Astra Control Center 24.02ドキュメント

Astra Control Center

NetApp
August 11, 2025

目次

Astra Control Center 24.02ドキュメント

リリースノート

最新リリースのAstra Control Centerを発表しました。

- "このリリースの Astra Control Center の内容"
- "既知の問題"
- "既知の制限"

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

最新リリースのAstra Control Centerを発表しました。

2024年3月15日（24.02.0）

新機能とサポート

- プライベートレジストリを使用せずに**Astra Control Center**を導入：Astra Control Centerのイメージをプライベートレジストリにプッシュしたり、Astra Control環境の一部として使用したりする必要がなくなりました。
- マイナーバグ修正

既知の問題および制限事項

- "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"

（テクニカルプレビュー）宣言型**Kubernetes**ワークフロー

このAstra Control Centerリリースには、ネイティブのKubernetesカスタムリソース（CR）からデータ管理を実行できるKubernetesの宣言機能が含まれています。

のインストール後 "Astra Connector" 管理するクラスタでは、UIまたはCRから次のCRベースのクラスタ操作を実行できます。

- "カスタムリソースを使用したアプリケーションの定義"
- "バケットを定義する"
- "クラスタ全体の保護"
- "アプリケーションのバックアップ"
- "Snapshot を作成します"
- "スナップショットまたはバックアップのスケジュールを作成する"
- "Snapshotまたはバックアップからアプリケーションをリストアする"

2023年11月7日（23.10.0）

新機能とサポート

- * ONTAP NAS経済性に優れたドライバベースのストレージバックエンドを使用したアプリケーションのバックアップとリストア機能*：バックアップとリストアの処理を `ontap-nas-economy` いくつかの "シンプルなステップ"。
- 変更不可のバックアップ：Astra Controlが新たにサポート "変更不可の読み取り専用バックアップ" マルウェアやその他の脅威に対する追加のセキュリティレイヤとして。
- * Astra Control Provisionerのご紹介*

  23.10リリースでは、Astra ControlにAstra Control Provisionerという新しいソフトウェアコンポーネントが導入されています。このソフトウェアコンポーネントは、Astra Controlのライセンスを取得したすべてのユーザが使用できます。Astra Control Provisionerでは、Astra Tridentよりも高度な管理機能とストレージプロビジョニング機能を利用できます。これらの機能は、Astra Controlをご利用のすべてのお客様が追加コストなしで利用できます。

  - * Astra Control Provisionerの利用を開始*
    可能です "Astra Control Provisionerを有効にする" Astra Trident 23.10を使用するように環境をインストールして設定している場合。
  - * Astra Control Provisionerの機能*

    Astra Control Provisioner 23.10リリースでは、次の機能を使用できます。

    - * Kerberos 5暗号化によるストレージバックエンドセキュリティの強化*：ストレージセキュリティを "暗号化の有効化" 管理対象クラスタとストレージバックエンド間のトラフィック用。Astra Control Provisionerは、Red Hat OpenShiftクラスタからAzure NetApp FilesおよびオンプレミスのONTAPボリュームへのNFSv4.1接続でKerberos 5暗号化をサポートします。
    - * Snapshotを使用したデータのリカバリ*：Astra Control Provisionerを使用すると、Snapshotからインプレースで迅速にボリュームをリストアできます。 `TridentActionSnapshotRestore` （TASR）CR。
    - * SnapMirrorの機能拡張*：Astra ControlからONTAPクラスタに直接接続できない環境やONTAPクレデンシャルにアクセスできない環境では、アプリケーションレプリケーション機能を使用できます。この機能を使用すると、Astra Controlでストレージバックエンドやそのクレデンシャルを管理することなく、レプリケーションを使用できます。
    - アプリケーションのバックアップおよびリストア機能 **`ontap-nas-economy`** ドライバ・バックアップ・ストレージ・バックエンド：説明どおり 上。
- * NVMe/TCPストレージを使用するアプリケーションの管理のサポート*
  NVMe / TCPを使用して接続された永続ボリュームでバックアップされたアプリケーションをAstra Controlで管理できるようになりました。
- 実行フックはデフォルトでオフになっています:このリリース以降、実行フック機能は次のようになります。 "有効" または、セキュリティを強化するために無効にします（デフォルトでは無効になっています）。Astra Controlで使用する実行フックをまだ作成していない場合は、 "実行フック機能を有効にする" フックの作成を開始します。このリリースより前に実行フックを作成した場合、実行フック機能は有効なままになり、通常どおりフックを使用できます。

既知の問題および制限事項

- "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"

2023年7月31日（23.07.0）

新機能とサポート

- "ストレッチ構成でのNetApp MetroClusterのストレージバックエンドとしての使用のサポート"
- "Longhornをストレージバックエンドとして使用するためのサポート"
- "同じKubernetesクラスタのONTAPバックエンド間でアプリケーションをレプリケートできるようになりました。"
- "Astra Control Centerで、リモート（LDAP）ユーザのログイン属性として「userPrincipalName」がサポートされるようになりました。"
- "Astra Control Centerを使用してレプリケーションのフェイルオーバー後に新しい実行フックタイプ「post-failover」を実行可能"
- クローンワークフローでは、ライブクローンのみ（管理対象アプリケーションの現在の状態）がサポートされるようになりました。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には、 "リストアのワークフロー"。

既知の問題および制限事項

- "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"

2023年5月18日（23.04.2）

Astra Control Center（23.04.0）向けのこのパッチリリース（23.04.2）では、がサポートされます "Kubernetes CSI外部Snapshotコピーv6.1.0" およびは、次の項目を修正します。

- 実行フックを使用する場合のインプレースアプリケーションリストアのバグ
- バケットサービスとの接続に問題があります

2023年4月25日（23.04.0）

新機能とサポート

- "Astra Control Centerの新規インストールでは、90日間の評価用ライセンスがデフォルトで有効になります"
- "強化された実行フック機能と追加のフィルタオプション"
- "Astra Control Centerでレプリケーションのフェイルオーバー後に実行フックを実行できるようになりました"
- "「ontap-nas-economy storage」クラスから「ontap-nas」ストレージクラスへのボリュームの移行がサポートされます"
- "リストア処理中のアプリケーションリソースの追加または除外がサポートされます"
- "データ専用アプリケーションの管理がサポートされます"

既知の問題および制限事項

- "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"

2022年11月22日（22.11.0）

新機能とサポート

- "複数のネームスペースにまたがるアプリケーションのサポート"
- "アプリケーション定義にクラスタリソースを含めることができます"
- "ロールベースアクセス制御（RBAC）を統合してLDAP認証を強化"
- "Kubernetes 1.25およびポッドセキュリティアドミッション（PSA）のサポートを追加"
- "バックアップ、リストア、クローニングの各処理の進捗状況レポートが強化されました"

既知の問題および制限事項

- "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"

2022年9月8日（22.08.1）

このパッチリリース（22.08.1）for Astra Control Center（22.08.0）では、NetApp SnapMirrorを使用したアプリケーションレプリケーションの小さなバグが修正されています。

2022年8月10日（22.08.0）

新機能とサポート

- "NetApp SnapMirrorテクノロジを使用したアプリケーションのレプリケーション"
- "アプリ管理ワークフローの改善"
- "拡張された独自の実行フック機能"

> ネットアップが提供している、特定のアプリケーションのデフォルトのPre-snapshot実行フックとPost-Snapshot実行フックは、このリリースでは削除されています。このリリースにアップグレードし、スナップショットの実行フックを独自に提供しない場合、Astra Controlはクラッシュコンシステントスナップショットのみを作成します。にアクセスします "ネットアップのVerda" GitHubリポジトリ：サンプルの実行フックスクリプトを使用します。環境に合わせて変更できます。

- "VMware Tanzu Kubernetes Grid Integrated Edition（TKGI）のサポート"
- "Google Anthosに対応しています"
- "LDAP設定（Astra Control API経由）"

既知の問題および制限事項

- "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"

2022 年 4 月 26 日（ 22.04.0 ）

新機能とサポート

- "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"
- "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 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"
- "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"
- "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。

- クラスタの管理後にボリュームnapshotclassを追加すると、アプリケーションのバックアップとSnapshotが失敗します
- kubeconfigファイルに複数のコンテキストが含まれている場合にAstra Control Centerでクラスタの管理が失敗する
- 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
- Kerberos転送中暗号化を使用する場合にバックアップからのリストアが失敗することがある
- [保持ポリシーの期限が切れたバケットでは、削除後もバックアップデータがバケットに残る]

クラスタの管理後にボリューム**napshotclass**を追加すると、アプリケーションのバックアップと**Snapshot**が失敗します

でバックアップとSnapshotの作成が失敗する `UI 500 error` このシナリオでは、回避策 として、アプリリストを更新します。

kubeconfigファイルに複数のコンテキストが含まれている場合に**Astra Control Center**でクラスタの管理が失敗する

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。

Kerberos転送中暗号化を使用する場合にバックアップからのリストアが失敗することがある

Kerberos転送中暗号化を使用しているストレージバックエンドにバックアップからアプリケーションをリストアすると、リストア処理が失敗することがあります。この問題は、Snapshotからのリストアや、NetApp SnapMirrorを使用したアプリケーションデータのレプリケートには影響しません。

NFSv4ボリュームでKerberos転送中暗号化を使用する場合は、NFSv4ボリュームで正しい設定が使用されていることを確認してください。『NetApp NFSv4ドメイン設定』のセクション（13ページ）を参照してください。"『NetApp NFSv4 Enhancements and Best Practices Guide』"。

保持ポリシーの期限が切れたバケットでは、削除後もバックアップデータがバケットに残る

バケットの保持ポリシーの期限が切れたあとにアプリケーションの変更不可のバックアップを削除すると、バックアップはバケットではなくAstra Controlから削除されます。この問題は今後のリリースで修正される予定です。

詳細については、こちらをご覧ください

- "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]
- [コネクタ以外のクラスタのリソースでは、制限的なロール制約は無視できます。]

アプリケーション管理の制限

- [1つのネームスペース内の複数のアプリケーションをまとめて別のネームスペースにリストアすることはできません]
- Astra Controlでは、ネームスペースごとに複数のストレージクラスを使用するアプリケーションはサポートされていません
- Astra Controlでは、クラウドインスタンスにデフォルトのバケットは自動的に割り当てられません
- [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

- 特定のバージョンのSnapshotコントローラを含むKubernetes 1.25以降のクラスタでは、Snapshotが失敗することがあります
- 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
- ONTAP NASエコノミーストレージクラスへのIn Placeリストア処理が失敗する

一般的な制限事項

- LDAPユーザおよびグループの制限事項
- 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
- <<[Activity]ページには、最大10万件のイベントが表示されます>>
- SnapMirrorはストレージバックエンドにNVMe over TCPを使用するアプリケーションをサポートしない

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. を編集します `kubeadm-config` 構成マップ：

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

2. を変更します `clusterName` フィールド値の開始値 `kubernetes` （Kubernetesのデフォルト名）を一意のカスタム名に変更します。
3. kubeconfigを編集します (`.kube/config`）。
4. からクラスタ名を更新します `kubernetes` を使用して一意のカスタム名を指定します (`xyz-cluster` は、以下の例で使用されています）。両方で更新を行います `clusters` および `contexts` 次の例に示すように、セクションを示します。

```
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

任意 `member` ネームスペース名/ IDによるRBACの制約があるユーザは、同じクラスタまたは組織のアカウントにある他のクラスタの新しいネームスペースにアプリケーションをクローニング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。クローン処理またはリストア処理で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 `member` 影響を受けるユーザーが新しい名前空間へのアクセスを許可するためのユーザーアカウントの制約を更新します。

コネクタ以外のクラスタのリソースでは、制限的なロール制約は無視できます。

- アクセス対象のリソースが最新の**Astra Connector**がインストールされているクラスタに属している場合：LDAPグループメンバーシップを通じてユーザに複数のロールが割り当てられると、ロールの制約が結合されます。たとえば、ローカルのビューアロールを持つユーザーが、メンバーロールにバインドされている3つのグループに参加した場合、ユーザーは元のリソースへのビューアロールアクセス権と、グループメンバーシップによって取得されたリソースへのメンバーロールアクセス権を持つようになります。
- アクセス対象のリソースが**Astra Connector**がインストールされていないクラスタに属している場合：ユーザにLDAPグループメンバーシップを通じて複数のロールが割り当てられている場合、最も権限の厳しいロールの制約のみが有効になります。

1つのネームスペース内の複数のアプリケーションをまとめて別のネームスペースにリストアすることはできません

複数のアプリケーションを1つのネームスペースで管理する場合（Astra Controlで複数のアプリケーション定義を作成する）、すべてのアプリケーションを別の1つのネームスペースにリストアすることはできません。

各アプリケーションを専用のネームスペースにリストアする必要があります。

Astra Controlでは、ネームスペースごとに複数のストレージクラスを使用するアプリケーションはサポートされていません

Astra Controlは、ネームスペースごとに単一のストレージクラスを使用するアプリケーションをサポートします。ネームスペースにアプリケーションを追加するときは、そのアプリケーションのストレージクラスがネームスペース内の他のアプリケーションと同じであることを確認してください。

Astra Controlでは、クラウドインスタンスにデフォルトのバケットは自動的に割り当てられません

Astra Controlでは、どのクラウドインスタンスに対してもデフォルトのバケットが自動的に割り当てられることはありません。クラウドインスタンスのデフォルトバケットは手動で設定する必要があります。デフォルトのバケットが設定されていないと、2つのクラスタ間でアプリケーションのクローニング処理を実行できません。

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

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。

> クローン処理中に、IngressClassリソースまたはwebhookを必要とするアプリケーションが正常に機能するためには、これらのリソースがデスティネーションクラスタですでに定義されていない必要があります。

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"。

特定のバージョンのSnapshotコントローラを含むKubernetes 1.25以降のクラスタでは、Snapshotが失敗することがあります

バージョン1.25以降を実行しているKubernetesクラスタのSnapshotは、クラスタにSnapshotコントローラAPIのバージョンv1beta1がインストールされている場合に失敗することがあります。

既存のKubernetes 1.25以降のインストールをアップグレードする場合は、回避策 として次の手順を実行します。

1. 既存のSnapshot CRDと既存のSnapshotコントローラをすべて削除します。
2. "Astra Trident をアンインストール"。
3. "スナップショットCRDとスナップショットコントローラをインストールします"。
4. "最新バージョンのAstra Tridentをインストール"。
5. "VolumeSnapshotClassを作成します"。

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。

ONTAP NASエコノミーストレージクラスへのIn Placeリストア処理が失敗する

アプリケーションのインプレースリストアを実行し（アプリケーションを元のネームスペースにリストア）、アプリケーションのストレージクラスで `ontap-nas-economy` ドライバの場合、スナップショットディレクトリが非表示になっていないとリストア処理が失敗することがあります。インプレースでリストアする前に、の手順に従ってください。 "ONTAP NAS経済性に優れた運用向けのバックアップとリストアを実現" Snapshotディレクトリを非表示にします。

LDAPユーザおよびグループの制限事項

Astra Control Centerは、最大5,000のリモートグループと10,000のリモートユーザをサポートします。

Astra Controlでは、末尾にスペースがあるRDNを含むDNを持つLDAPエンティティ（ユーザまたはグループ）はサポートされません。

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。

[Activity]ページには、最大10万件のイベントが表示されます

[Astra Control Activity]ページには、最大10、000件のイベントを表示できます。ログに記録されたすべてのイベントを表示するには、を使用してイベントを取得します "Astra Control API の略"。

SnapMirrorはストレージバックエンドにNVMe over TCPを使用するアプリケーションをサポートしない

Astra Control Centerでは、NVMe over TCPプロトコルを使用するストレージバックエンドのNetApp SnapMirrorレプリケーションはサポートされません。

詳細については、こちらをご覧ください

- "既知の問題"

はじめに

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、複製、移行を簡易化し、作業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
- NetApp SnapMirrorテクノロジを使用してアプリケーションをリモートシステムにレプリケート（Astra Control Center）
- ステージング環境から本番環境へのアプリケーションのクローニング
- 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
- Web UIまたはAPIを使用して、バックアップと移行のワークフローを実装します

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 *：ネットアップが管理するサービス。複数のクラウドプロバイダ環境や自己管理型のKubernetesクラスタで、アプリケーションに対応したデータ管理を提供します。
- * 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。NetApp Cloud Volumes ONTAP ストレージバックエンドを使用して、複数のクラウドプロバイダ環境にAstra Control Centerをインストールすることもできます。

	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、および管理できます
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	自社所有のKubernetesクラスタ
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
サポートされている**Kubernetes**ディストリビューションを教えてください。	• クラウドプロバイダ ◦ Amazon Web Services の ▪ Amazon Elastic Kubernetes Service（EKS） ◦ Google Cloud ▪ Google Kubernetes Engine （ GKE ） ◦ Microsoft Azure ▪ Azure Kubernetes Service （ AKS ） • 自己管理クラスタ ◦ Kubernetes（アップストリーム） ◦ Rancher Kubernetes Engine （ RKE ） ◦ Red Hat OpenShift Container Platform • オンプレミスクラスタ ◦ オンプレミスのRed Hat OpenShift Container Platform	• Azure Stack HCIで実行されるAzure Kubernetes Service • Google Anthos • Kubernetes（アップストリーム） • Rancher Kubernetes Engine （ RKE ） • Red Hat OpenShift Container Platform

	Astra 制御サービス	Astra Control Center の略
サポートされているストレージバックエンドは何ですか。	• クラウドプロバイダ ◦ Amazon Web Services の ▪ Amazon EBSのことです ▪ NetApp ONTAP 対応の Amazon FSX ▪ "Cloud Volumes ONTAP" ◦ Google Cloud ▪ Google Persistent Disk のことです ▪ NetApp Cloud Volumes Service の略 ▪ "Cloud Volumes ONTAP" ◦ Microsoft Azure ▪ Azure Managed Disksの略 ▪ Azure NetApp Files の特長 ▪ "Cloud Volumes ONTAP" • 自己管理クラスタ ◦ Amazon EBSのことです ◦ Azure Managed Disksの略 ◦ Google Persistent Disk のことです ◦ "Cloud Volumes ONTAP" ◦ NetApp MetroCluster ◦ "ロングホーン"" • オンプレミスクラスタ ◦ NetApp MetroCluster ◦ NetApp ONTAP AFF および FAS システム ◦ NetApp ONTAP Select の略 ◦ "Cloud Volumes ONTAP" ◦ "ロングホーン""	• NetApp ONTAP AFF および FAS システム • NetApp ONTAP Select の略 • "Cloud Volumes ONTAP" • "ロングホーン""

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で管理されているディスクがあります。
  - Amazon EKSクラスタの場合、Astra Control Serviceはを使用します "Amazon Elastic Block Store" または "NetApp ONTAP 対応の Amazon FSX" 永続ボリュームのストレージバックエンドとして。
- 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。
  - この新しい管理者ロールを使用して、link ../concepts/architecture#astra-control-components [Astra Control Provisioner ^]をクラスタにインストールし、1つ以上のストレージクラスを作成します。
  - NetAppクラウドサービスのストレージサービスをストレージバックエンドとして使用する場合、Astra Control ServiceはAstra Control Provisioner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。Amazon EBSまたはAzureで管理されているディスクをストレージバックエンドとして使用している場合は、プロバイダ固有のCSIドライバをインストールする必要があります。インストール手順については、を参照してください "Amazon Web Servicesをセットアップする" および "Azure で管理されているディスクを使用して Microsoft Azure をセットアップする"。
- 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の無料プランを使用すると、最大10個のネームスペースをアカウントで管理できます。10以上を管理する場合は、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、ONTAPストレージバックエンドを使用するAstra Control Provisionerが設定されたストレージクラスでKubernetesクラスタをサポートします。

Astra Control Centerでは、限定的な（7日間の指標）監視とテレメトリを使用できます。また、オープンな指標エンドポイントを介してKubernetesネイティブの監視ツール（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のデジタルアドバイザ（デジタルアドバイザとも呼ばれます）エコシステムに完全に統合されており、ユーザとNetAppサポートにトラブルシューティングと使用状況の情報を提供します。

90日間の組み込み評価用ライセンスを使用して、Astra Control Centerを試用できます。Astra Control Centerの評価中は、Eメールとコミュニティのオプションでサポートを受けることができます。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"。

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を使用してそれらを管理できます。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、およびレプリケーション関係を作成します。

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

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。Astra Control Centerを導入して運用するために、お客様の環境が上記の要件を満たしていることを確認してください。

サポート対象のホストクラスタ**Kubernetes**環境

Astra Control Centerは、次のKubernetesホスト環境で検証済みです。

> Astra Control Centerをホストする**Kubernetes**環境が、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。

ホストクラスタ上の**Kubernetes**ディストリビューション	サポートされるバージョン
Azure Stack HCIで実行されるAzure Kubernetes Service	Azure Stack HCI 21H2および22H2（AKS 1.24.11~1.26.6を使用）
Google Anthos	1.15～1.16（を参照） Google Anthos Ingressの要件)
Kubernetes（アップストリーム）	1.27～1.29
Rancher Kubernetes Engine （ RKE ）	RKE 1：バージョン1.24.17、1.25.13、1.26.8（Rancher Manager 2.7.9を使用） RKE 2：Rancher Manager 2.6.13を使用したバージョン1.23.16および1.24.13 RKE 2：バージョン1.24.17、1.25.14、1.26.9（Rancher Manager 2.7.9を使用）
Red Hat OpenShift Container Platform	4.12~4.14

ホストクラスタリソースの要件

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。

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。

- * CPU拡張機能*：ホスティング環境のすべてのノードのCPUでAVX拡張機能が有効になっている必要があります。
- ワーカーノード:少なくとも3つのワーカーノードで、それぞれ4つのCPUコアと12GBのRAMを備えています
- * 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をホストする場合は、次の考慮事項に注意してください。
  - 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"
  - を使用します `kubectl get nsxlbmonitors -A` 入力トラフィックを受け入れるように設定されたサービスモニタがすでにあるかどうかを確認するコマンド。MetalLB が存在する場合は、既存のサービスモニタが新しいロードバランサ設定を上書きするため、 MetalLB をインストールしないでください。
  - TKG または TKGi のデフォルト・ストレージ・クラス・エンフォースメントは、 Astra Control によって管理されるすべてのアプリケーション・クラスタで無効にします。これを行うには、を編集します `TanzuKubernetesCluster` ネームスペースクラスタ上のリソース。
  - TKGまたはTKGi環境にAstra Control Centerを導入する場合は、Astra Control Provisionerに固有の要件に注意してください。
    - クラスタが特権ワークロードをサポートしている必要があります。
    - 。 `--kubelet-dir` フラグはkubeletディレクトリの場所に設定する必要があります。デフォルト

はです `/var/vcap/data/kubelet`。

- を使用してkubeletの場所を指定します `--kubelet-dir` は、Trident Operator、Helm、およびで動作することがわかっています `tridentctl` 導入：

サービスメッシュの要件

サポートされているバニラバージョンのIstioサービスメッシュをAstra Control Centerホストクラスタにインストールすることを強く推奨します。を参照してください "サポートされるリリース" サポートされているバージョンのIstioの場合。OpenShift Service MeshなどのIstioサービスメッシュのブランドリリースは、Astra Control Centerでは検証されていません。

ホストクラスタにインストールされているIstioサービスメッシュとAstra Control Centerを統合するには、Astra Control Centerの一部として統合を行う必要があります。 "インストール" そしてこのプロセスから独立していません。

ホストクラスタにサービスメッシュを設定せずにAstra Control Centerをインストールして使用すると、セキュリティに重大な影響を及ぼす可能性があります。

Astra Trident

本リリースでAstra Control Provisionerの代わりにAstra Tridentを使用する場合は、Astra Trident 23.04以降のバージョンがサポートされます。Astra Control Centerには Astra Controlプロビジョニングツール 今後のリリースでは、

Astra Controlプロビジョニングツール

Astra Control Provisionerの高度なストレージ機能を使用するには、Astra Trident 23.10以降をインストールし、 "Astra Control Provisionerの機能"。最新のAstra Control Provisioner機能を使用するには、Astra TridentとAstra Control Centerの最新バージョンが必要です。

- * Astra Control Centerで使用するAstra Control Provisionerの最小バージョン*：Astra Control Provisioner 23.10以降がインストールおよび設定されています。

Astra Tridentを使用した**ONTAP**の設定

- ストレージクラス：クラスタに少なくとも1つのストレージクラスを設定します。デフォルトのストレージクラスが設定されている場合は、そのストレージクラスがデフォルトで指定された唯一のストレージクラスであることを確認します。
- ストレージドライバとワーカーノード:ポッドがバックエンドストレージと対話できるように、クラスタ内のワーカーノードを適切なストレージドライバで構成します。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-economy` （このストレージクラスタイプではアプリケーションレプリケーションは使用できません）
  - `ontap-nas-economy` （このストレージクラスタイプではSnapshotおよびアプリケーションレプリケーションポリシーは使用できません）

ストレージバックエンド

十分な容量を備えたサポート対象のバックエンドがあることを確認してください。

- 必要なストレージバックエンド容量：500GB以上の空き容量
- サポートされるバックエンド：Astra Control Centerは次のストレージバックエンドをサポートします。
  - NetApp ONTAP 9.9.1以降のAFF、FAS、ASAシステム
  - NetApp ONTAP Select 9.9.1以降
  - NetApp Cloud Volumes ONTAP 9.9.1以降
  - （Astra Control Centerのテクニカルプレビュー）技術プレビューとして提供されるデータ保護処理用のNetApp ONTAP 9.10.1以降
  - Longhorn 1.5.0以降
    - VolumeSnapshotClassオブジェクトを手動で作成する必要があります。を参照してください "Longhornドキュメント" 手順については、を参照し
  - NetApp MetroCluster
    - 管理対象のKubernetesクラスタはストレッチ構成に含まれている必要があります。
  - サポート対象のクラウドプロバイダで利用可能なストレージバックエンド

ONTAP ライセンス

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

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

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

NetApp MetroCluster

NetApp MetroClusterをストレージバックエンドとして使用する場合は、次の作業を行う必要があります。

- 使用するAstra Tridentドライバで、バックエンドオプションとしてSVM管理LIFを指定する
- 適切なONTAPライセンスがあることを確認します。

MetroCluster LIFを設定するには、各ドライバについて次のオプションと例を参照してください。

- "SAN"
- "NAS"

Astra Control Centerのライセンス

Astra Control CenterにはAstra Control Centerライセンスが必要です。Astra Control Centerをインストールすると、4、800 CPUユニットの90日間の評価用ライセンスがすでにアクティブ化されています。容量の追加や評価期間の変更が必要な場合や、フルライセンスにアップグレードする場合は、ネットアップから別の評価用ライセンスまたはフルライセンスを取得できます。アプリケーションとデータを保護するにはライセンスが必要です。

Astra Control Centerは無償トライアルにサインアップして試すことができます。登録することでサインアップできます "こちらをご覧ください"。

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

ライセンスの機能の詳細については、を参照してください "ライセンス"。

ネットワーク要件

Astra Control Centerが適切に通信できるように運用環境を設定します。次のネットワーク設定が必要です。

- * FQDNアドレス*：Astra Control CenterのFQDNアドレスが必要です。
- インターネットへのアクセス：インターネットに外部からアクセスできるかどうかを判断する必要があります。そうしないと、サポートバンドルをへ送信するなど、一部の機能が制限されることがあります。 "NetApp Support Site"。
- ポートアクセス：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とAstra Control Centerへのアクセスに使用するシステムの間で、このポートが両方向で開いていることを確認する

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	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
Astra Control Center の略	管理対象のKubernetesクラスタ	443/6443 注：管理対象クラスタが使用するポートは、クラスタによって異なる場合があります。クラスタソフトウェアベンダーのドキュメントを参照してください。	HTTPS	管理対象クラスタとの通信- Astra Control Centerをホストす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。

オンプレミス **Kubernetes** クラスタへの入力

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

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

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"。

Google Anthos Ingressの要件

Google AnthosクラスタでAstra Control Centerをホストする場合、Google AnthosにはMetalLBロードバランサとIstio Ingressサービスがデフォルトで含まれているため、インストール時にAstra Control Centerの一般的な入力機能を簡単に使用できます。を参照してください "Astra Control Centerのインストールドキュメント" を参照してください。

サポートされている Web ブラウザ

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

アプリケーションクラスタのその他の要件

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

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

次のステップ

を表示します "クイックスタート" 概要（ Overview ）：

Astra Control Center のクイックスタート

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

1 **Kubernetes** クラスタの要件を確認

環境が次の要件を満たしていることを確認します。

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

ストレージ統合

- "環境にAstra Control Provisionerが含まれていることを確認する"
- "Astra Control Provisionerの高度な管理機能とストレージプロビジョニング機能を有効にする"
- "クラスタワーカーノードを準備"
- "ストレージバックエンドの設定"
- "ストレージクラスを設定"
- "ボリュームSnapshotコントローラのインストール"
- "ボリュームSnapshotクラスを作成します"
- ONTAP クレデンシャル*
- "ONTAP クレデンシャルを設定する"

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

次のインストールタスクを実行します。

- "NetApp Support Site のダウンロードページからAstra Control Centerをダウンロードします"
- ネットアップライセンスファイルを入手します。
  - Astra Control Centerを評価する場合は、組み込みの評価用ライセンスがすでに付属しています
  - "Astra Control Centerをすでに購入している場合は、ライセンスファイルを生成します"
- "Astra Control Center をインストールします"
- "追加のオプション設定手順を実行します"

3 いくつかの初期セットアップ作業を完了します

開始するには、いくつかの基本的なタスクを実行します。

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

4 Astra Control Center を使用

Astra Control Centerのセットアップが完了したら、Astra Control UIまたはを使用します "Astra Control API の略" アプリの管理と保護を開始するには：

- "アカウントを管理"：ユーザ、ロール、LDAP、クレデンシャルなど。
- "通知を管理します"
- "アプリの管理":管理するリソースを定義します。
- "アプリを保護します"：保護ポリシーを構成し、アプリケーションのレプリケーション、クローニング、移行を行います。

を参照してください。

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

インストールの概要

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

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

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

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

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

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

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

作業を開始する前に

- 環境条件を満たしている： "インストールを開始する前に、 Astra Control Center の導入環境を準備します"。

> 3つ目の障害ドメインまたはセカンダリサイトにAstra Control Centerを導入これは、アプリケーションのレプリケーションとシームレスなディザスタリカバリに推奨されます。

- 正常なサービスを確認：すべてのAPIサービスが正常な状態で使用可能であることを確認します。

