Center のインストールのサイズと、要求されたサポートバンドルのパラメータによって異なります。サポートバンドルの生成要求時に指定した期間によって、バンドルの生成にかかる時間が決まります（たとえば、期間を短くするとバンドルの生成にかかる時間が短縮されます）。

作業を開始する前に

NSS へのバンドルのアップロードにプロキシ接続が必要かどうかを確認します。プロキシ接続が必要な場合は、 Astra Control Center がプロキシサーバーを使用するように設定されていることを確認します。

1. [* アカウント * > * 接続 *] を選択します。
2. * 接続設定 * でプロキシ設定を確認します。

手順

1. NSS ポータルで、 Astra Control Center UI の * Support * ページに記載されているライセンスシリアル番号を使用してケースを作成します。
2. Astra Control Center UI を使用して、サポートバンドルを生成するには、次の手順を実行します。
   a. [サポート * （ Support * ）] ページの [サポートバンドル（ Support bundle ）] タイルで、 [* 生成（ Generate ）] を選択します。
   b. [* サポートバンドルの生成 * （ Generate a Support Bundle * ）] ウィンドウで、期間を選択します。

      クイックタイムフレームまたはカスタムタイムフレームのいずれかを選択できます。

      > カスタムの日付範囲を選択したり、期間にカスタムの期間を指定したりできます。

   c. 選択したら、 * 確認 * を選択します。
   d. [生成時にネットアップサポートサイトにバンドルをアップロードする *] チェックボックスをオンにします。
   e. [* バンドルの生成 *] を選択します。

      サポートバンドルの準備が完了すると、アラート領域の * アカウント * > * 通知 * ページ、 * アクティビティ * ページ、および通知リストに通知が表示されます（ UI の右上にあるアイコンを選択してアクセスできます）。

生成が失敗した場合は、バンドルの生成ページにアイコンが表示されます。アイコンを選択すると、メッセージが表示されます。

> UI の右上にある通知アイコンには、バンドルの作成に成功したタイミング、バンドルの作成に失敗したタイミング、バンドルをアップロードできなかったタイミング、バンドルをダウンロードできなかったタイミングなど、サポートバンドルに関連するイベントに関する情報が表示されます。

エアギャップ設置がある場合

エアギャップのある設置の場合は、サポートバンドルが生成されたあとに次の手順を実行します。バンドルがダウンロード可能になると、 [サポート *] ページの [サポートバンドル *] セクションの [生成 *] の横に [ダウンロード] アイコンが表示されます。

手順

1. ダウンロードアイコンを選択して、バンドルをローカルにダウンロードします。
2. 手動で NSS にバンドルをアップロードします。

   これを行うには、次のいずれかの方法を使用します。

   - 使用 "NetApp Authenticated File Upload （ログインが必要）"。
   - NSS でケースにバンドルを直接添付します。
   - NetApp Active IQ を使用する。

詳細については、こちらをご覧ください

- "ネットアップにファイルをアップロードする方法（ログインが必要）"
- "ネットアップにファイルを手動でアップロードする方法（ログインが必要）"

以前のバージョンの Astra Control Center ドキュメント

以前のリリースのドキュメントを参照できます。

- "Astra Control Center 21.12 ドキュメント"
- "Astra Control Center 21.08 ドキュメント"

法的通知

著作権に関する声明、商標、特許などにアクセスできます。

著作権

"https://www.netapp.com/company/legal/copyright/"

商標

NetApp 、 NetApp のロゴ、および NetApp の商標ページに記載されているマークは、 NetApp, Inc. の商標です。その他の会社名および製品名は、それぞれの所有者の商標である場合があります。

"https://www.netapp.com/company/legal/trademarks/"

特許

ネットアップが所有する特許の最新リストは、次のサイトで入手できます。

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

プライバシーポリシー

"https://www.netapp.com/company/legal/privacy-policy/"

オープンソース

通知ファイルには、ネットアップソフトウェアで使用されるサードパーティの著作権およびライセンスに関する情報が記載されています。

- "Astra Control Center へのお知らせ"
- "Astraデータストアに関する注意事項"

Astra Control API ライセンス

https://docs.netapp.com/us-en/astra-automation-2204/media/astra-api-license.pdf