```
kubectl get apiservices
```

- *ルーティング可能なFQDN *：使用するAstra FQDNをクラスタにルーティングできることを確認します。つまり、内部 DNS サーバに DNS エントリがあるか、すでに登録されているコア URL ルートを使用しています。
- 証明書マネージャの設定:クラスタに証明書マネージャがすでに存在する場合は、一部の証明書マネージャを実行する必要があります。 "事前に必要な手順" そのため、Astra Control Centerは独自の証明書マネージャのインストールを試みません。デフォルトでは、Astra Control Centerはインストール時に独自の証明書マネージャをインストールします。
- （**ONTAP SAN**ドライバのみ）マルチパスの有効化：ONTAP SANドライバを使用している場合は、すべてのKubernetesクラスタでマルチパスが有効になっていることを確認してください。

また、次の点も考慮する必要があります。

- * NetApp Astra Controlイメージレジストリへのアクセス*：

  Astra Control Provisionerなど、Astra Controlのインストールイメージや機能強化された機能をNetAppイメージレジストリから取得することができます。

  a. レジストリへのログインに必要なAstra ControlアカウントIDを記録します。

アカウントIDはAstra Control Service Web UIで確認できます。ページ右上の図アイコンを選択し、* APIアクセス*を選択して、アカウントIDを書き留めます。

b. 同じページから* APIトークンの生成*を選択し、APIトークン文字列をクリップボードにコピーしてエディターに保存します。

c. Astra Controlレジストリにログインします。

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

- セキュアな通信のためのサービスメッシュをインストール：Astra Controlホストクラスタの通信チャネルは、 "サポートされるサービスメッシュ"。

> Astra Control Centerとサービスメッシュの統合は、Astra Control Centerでのみ実行可能 "インストール" そしてこのプロセスから独立していません。メッシュ環境から非メッシュ環境に戻すことはサポートされていません。

Istioサービスメッシュを使用するには、次の手順を実行する必要があります。

  - を追加します。 `istio-injection:enabled` ラベル にアクセスしてからAstra Control Centerを導入する必要があります。
  - を使用します `Generic` 入力設定 別のイングレスを提供します。 外部ロードバランシング。
  - Red Hat OpenShiftクラスタの場合は、 `NetworkAttachmentDefinition` 関連付けられているすべてのAstra Control Centerネームスペース (`netapp-acc-operator`、 `netapp-acc`、 `netapp-monitoring` アプリケーションクラスタの場合、または置換されたカスタムネームスペースの場合）。

```
cat <<EOF | oc -n netapp-acc-operator create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-acc create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-monitoring create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF
```

手順

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

- Astra Control Centerをダウンロードして展開します
- [ローカルレジストリを使用する場合は、追加の手順を実行します。]
- [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オペレータ（たとえば、 `kubectl delete -f astra_control_center_operator_deploy.yaml`）Astra Control Centerのインストール中または操作中はいつでも、ポッドを削除しないようにします。

Astra Control Centerをダウンロードして展開します

次のいずれかの場所からAstra Control Centerのイメージをダウンロードします。

- * Astra Controlサービスのイメージレジストリ*：Astra Control Centerのイメージでローカルレジストリを使用しない場合や、NetApp Support Siteからバンドルをダウンロードするよりもこの方法を使用する場合は、このオプションを使用します。
- * NetApp Support Site *：このオプションは、Astra Control Centerのイメージでローカルレジストリを使用する場合に使用します。

Astra Controlイメージレジストリ

1. Astra Control Serviceにログインします。
2. ダッシュボードで、*[Deploy a self-managed instance of Astra Control]*を選択します。
3. 手順に従ってAstra Controlイメージのレジストリにログインし、Astra Control Centerのインストールイメージを取得してイメージを展開します。

NetApp Support Site

1. Astra Control Centerを含むバンドルをダウンロードします (`astra-control-center-[version].tar.gz`）をクリックします "Astra Control Centerのダウンロードページ"。
2. （推奨ですがオプション）Astra Control Centerの証明書と署名のバンドルをダウンロードします (`astra-control-center-certs-[version].tar.gz`）をクリックして、バンドルのシグネチャを確認します。

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

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

出力にはと表示されます `Verified OK` 検証が成功したあとに、

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

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

ローカルレジストリを使用する場合は、追加の手順を実行します。

Astra Control Centerバンドルをローカルのレジストリにプッシュする場合は、NetApp Astra kubectlコマンドラインプラグインを使用する必要があります。

ネットアップ**Astra kubectl**プラグインをインストール

次の手順を実行して、最新のNetApp Astra kubectlコマンドラインプラグインをインストールします。

作業を開始する前に

ネットアップでは、CPUアーキテクチャやオペレーティングシステム別にプラグインのバイナリを提供して

います。このタスクを実行する前に、使用しているCPUとオペレーティングシステムを把握しておく必要があります。

以前のインストールからプラグインがインストールされている場合は、 "最新バージョンがインストールされていることを確認してください" これらの手順を実行する前に。

手順

1. 使用可能なNetApp Astra kubectlプラグインのバイナリを一覧表示します。

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

   ```
   ls kubectl-astra/
   ```

2. 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を現在のパスに移動し、次の名前に変更します。 `kubectl-astra`：

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

イメージをレジストリに追加する

1. Astra Control Centerバンドルをローカルのレジストリにプッシュする場合は、コンテナエンジンに応じた手順を実行します。

Docker です

a. tarballのルートディレクトリに移動します。次のように表示されます。 `acc.manifest.bundle.yaml` ファイルと次のディレクトリ：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. Astra Control Centerのイメージディレクトリにあるパッケージイメージをローカルレジストリにプッシュします。を実行する前に、次の置換を行ってください `push-images` コマンドを実行します

- <BUNDLE_FILE> をAstra Controlバンドルファイルの名前に置き換えます（`acc.manifest.bundle.yaml`）。
- <MY_FULL_REGISTRY_PATH> をDockerリポジトリのURLに置き換えます。次に例を示します。 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
- <MY_REGISTRY_USER> をユーザ名に置き換えます。
- <MY_REGISTRY_TOKEN> をレジストリの認証済みトークンに置き換えます。

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

ポドマン

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

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. レジストリにログインします。

```
podman login <YOUR_REGISTRY>
```

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

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

レジストリ設定に応じて、スクリプトが作成するイメージパスは次のようになります。

```
https://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version
```

2. ディレクトリを変更します。

```
cd manifests
```

認証要件を持つレジストリのネームスペースとシークレットを設定します

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

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

インストールを完了する前に、Astra Control Centerをインストールするクラスタをkubeconfigで指定していることを確認してください。

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

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

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

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

プレースホルダ `your_registry_path` 以前にアップロードした画像の場所と一致する必要があります（例： `[Registry_URL]/netapp/astra/astracc/24.02.0-69` ）。

```
kubectl create secret docker-registry astra-registry-cred -n netapp-acc-operator --docker-server=cr.astra.netapp.io --docker-username=[astra_account_id] --docker-password=[astra_api_token]
```

+

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

+

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

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

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

b. のシークレットを作成します `netapp-acc` （またはカスタム名）ネームスペース。Docker情報を追

加し、レジストリの設定に応じて適切なコマンドのいずれかを実行します。

```
kubectl create secret docker-registry astra-registry-cred -n [netapp-acc or custom namespace] --docker-server=cr.astra.netapp.io --docker-username=[astra_account_id] --docker-password=[astra_api_token]
```

```
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]
```

Astra Control Center オペレータを設置します

1. （ローカルレジストリのみ）ローカルレジストリを使用している場合は、次の手順を実行します。

   a. Astra Control Centerオペレータによる導入YAMLを開きます。

   ```
   vim astra_control_center_operator_deploy.yaml
   ```

   注釈付きサンプルYAMLは以下の手順に従います。

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

   ```
   imagePullSecrets: [{name: astra-registry-cred}]
   ```

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

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

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
```

```
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: ASTRA_IMAGE_REGISTRY/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_INSTALLTIMEOUT
          value: 5m
        image: ASTRA_IMAGE_REGISTRY/acc-operator:24.02.68
        imagePullPolicy: IfNotPresent
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8081
          initialDelaySeconds: 15
          periodSeconds: 20
        name: manager
        readinessProbe:
          httpGet:
            path: /readyz
            port: 8081
          initialDelaySeconds: 5
          periodSeconds: 10
        resources:
          limits:
            cpu: 300m
            memory: 750Mi
```

```
          requests:
            cpu: 100m
            memory: 75Mi
        securityContext:
          allowPrivilegeEscalation: false
      imagePullSecrets: []
      securityContext:
        runAsUser: 65532
      terminationGracePeriodSeconds: 10
```

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
```

回答例を表示するには展開します。

```
namespace/netapp-acc-operator created
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.as
tra.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
```

3. ポッドが実行中であることを確認します

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

Astra Control Center を設定します

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

```
vim astra_control_center.yaml
```

注釈付きサンプルYAMLは以下の手順に従います。

2. 次の設定を変更または確認します。

アカウント名

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

astraVersion

設定	ガイダンス（**Guidance**）	を入力します	例
`astraVersion`	導入するAstra Control Centerのバージョン。この設定には値があらかじめ入力されているため、対処は不要です。	文字列	`24.02.0-69`

astraitAddress

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

AutoSupport

このセクションで選択した内容によって、NetAppのプロアクティブサポートアプリケーション、デジタルアドバイザ、およびデータの送信先が決まります。インターネット接続が必要です（ポート442）。サポートデータはすべて匿名化されます。

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

E メール

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

FirstName

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

姓

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

imageRegistryのこと

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

設定	使用	ガイダンス（**Guidance**）	を入力します	例
`imageRegistry.name`	必須	Astra Control Centerの導入に必要なすべてのイメージをホストするAstra Controlイメージレジストリの名前。値は事前に入力されます。ローカルレジストリを設定しないかぎり、アクションは必要ありません。ローカルレジストリの場合は、この既存の値を、前の手順。使用しないでください `http://` または `https://` をレジストリ名に追加します。	文字列	`cr.astra.netapp.io` （デフォルト） `example.registry.com/astra` （ローカルレジストリの例）

設定	使用	ガイダンス（**Guidance**）	を入力します	例
`imageRegistry.secret`	任意。	イメージレジストリでの認証に使用するKubernetesシークレットの名前。値は事前に入力されます。ローカルレジストリを設定し、でそのレジストリに入力した文字列を設定しない限り、アクションは必要ありません。`imageRegistry.name` シークレットが必要です。 重要：認証を必要としないローカルレジストリを使用している場合は、これを削除する必要があります。`secret` ラインの内側 `imageRegistry` または、インストールが失敗します。	文字列	`astra-registry-cred`

ストレージクラス

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

volumeReclaimPolicyのように指定します

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

ingressType

設定	ガイダンス（**Guidance**）	を入力します	オプション（ **Options** ）
`ingressType`	次の入力タイプのいずれかを使用します。 汎用 (`ingressType: "Generic"`）（デフォルト） このオプションは、別の入力コントローラを使用している場合、または独自の入力コントローラを使用する場合に使用します。Astra Control Centerを導入したら、 "入力コントローラ" URLを使用してAstra Control Centerを公開します。 重要：Astra Control Centerでサービスメッシュを使用する場合は、 `Generic` 入力タイプとして入力し、独自の設定を行います。 "入力コントローラ"。 * AccTraefik * (`ingressType: "AccTraefik"`) 入力コントローラを設定しない場合は、このオプションを使用します。これにより、Astra Control Centerが導入されます `traefik` Gateway as a Kubernetes LoadBalancer type serviceの略。 Astra Control Centerは、タイプ「LoadBalancer」のサービスを使用します。 (`svc/traefik` Astra Control Centerの名前空間）で、アクセス可能な外部IPアドレスが割り当てられている必要があります。お使いの環境でロードバランサが許可されていて、設定されていない	文字列	• `Generic` （デフォルト値） • `AccTraefik`

スケールサイズ

設定	ガイダンス（**Guidance**）	を入力します	オプション（ **Options** ）
`scaleSize`	デフォルトでは、Astra で高可用性（HA）が使用されます。 `scaleSize` の `Medium`ほとんどのサービスをHAに導入し、冗長性を確保するために複数のレプリカを導入します。を使用 `scaleSize` として `Small`Astraは、消費量を削減するための必須サービスを除き、すべてのサービスのレプリカ数を削減します。ヒント： `Medium` 環境は約100個のポッドで構成されています（一時的なワークロードは含まれません）。100個のポッドは、3つのマスターノードと3つのワーカーノード構成に基づいています）。特にディザスタリカバリのシナリオを検討する場合は、環境で問題 となる可能性があるポッド単位のネットワーク制限に注意してください。	文字列	• `Small` • `Medium` （デフォルト値）

astraitcesScaler

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

追加値

インストール時に既知の問題が表示されないように、Astra Control CenterのCRに次の値を追加します。

```
additionalValues:
    keycloak-operator:
      livenessProbe:
        initialDelaySeconds: 180
      readinessProbe:
        initialDelaySeconds: 180
```

CRD

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

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

インストールを完了する前に、構成に適したストレージクラスと入力タイプを選択していることを確認してください。

astra_control_center.yamlの例

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[cr.astra.netapp.io or your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
  volumeReclaimPolicy: "Retain"
  ingressType: "Generic"
  scaleSize: "Medium"
  astraResourcesScaler: "Default"
  additionalValues:
    keycloak-operator:
      livenessProbe:
        initialDelaySeconds: 180
      readinessProbe:
        initialDelaySeconds: 180
  crds:
    externalTraefik: false
    externalCertManager: false
```

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

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

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

2. Astra Control Centerでサービスメッシュを使用している場合は、 `netapp-acc` またはカスタムネームスペース：

> 入力タイプ (`ingressType`）をに設定する必要があります。 `Generic` このコマンドを実行する前に、Astra Control Center CRで確認する必要があります。

```
kubectl label ns [netapp-acc or custom namespace] istio-
injection:enabled
```

3. （推奨） "厳密なMTLを有効にする" Istioサービスメッシュの場合：

```
kubectl apply -n istio-system -f - <<EOF
apiVersion: security.istio.io/v1beta1
kind: PeerAuthentication
metadata:
  name: default
spec:
  mtls:
    mode: STRICT
EOF
```

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

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

Astra Control Centerのオペレータが環境要件の自動チェックを実行ありません "要件" 原因 でインストールが失敗するか、Astra Control Centerが正常に動作しない可能性があります。を参照してください 次のセクション 自動システムチェックに関連する警告メッセージをチェックします。

システムステータスを確認します

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

手順

1. インストールプロセスで検証チェックに関連する警告メッセージが生成されなかったことを確認します。

```
kubectl get acc [astra or custom Astra Control Center CR name] -n
[netapp-acc or custom namespace] -o yaml
```

その他の警告メッセージは、Astra Control Centerのオペレータログでも報告されます。

2. 自動化された要件チェックによって報告された環境の問題を修正します。

問題を解決するには、環境がを満たしていることを確認します "要件" （Astra Control Center向け）。

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

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

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

サンプル応答のために展開

```
acc-helm-repo-5bd77c9ddd-8wxm2              1/1     Running     0
1h
activity-5bb474dc67-8l9ss                   1/1     Running     0
1h
activity-5bb474dc67-qbrtq                   1/1     Running     0
1h
api-token-authentication-6wbj2              1/1     Running     0
1h
api-token-authentication-9pgw6              1/1     Running     0
1h
api-token-authentication-tqf6d              1/1     Running     0
1h
asup-5495f44dbd-z4kft                       1/1     Running     0
1h
authentication-6fdd899858-5x45s             1/1     Running     0
1h
bucketservice-84d47487d-n9xgp               1/1     Running     0
1h
bucketservice-84d47487d-t5jhm               1/1     Running     0
1h
cert-manager-5dcb7648c4-hbldc               1/1     Running     0
1h
cert-manager-5dcb7648c4-nr9qf               1/1     Running     0
1h
cert-manager-cainjector-59b666fb75-bk2tf    1/1     Running     0
1h
cert-manager-cainjector-59b666fb75-pfnck    1/1     Running     0
1h
cert-manager-webhook-c6f9b6796-ngz2x        1/1     Running     0
1h
cert-manager-webhook-c6f9b6796-rwtbn        1/1     Running     0
1h
certificates-5f5b7b4dd-52tnj                1/1     Running     0
1h
certificates-5f5b7b4dd-gtjbx                1/1     Running     0
1h
certificates-expiry-check-28477260-dz5vw    0/1     Completed   0
1h
cloud-extension-6f58cc579c-lzfmv            1/1     Running     0
1h
cloud-extension-6f58cc579c-zw2km            1/1     Running     0
1h
cluster-orchestrator-79dd5c8d95-qjg92       1/1     Running     0
```

```
1h
composite-compute-85dc84579c-nz82f          1/1     Running     0
1h
composite-compute-85dc84579c-wx2z2          1/1     Running     0
1h
composite-volume-bff6f4f76-789nj            1/1     Running     0
1h
composite-volume-bff6f4f76-kwnd4            1/1     Running     0
1h
credentials-79fd64f788-m7m8f                1/1     Running     0
1h
credentials-79fd64f788-qnc6c                1/1     Running     0
1h
entitlement-f69cdbd77-4p2kn                 1/1     Running     0
1h
entitlement-f69cdbd77-hswm6                 1/1     Running     0
1h
features-7b9585444c-7xd7m                   1/1     Running     0
1h
features-7b9585444c-dcqwc                   1/1     Running     0
1h
fluent-bit-ds-crq8m                         1/1     Running     0
1h
fluent-bit-ds-gmgq8                         1/1     Running     0
1h
fluent-bit-ds-gzr4f                         1/1     Running     0
1h
fluent-bit-ds-j6sf6                         1/1     Running     0
1h
fluent-bit-ds-v4t9f                         1/1     Running     0
1h
fluent-bit-ds-x7j59                         1/1     Running     0
1h
graphql-server-6cc684fb46-2x8lr             1/1     Running     0
1h
graphql-server-6cc684fb46-bshbd             1/1     Running     0
1h
hybridauth-84599f79fd-fjc7k                 1/1     Running     0
1h
hybridauth-84599f79fd-s9pmn                 1/1     Running     0
1h
identity-95df98cb5-dvlmz                    1/1     Running     0
1h
identity-95df98cb5-krf59                    1/1     Running     0
1h
influxdb2-0                                 1/1     Running     0
```

```
1h
keycloak-operator-6d4d688697-cfq8b          1/1     Running     0
1h
krakend-5d5c8f4668-7bq8g                    1/1     Running     0
1h
krakend-5d5c8f4668-t8hbn                    1/1     Running     0
1h
license-689cdd4595-2gsc8                    1/1     Running     0
1h
license-689cdd4595-g6vwk                    1/1     Running     0
1h
login-ui-57bb599956-4fwgz                   1/1     Running     0
1h
login-ui-57bb599956-rhztb                   1/1     Running     0
1h
loki-0                                      1/1     Running     0
1h
metrics-facade-846999bdd4-f7jdm             1/1     Running     0
1h
metrics-facade-846999bdd4-lnsxl             1/1     Running     0
1h
monitoring-operator-6c9d6c4b8c-ggkrl        2/2     Running     0
1h
nats-0                                      1/1     Running     0
1h
nats-1                                      1/1     Running     0
1h
nats-2                                      1/1     Running     0
1h
natssync-server-6df7d6cc68-9v2gd            1/1     Running     0
1h
nautilus-64b7fbdd98-bsgwb                   1/1     Running     0
1h
nautilus-64b7fbdd98-djlhw                   1/1     Running     0
1h
openapi-864584bccc-75nlv                    1/1     Running     0
1h
openapi-864584bccc-zh6bx                    1/1     Running     0
1h
polaris-consul-consul-server-0              1/1     Running     0
1h
polaris-consul-consul-server-1              1/1     Running     0
1h
polaris-consul-consul-server-2              1/1     Running     0
1h
polaris-keycloak-0                          1/1     Running     2 (1h
```

```
ago)      1h
polaris-keycloak-1                            1/1     Running     0
1h
polaris-keycloak-db-0                         1/1     Running     0
1h
polaris-keycloak-db-1                         1/1     Running     0
1h
polaris-keycloak-db-2                         1/1     Running     0
1h
polaris-mongodb-0                             1/1     Running     0
1h
polaris-mongodb-1                             1/1     Running     0
1h
polaris-mongodb-2                             1/1     Running     0
1h
polaris-ui-66476dcf87-f6s8j                   1/1     Running     0
1h
polaris-ui-66476dcf87-ztjk7                   1/1     Running     0
1h
polaris-vault-0                               1/1     Running     0
1h
polaris-vault-1                               1/1     Running     0
1h
polaris-vault-2                               1/1     Running     0
1h
public-metrics-bfc4fc964-x4m79                1/1     Running     0
1h
storage-backend-metrics-7dbb88d4bc-g78cj      1/1     Running     0
1h
storage-provider-5969b5df5-hjvcm              1/1     Running     0
1h
storage-provider-5969b5df5-r79ld              1/1     Running     0
1h
task-service-5fc9dc8d99-4q4f4                 1/1     Running     0
1h
task-service-5fc9dc8d99-8l5zl                 1/1     Running     0
1h
task-service-task-purge-28485735-fdzkd        1/1     Running     0
12m
telegraf-ds-2rgm4                             1/1     Running     0
1h
telegraf-ds-4qp6r                             1/1     Running     0
1h
telegraf-ds-77frs                             1/1     Running     0
1h
telegraf-ds-bc725                             1/1     Running     0
```

```
1h
telegraf-ds-cvmxf                              1/1     Running     0
1h
telegraf-ds-tqzgj                              1/1     Running     0
1h
telegraf-rs-5wtd8                              1/1     Running     0
1h
telemetry-service-6747866474-5djnc             1/1     Running     0
1h
telemetry-service-6747866474-thb7r             1/1     Running     1 (1h
ago)      1h
tenancy-5669854fb6-gzdzf                       1/1     Running     0
1h
tenancy-5669854fb6-xvsm2                       1/1     Running     0
1h
traefik-8f55f7d5d-4lgfw                        1/1     Running     0
1h
traefik-8f55f7d5d-j4wt6                        1/1     Running     0
1h
traefik-8f55f7d5d-p6gcq                        1/1     Running     0
1h
trident-svc-7cb5bb4685-54cnq                   1/1     Running     0
1h
trident-svc-7cb5bb4685-b28xh                   1/1     Running     0
1h
vault-controller-777b9bbf88-b5bqt              1/1     Running     0
1h
vault-controller-777b9bbf88-fdfd8              1/1     Running     0
1h
```

4. （オプション） `acc-operator` 進捗状況を監視するログ：

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

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

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

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

対応：

```
NAME    UUID                                   VERSION     ADDRESS
READY
astra   9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f  24.02.0-69
10.111.111.111  True
```

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

ロードバランシング用の入力を設定します

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

Astra Control Centerを導入したら、URLを使用してAstra Control Centerを公開するように入力コントローラを設定する必要があります。

セットアップ手順は、使用する入力コントローラのタイプによって異なります。Astra Control Centerは、多くの入力コントローラタイプをサポートしています。ここでは、一部の一般的な入力コントローラタイプの設定手順の例を示します。

作業を開始する前に

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

Istio Ingressの手順

1. Istio Ingressを設定します。

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

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

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

   コマンド例：

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

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

   コマンド例：

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

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

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

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

5. 変更を適用します。

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

6. 入力のステータスを確認します。

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

対応：

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

7. Astra Control Centerのインストールを完了します。

Nginx Ingress Controller の手順

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

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

3. 変更を適用します。

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

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

OpenShift 入力コントローラの手順

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

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

Astra Control Center UI にログインします

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

手順

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

   カスタム証明書はログイン後に作成できます。

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

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

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

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

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

インストールのトラブルシューティングを行います

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

オプション（ **Options** ）

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

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

- Astra Control Center CRの出力を確認するには、次の手順を実行します。

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

別のインストール手順

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

次のステップ

- （オプション）お使いの環境に応じて、インストール後に実行します "設定手順"。
- "Astra Control Centerをインストールし、UIにログインしてパスワードを変更したら、ライセンスのセットアップ、クラスタの追加、認証の有効化、ストレージの管理、バケットの追加を行うことができます。"。

外部証明書マネージャを設定します

Kubernetesクラスタに証明書マネージャがすでに存在する場合は、Astra Control Centerで独自の証明書マネージャがインストールされないように、いくつかの前提条件となる手順を実行する必要があります。

手順

1. 証明書マネージャがインストールされていることを確認します。

```
kubectl get pods -A | grep 'cert-manager'
```

回答例：

```
cert-manager   essential-cert-manager-84446f49d5-sf2zd              1/1
Running    0      6d5h
cert-manager   essential-cert-manager-cainjector-66dc99cc56-9ldmt   1/1
Running    0      6d5h
cert-manager   essential-cert-manager-webhook-56b76db9cc-fjqrq      1/1
Running    0      6d5h
```

2. の証明書とキーのペアを作成します `astraAddress` FQDN：

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

回答例：

```
Generating a 2048 bit RSA private key
...................+++
.......................................+++
writing new private key to 'tls.key'
```

3. 以前に生成したファイルを使用してシークレットを作成します。

```
kubectl create secret tls selfsigned-tls --key tls.key --cert tls.crt -n
<cert-manager-namespace>
```

回答例：

```
secret/selfsigned-tls created
```

4. を作成します `ClusterIssuer` *とまったく同じ*のファイル。ただし、の名前空間の場所が含まれます `cert-manager` ポッドがインストールされます。

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: astra-ca-clusterissuer
  namespace: <cert-manager-namespace>
spec:
  ca:
    secretName: selfsigned-tls
```

```
kubectl apply -f ClusterIssuer.yaml
```

回答例：

```
clusterissuer.cert-manager.io/astra-ca-clusterissuer created
```

5. を確認します `ClusterIssuer` が正常に起動しました。 `Ready` はである必要があります `True` 次の手順に進む前に、次の手順

```
kubectl get ClusterIssuer
```

回答例：

```
NAME                        READY   AGE
astra-ca-clusterissuer      True    9s
```

6. を実行します "Astra Control Center のインストールプロセス"。があります "Astra Control CenterクラスタYAMLの必須の設定手順" CRD値を変更して、証明書マネージャが外部にインストールされていることを示します。Astra Control Centerが外部証明書マネージャを認識するように、インストール時にこの手順を完了する必要があります。

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。

作業を開始する前に

- 環境条件を満たしている： "インストールを開始する前に、 Astra Control Center の導入環境を準備します"。

> 3つ目の障害ドメインまたはセカンダリサイトにAstra Control Centerを導入これは、アプリケーションのレプリケーションとシームレスなディザスタリカバリに推奨されます。

- 正常なクラスタオペレータと**API**サービスを確保：
  - OpenShiftクラスタから、すべてのクラスタオペレータが正常な状態にあることを確認します。

```
oc get clusteroperators
```

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

```
oc get apiservices
```

- *ルーティング可能なFQDN *：使用するAstra FQDNをクラスタにルーティングできることを確認します。つまり、内部 DNS サーバに DNS エントリがあるか、すでに登録されているコア URL ルートを使用しています。
- * OpenShiftの権限を取得する*:説明されているインストール手順を実行するには、Red Hat OpenShift Container Platformに必要なすべての権限とアクセス権が必要です。
- 証明書マネージャの設定:クラスタに証明書マネージャがすでに存在する場合は、一部の証明書マネージャを実行する必要があります。 "事前に必要な手順" そのため、Astra Control Centerは独自の証明書管理ツールをインストールしません。デフォルトでは、Astra Control Centerはインストール時に独自の証明書マネ

ージャをインストールします。

- * Kubernetes Ingressコントローラのセットアップ*：クラスタ内のロードバランシングなど、サービスへの外部アクセスを管理するKubernetes Ingressコントローラがある場合は、Astra Control Centerで使用するようにセットアップする必要があります。
  a. operatorネームスペースを作成します。

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

  b. "セットアップを完了" 入力コントローラのタイプ。
- （**ONTAP SAN**ドライバのみ）マルチパスの有効化：ONTAP SANドライバを使用している場合は、すべてのKubernetesクラスタでマルチパスが有効になっていることを確認してください。

また、次の点も考慮する必要があります。

- * NetApp Astra Controlイメージレジストリへのアクセス*：

  Astra Control Provisionerなど、Astra Controlのインストールイメージや機能強化された機能をNetAppイメージレジストリから取得することができます。

  a. レジストリへのログインに必要なAstra ControlアカウントIDを記録します。

     アカウントIDはAstra Control Service Web UIで確認できます。ページ右上の図アイコンを選択し、* APIアクセス*を選択して、アカウントIDを書き留めます。

  b. 同じページから* APIトークンの生成*を選択し、APIトークン文字列をクリップボードにコピーしてエディターに保存します。

  c. Astra Controlレジストリにログインします。

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

- セキュアな通信のためのサービスメッシュをインストール：Astra Controlホストクラスタの通信チャネルは、 "サポートされるサービスメッシュ"。

> Astra Control Centerとサービスメッシュの統合は、Astra Control Centerでのみ実行可能 "インストール" そしてこのプロセスから独立していません。メッシュ環境から非メッシュ環境に戻すことはサポートされていません。

  Istioサービスメッシュを使用するには、次の手順を実行する必要があります。

  - を追加します。 `istio-injection:enabled` Astra Control Centerを導入する前に、Astraネームスペースにラベルを付けます。
  - を使用します `Generic` 入力設定 別のイングレスを提供します。 "外部ロードバランシング"。
  - Red Hat OpenShiftクラスタの場合は、 `NetworkAttachmentDefinition` 関連付けられているすべてのAstra Control Centerネームスペース (`netapp-acc-operator`、 `netapp-acc`、 `netapp-monitoring` アプリケーションクラスタの場合、または置換されたカスタムネームスペースの場合）。

```
cat <<EOF | oc -n netapp-acc-operator create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-acc create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-monitoring create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF
```

手順

- Astra Control Centerをダウンロードして展開します
- [ローカルレジストリを使用する場合は、追加の手順を実行します。]
- [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オペレータ（たとえば、 `kubectl delete -f astra_control_center_operator_deploy.yaml`）Astra Control Centerのインストール中または操作中はいつでも、ポッドを削除しないようにします。

Astra Control Centerをダウンロードして展開します

次のいずれかの場所からAstra Control Centerのイメージをダウンロードします。

- * Astra Controlサービスのイメージレジストリ*：Astra Control Centerのイメージでローカルレジストリを使用しない場合や、NetApp Support Siteからバンドルをダウンロードするよりもこの方法を使用する場合は、このオプションを使用します。
- * NetApp Support Site *：このオプションは、Astra Control Centerのイメージでローカルレジストリを使用する場合に使用します。

Astra Controlイメージレジストリ

1. Astra Control Serviceにログインします。
2. ダッシュボードで、*[Deploy a self-managed instance of Astra Control]*を選択します。
3. 手順に従ってAstra Controlイメージのレジストリにログインし、Astra Control Centerのインストールイメージを取得してイメージを展開します。

NetApp Support Site

1. Astra Control Centerを含むバンドルをダウンロードします (`astra-control-center-[version].tar.gz`）をクリックします "Astra Control Centerのダウンロードページ"。
2. （推奨ですがオプション）Astra Control Centerの証明書と署名のバンドルをダウンロードします (`astra-control-center-certs-[version].tar.gz`）をクリックして、バンドルのシグネチャを確認します。

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

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

出力にはと表示されます `Verified OK` 検証が成功したあとに、

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

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

ローカルレジストリを使用する場合は、追加の手順を実行します。

Astra Control Centerバンドルをローカルのレジストリにプッシュする場合は、NetApp Astra kubectlコマンドラインプラグインを使用する必要があります。

ネットアップ**Astra kubectl**プラグインをインストール

次の手順を実行して、最新のNetApp Astra kubectlコマンドラインプラグインをインストールします。

作業を開始する前に

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

以前のインストールからプラグインがインストールされている場合は、 "最新バージョンがインストールされていることを確認してください" これらの手順を実行する前に。

手順

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

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

   ```
   ls kubectl-astra/
   ```

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

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

イメージをレジストリに追加する

1. Astra Control Centerバンドルをローカルのレジストリにプッシュする場合は、コンテナエンジンに応じた手順を実行します。

Docker です

a. tarballのルートディレクトリに移動します。次のように表示されます。 `acc.manifest.bundle.yaml` ファイルと次のディレクトリ：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. Astra Control Centerのイメージディレクトリにあるパッケージイメージをローカルレジストリにプッシュします。を実行する前に、次の置換を行ってください `push-images` コマンドを実行します

- <BUNDLE_FILE> をAstra Controlバンドルファイルの名前に置き換えます（`acc.manifest.bundle.yaml`）。
- <MY_FULL_REGISTRY_PATH> をDockerリポジトリのURLに置き換えます。次に例を示します。 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
- <MY_REGISTRY_USER> をユーザ名に置き換えます。
- <MY_REGISTRY_TOKEN> をレジストリの認証済みトークンに置き換えます。

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

ポドマン

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

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. レジストリにログインします。

```
podman login <YOUR_REGISTRY>
```

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

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

ⓘ レジストリ設定に応じて、スクリプトが作成するイメージパスは次のようになります。

```
https://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version
```

2. ディレクトリを変更します。

```
cd manifests
```

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。

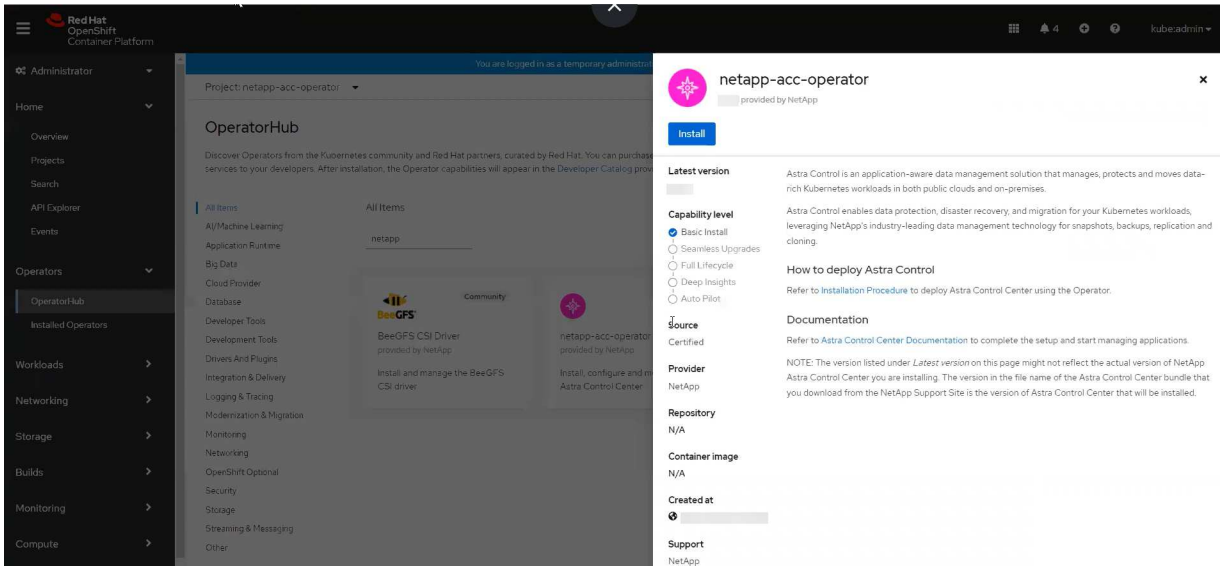
Red Hat OpenShift Webコンソール

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

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

> このオペレータを使用している場合は、Astra Control Centerの最新バージョンにのみアップグレードできます。

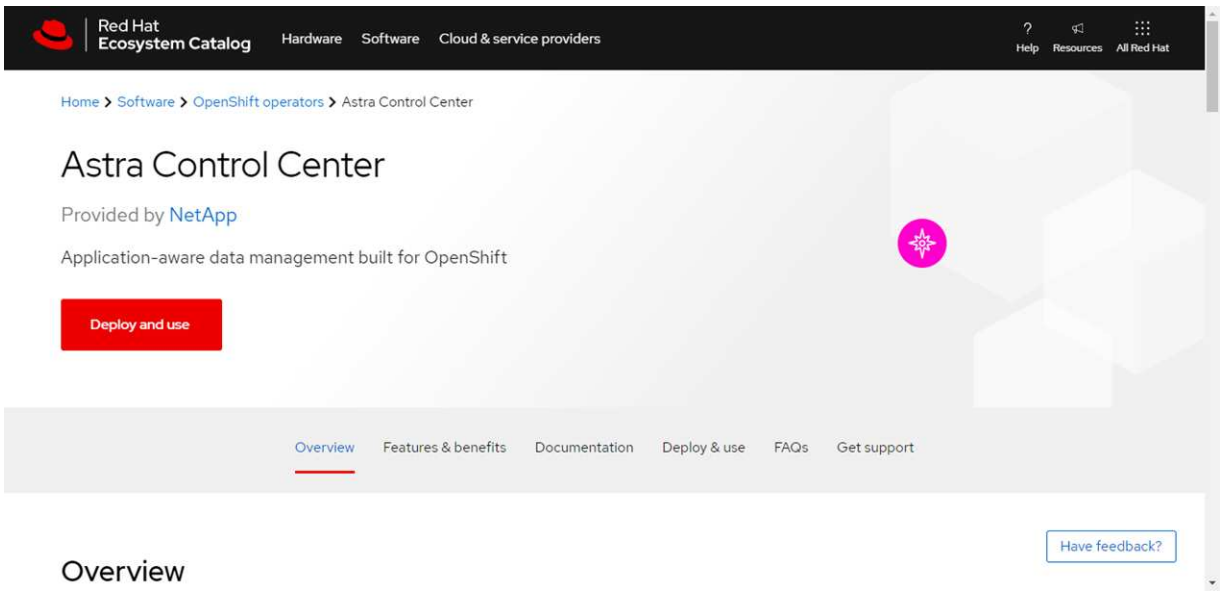
c. を検索します `netapp-acc` にアクセスし、NetApp Astra Control Centerオペレータを選択します。



Red Hat エコシステムカタログ

a. NetApp Astra Control Center を選択します "演算子"。

b. [Deploy and Use]*を選択します。



オペレータをインストールします

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

   オペレータはすべてのクラスタネームスペースで使用できます。

   a. operator名前空間またはを選択します `netapp-acc-operator` オペレータのインストールの一環として、名前空間が自動的に作成されます。

   b. 手動または自動の承認方法を選択します。

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

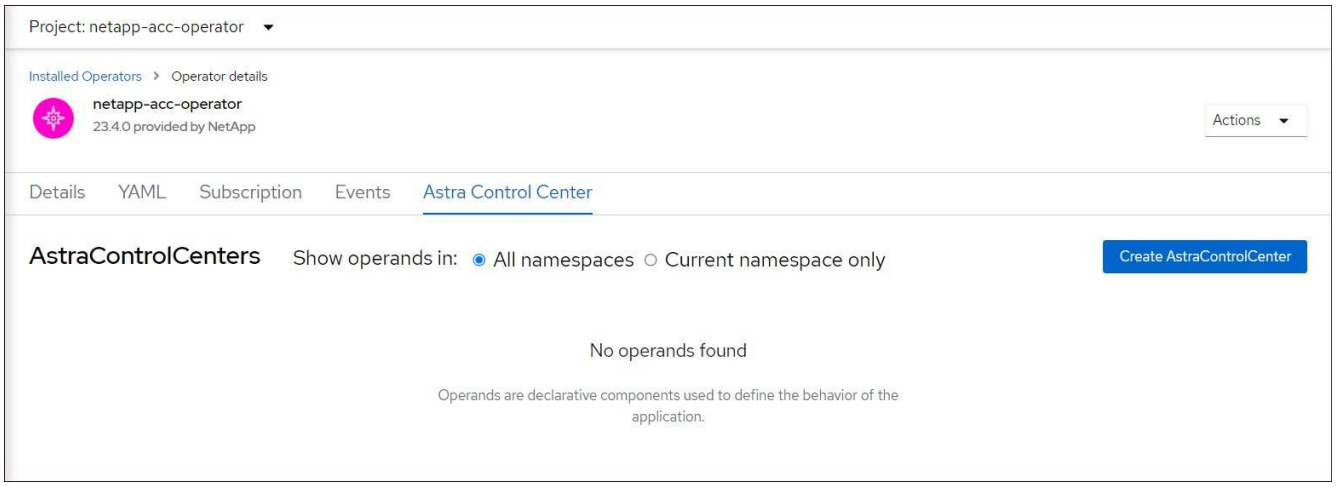
   c. 「 * Install * 」を選択します。

      手動承認ストラテジーを選択した場合は、このオペレーターの手動インストール計画を承認するように求められます。

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

Astra Control Center をインストールします

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



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

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

   b. Astra Control Centerのラベルを追加します。

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

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

   e. Astra Control Centerのバージョン（例：24.02.0-69）を入力します。

f. アカウント名、 E メールアドレス、および管理者の姓を入力します。

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

h. インストールのスケールサイズを選択します。

> デフォルトでは、Astraで高可用性（HA）が使用されます。 `scaleSize` の `Medium`ほとんどのサービスをHAに導入し、冗長性を確保するために複数のレプリカを導入します。を使用 `scaleSize` として `Small`Astraは、消費量を削減するための必須サービスを除き、すべてのサービスのレプリカ数を削減します。

i. 入力タイプを選択します。

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

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

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

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

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

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

a. * Image Registry*では、ローカルレジストリを構成していない限り、デフォルト値を使用します。ローカルレジストリの場合は、この値を、前の手順でイメージをプッシュしたローカルイメージレジストリパスに置き換えます。入らないでください `http://` または `https://` をクリックします。

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

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

c. 管理者の名を入力します。

d. リソースの拡張を構成する。

e. デフォルトのストレージクラスを指定します。

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

f. CRD 処理の環境設定を定義します。

3. YAMLビューを選択して、選択した設定を確認します。
4. 選択するオプション `Create`。

レジストリシークレットを作成します

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

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

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

2. この名前空間にシークレットを作成します。

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

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

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

次のステップ

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

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

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

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

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

- Amazon Web Services に Astra Control Center を導入

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

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

Amazon Web Services に Astra Control Center を導入

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

AWSに必要なもの

AWSにAstra Control Centerを導入する前に、次の項目が必要になります。

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

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 4.11~4.13

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

Astra Control Centerでは、環境のリソース要件に加えて、特定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"。

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

AWS の導入の概要を参照してください

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

これらの各手順については、以下で詳しく説明します。

1. 十分な IAM 権限があることを確認します。

2. AWS に Red Hat OpenShift クラスタをインストールします。
3. AWS を設定します。
4. NetApp BlueXP for AWSを構成します。
5. Astra Control Center for AWSをインストール。

十分な **IAM** 権限があることを確認します

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

を参照してください "AWS の初期クレデンシャル"。

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

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

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

AWS を設定します

次に、仮想ネットワークを作成するようにAWSを設定し、EC2コンピューティングインスタンスをセットアップし、AWS S3バケットを作成します。NetApp Astra Control Centerのイメージレジストリにアクセスできない場合は、Astra Control Centerのイメージをホストする Elastic Container Registry（ECR）を作成し、イメージをこのレジストリにプッシュする必要もあります。

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"。

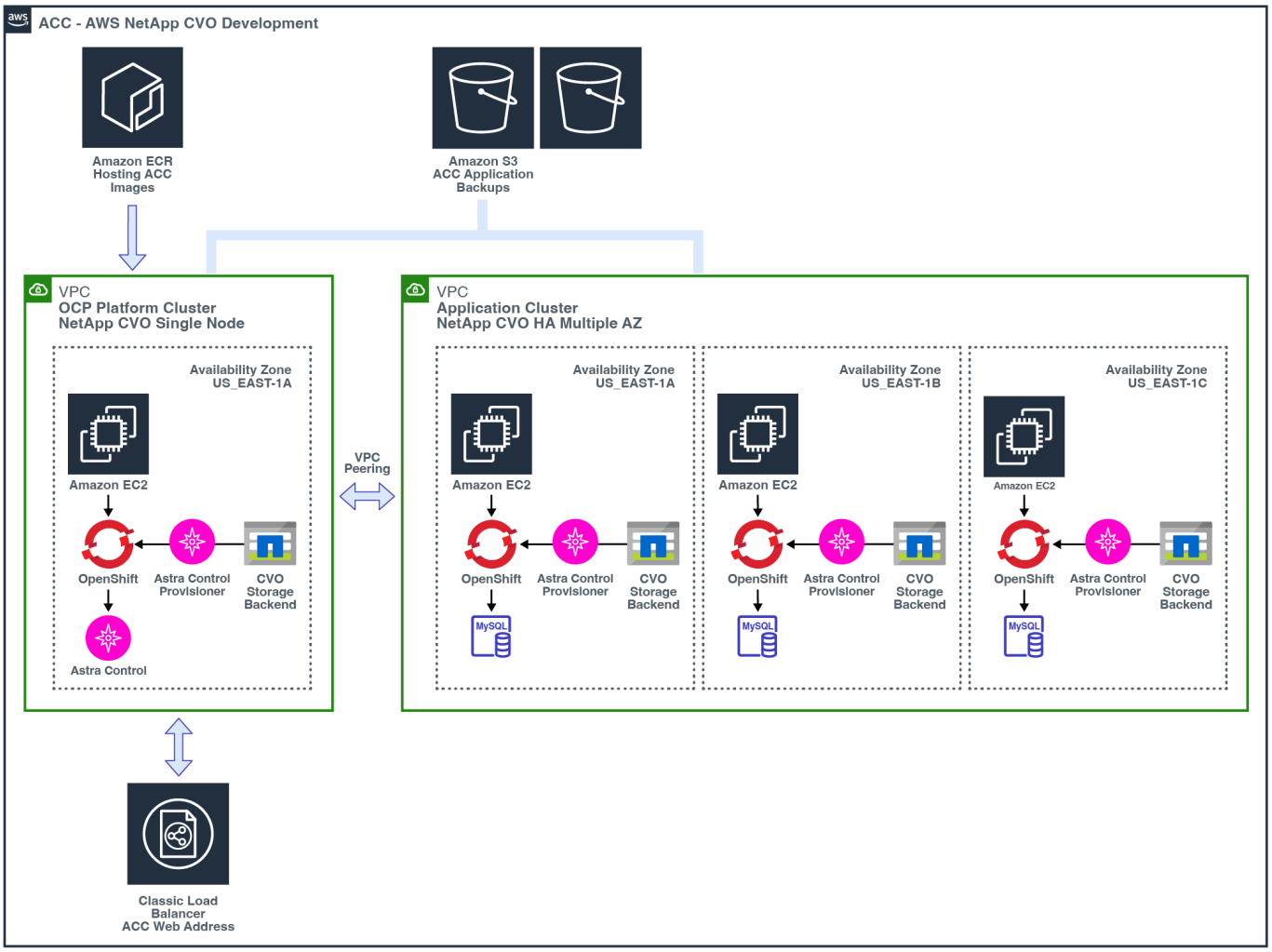
1. AWS仮想ネットワークを作成します。
2. EC2 コンピューティングインスタンスを確認します。AWS ではベアメタルサーバまたは VM を使用できます。
3. インスタンスタイプが、マスターノードとワーカーノードの Astra の最小リソース要件に一致していない場合は、 Astra の要件に合わせて AWS でインスタンスタイプを変更します。 を参照してください "Astra Control Center の要件"。
4. バックアップを格納する AWS S3 バケットを少なくとも 1 つ作成します。
5. （オプション）NetAppイメージレジストリにアクセスできない場合は、次の手順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

      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。

   b. Astra Control Centerのイメージを定義済みのレジストリにプッシュ

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

AWS 環境の例を次に示します。



NetApp BlueXP for AWSを構成します

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

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

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

手順

1. 資格情報をBlueXPに追加します。
2. ワークスペースを作成します。

3. AWS 用のコネクタを追加します。プロバイダとして AWS を選択します。
4. クラウド環境の作業環境を構築
   a. 場所：「 Amazon Web Services （ AWS ）」
   b. 「 Cloud Volumes ONTAP HA 」と入力します。
5. OpenShift クラスタをインポートします。作成した作業環境にクラスタが接続されます。
   a. ネットアップクラスタの詳細を表示するには、 * K8s * > * Cluster list * > * Cluster Details * を選択します。
   b. 右上にあるAstra Control Provisionerのバージョンを確認します。
   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。
      Astra Control Provisionerは、インポートと検出のプロセスで自動的にインストールされます。
6. このCloud Volumes ONTAP 環境内のすべての永続ボリュームとボリュームをメモします。

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

Astra Control Center for AWSをインストール

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

AWSでは汎用のS3バケットタイプが使用されます。

Astra Control CenterをGoogle Cloud Platformに導入

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

GCPに必要なもの

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

- Astra Control Center ライセンス。を参照してください "Astra Control Center のライセンス要件"。
- "Astra Control Center の要件を満たす"。
- NetApp Cloud Central アカウント
- OCPを使用している場合、Red Hat OpenShift Container Platform（OCP）4.11~4.13
- OCPを使用する場合は、Red Hat OpenShift Container Platform（OCP）権限（ポッドを作成するためのネームスペースレベル）
- バケットとコネクタの作成を可能にする権限を持つGCPサービスアカウント

GCPの運用環境の要件

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

Astra Control Centerでは、環境のリソース要件に加えて、特定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"。

GCPの導入の概要

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

これらの各手順については、以下で詳しく説明します。

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

GCPにRed Hat OpenShiftクラスタをインストールします

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

インストール手順については、次を参照してください。

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

GCPプロジェクトとVirtual Private Cloudを作成します

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

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

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

十分な IAM 権限があることを確認します

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

を参照してください "GCPの初期資格情報と権限"。

GCPを設定します

次に、GCPを設定してVPCを作成し、コンピューティングインスタンスをセットアップし、Google Cloud Object Storageを作成します。NetApp Astra Control Centerのイメージレジストリにアクセスできない場合は、Astra Control Centerのイメージをホストするコンテナレジストリを作成し、このレジストリにイメージをプッシュする必要もあります。

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

1. GCPでGCPプロジェクトとVPCを作成します。GCPでは、CVOバックエンドでOCPクラスタ用にを使用する予定です。
2. コンピューティングインスタンスを確認します。GCP内のベアメタルサーバまたはVMです。
3. インスタンスタイプが、マスターノードとワーカーノードのAstra最小リソース要件と一致していない場合は、GCPでインスタンスタイプを変更してAstraの要件を満たします。を参照してください "Astra Control Center の要件"。
4. バックアップを保存するGCP Cloud Storageバケットを少なくとも1つ作成します。
5. バケットへのアクセスに必要なシークレットを作成します。
6. （オプション）NetAppイメージレジストリにアクセスできない場合は、次の手順を実行します。

   a. Astra Control Centerのイメージをホストする Google Container Registryを作成します。

   b. すべてのAstra Control Centerイメージに対して、Dockerプッシュ/プル用のGoogle Container Registryアクセスを設定します。

      例：次のスクリプトを入力して、Astra Control Centerのイメージをこのレジストリにプッシュできます。

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

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

```
manifestfile=acc.manifest.bundle.yaml
GCP_CR_REGISTRY=<target GCP image registry>
ASTRA_REGISTRY=<source Astra Control Center image registry>

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

7. DNS ゾーンを設定します。

NetApp BlueXP for GCPを構成します

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

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

作業を開始する前に

- 必要なIAM権限と役割を持つGCPサービスアカウントにアクセスします

手順

1. 資格情報をBlueXPに追加します。を参照してください "GCP アカウントの追加"。
2. GCPのコネクターを追加します。
   a. プロバイダーとして[GCP]を選択します。
   b. GCP資格情報を入力します。を参照してください "BlueXPからGCPでコネクタを作成する"。
   c. コネクタが動作していることを確認し、コネクタに切り替えます。
3. クラウド環境の作業環境を構築
   a. 場所："GCP"
   b. 「 Cloud Volumes ONTAP HA 」と入力します。
4. OpenShift クラスタをインポートします。作成した作業環境にクラスタが接続されます。
   a. ネットアップクラスタの詳細を表示するには、 * K8s * > * Cluster list * > * Cluster Details * を選択します。
   b. 右上にあるAstra Control Provisionerのバージョンを確認します。
   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。
      Astra Control Provisionerは、インポートと検出のプロセスで自動的にインストールされます。
5. このCloud Volumes ONTAP 環境内のすべての永続ボリュームとボリュームをメモします。

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

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

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

> GCPでは汎用S3バケットタイプが使用されます。

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

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

Microsoft Azure に Astra Control Center を導入

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

Azureに必要なもの

AzureにAstra Control Centerを導入する前に、次の項目が必要になります。

- Astra Control Center ライセンス。を参照してください "Astra Control Center のライセンス要件"。
- "Astra Control Center の要件を満たす"。
- NetApp Cloud Central アカウント
- OCPを使用している場合、Red Hat OpenShift Container Platform（OCP）4.11~4.13
- OCPを使用する場合は、Red Hat OpenShift Container Platform（OCP）権限（ポッドを作成するためのネームスペースレベル）
- バケットとコネクタの作成を可能にする権限を持つ Azure クレデンシャル

Azure の運用環境の要件

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

Astra Control Centerでは、環境のリソース要件に加えて、特定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"。

Azure の導入の概要

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

これらの各手順については、以下で詳しく説明します。

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

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

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

インストール手順については、次を参照してください。

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

Azure リソースグループを作成する

Azure リソースグループを少なくとも 1 つ作成します。

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

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

十分な IAM 権限があることを確認します

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

を参照してください "Azure のクレデンシャルと権限"。

Azure を設定

次に、仮想ネットワークを作成し、コンピューティングインスタンスをセットアップし、Azure Blobコンテナを作成するようにAzureを設定します。NetApp Astra Control Centerのイメージレジストリにアクセスできない場合は、Astra Control Centerのイメージをホストする Azure Container Registry（ACR）を作成し、イメージをこのレジストリにプッシュする必要もあります。

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. （オプション）NetAppイメージレジストリにアクセスできない場合は、次の手順を実行します。
    a. Astra Control Centerのイメージをホストする Azure Container Registry（ACR）を作成します。

b. Astra Control Centerのすべてのイメージに対して、Dockerによるプッシュ/プルのACRアクセスをセットアップします。

c. 次のスクリプトを使用して、Astra Control Centerの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 the Astra Control Center manifest file and your
Azure ACR location.
```

- 例 * ：

```
manifestfile=acc.manifest.bundle.yaml
AZ_ACR_REGISTRY=<target Azure ACR image registry>
ASTRA_REGISTRY=<source Astra Control Center image registry>

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 $AZ_ACR_REGISTRY/$image
    docker push $AZ_ACR_REGISTRY/$image
done < acc.manifest.bundle.yaml
```

8. DNS ゾーンを設定します。

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

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

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

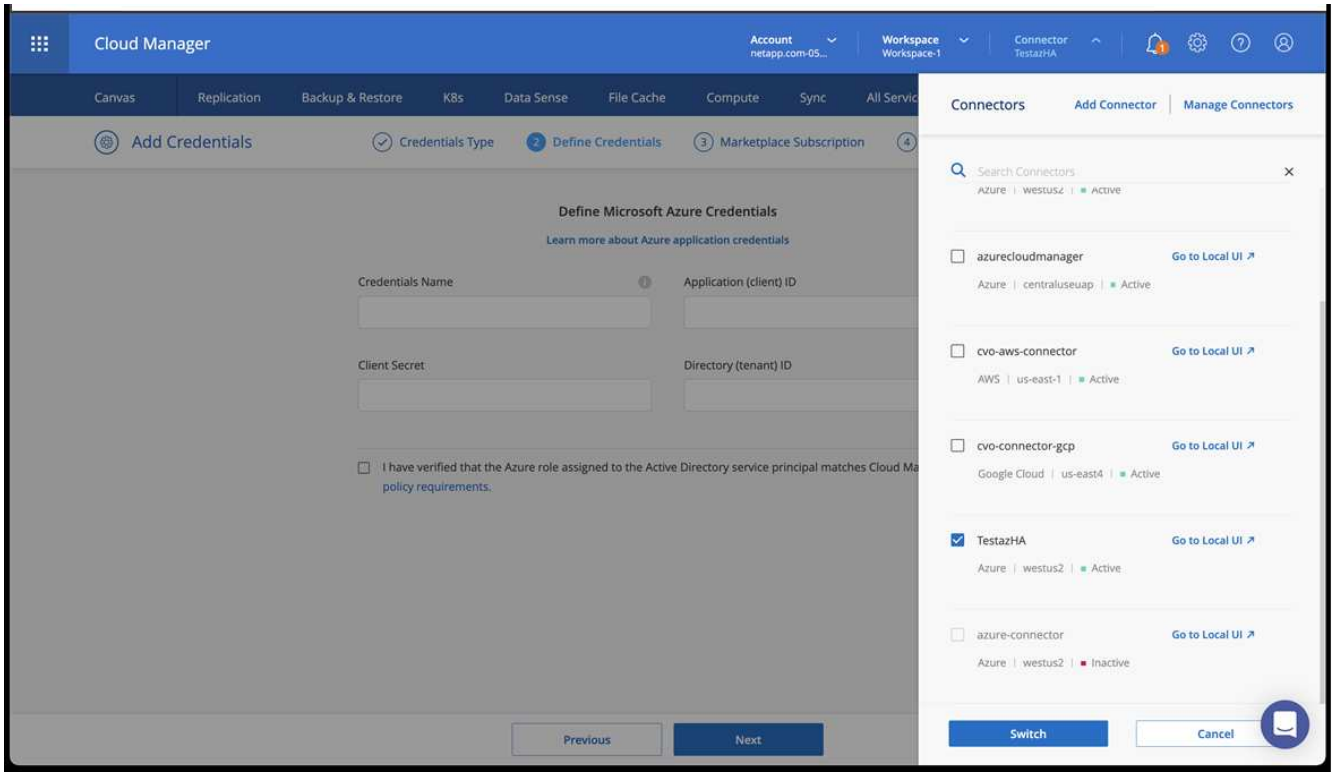
作業を開始する前に

必要な IAM 権限とロールを持つ Azure アカウントにアクセスします

手順

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

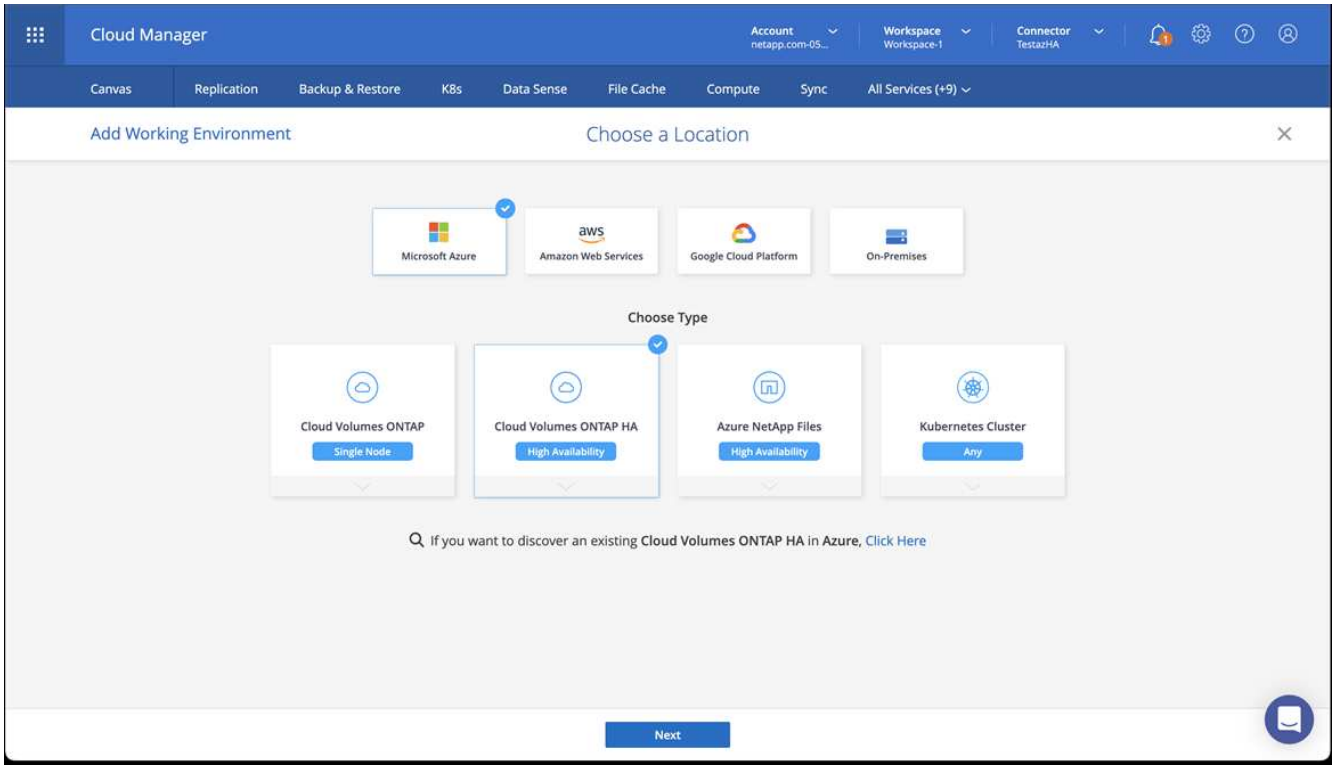
      を参照してください "BlueXPrからAzureでコネクタを作成しています"。
3. コネクタが動作していることを確認し、コネクタに切り替えます。



4. クラウド環境の作業環境を構築

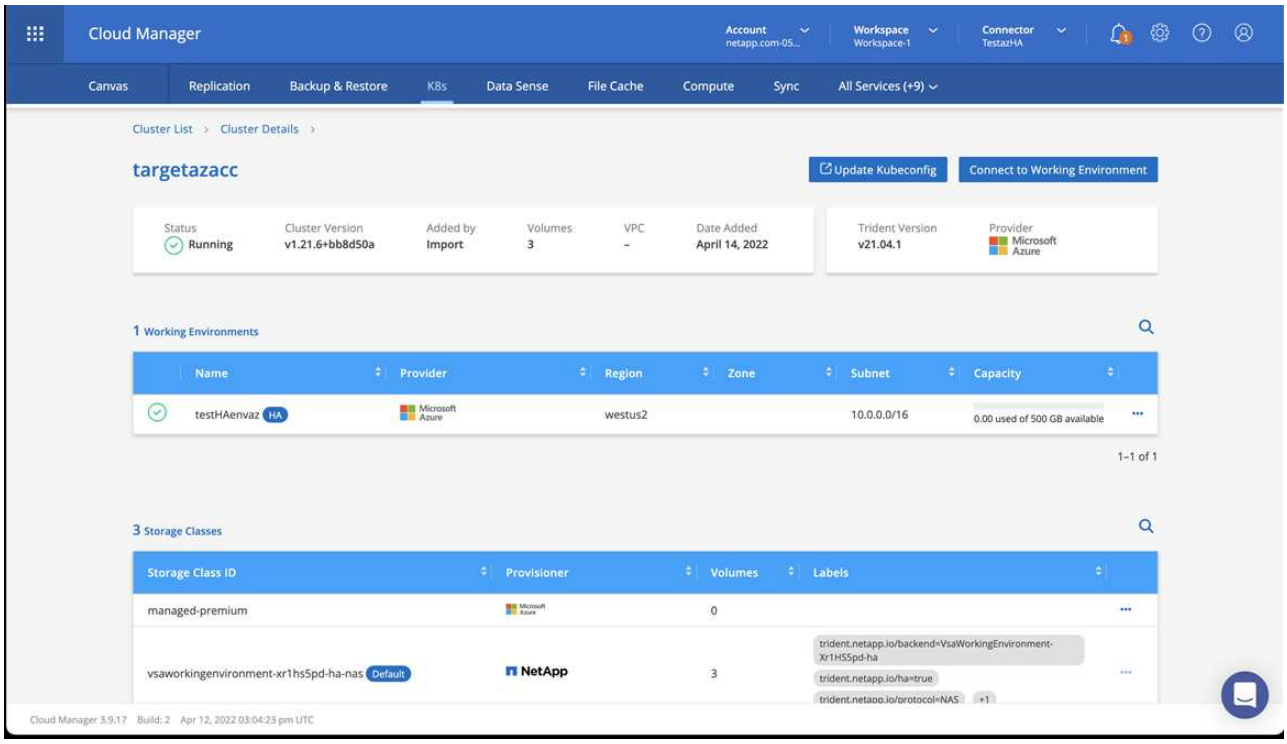
   a. 場所：「 Microsoft Azure 」。

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



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

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



b. 右上にあるAstra Control Provisionerのバージョンを確認します。

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。
Astra Control Provisionerは、インポートと検出のプロセスで自動的にインストールされます。

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

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

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

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

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

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

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

リソースの制限を解除します

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

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

手順

1. でリソースクォータを取得します `netapp-acc` （またはカスタム名）ネームスペース：

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

対応：

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

2. 名前別にすべてのリソースクォータを削除します。

```
kubectl delete resourcequota pods-high -n [netapp-acc or custom
namespace]
```

```
kubectl delete resourcequota pods-low -n [netapp-acc or custom
namespace]
```

```
kubectl delete resourcequota pods-medium -n [netapp-acc or custom
namespace]
```

3. で制限範囲を取得します `netapp-acc` （またはカスタム名）ネームスペース：

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

対応：

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

4. 制限範囲を名前で削除します。

```
kubectl delete limitrange cpu-limit-range -n [netapp-acc or custom
namespace]
```

カスタム **TLS** 証明書を追加します

Astra Control Centerは、入力コントローラトラフィック（一部の設定のみ）およびWebブラウザでのWeb UI認証に、デフォルトで自己署名TLS証明書を使用します。本番環境で使用する場合は、既存の自己署名TLS証明書を削除して、認証局（CA）によって署名されたTLS証明書に置き換える必要があります。

デフォルトの自己署名証明書は、次の2種類の接続に使用されます。

- Astra Control Center Web UIへのHTTPS接続
- 入力コントローラトラフィック（がの場合のみ） `ingressType: "AccTraefik"` プロパティはで設定されました `astra_control_center.yaml` Astra Control Centerのインストール中にファイルを作成）

これらの接続の認証に使用される証明書は、デフォルトのTLS証明書に置き換えられます。

作業を開始する前に

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

自己署名証明書を削除します

既存の自己署名 TLS 証明書を削除します。

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

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

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

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

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

コマンドラインを使用して新しい証明書を追加します

CA によって署名された新しい TLS 証明書を追加します。

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

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

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

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

CRD：

```
#spec:
#  selfSigned: {}

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

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

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

対応：

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

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

> この例では、 `dnsNames` Astra Control CenterのDNSアドレスを指定するプロパティ。Astra Control Centerでは、Common Name（CN）プロパティを使用したDNSアドレスの指定はサポートされていません。

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  <strong>name: <certificate-name></strong>
  namespace: <ACC-deployment-namespace>
spec:
  <strong>secretName: <certificate-secret-name></strong>
  duration: 2160h # 90d
  renewBefore: 360h # 15d
  dnsNames:
  <strong>- <astra.dnsname.example.com></strong> #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>
```

対応：

```
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. 次のコマンドと例を使用してTLS Stores CRDを編集し、括弧<>のプレースホルダ値を適切な情報に置き換えます。

```
kubectl edit tlsstores.traefik.io -n <ACC-deployment-namespace>
```

CRD：

```
...
spec:
  defaultCertificate:
    secretName: <certificate-secret-name>
```

8. 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.io -n <ACC-deployment-namespace>
```

CRD：

```
...
 tls:
    secretName: <certificate-secret-name>
```

9. 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。
10. 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。
11. 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。

Astra Control Center をセットアップします

Astra Control Center のライセンスを追加します

Astra Control Centerをインストールすると、組み込みの評価用ライセンスがすでにインストールされています。Astra Control Centerを評価する場合は、この手順を省略できます。

新しいライセンスは、Astra Control UIまたはを使用して追加できます "Astra Control API の略"。

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

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

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

作業を開始する前に

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

手順

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

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

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

Astra Control Provisionerを有効にする

Astra Tridentバージョン23.10以降には、Astra Control Provisionerを使用するオプションが用意されています。このオプションを使用すると、ライセンスを取得したAstra Controlユーザは、高度なストレージプロビジョニング機能にアクセスできます。Astra Control Provisionerは、Astra Trident CSIベースの標準機能に加えて、この拡張機能を提供します。

今後のAstra Controlの更新では、Astra Control ProvisionerがAstra Tridentの代わりにストレージプロビジョニングツールおよびオーケストレータとして使用され、Astra Controlでは必須となります。そのため、Astra ControlのユーザはAstra Control Provisionerを有効にすることを強く推奨します。Astra Tridentは引き続きオープンソースであり、NetAppの新しいCSIやその他の機能でリリース、メンテナンス、サポート、更新されます。

このタスクについて

Astra Control Centerのライセンスを取得していて、Astra Control Provisioner機能を使用する場合は、この手順に従う必要があります。また、Astra Tridentを使用していて、Astra Control Provisionerが提供する追加機能をAstra Controlを使用せずに使用する場合も、この手順に従う必要があります。

どちらの場合も、Astra Trident 24.02ではプロビジョニングツール機能はデフォルトでは有効になっていないため、有効にする必要があります。

作業を開始する前に

Astra Control Provisionerを有効にする場合は、まず次の手順を実行します。

Astra Controlプロビジョニングツールユーザと**Astra Control Center**

- * Astra Control Centerライセンスの取得*： "Astra Control Centerのライセンス" Astra Control Provisionerを有効にし、提供される機能にアクセスするため。
- * Astra Control Center 23.10以降をインストールまたはアップグレード*：Astra Control Provisionerの最新機能（24.02）をAstra Controlで使用する場合は、最新のAstra Control Centerバージョン（24.02）が必要です。
- クラスタに**AMD64**システムアーキテクチャがあることを確認する：Astra Control ProvisionerイメージはAMD64とARM64の両方のCPUアーキテクチャで提供されますが、Astra Control Centerでサポートされるのはamd64のみです。
- レジストリアクセス用の**Astra Control**サービスアカウントを取得：NetApp Support Siteではなく Astra Controlレジストリを使用してAstra Control Provisionerイメージをダウンロードする場合は、"Astra Controlサービスのアカウント"。フォームに必要事項を入力して送信し、BlueXPアカウントを作成すると、Astra Control ServiceのようこそEメールが届きます。
- * Astra Tridentがインストールされている場合は、バージョンが4リリース期間内であることを確認してください*：Astra Tridentがバージョン24.02の4リリース期間内であれば、Astra Control Provisionerを使用してAstra Trident 24.02への直接アップグレードを実行できます。たとえば、Astra Trident 23.04から24.02に直接アップグレードできます。

Astra Control Provisionerのみユーザ

- * Astra Control Centerライセンスの取得*： "Astra Control Centerのライセンス" Astra Control Provisionerを有効にし、提供される機能にアクセスするため。
- * Astra Tridentがインストールされている場合は、バージョンが4リリース期間内であることを確認してください*：Astra Tridentがバージョン24.02の4リリース期間内であれば、Astra Control Provisionerを使用してAstra Trident 24.02への直接アップグレードを実行できます。たとえば、Astra Trident 23.04から24.02に直接アップグレードできます。
- レジストリアクセス用の**Astra Control**サービスアカウントを取得：Astra Control Provisionerイメージをダウンロードするには、レジストリへのアクセスが必要です。使用を開始するには、 "Astra Controlサービスのアカウント"。フォームに必要事項を入力して送信し、BlueXPアカウントを作成すると、Astra Control ServiceのようこそEメールが届きます。

（手順1）**Astra Control Provisioner**の画像を取得

Astra Control Centerのユーザは、Astra ControlのレジストリまたはNetApp Support SiteメソッドをAstra Control Provisionerイメージを取得できます。Astra Controlを使用せずにAstra Control Provisionerを使用する場合は、レジストリ方式を使用する必要があります。

Astra Controlイメージレジストリ

この手順のコマンドには、Dockerの代わりにPodmanを使用できます。Windows環境を使用している場合は、PowerShellを推奨します。

1. NetApp Astra Controlイメージのレジストリにアクセスします。
   a. Astra Control Service Web UIにログオンし、ページの右上にある図アイコンを選択します。
   b. [API access*]を選択します。
   c. アカウントIDを書き留めます。
   d. 同じページから* APIトークンの生成*を選択し、APIトークン文字列をクリップボードにコピーしてエディターに保存します。
   e. 任意の方法でAstra Controlレジストリにログインします。

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

```
crane auth login cr.astra.netapp.io -u <account-id> -p <api-token>
```

2. (カスタムレジストリのみ)イメージをカスタムレジストリに移動するには、次の手順に従います。レジストリを使用していない場合は、 "次のセクション"。
   a. Astra Control Provisionerのイメージをレジストリから取得します。

プルされたイメージは複数のプラットフォームをサポートせず、Linux AMD64など、イメージをプルしたホストと同じプラットフォームのみをサポートします。

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0 --platform <cluster platform>
```

例

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0 --platform linux/amd64
```

   a. 画像にタグを付けます。

```
docker tag cr.astra.netapp.io/astra/trident-acp:24.02.0 <my_custom_registry>/trident-acp:24.02.0
```

b. イメージをカスタムレジストリにプッシュします。

```
docker push <my_custom_registry>/trident-acp:24.02.0
```

次のDockerコマンドを実行する代わりに、クレーンコピーを使用できます。
`crane copy cr.astra.netapp.io/astra/trident-acp:24.02.0 <my_custom_registry>/trident-acp:24.02.0`

NetApp Support Site

1. Astra Control Provisionerバンドルをダウンロード (`trident-acp-[version].tar`）をクリックします "Astra Control Centerのダウンロードページ"。
2. （推奨、ただしオプション）Astra Control Centerの証明書とシグネチャバンドル（astra-control-center-certs-[version].tar.gz）をダウンロードして、trident-acp-[version] tarバンドルのシグネチャを確認します。

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

```
openssl dgst -sha256 -verify certs/AstraControlCenterDockerImages-public.pub -signature certs/trident-acp-[version].tar.sig trident-acp-[version].tar
```

3. Astra Control Provisionerのイメージをロードします。

```
docker load < trident-acp-24.02.0.tar
```

対応：

```
Loaded image: trident-acp:24.02.0-linux-amd64
```

4. 画像にタグを付けます。

```
docker tag trident-acp:24.02.0-linux-amd64 <my_custom_registry>/trident-acp:24.02.0
```

5. イメージをカスタムレジストリにプッシュします。

```
docker push <my_custom_registry>/trident-acp:24.02.0
```

（ステップ2）Astra TridentでAstra Control Provisionerを有効にする

元のインストール方法で "オペレータ（手動またはHelmを使用）またはtridentctl" そして、元の方法に従って適切な手順を完了します。

Astra Trident運用者

1. "Astra Tridentインストーラをダウンロードして展開"。
2. Astra Tridentをまだインストールしていない場合、または元のAstra Trident環境からオペレータを削除した場合は、次の手順を実行します。
   a. CRDを作成します。

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.y
aml
```

   b. Tridentネームスペースを作成 (`kubectl create namespace trident`）またはTridentネームスペースがまだ存在することを確認 (`kubectl get all -n trident`）。ネームスペースが削除されている場合は、もう一度作成します。
3. Astra Tridentを24.02.0に更新：

> クラスタでKubernetes 1.24以前を実行している場合は、を使用します `bundle_pre_1_25.yaml`。クラスタでKubernetes 1.25以降を実行している場合は、を使用します `bundle_post_1_25.yaml`。

```
kubectl -n trident apply -f trident-installer/deploy/<bundle-
name.yaml>
```

4. Astra Tridentが実行されていることを確認します。

```
kubectl get torc -n trident
```

   対応：

```
NAME      AGE
trident   21m
```

5. [pull-secrets]]シークレットを使用するレジストリがある場合は、Astra Control Provisionerイメージの取得に使用するシークレットを作成します。

```
kubectl create secret docker-registry <secret_name> -n trident
--docker-server=<my_custom_registry> --docker-username=<username>
--docker-password=<token>
```

6. TridentOrchestrator CRを編集し、次の編集を行います。

```
kubectl edit torc trident -n trident
```

a. Astra Tridentイメージのカスタムレジストリの場所を設定するか、Astra Controlレジストリから取得 (`tridentImage: <my_custom_registry>/trident:24.02.0` または `tridentImage: netapp/trident:24.02.0`）。

b. Astra Control Provisionerを有効にする (`enableACP: true`）。

c. Astra Control Provisionerイメージのカスタムレジストリの場所を設定するか、Astra Controlレジストリから取得 (`acpImage: <my_custom_registry>/trident-acp:24.02.0` または `acpImage: cr.astra.netapp.io/astra/trident-acp:24.02.0`）。

d. もしあなたが 画像プルシークレット この手順では、ここで設定できます。(`imagePullSecrets: - <secret_name>`）。前の手順で設定した名前と同じシークレット名を使用します。

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  tridentImage: <registry>/trident:24.02.0
  enableACP: true
  acpImage: <registry>/trident-acp:24.02.0
  imagePullSecrets:
  - <secret_name>
```

7. ファイルを保存して終了します。導入プロセスが自動的に開始されます。

8. operator、deployment、およびReplicaSetsが作成されていることを確認します。

```
kubectl get all -n trident
```

Kubernetes クラスタには、オペレータのインスタンスが * 1 つしか存在しないようにしてください。Astra Tridentオペレータを複数の環境に導入することは避けてください。

9. を確認します `trident-acp` コンテナが実行中で、 `acpVersion` はです `24.02.0` ステータス：`Installed`：

```
kubectl get torc -o yaml
```

対応：

```
status:
  acpVersion: 24.02.0
  currentInstallationParams:
    ...
    acpImage: <registry>/trident-acp:24.02.0
    enableACP: "true"
    ...
  ...
  status: Installed
```

Tridentctl

1. "Astra Tridentインストーラをダウンロードして展開"。
2. "既存のAstra Tridentがある場合は、そのTridentをホストしているクラスタからアンインストール"。
3. Astra Control Provisionerを有効にしてAstra Tridentをインストール (`--enable-acp=true`）：

```
./tridentctl -n trident install --enable-acp=true --acp
-image=mycustomregistry/trident-acp:24.02
```

4. Astra Control Provisionerが有効になっていることを確認します。

```
./tridentctl -n trident version
```

対応：

```
+----------------+----------------+-------------+ | SERVER VERSION |
CLIENT VERSION | ACP VERSION | +----------------+----------------
+-------------+ | 24.02.0 | 24.02.0 | 24.02.0. | +----------------
+----------------+-------------+
```

Helm

1. Astra Trident 23.07.1以前がインストールされている場合は、 "をアンインストールします" オペレータおよびその他のコンポーネント。
2. Kubernetesクラスタが1.24以前を実行している場合は、pspを削除します。

```
kubectl delete psp tridentoperatorpod
```

3. Astra Trident Helmリポジトリを追加します。

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-
chart
```

4. Helmチャートを更新します。

```
helm repo update netapp-trident
```

対応：

```
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "netapp-trident" chart
repository
Update Complete. □Happy Helming!□
```

5. 画像を一覧表示します。

```
./tridentctl images -n trident
```

対応：

```
| v1.28.0            | netapp/trident:24.02.0|
|                    | docker.io/netapp/trident-autosupport:24.02|
|                    | registry.k8s.io/sig-storage/csi-
provisioner:v4.0.0|
|                    | registry.k8s.io/sig-storage/csi-
attacher:v4.5.0|
|                    | registry.k8s.io/sig-storage/csi-
resizer:v1.9.3|
|                    | registry.k8s.io/sig-storage/csi-
snapshotter:v6.3.3|
|                    | registry.k8s.io/sig-storage/csi-node-driver-
registrar:v2.10.0 |
|                    | netapp/trident-operator:24.02.0 (optional)
```

6. trident-operator 24.02.0が使用可能であることを確認します。

```
helm search repo netapp-trident/trident-operator --versions
```

対応：

```
NAME                                CHART VERSION   APP VERSION
DESCRIPTION
netapp-trident/trident-operator 100.2402.0      24.02.0         A
```

7. 使用 `helm install` これらの設定を含む次のいずれかのオプションを実行します。
   - 導入場所の名前
   - Astra Tridentバージョン
   - Astra Control Provisionerの名前の画像
   - プロビジョニングツールを有効にするフラグ
   - （任意）ローカルレジストリパス。ローカルレジストリを使用している場合は、 "Tridentの画像" 1つのレジストリまたは別のレジストリに配置できますが、すべてのCSIイメージは同じレジストリに配置する必要があります。
   - Tridentネームスペース

オプション（ **Options** ）

- レジストリなしのイメージ

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=cr.astra.netapp.io/astra/trident-acp:24.02.0
--set enableACP=true --set operatorImage=netapp/trident-
operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

- 1つまたは複数のレジストリ内の画像

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=<your-registry>:<acp image> --set
enableACP=true --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=netapp/trident-operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

を使用できます `helm list` 名前、ネームスペース、グラフ、ステータス、アプリケーションバージョンなどのインストールの詳細を確認するには、次の手順を実行します。とリビジョン番号。

Helmを使用したTridentの導入で問題が発生した場合は、次のコマンドを実行してAstra Tridentを完全にアンインストールします。

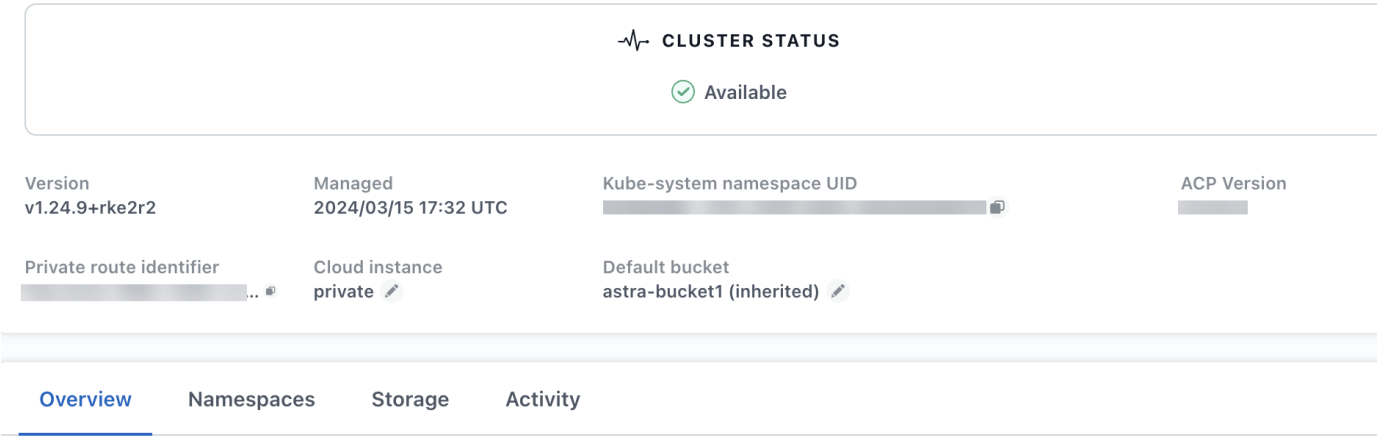
```
./tridentctl uninstall -n trident
```

禁止 "Astra TridentのCRDを完全に削除" Astra Control Provisionerを再度有効にする前に、アンインストールの一環として実行します。

結果

Astra Control Provisionerの機能が有効になっており、実行しているバージョンで使用可能なすべての機能を使用できます。

（Astra Control Centerユーザのみ）Astra Control Provisionerをインストールすると、Astra Control Center UIでプロビジョニングツールをホストしているクラスタに `ACP version` 代わりに `Trident version` フィールドと現在インストールされているバージョン番号。



を参照してください。

- "Astra Tridentのアップグレードに関するドキュメント"

Astra Controlを使用して、クラスタ管理のための環境を準備する

クラスタを追加する前に、次の前提条件を満たしていることを確認する必要があります。また、適性チェックを実行してクラスタをAstra Control Centerに追加する準備ができていることを確認し、必要に応じてkubeconfigクラスタロールを作成する必要があります。

Astra Controlでは、環境や設定に応じて、カスタムリソース（CR）またはkubeconfigで管理されるクラスタを追加できます。

作業を開始する前に

- 環境の前提条件を満たす：環境が次の条件を満たしている "運用環境の要件" （Astra Control Center向け）。
- ワーカーノードの構成: "ワーカーノードの設定" Podがバックエンドストレージと通信できるように、適切なストレージドライバを使用してクラスタを構成します。
- * PSA制限を有効にする*：Kubernetes 1.25以降のクラスタで標準であるポッドセキュリティアドミッション強制が有効になっている場合は、次の名前空間でPSA制限を有効にする必要があります。
  - `netapp-acc-operator` ネームスペース：

```
kubectl label --overwrite ns netapp-acc-operator pod-
security.kubernetes.io/enforce=privileged
```

- `netapp monitoring` ネームスペース：

```
kubectl label --overwrite ns netapp-monitoring pod-
security.kubernetes.io/enforce=privileged
```

- * ONTAP クレデンシャル*：Astra Control Centerを使用してアプリケーションをバックアップおよびリストアするには、バックアップONTAP システムでONTAP クレデンシャルとスーパーユーザーIDを設定する必要があります。

  ONTAP コマンドラインで次のコマンドを実行します。

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

- **kubeconfig**が管理するクラスタ要件:これらの要件は、kubeconfigが管理するアプリケーションクラスタに固有のものです。
  - * kubeconfigをアクセス可能にする*: "デフォルトのクラスタkubeconfig" それは "インストール時に設定"。
  - 認証局に関する考慮事項：プライベート認証局（CA）を参照するkubeconfigファイルを使用してクラスタを追加する場合は、 `cluster` kubeconfigファイルのセクションを参照してください。これにより、Astra Controlでクラスタを追加できます。

```
insecure-skip-tls-verify: true
```

  - **rancher**のみ: Rancher環境でアプリケーションクラスタを管理する場合、rancherから提供されたkubeconfigファイルでアプリケーションクラスタのデフォルトコンテキストを変更して、rancher APIサーバコンテキストではなくコントロールプレーンコンテキストを使用します。これにより、Rancher API サーバの負荷が軽減され、パフォーマンスが向上します。
- * Astra Control Provisionerの要件*：クラスタを管理するには、Astra Tridentコンポーネントを含むAstra Control Provisionerを適切に設定する必要があります。
  - * Astra Trident環境要件の確認*：Astra Control Provisionerをインストールまたはアップグレードする前に、 "サポートされるフロントエンド、バックエンド、およびホスト構成"。
  - * Astra Control Provisioner機能を有効にする*：Astra Trident 23.10以降をインストールして有効にすることを強く推奨します。 "Astra Control Provisionerの高度なストレージ機能"。今後のリリースでは、Astra Control Provisionerが有効になっていない場合、Astra ControlはAstra Tridentをサポートしません。
  - ストレージバックエンドの構成:少なくとも1つのストレージバックエンドが "Astra Tridentで設定" ク

ラスタのポリシーを確認してください。

- ストレージクラスの設定：少なくとも1つのストレージクラスが "Astra Tridentで設定" クラスタのポリシーを確認してください。デフォルトのストレージクラスが設定されている場合は、デフォルトのアノテーションが設定されている*唯一の*ストレージクラスであることを確認します。
- ボリュームスナップショットコントローラを設定し、ボリュームスナップショットクラスをインストールする： "ボリュームSnapshotコントローラのインストール" Astra Controlでスナップショットを作成できるようにします。 "作成" 1つ以上 `VolumeSnapshotClass` Astra Tridentを使用：

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. 実行しているAstra Tridentのバージョンを確認します。

```
kubectl get tridentversion -n trident
```

Astra 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   24.02.0
```

Astra 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"
```

2. 次のいずれかを実行します。
   - Astra Trident 23.01以前を実行している場合は、以下を使用 "手順" Astra Control Provisionerにアップグレードする前に、Astra Tridentの最新バージョンにアップグレードすること。可能です "直接アップグレードを実行する" Astra Tridentがバージョン24.02の4リリース期間内にある場合は、Astra Control Provisioner 24.02をダウンロードします。たとえば、Astra Trident 23.04からAstra Control Provisioner 24.02に直接アップグレードできます。
   - Astra Trident 23.10以降を実行している場合は、Astra Control Provisionerが "有効"。Astra Control Provisionerは、23.10より前のリリースのAstra Control Centerでは機能しません。 "Astra Control Provisionerのアップグレード" 最新の機能にアクセスするには、アップグレードするAstra Control Centerと同じバージョンを使用する必要があります。
3. すべてのポッド（ `trident-acp`）を実行しています：

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

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

```
kubectl get sc
```

回答例：

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

クラスタロール**kubeconfig**を作成します。

kubeconfigを使用して管理されるクラスタの場合は、必要に応じて、制限された権限または拡張された権限管理者ロールをAstra Control Centerに対して作成できます。kubeconfigはAstra Control Centerのセットアップですでに設定されているため、これは必須の手順ではありません。 "インストールプロセス"。

この手順を使用すると、次のいずれかのシナリオで環境を環境化する場合に、別のkubeconfigを作成できます。

- 管理対象のクラスタに対するAstra Controlの権限を制限する
- 複数のコンテキストを使用し、インストール時に設定されたデフォルトのAstra Control kubeconfigは使用できません。また、単一のコンテキストを持つ限定されたロールは環境では機能しません。

作業を開始する前に

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

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

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

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

手順

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

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

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

b. サービスアカウントを適用します。

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

2. 次のいずれかのクラスタロールを作成し、Astra Controlで管理するクラスタに必要な権限を割り当てます。

クラスタロールの制限

このロールには、Astra Controlでクラスタを管理するために必要な最小限の権限が含まれています。

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

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

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

# Get, List, Create, and Update all resources
# Necessary to backup and restore all resources in an app
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - get
  - list
  - create
  - patch

# Delete Resources
# Necessary for in-place restore and AppMirror failover
- apiGroups:
  - ""
  - apps
  - autoscaling
  - batch
  - crd.projectcalico.org
  - extensions
  - networking.k8s.io
  - policy
  - rbac.authorization.k8s.io
  - snapshot.storage.k8s.io
  - trident.netapp.io
  resources:
  - configmaps
  - cronjobs
  - daemonsets
  - deployments
```

```
  - horizontalpodautoscalers
  - ingresses
  - jobs
  - namespaces
  - networkpolicies
  - persistentvolumeclaims
  - poddisruptionbudgets
  - pods
  - podtemplates
  - replicasets
  - replicationcontrollers
  - replicationcontrollers/scale
  - rolebindings
  - roles
  - secrets
  - serviceaccounts
  - services
  - statefulsets
  - tridentmirrorrelationships
  - tridentsnapshotinfos
  - volumesnapshots
  - volumesnapshotcontents
  verbs:
  - delete

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

# Update resources
- apiGroups:
  - ""
  - build.openshift.io
  - image.openshift.io
  resources:
  - builds/details
  - replicationcontrollers
  - replicationcontrollers/scale
  - imagestreams/layers
```

```
  - imagestreamtags
  - imagetags
  verbs:
  - update
```

b. （OpenShiftクラスタの場合のみ） `astra-admin-account.yaml` ファイル：

```
# OpenShift security
- apiGroups:
  - security.openshift.io
  resources:
  - securitycontextconstraints
  verbs:
  - use
  - update
```

c. クラスタロールを適用します。

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

クラスタロールの拡張

このロールには、Astra Controlで管理するクラスタに対する権限が拡張されています。このロールは、複数のコンテキストを使用し、インストール時に設定されたデフォルトのAstra Control kubeconfigを使用できない場合や、単一のコンテキストを持つ限定されたロールが環境で機能しない場合に使用できます。

次のようになります `ClusterRole` 手順はKubernetesの一般的な例です。ご使用の環境に固有の手順については、ご使用のKubernetesディストリビューションのドキュメントを参照してください。

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

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

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: astra-admin-account
rules:
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - '*'
- nonResourceURLs:
  - '*'
  verbs:
  - '*'
```

b. クラスタロールを適用します。

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

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

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

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

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

b. クラスタロールバインドを適用します。

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

4. トークンシークレットを作成して適用します。
   a. という名前のトークンシークレットファイルを作成します。 `secret-astracontrol-service-account.yaml`。

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

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-astracontrol-service-account
  namespace: default
  annotations:
    kubernetes.io/service-account.name: "astracontrol-service-
account"
type: kubernetes.io/service-account-token
```

   b. トークンシークレットを適用します。

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

5. トークンシークレットの名前を `secrets` Array（次の例の最後の行）：

```
kubectl edit sa astracontrol-service-account
```

```
apiVersion: v1
imagePullSecrets:
- name: astracontrol-service-account-dockercfg-48xhx
kind: ServiceAccount
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |

{"apiVersion":"v1","kind":"ServiceAccount","metadata":{"annotations":{},
"name":"astracontrol-service-account","namespace":"default"}}
  creationTimestamp: "2023-06-14T15:25:45Z"
  name: astracontrol-service-account
  namespace: default
  resourceVersion: "2767069"
  uid: 2ce068c4-810e-4a96-ada3-49cbf9ec3f89
secrets:
- name: astracontrol-service-account-dockercfg-48xhx
<strong>- name: secret-astracontrol-service-account</strong>
```

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

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

出力の末尾は次のようになります。

```
"secrets": [
{ "name": "astracontrol-service-account-dockercfg-48xhx"},
{ "name": "secret-astracontrol-service-account"}
]
```

内の各要素のインデックス `secrets` アレイは0から始まります。上記の例では、のインデックスです `astracontrol-service-account-dockercfg-48xhx` は0、のインデックスです `secret-astracontrol-service-account` は1です。出力で、サービスアカウントシークレットのインデックス番号をメモします。このインデックス番号は次の手順で必要になります。

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

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

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

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

CONTEXT=$(kubectl config current-context)

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

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

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

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

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

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

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

# Set context to use token user
```

```
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --user ${CONTEXT}-${NAMESPACE}-token-
user

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

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

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

c. 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
```

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

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

（技術プレビュー）管理対象クラスタ用の**Astra Connector**のインストール

Astra Control Centerで管理されるクラスタは、Astra Connectorを使用して、管理対象クラスタとAstra Control Centerの間の通信を有効にします。管理するすべてのクラスタにAstra Connectorをインストールする必要があります。

Astra Connectorのインストール

KubernetesコマンドとCustom Resource（CR）ファイルを使用してAstra Connectorをインストールします。

このタスクについて

- これらの手順を実行する場合は、Astra Controlで管理するクラスタでこれらのコマンドを実行します。
- 要塞ホストを使用している場合は、要塞ホストのコマンドラインからこれらのコマンドを問題 で実行します。

作業を開始する前に

- Astra Controlで管理するクラスタへのアクセスが必要です。
- クラスタにAstra Connectorオペレータをインストールするには、Kubernetes管理者の権限が必要です。

Kubernetes 1.25以降のクラスタのデフォルトであるポッドセキュリティアドミッション適用がクラスタに設定されている場合は、適切なネームスペースに対してPSA制限を有効にする必要があります。を参照してください "Astra Controlを使用して、クラスタ管理のための環境を準備する" 手順については、を参照し

手順

1. Astra Controlで管理するクラスタにAstra Connectorオペレータをインストールします。このコマンドを実行すると、ネームスペース `astra-connector-operator` が作成され、設定がネームスペースに適用されます。

```
kubectl apply -f https://github.com/NetApp/astra-connector-operator/releases/download/24.02.0-202403151353/astraconnector_operator.yaml
```

2. オペレータが設置され、準備ができていることを確認します。

```
kubectl get all -n astra-connector-operator
```

3. Astra ControlからAPIトークンを取得を参照してください "Astra Automationのドキュメント" 手順については、を参照し

4. トークンを使用してシークレットを作成します。<API_TOKEN>を、Astra Controlから受け取ったトークンに置き換えます。

```
kubectl create secret generic astra-token \
--from-literal=apiToken=<API_TOKEN> \
-n astra-connector
```

5. Astra Connectorイメージの取得に使用するDockerシークレットを作成します。括弧<>の値は、環境の情報で置き換えます。

<ASTRA_CONTROL_ACCOUNT_ID>はAstra Control Web UIで確認できます。Web UIで、ページの右上にある図アイコンを選択し、*[API access]*を選択します。

```
kubectl create secret docker-registry regcred \
--docker-username=<ASTRA_CONTROL_ACCOUNT_ID> \
--docker-password=<API_TOKEN> \
-n astra-connector \
--docker-server=cr.astra.netapp.io
```

6. Astra Connector CRファイルを作成してという名前を付ける `astra-connector-cr.yaml`。カッコ内の値を、Astra Controlの環境とクラスタの構成に合わせて更新します。

   - <ASTRA_CONTROL_ACCOUNT_ID>：前の手順でAstra Control Web UIから取得。

- <CLUSTER_NAME>：このクラスタをAstra Controlで割り当てる名前。
- <ASTRA_CONTROL_URL>：Astra ControlのWeb UI URL。例：

```
https://astra.control.url
```

```
apiVersion: astra.netapp.io/v1
kind: AstraConnector
metadata:
  name: astra-connector
  namespace: astra-connector
spec:
  astra:
    accountId: <ASTRA_CONTROL_ACCOUNT_ID>
    clusterName: <CLUSTER_NAME>
    #Only set `skipTLSValidation` to `true` when using the default
self-signed
    #certificate in a proof-of-concept environment.
    skipTLSValidation: false #Should be set to false in production
environments
    tokenRef: astra-token
  natsSyncClient:
    cloudBridgeURL: <ASTRA_CONTROL_HOST_URL>
  imageRegistry:
    name: cr.astra.netapp.io
    secret: regcred
```

7. データを入力した後、 `astra-connector-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -n astra-connector -f astra-connector-cr.yaml
```

8. Astra Connectorが完全に導入されたことを確認します。

```
kubectl get all -n astra-connector
```

9. クラスタがAstra Controlに登録されたことを確認します。

```
kubectl get astraconnectors.astra.netapp.io -A
```

次のような出力が表示されます。

```
NAMESPACE          NAME                REGISTERED   ASTRACONNECTORID
STATUS
astra-connector    astra-connector     true         00ac8-2cef-41ac-8777-
ed0583e    Registered with Astra
```

10. Astra Control Web UIの*[Clusters]*ページで、管理対象クラスタのリストにクラスタが表示されることを確認します。

クラスタを追加

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

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

作業を開始する前に

- 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"。
- ONTAP SANドライバを使用している場合は、すべてのKubernetesクラスタでマルチパスが有効になっていることを確認します。

手順

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

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

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

3. クレデンシャル名を指定します。デフォルトでは、クレデンシャル名がクラスタの名前として自動的に入力されます。
4. 「 * 次へ * 」を選択します。

5. このKubernetesクラスタに使用するデフォルトのストレージクラスを選択し、* Next *を選択します。

Astra Control Provisionerで設定され、ONTAPストレージでバックアップされるストレージクラスを選択してください。

6. 情報を確認し、すべてが良好な場合は、「*追加」を選択します。

結果

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

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

ONTAPストレージバックエンドで認証を有効にする

Astra Control Centerには、ONTAP バックエンドの認証に次の2つのモードがあります。

- クレデンシャルベースの認証：必要な権限を持つONTAP ユーザのユーザ名とパスワード。ONTAP のバージョンとの互換性を最大限に高めるには、adminやvsadminなどの事前定義されたセキュリティログインロールを使用する必要があります。
- 証明書ベースの認証：Astra Control Centerは、バックエンドにインストールされている証明書を使用してONTAP クラスタと通信することもできます。クライアント証明書、キー、および信頼されたCA証明書を使用する（推奨）。

後で既存のバックエンドを更新して、あるタイプの認証から別の方法に移行することができます。一度にサポートされる認証方式は1つだけです。

クレデンシャルベースの認証を有効にします

Astra Control Centerには、クラスタを対象としたクレデンシャルが必要です `admin` ONTAP バックエンドと通信するため。事前定義された標準のロール（など）を使用する必要があります `admin`。これにより、Astra Control Centerの今後のリリースで使用する機能APIが公開される可能性がある、将来のONTAP リリースとの前方互換性が確保されます。

カスタムのセキュリティログインロールはAstra Control Centerで作成して使用できますが、推奨されません。

バックエンド定義の例を次に示します。

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "admin",
  "password": "secret"
}
```

クレデンシャルがプレーンテキストで保存されるのは、バックエンド定義のみです。クレデンシャルの知識が必要なのは、バックエンドの作成または更新だけです。そのため、Kubernetes管理者またはストレージ管理者が実行するのは管理者専用の操作です。

証明書ベースの認証を有効にします

Astra Control Centerでは、証明書を使用して新規および既存のONTAP バックエンドと通信できます。バックエンド定義には、次の情報を入力する必要があります。

- `clientCertificate`:クライアント証明書。
- `clientPrivateKey`:関連付けられた秘密鍵。
- `trustedCACertificate`:信頼されたCA証明書。信頼された CA を使用する場合は、このパラメータを指定する必要があります。信頼された CA が使用されていない場合は無視してかまいません。

次のいずれかのタイプの証明書を使用できます。

- 自己署名証明書
- サードパーティの証明書

自己署名証明書による認証を有効にします

一般的なワークフローは次の手順で構成されます。

手順

1. クライアント証明書とキーを生成します。生成時に、認証に使用するONTAP ユーザに共通名（CN）を設定します。

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=<common-name>"
```

2. タイプがのクライアント証明書をインストールします `client-ca` とキーをONTAP 入力します。

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

3. ONTAP のセキュリティログインロールが証明書認証方式をサポートしていることを確認します。

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

4. 生成した証明書を使用して認証をテストします。ONTAP 管理LIF>と<vserver name> を管理のIPと名前に置き換えてください。LIFのサービスポリシーがに設定されていることを確認する必要があります `default-data-management`。

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns=http://www.netapp.com/filer/admin version="1.21" vfiler="<vserver-
name>"><vserver-get></vserver-get></netapp>
```

5. 前の手順で得た値を使用して、Astra Control CenterのUIでストレージバックエンドを追加します。

サードパーティの証明書による認証を有効にします

サードパーティの証明書がある場合は、次の手順で証明書ベースの認証を設定できます。

手順

1. 秘密鍵とCSRを生成します。

```
openssl req -new -newkey rsa:4096 -nodes -sha256 -subj "/" -outform pem
-out ontap_cert_request.csr -keyout ontap_cert_request.key -addext
"subjectAltName = DNS:<ONTAP_CLUSTER_FQDN_NAME>,IP:<ONTAP_MGMT_IP>”
```

2. CSRをWindows CA（サードパーティCA）に渡し、署名済み証明書を問題 します。
3. 署名済み証明書をダウンロードし、「ontap_signed_cert.crt」という名前を付けます。
4. Windows CA（サードパーティCA）からルート証明書をエクスポートします。
5. このファイルに名前を付けます `ca_root.crt`

   これで、次の3つのファイルが作成されました。

   - 秘密鍵： `ontap_signed_request.key` （これは、ONTAP のサーバ証明書に対応するキーです。サ

ーバ証明書のインストール時に必要です）。

- 署名済み証明書： `ontap_signed_cert.crt` （これは、ONTAP の_server certificate_inとも呼ばれます）。
- ルート**CA**証明書： `ca_root.crt` （これは、ONTAP の_server-ca certificate_inとも呼ばれます）。

6. これらの証明書をONTAP にインストールします。生成してインストールします `server` および `server-ca` ONTAP の証明書。

sample.yamlの展開

```
# Copy the contents of ca_root.crt and use it here.

security certificate install -type server-ca

Please enter Certificate: Press <Enter> when done

-----BEGIN CERTIFICATE-----
<certificate details>
-----END CERTIFICATE-----


You should keep a copy of the CA-signed digital certificate for
future reference.

The installed certificate's CA and serial number for reference:

CA:
serial:

The certificate's generated name for reference:


===

# Copy the contents of ontap_signed_cert.crt and use it here. For
key, use the contents of ontap_cert_request.key file.
security certificate install -type server
Please enter Certificate: Press <Enter> when done

-----BEGIN CERTIFICATE-----
<certificate details>
-----END CERTIFICATE-----

Please enter Private Key: Press <Enter> when done

-----BEGIN PRIVATE KEY-----
<private key details>
-----END PRIVATE KEY-----

Enter certificates of certification authorities (CA) which form the
certificate chain of the server certificate. This starts with the
issuing CA certificate of the server certificate and can range up to
the root CA certificate.
Do you want to continue entering root and/or intermediate
```

```
certificates {y|n}: n

The provided certificate does not have a common name in the subject
field.
Enter a valid common name to continue installation of the
certificate: <ONTAP_CLUSTER_FQDN_NAME>

You should keep a copy of the private key and the CA-signed digital
certificate for future reference.
The installed certificate's CA and serial number for reference:
CA:
serial:
The certificate's generated name for reference:


==
# Modify the vserver settings to enable SSL for the installed
certificate

ssl modify -vserver <vserver_name> -ca <CA>  -server-enabled true
-serial <serial number>       (security ssl modify)

==
# Verify if the certificate works fine:

openssl s_client -CAfile ca_root.crt -showcerts -servername server
-connect <ONTAP_CLUSTER_FQDN_NAME>:443
CONNECTED(00000005)
depth=1 DC = local, DC = umca, CN = <CA>
verify return:1
depth=0
verify return:1
write W BLOCK
---
Certificate chain
0 s:
   i:/DC=local/DC=umca/<CA>

-----BEGIN CERTIFICATE-----
<Certificate details>
```

7. パスワードを使用しない通信用に同じホストのクライアント証明書を作成します。Astra Control Center は、このプロセスを使用してONTAP と通信します。
8. クライアント証明書を生成してONTAP にインストールします。

sample.yamlの展開

```
# Use /CN=admin or use some other account which has privileges.
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout
ontap_test_client.key -out ontap_test_client.pem -subj "/CN=admin"

Copy the content of ontap_test_client.pem file and use it in the
below command:
security certificate install -type client-ca -vserver <vserver_name>

Please enter Certificate: Press <Enter> when done

-----BEGIN CERTIFICATE-----
<Certificate details>
-----END CERTIFICATE-----

You should keep a copy of the CA-signed digital certificate for
future reference.
The installed certificate's CA and serial number for reference:

CA:
serial:
The certificate's generated name for reference:


==

ssl modify -vserver <vserver_name> -client-enabled true
(security ssl modify)

# Setting permissions for certificates
security login create -user-or-group-name admin -application ontapi
-authentication-method cert -role admin -vserver <vserver_name>

security login create -user-or-group-name admin -application http
-authentication-method cert -role admin -vserver <vserver_name>

==

#Verify passwordless communication works fine with the use of only
certificates:

curl --cacert ontap_signed_cert.crt  --key ontap_test_client.key
--cert ontap_test_client.pem
https://<ONTAP_CLUSTER_FQDN_NAME>/api/storage/aggregates
{
```

```
"records": [
{
"uuid": "f84e0a9b-e72f-4431-88c4-4bf5378b41bd",
"name": "<aggr_name>",
"node": {
"uuid": "7835876c-3484-11ed-97bb-d039ea50375c",
"name": "<node_name>",
"_links": {
"self": {
"href": "/api/cluster/nodes/7835876c-3484-11ed-97bb-d039ea50375c"
}
}
},
"_links": {
"self": {
"href": "/api/storage/aggregates/f84e0a9b-e72f-4431-88c4-
4bf5378b41bd"
}
}
}
],
"num_records": 1,
"_links": {
"self": {
"href": "/api/storage/aggregates"
}
}
}%
```


9. Astra Control CenterのUIでストレージバックエンドを追加し、次の値を指定します。
   - クライアント証明書：ontap_test_client.pem
   - 秘密鍵：ontap_test_client.key
   - 信頼された**CA**証明書：ontap_signed_cert.crt

ストレージバックエンドを追加します

クレデンシャルまたは証明書認証情報を設定したら、Astra Control Centerに既存のONTAP ストレージバックエンドを追加してリソースを管理できます。

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

Control Provisionerを有効にしている場合、NetApp SnapMirrorテクノロジを使用する場合、Astra Control CenterでONTAPストレージバックエンドの追加と管理はオプションです。

手順

1. 左側のナビゲーション領域のダッシュボードで、* Backends *を選択します。
2. 「 * 追加」を選択します。
3. [Add storage backend]ページの[Use existing]セクションで、* ONTAP *を選択します。
4. 次のいずれかを選択します。
   - 管理者のクレデンシャルを使用：ONTAP クラスタ管理IPアドレスと管理者のクレデンシャルを入力します。クレデンシャルはクラスタ全体のクレデンシャルである必要があります。

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

   - 証明書を使用：証明書をアップロードします `.pem` ファイル、証明書キー `.key` ファイルを指定し、必要に応じて認証局ファイルを指定します。
5. 「 * 次へ * 」を選択します。
6. バックエンドの詳細を確認し、 * Manage * を選択します。

結果

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

バックエンドが表示されるようにページを更新する必要がある場合があります。

バケットを追加します

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

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

作業を開始する前に

- Astra Control Centerで管理されているクラスタから到達できるバケットを用意します。
- バケットのクレデンシャルがあることを確認します。
- バケットが次のいずれかのタイプであることを確認します。
  - NetApp ONTAP S3
  - NetApp StorageGRID S3 の略
  - Microsoft Azure

- 汎用 S3

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

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

手順

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

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

4. 既存のバケット名とオプションの概要 を入力します。

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

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

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

概念

アーキテクチャとコンポーネント

Astra Control は、ステートフル アプリケーションの操作を簡素化し、ハイブリッド環境全体で Kubernetes ワークロードを保存、保護、移動できるようにする Kubernetes アプリケーション データ ライフサイクル管理ソリューションです。

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

ストア：

- コンテナ化されたワークロード向けの動的なストレージプロビジョニング
- コンテナから永続的ボリュームへの転送中データの暗号化
- リージョン間、ゾーン間レプリケーション

保護：

- アプリケーション全体とそのデータを自動検出し、アプリケーション対応の保護を実現
- 組織のニーズに基づいて’任意のスナップショット・バージョンからアプリケーションを即座にリカバリ
- ゾーン、リージョン、クラウドプロバイダ間で高速フェイルオーバー

移動：

- Kubernetesクラスタとクラウド間でアプリケーションとデータを完全に移動
- アプリケーションとデータ全体を瞬時にクローニング
- 一貫したWeb UIとAPIでワンクリックでアプリケーションを移行

アーキテクチャ

IT部門はAstra Controlのアーキテクチャにより、高度なデータ管理機能を提供できるようになり、Kubernetesアプリケーションの機能と可用性の両方を強化し、パブリッククラウドとオンプレミス環境にわたるコンテナ化されたワークロードの管理、保護、移動を簡易化できます。 また、REST APIとSDKを使用した自動化機能を提供し、プログラム経由でアクセスして既存のワークフローとシームレスに統合できます。

Astra ControlはKubernetesネイティブであるため、既存のAPIやSDKとの後方互換性を維持しながら、カスタムリソースを活用するデータ保護ワークフローを実現できます。Kubernetesネイティブのデータ保護には大きなメリットがあります。KubernetesのAPIやリソースとシームレスに統合することで、組織の既存のCI / CDツールやGitOpsツールを通じて、データ保護がアプリケーションのライフサイクルに本質的に組み込まれるようになります。

Astra Controlは、次の4つのコンポーネントを基盤として構築されています。

- **Astra Control**: Astra Control は、すべての管理対象クラスターの集中管理サービスであり、オンプレミスのアプリケーション保護とモビリティのためのオーケストレーションされたワークロードと次の機能を提

供します。

- 複数のクラスターを組み合わせたビュー
- オーケストレーションされたワークフローの保護
- きめ細かなリソースの視覚化と選択

- * Astra Connector *：Astra Connectorは、Astra Controlと連携して各管理対象クラスタへのセキュアな接続を提供し、接続ステータスに関係なくスケジュールされた処理をローカルで実行できるようにします。また、次の機能も提供します。
  - 接続ステータスに関係なく、スケジュールされた操作をローカルで実行
  - Astraのシステムリソース使用量をクラスタ間で分散、最適化するローカル運用
  - セキュリティを向上させるために、クラスタへの最小限の権限でのアクセスを可能にするローカルインストール
- * Astra Control Provisioner *：Astra Control Provisionerは、CSIのコアプロビジョニング機能と高度なストレージ管理機能を提供し、セキュリティとディザスタリカバリの設定を強化します。さらに、以下の機能も提供します。
  - コンテナ化されたワークロード向けの動的なストレージプロビジョニング
  - 高度なストレージ管理：
    - コンテナからPVへのデータの転送中暗号化
    - リージョン間、ゾーン間レプリケーションを使用したSnapMirror Cloud機能
- * Astraカスタムリソース*：各クラスタで使用されるカスタムリソースにより、Kubernetesネイティブのアプローチでローカルで運用を実行できるため、Kubernetesに対応した他のツールや自動化との統合が簡易化され、次の機能が提供されます。
  - エコシステムツールの直接統合と自動化のワークフロー
  - カスタムワークフローを可能にする下位レベルのプリミティブ

導入モデル

Astra Control は単一の展開モデルで利用できます。

- 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。NetApp Cloud Volumes ONTAP ストレージバックエンドを使用して、複数のクラウドプロバイダ環境にAstra Control Centerをインストールすることもできます。

"Astra Control Center のドキュメント"

	Astra Control Center の略
どのような方法で提供されますか？	ソフトウェアとしてダウンロード、インストール、および管理できます
ホストされているのはどこですか？	自社所有のKubernetesクラスタ
更新方法	更新を管理します

	Astra Control Center の略
サポートされている**Kubernetes**ディストリビューションを教えてください。	• Azure Stack HCIで実行されるAzure Kubernetes Service • Google Anthos • Kubernetes（アップストリーム） • Rancher Kubernetes Engine （ RKE ） • Red Hat OpenShift Container Platform
サポートされているストレージバックエンドは何ですか。	• NetApp ONTAP AFF および FAS システム • NetApp ONTAP Select の略 • "Cloud Volumes ONTAP" • "ロングホーン""

を参照してください。

- "Astra Control Center のドキュメント"
- "Astra Trident のドキュメント"
- "Astra Control API の略"
- "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

Snapshotとバックアップのどちらも、次のタイプのデータを保護します。

- アプリケーション自体
- アプリケーションに関連付けられている永続的データボリューム
- アプリケーションに属するリソースアーティファクト

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。スナップショットは、同じクラスタ内でアプリケーションをクローニングまたはリストアする場合に便利です。

_backup_はSnapshotに基づいています。外部のオブジェクトストアに格納されるため、ローカルSnapshotに比べて取得に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。また、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を選択したり、さまざまなスケジュールレベルを設定したりすることもできます。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は、組織のニーズやSLA（Service Level Agreement）の要件に応じて各アプリケーションを確実に保護するための最良の方法です。

_ 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。

変更不可のバックアップ

変更不可のバックアップとは、指定した期間中に変更または削除できないバックアップです。書き換え不可のバックアップを作成すると、Astra Controlは使用しているバケットがWrite Once Read Many（WORM）バケットであるかどうかを確認し、使用している場合はAstra Control内からバックアップを変更できないかどうかを確認します。
Astra Control Centerでは、次のプラットフォームおよびバケットタイプを使用して、変更不可のバックアップを作成できます。

- S3オブジェクトロックが設定されたAmazon S3バケットを使用するAmazon Web Services
- S3オブジェクトロックが設定されたS3バケットを使用するNetApp StorageGRID

変更不可のバックアップを使用する場合は、次の点に注意してください。

- サポートされていないプラットフォームまたはサポートされていないバケットタイプのWORMバケットにバックアップすると、保持期間が経過してもバックアップの削除が失敗するなど、予期しない結果が発生する可能性があります。
- Astra Controlでは、データのライフサイクル管理ポリシーや、変更不可のバックアップで使用するバケット上のオブジェクトの手動削除はサポートされていません。Astra ControlのSnapshotやバックアップデータのライフサイクルを管理するようにストレージバックエンドが設定されていないことを確認します。

クローン

a_clone_は、アプリケーション、その設定、および永続データボリュームの完全な複製です。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。

ストレージバックエンド間のレプリケーション

Astra Controlを使用すると、NetApp SnapMirrorテクノロジの非同期レプリケーション機能を使用して、RPO（目標復旧時点）とRTO（目標復旧時間）の低いアプリケーションのビジネス継続性を構築できます。設定が完了すると、アプリケーションは、ストレージバックエンド間、同じクラスタ上、または異なるクラスタ間でデータやアプリケーションの変更をレプリケートできるようになります。

レプリケートは、同じONTAPクラスタ上または異なるONTAPクラスタ上の2つのONTAP SVM間で実行できます。

Astra Controlは、アプリケーションのSnapshotコピーをデスティネーションクラスタに非同期でレプリケートします。レプリケーションプロセスには、SnapMirrorでレプリケートされた永続ボリュームのデータと、Astra Controlで保護されたアプリケーションメタデータが含まれます。

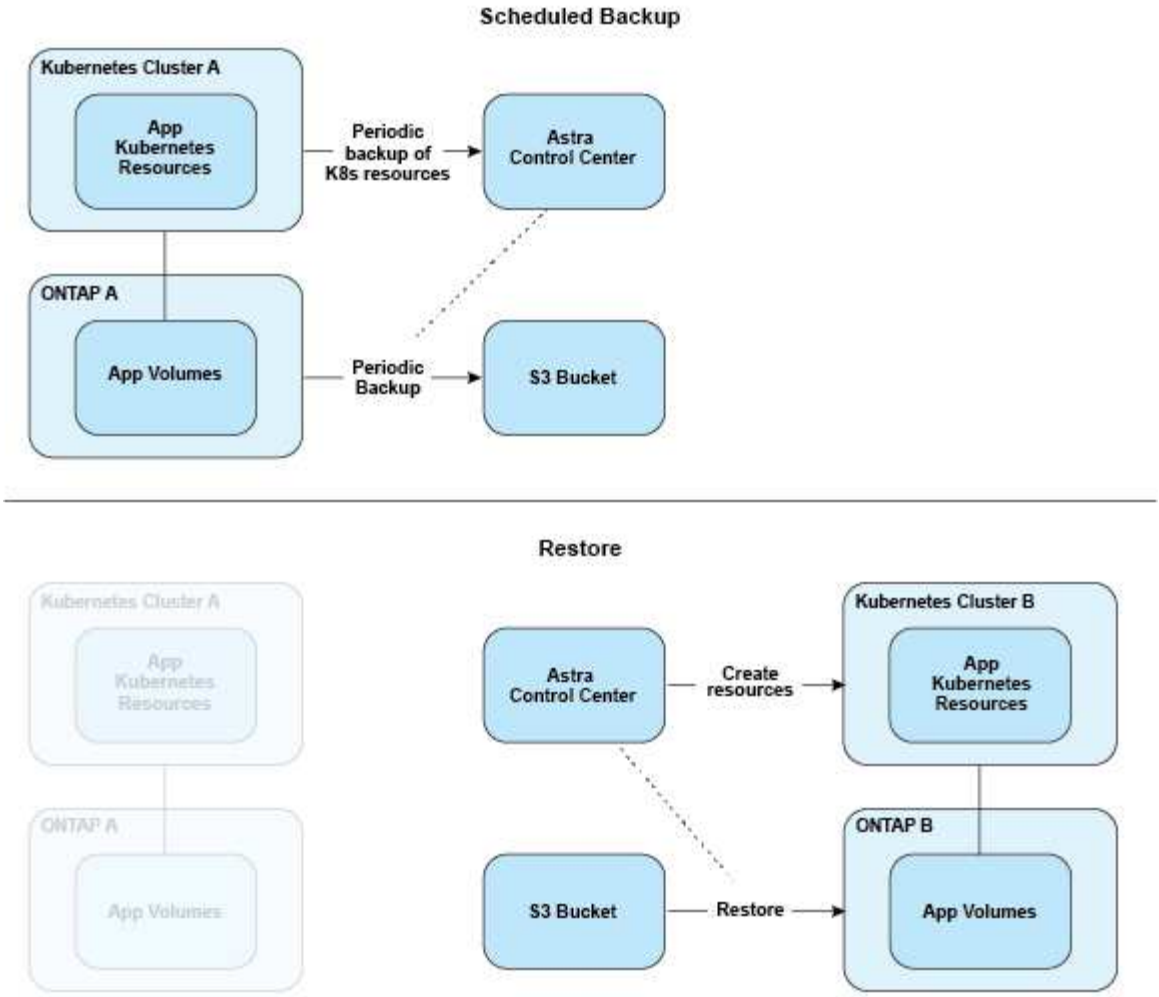
アプリケーションのレプリケーションは、次のようにアプリケーションのバックアップとリストアとは異なります。

- アプリケーションレプリケーション：Astra Controlを使用するには、ソースとデスティネーションのKubernetesクラスタ（同じクラスタでも可）が使用可能で、それぞれのONTAPストレージバックエンドでNetApp SnapMirrorを有効にするように設定されている必要があります。Astra Controlは、ポリシーベースのアプリケーションSnapshotを作成し、デスティネーションストレージバックエンドにレプリケートします。NetApp SnapMirrorテクノロジは、永続ボリュームのデータのレプリケートに使用されます。フェイルオーバーのために、デスティネーションONTAP クラスタ上のレプリケートされたボリュームを含むデスティネーションKubernetesクラスタにアプリケーションオブジェクトを再作成することで、レプリケーションされたアプリケーションをオンラインにすることができます。永続ボリュームのデータはデスティネーションのONTAPクラスタにすでに存在するため、Astra Controlを使用してフェイルオーバー時に短時間でリカバリできます。
- アプリケーションのバックアップとリストア：アプリケーションのバックアップ時に、Astra Controlによってアプリケーションデータのスナップショットが作成され、オブジェクトストレージバケットに格納されます。リストアが必要な場合は、バケット内のデータをONTAP クラスタ上の永続ボリュームにコピーする必要があります。バックアップ/リストア処理では、セカンダリKubernetes / ONTAPクラスタを使用可能にして管理する必要はありませんが、データコピーを追加するとリストア時間が長くなる可能性があります。

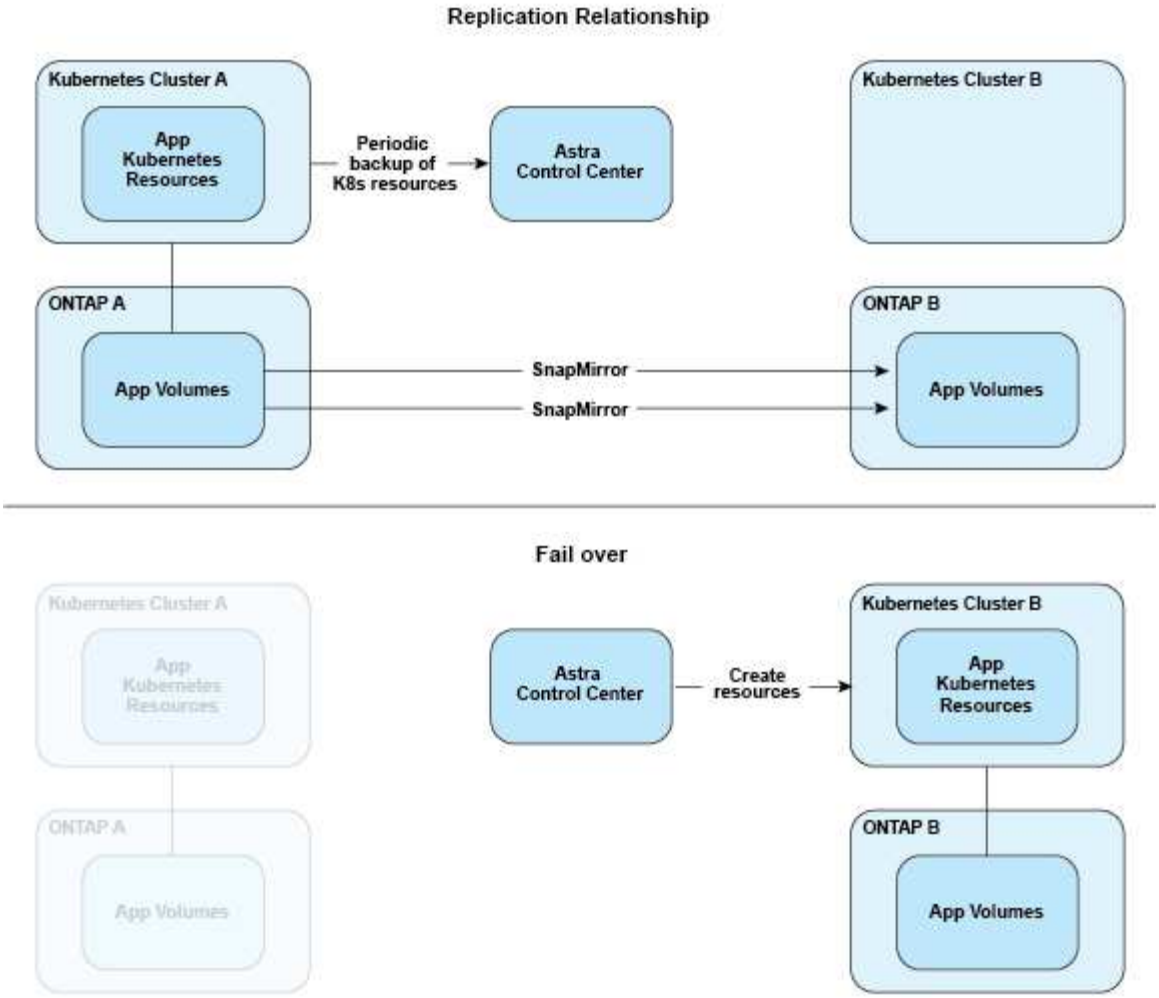
アプリケーションを複製する方法については、を参照してください "SnapMirrorテクノロジを使用してアプリケーションをリモートシステムにレプリケート"。

次の図は、スケジュールされたバックアップおよびリストアのプロセスをレプリケーションプロセスと比較したものです。

バックアッププロセスでは、S3バケットにデータをコピーし、S3バケットからリストアします。



一方、レプリケーションはONTAPにレプリケートすることで実行され、フェイルオーバーによってKubernetesリソースが作成されます。



ライセンスの有効期限が切れたバックアップ、スナップショット、クローン

ライセンスの有効期限が切れた場合、追加または保護するアプリケーションが別のAstra Control Centerインスタンスである場合にのみ、新しいアプリケーションの追加やアプリケーションの保護処理（Snapshot、バックアップ、クローン、リストア処理など）を実行できます。

ライセンス

Astra Control Centerを導入すると、4、800 CPUユニットの90日間の評価ライセンスが組み込まれてインストールされます。容量の追加や評価期間の延長が必要な場合、またはフルライセンスにアップグレードする場合は、ネットアップから別の評価用ライセンスまたはフルライセンスを取得できます。

次のいずれかの方法でライセンスを取得します。

- Astra Control Centerを評価する際に、組み込みの評価用ライセンスと異なる評価条件が必要な場合は、ネットアップに連絡して別の評価用ライセンスファイルをリクエストしてください。
- "Astra Control Centerを購入済みの場合は、ネットアップライセンスファイル（NLF）を生成する" NetApp Support Site にログインし、[Systems]メニューからソフトウェアライセンスに移動します。

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

ライセンスで有効になっているCPUユニットが必要な数以上であることを確認してください。Astra Control Centerで現在管理しているCPUユニット数が、適用する新しいライセンスで使用可能なCPUユニット数を超えると、新しいライセンスを適用できません。

評価用ライセンスとフルライセンス

Astra Control Centerの新しいインストールには、評価用ライセンスが組み込まれています。評価用ライセンスでは、フルライセンスと同じ機能を制限付き（90日間）で使用できます。評価期間が終了したら、フル機能を使用するにはフルライセンスが必要です。

ライセンスの有効期限

アクティブなAstra Control Centerのライセンスが期限切れになると、次の機能のUIおよびAPI機能は使用できなくなります。

- ローカルのスナップショットとバックアップを手動で実行します
- スケジュールされたローカルSnapshotおよびバックアップ
- 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
- 新しいアプリケーションの管理
- レプリケーションポリシーを設定しています

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。

クラスタでアプリケーションの管理を開始すると、そのクラスタのすべてのCPUユニットがAstra Control Centerのライセンス消費に含まれます。ただし、でラベルを使用して報告されるRed Hat OpenShiftクラスタノードのCPUユニットは除きます `node-role.kubernetes.io/infra: ""`。

Red Hat OpenShiftインフラノードでは、Astra Control Centerのライセンスは使用されません。ノードをインフラストラクチャノードとしてマークするには、ラベルを適用します `node-role.kubernetes.io/infra: ""` ノードに追加します。

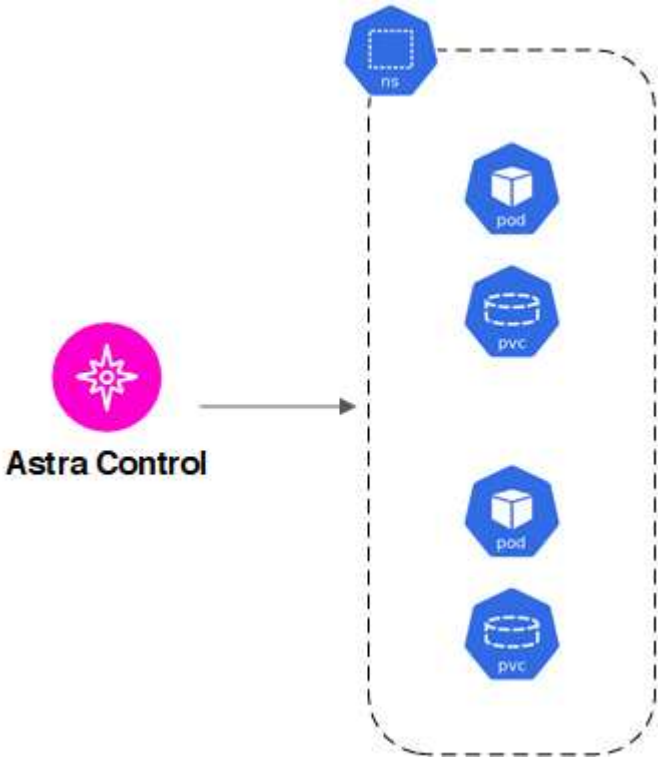
詳細については、こちらをご覧ください

- "Astra Control Centerの初回セットアップ時にライセンスを追加します"
- "既存のライセンスを更新する"

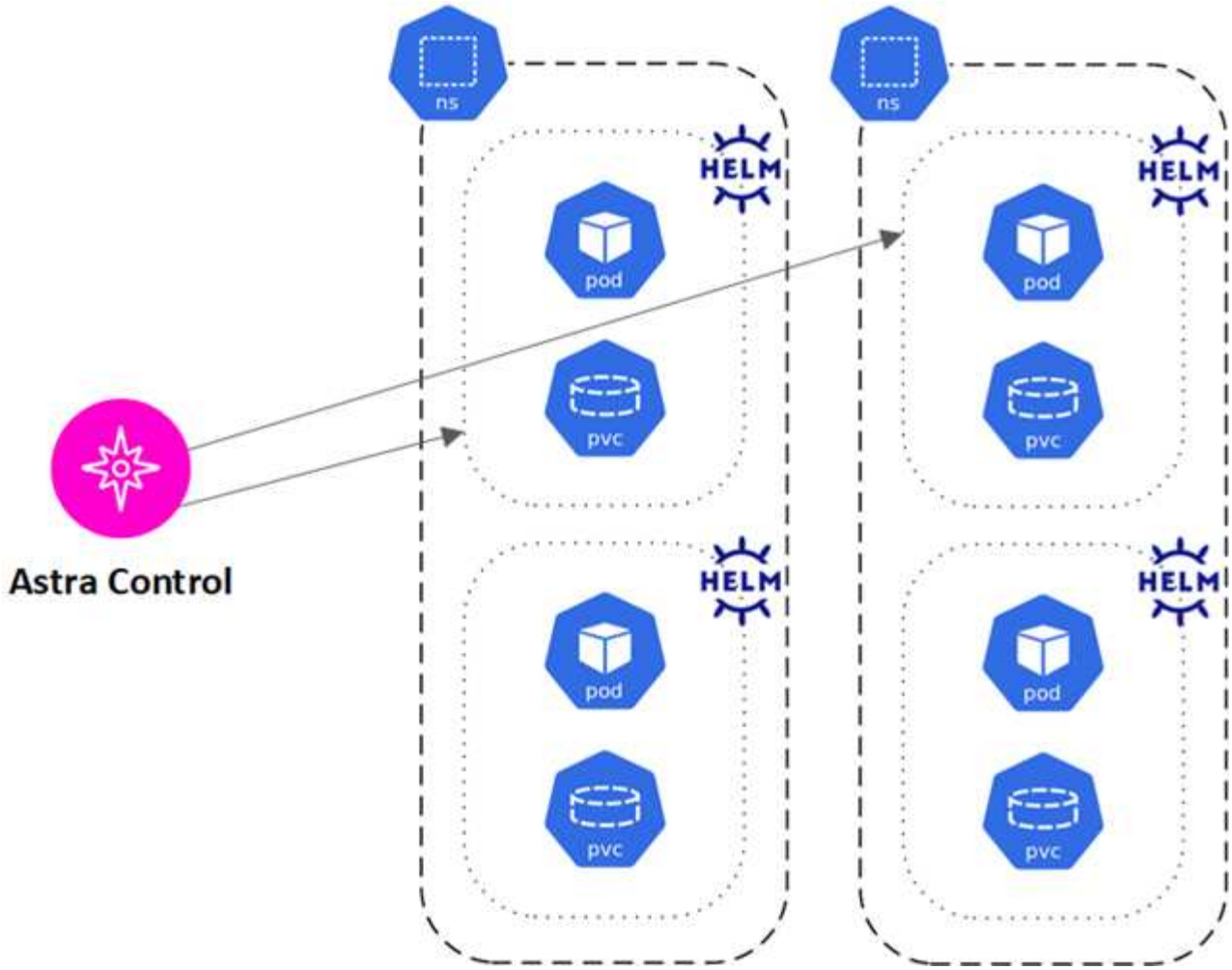
アプリケーション管理

Astra Controlがクラスタを検出すると、それらのクラスタ上のアプリケーションは、管理方法を選択するまで管理されません。Astra Control のマネージドアプリケーションには、次のいずれかを使用できます。

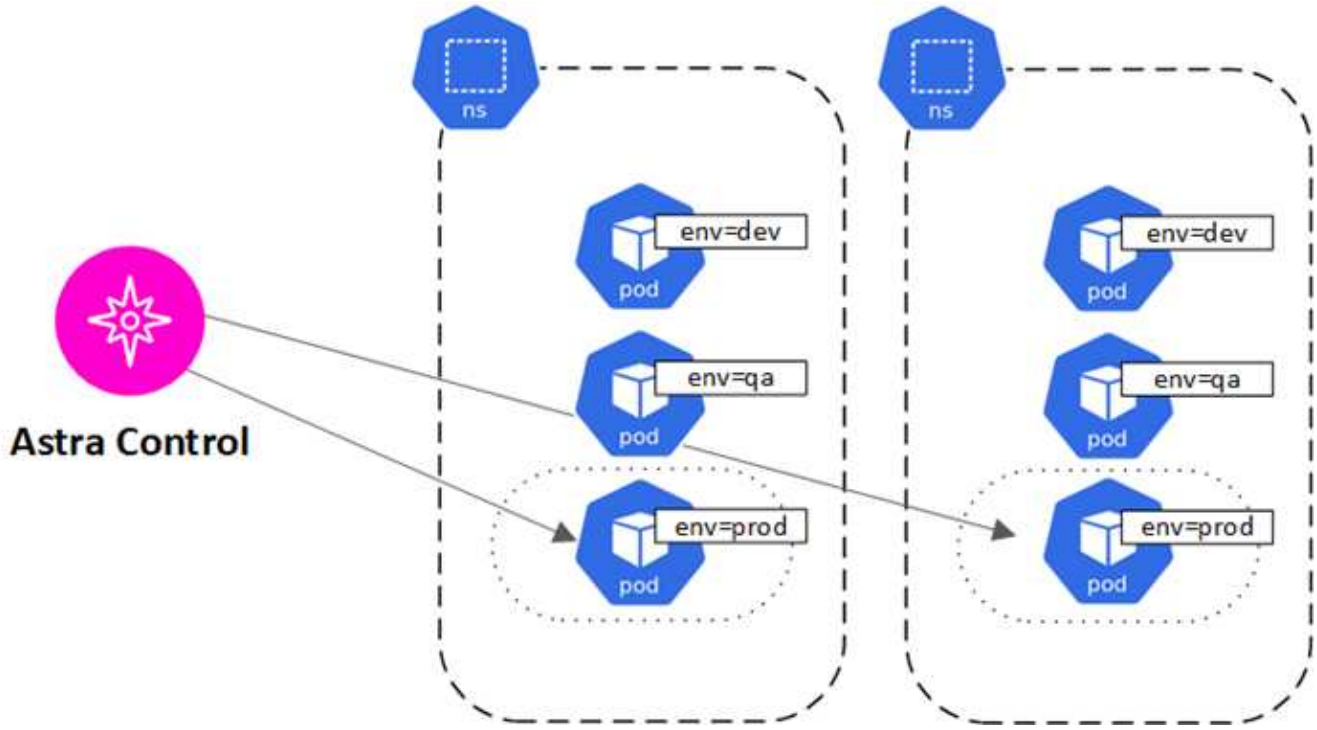
- ネームスペース。ネームスペース内のすべてのリソースを含みます



- 1つ以上のネームスペース内に導入された個 々 のアプリケーション（この例では、helm3を使用）



- 1つ以上のネームスペース内のKubernetesラベルで識別されるリソースのグループ



ストレージクラスと永続的ボリュームサイズ

Astra Control Centerでは、NetApp ONTAPとLonghornがストレージバックエンドとしてサポートされます。

概要

Astra Control Center は、次の機能をサポートします。

- * ONTAPストレージでサポートされるストレージクラス*：ONTAPバックエンドを使用している場合、Astra Control CenterでONTAPバックエンドをインポートして監視情報をレポートすることができます。
- **Longhorn**によってサポートされる**CSI**ベースのストレージクラス: Longhorn Container Storage Interface (CSI)ドライバでLonghornを使用できます。

ストレージクラスは "を設定します" Astra Control Provisionerを使用します。

ストレージクラス

Astra Control Centerにクラスタを追加する場合は、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。

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

"ローカルユーザとロールを管理します"

Astra Control Center を使用

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[アプリケーション]ページに移動して、アプリケーションとそのリソースを定義できます。

実行中のポッドを使用してストレージリソースを含むアプリケーション、または実行中のポッドを使用しないストレージリソースを含むアプリケーションを定義および管理できます。ポッドが実行されていないアプリケーションは、データ専用アプリケーションと呼ばれます。

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、組み込みのAstra Control Center評価ライセンスまたはフルライセンスのいずれかが必要です。
- 名前空間：アプリケーションは、Astra Controlを使用して、単一クラスタ上の1つ以上の指定された名前空間内で定義できます。アプリケーションには、同じクラスタ内の複数のネームスペースにまたがるリソースを含めることができます。Astra Controlでは、複数のクラスタ間でアプリケーションを定義する機能はサポートされていません。
- ストレージクラス：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インストールし、アプリケーションのクローンを作成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は、元 々 指定されたストレージクラスが必要です。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。
- * 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。

ClusterRole	ClusterRoleBinding	ConfigMap
CronJob	CustomResourceDefinition	CustomResource
DaemonSet	DeploymentConfig	HorizontalPodAutoscaler
Ingress	MutatingWebhook	NetworkPolicy
PersistentVolumeClaim	Pod	PodDisruptionBudget
PodTemplate	ReplicaSet	Role
RoleBinding	Route	Secret
Service	ServiceAccount	StatefulSet
ValidatingWebhook		

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。
- オペレータが導入するアプリ：Astra Controlは、ネームスペーススコープの演算子を使用してインストールされたアプリをサポートします。これらの演算子は、一般的に「pass-by-reference」アーキテクチャではなく「pass-by-value」アーキテクチャで設計されています。オペレーターとそれがインストールするアプリは、同じ名前空間を使用する必要があります。これを確実にするために、オペレーターの配置YAMLファイルを変更する必要がある場合があります。

  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。

クラスタにアプリをインストールします

お先にどうぞ "クラスタが追加されました" Astra Controlを使用すると、アプリケーションをインストールしたり、クラスタ上の既存のアプリケーションを管理したりできます。1つ以上の名前空間にスコープされているすべてのアプリケーションを管理できます。

アプリケーションを定義します

Astra Controlがクラスタ上のネームスペースを検出したら、管理するアプリケーションを定義できます。を選択できます 1つ以上のネームスペースにまたがるアプリケーションを管理します または ネームスペース全体を単一のアプリケーションとして管理。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。

Astra Controlを使用すると、階層の両方のレベル（ネームスペースとそのネームスペースまたはスパニングネームスペース内のアプリケーション）を別 々 に管理できますが、いずれか一方を選択することを推奨します。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。

たとえば、「Maria」に対して、毎週同じ頻度でバックアップを作成するように設定することもできますが、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必要があるとします。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ます。また、シングルネームスペースアプリケーションとして管理する必要はありません。

作業を開始する前に

- KubernetesクラスタをAstra Controlに追加。
- クラスタにインストールされているアプリケーションが1つ以上あります。 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については、こちらをご覧ください。
- Astra Controlに追加したKubernetesクラスタ上の既存のネームスペース。
- （オプション）すべてのにKubernetesラベルを付けます "サポートされるKubernetesリソース"。

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"。

このタスクについて

- 開始する前に、を理解しておく必要があります "標準ネームスペースとシステムネームスペースの管理"。
- Astra Controlのアプリケーションで複数の名前空間を使用する場合は、 "ネームスペースの制約を持つユーザロールを変更します" 複数の名前空間をサポートするAstra Control Centerバージョンにアップグレードした後。
- 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"。

アプリケーション管理オプション

- [アプリケーションとして管理するリソースを定義します]
- [アプリケーションとして管理するネームスペースを定義します]
- "（テクニカルプレビュー）Kubernetesのカスタムリソースを使用したアプリケーションの定義"

アプリケーションとして管理するリソースを定義します

を指定できます "アプリケーションを構成するKubernetesリソース" Astra Controlで管理したい。アプリケーションを定義すると、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。このKubernetesリソースの集まりは、ネームスペースとラベル選択条件によって分類されます。

アプリケーションを定義することで、クローン、スナップショット、バックアップなどのAstra Control操作に含めるものをより細かく制御できます。

アプリケーションを定義するときは、保護ポリシーを使用して複数のアプリケーションにKubernetesリソースを含めないようにしてください。Kubernetesリソースの保護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 詳細については、例を参照してください。

を展開して、アプリケーションネームスペースへのクラスタを対象としたリソースの追加について詳しく説明します。

ネームスペースリソースに関連付けられているクラスタリソースを、自動的に含まれるアストラコントロールに加えてインポートできます。特定のグループ、種類、バージョンのリソースを含むルールを追加し、必要に応じてラベルを付けることができます。この処理は、Astra Controlに自動的に含まれないリソースがある場合などに実行します。

Astra Controlに自動的に含まれる、クラスタを対象としたリソースを除外することはできません。

以下を追加できます `apiVersions` （APIバージョンと組み合わせたグループ）。

リソースの種類	1回あたりのバージョン（グループ+バージョン）
`ClusterRole`	rbac.authorization.k8s.io/v1
`ClusterRoleBinding`	rbac.authorization.k8s.io/v1
`CustomResource`	apiextensions.k8s.io/v1、apiextensions.k8s.io/v1beta1
`CustomResourceDefinition`	apiextensions.k8s.io/v1、apiextensions.k8s.io/v1beta1
`MutatingWebhookConfiguration`	admissionregistration.k8s.io/v1
`ValidatingWebhookConfiguration`	admissionregistration.k8s.io/v1

手順

1. [アプリケーション（Applications）]ページで、[定義（ Define）]を選択します
2. [アプリケーションの定義（* Define application）]ウィンドウで、アプリケーション名を入力します。
3. **[Cluster]**ドロップダウン・リストから、アプリケーションが実行されているクラスタを選択します。
4. 「名前空間」ドロップダウンリストからアプリケーションの名前空間を選択します。

アプリケーションは、Astra Controlを使用して、単一クラスタ上の1つ以上の指定された名前空間内で定義できます。アプリケーションには、同じクラスタ内の複数のネームスペースにまたがるリソースを含めることができます。Astra Controlでは、複数のクラスタ間でアプリケーションを定義する機能はサポートされていません。

5. （オプション）各ネームスペースにKubernetesリソースのラベルを入力します。ラベルまたはラベルの選択基準（クエリー）を1つ指定できます。

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"。

6. （オプション）「名前空間の追加」を選択し、ドロップダウンリストから名前空間を選択して、アプリケーションの名前空間を追加します。
7. （オプション）追加するネームスペースのラベルまたはラベルの選択基準を1つ入力します。
8. （オプション）Astra Controlに自動的に含まれるリソースに加えて、クラスタを対象としたリソースを含めるには、*クラスタを対象とした追加のリソースを含める*をチェックし、次の手順を実行します。
   a. 「含めるルールを追加」を選択します。
   b. グループ：ドロップダウンリストから、リソースのAPIグループを選択します。
   c. *kind *:ドロップダウンリストから'オブジェクトスキーマの名前を選択します
   d. バージョン：APIのバージョンを入力します。
   e. ラベルセレクタ：必要に応じて、ルールに追加するラベルを指定します。このラベルは、このラベルに一致するリソースのみを取得するために使用します。ラベルを指定しないと、Astra Controlは、そのクラスタに指定されている種類のリソースのすべてのインスタンスを収集します。
   f. エントリに基づいて作成されたルールを確認します。
   g. 「 * 追加」を選択します。

   クラスタを対象としたリソースルールは必要な数だけ作成できます。[アプリケーションの定義の概要]にルールが表示されます。

9. [* 定義（ Define ）] を選択します
10. [定義（Define *）]を選択した後、必要に応じて他のアプリケーションについても同じ手順を繰り返します。

アプリケーションの定義が完了すると、アプリケーションがに表示されます `Healthy` 「アプリケーション」ページのアプリケーションのリストに表示されます。クローンを作成し、バックアップとスナップショットを作成できるようになりました。

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。

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。

このアプリに追加されたリソースを表示するには、*リソース*タブを選択します。Resource列でリソース名のあとの番号を選択するか、Searchでリソース名を入力して、追加のクラスタを対象としたリソースを確認します。

アプリケーションとして管理するネームスペースを定義します

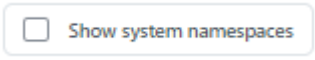
ネームスペースのリソースをアプリケーションとして定義することで、ネームスペース内のすべてのKubernetesリソースをAstra Control管理に追加できます。特定の名前空間内のすべてのリソースを同じような方法で、共通の間隔で管理および保護する場合は、アプリケーションを個別に定義することをお勧めします。

手順

1. クラスタページで、クラスタを選択します。
2. [名前空間]タブを選択します。

3. 管理するアプリケーションリソースを含む名前空間のアクションメニューを選択し、*アプリケーションとして定義*を選択します。

複数のアプリケーションを定義する場合は、名前空間リストから選択し、左上隅の*アクション*ボタンを選択して、*アプリケーションとして定義*を選択します。これにより、個 々のネームスペースに複数のアプリケーションが定義されます。マルチネームスペースアプリケーションについては、を参照してください [アプリケーションとして管理するリソースを定義します]。

[システムネームスペースを表示（Show system Namespaces）]チェックボックスを選択して、アプリケーション管理で通常はデフォルトで使用されないシステムネームスペースを表示します。 

"詳細はこちら"。

このプロセスが完了すると、ネームスペースに関連付けられているアプリケーションがに表示されます `Associated applications` 列（Column）：

[テクニカルプレビュー] Kubernetesのカスタムリソースを使用したアプリケーションの定義

カスタムリソース（CR）を使用してアプリケーションとして定義することで、Astra Controlで管理するKubernetesリソースを指定できます。たとえば、特定のネームスペース内のすべてのリソースを同様の方法で共通の間隔で管理および保護する場合は、それらのリソースを個別に管理するか、またはネームスペース内のすべてのKubernetesリソースを個別に管理する場合は、クラスタ対象のリソースを追加できます。

手順

1. カスタムリソース（CR）ファイルを作成し、という名前を付けます（例： `astra_mysql_app.yaml` ）。
2. アプリケーションに名前を付けます。 `metadata.name`。
3. 管理するアプリケーションリソースを定義します。

spec.includedClusterScopedResources

Astra Controlに自動的に含まれるもののほかに、クラスタを対象としたリソースタイプも含めます。

- * spec.includedClusterScopedResources*:_(オプション)_含めるクラスタスコープのリソースタイプのリスト。
  - *groupVersionKind *:_(オプション)_unambiguouslyは種類を識別します。
    - **group**:_(groupVersionKindが使用されている場合は必須)含めるリソースのAPIグループ。
    - **version**:_(groupVersionKindが使用されている場合は必須)_含めるリソースのAPIバージョン。
    - **kind**:_(groupVersionKindを使用する場合は必須)_kind含めるリソースの種類。
  - *labelSelector *:_(オプション)_リソースセットのラベルクエリ。ラベルに一致するリソースのみを取得するために使用されます。ラベルを指定しないと、Astra Controlは、そのクラスタに指定されている種類のリソースのすべてのインスタンスを収集します。matchLabelsとmatchExpressionsの結果はANDで処理されます。
    - **matchLabels**:_(省略可能)_{key, value}ペアのマップ。matchLabelsマップ内の1つの{key, value}は、keyフィールドが"key"、演算子が"in"、value"のみを含むvalues配列を持つmatchExpressionsの要素に相当します。要件はANDで処理されます。
    - **matchExpressions**:_(オプション)_ラベルセレクタの要件のリスト。要件はANDで処理されます。
      - *key *:_(matchExpressionsを使用する場合は必須)_ラベルセレクタに関連付けられたラベルキー。
      - 演算子:_(matchExpressionsが使用されている場合は必須)_値のセットに対するキーの関係を表します。有効な演算子： `In`、 `NotIn`、 `Exists` および `DoesNotExist`。
      - * values *：_（matchExpressionsを使用する場合は必須）_文字列値の配列。演算子が `In` または `NotIn`、values配列must_not_be empty。演算子が `Exists` または `DoesNotExist`values配列は空である必要があります。

spec.includedNamespaces

アプリケーション内のこれらのリソースに名前空間とリソースを含めます。

- * spec.includedNamespaces*:_(必須)_リソース選択のための名前空間とオプションのフィルタを定義します。
  - ネームスペース：（必須） Astra Controlで管理するアプリケーションリソースを含むネームスペース。
  - *labelSelector *:_(オプション)_リソースセットのラベルクエリ。ラベルに一致するリソースのみを取得するために使用されます。ラベルを指定しないと、Astra Controlは、そのクラスタに指定されている種類のリソースのすべてのインスタンスを収集します。matchLabelsとmatchExpressionsの結果はANDで処理されます。
    - **matchLabels**:_(省略可能)_{key, value}ペアのマップ。matchLabelsマップ内の1つの{key, value}は、keyフィールドが"key"、演算子が"in"、value"のみを含むvalues配列を持つmatchExpressionsの要素に相当します。要件はANDで処理されます。
    - **matchExpressions**:_(オプション)_ラベルセレクタの要件のリスト。 `key` および `operator` は必須です。要件はANDで処理されます。

- *key *:_(matchExpressionsを使用する場合は必須)_ラベルセレクタに関連付けられたラベルキー。
- 演算子:_(matchExpressionsが使用されている場合は必須)_値のセットに対するキーの関係を表します。有効な演算子： `In`、 `NotIn`、 `Exists` および `DoesNotExist`。
- * values *：_（matchExpressionsを使用する場合は必須）_文字列値の配列。演算子が `In` または `NotIn`、values配列must_not_be empty。演算子が `Exists` または `DoesNotExist`values配列は空である必要があります。

YAMLの例：

```
apiVersion: astra.netapp.io/v1
kind: Application
metadata:
  name: astra_mysql_app
spec:
  includedNamespaces:
    - namespace: astra_mysql_app
      labelSelector:
        matchLabels:
          app: nginx
          env: production
        matchExpressions:
          - key: tier
            operator: In
            values:
              - frontend
              - backend
```

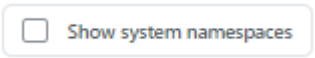
4. データを入力した後、 `astra_mysql_app.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra_mysql_app.yaml -n astra-connector
```

システムネームスペースについて教えてください。

Astra Controlは、Kubernetesクラスタ上のシステムネームスペースも検出します。これらのシステムネームスペースはデフォルトでは表示されません。システムアプリケーションリソースのバックアップが必要になることがまれです。

選択したクラスタの[ネームスペース]タブからシステムネームスペースを表示するには、[システムネームスペースを表示]チェックボックスをオンにします。



デフォルトでは、管理可能なアプリケーションとしてAstra Control Centerが表示されませんが、別のAstra Control Centerインスタンスを使用してAstra Control Centerインスタンスをバックアップおよびリストアできます。

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。チームのクラスタには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管理者がラベルを追加します `deployment=stable` を使用して、安定版リリースポッドに移動しますチームがラベルを追加します `deployment=canary` カナリアリリースポッドに移動します。

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リリースなので、ラベル付きのものは何でも短時間で、よりアグレッシブな保護ポリシーを作成したいと考えています `deployment=canary`。

データの競合を回避するために、管理者は「カナリア」リリース用と「stable」リリース用の2つのアプリケーションを作成します。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。

詳細については、こちらをご覧ください

- "Astra Control API を使用"
- "アプリの管理を解除します"

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"。

また、ディザスタリカバリに備えてアプリケーションをリモートクラスタにレプリケートすることもできます。

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保護

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] 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.] アプリケーションをリモートクラスタにレプリケートします

"アプリケーションをレプリケートします" NetApp SnapMirrorテクノロジを使用してリモートクラスタにバックアップする場合Astra Controlは、Snapshotをリモートクラスタにレプリケートし、非同期のディザスタリカバリ機能を提供します。

[5 つ] 災害が発生した場合は、最新のバックアップまたはレプリケーションを使用してアプリケーションをリモートシステムにリストアします

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）。または、リモートシステムへのレプリケーションを使用することもできます。

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、すべてのアプリケーションを保護します。Astra Control Center UI またはを使用できます "Astra Control API" アプリを保護します。

このタスクについて

- * Helmでアプリケーションを展開*：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**クラスタのみ）ポリシーの追加：OpenShiftクラスタでアプリケーションをホストするためのプロジェクトを作成すると、プロジェクト（またはKubernetes名前空間）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 wordpress
oc adm policy add-scc-to-group anyuid system:serviceaccounts:wordpress
oc adm policy add-scc-to-user privileged -z default -n wordpress
```

アプリケーションデータの保護に関連する次のタスクを実行できます。

- [保護ポリシーを設定します]
- Snapshot を作成します
- [バックアップを作成します]
- ONTAP NAS経済性に優れた運用向けのバックアップとリストアを実現
- [変更不可のバックアップの作成]
- Snapshot とバックアップを表示します
- Snapshot を削除します
- [バックアップをキャンセルします]
- [バックアップを削除します]

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。保護ポリシーは、Astra Control Web UIまたはカスタムリソース（CR）ファイルを使用して定義できます。

1 時間に 1 回以上の頻度でバックアップや Snapshot を実行する必要がある場合は、次の方法があります "Astra Control REST API を使用して、スナップショットとバックアップを作成"。

Write Once Read Many（WORM）バケットに書き換え不能なバックアップを作成する保護ポリシーを定義している場合は、バックアップの保持期間がバケットに設定されている保持期間よりも短くないようにしてください。

バックアップとレプリケーションのスケジュールをオフセットして、スケジュールの重複を回避します。たとえば、1時間ごとに1時間の最上部にバックアップを実行し、オフセットを5分、間隔を10分に設定してレプリケーションを開始するようにスケジュールを設定します。

Web UIを使用して保護ポリシーを設定する

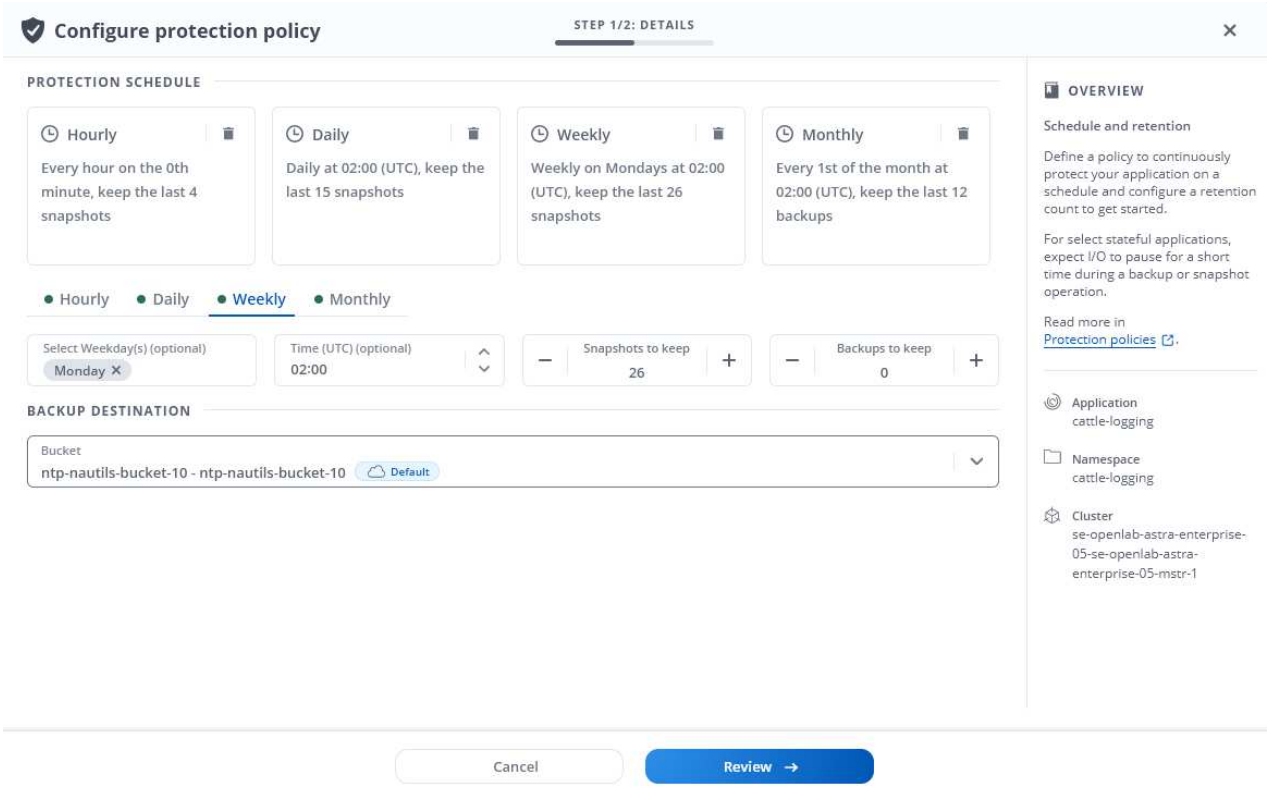
手順

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。

   バックアップの保持レベルを設定する際に、バックアップを格納するバケットを選択できます。

   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. [技術プレビュー]ストレージバケットのリストから、バックアップまたはスナップショットのデスティネーションバケットを選択します。
6. [* Review （レビュー）] を選択します
7. ［ * 保護ポリシーの設定 * ］ を選択します

[テクニカルプレビュー] CRを使用した保護ポリシーの設定

手順

1. カスタムリソース（CR）ファイルを作成して名前を付けます。 `astra-control-schedule-`

`cr.yaml`。Astra Control環境、クラスタ構成、データ保護のニーズに合わせて、かっこ<>の値を更新します。

- <CR_NAME>：このカスタムリソースの名前。環境に適した一意の適切な名前を選択します。
- <APPLICATION_NAME>：バックアップするアプリケーションのKubernetes名。
- <APPVAULT_NAME>：バックアップコンテンツを格納するAppVaultの名前。
- <BACKUPS_RETAINED>：保持するバックアップの数。ゼロは、バックアップを作成しないことを示します。
- <SNAPSHOTS_RETAINED>：保持するSnapshotの数。ゼロは、スナップショットを作成しないことを示します。
- <GRANULARITY>：スケジュールを実行する頻度。指定可能な値と必須の関連フィールドは次のとおりです。
  - `hourly`（次を指定する必要があります： `spec.minute`）
  - `daily`（次を指定する必要があります： `spec.minute` および `spec.hour`）
  - `weekly`（次を指定する必要があります： `spec.minute`、 `spec.hour`および `spec.dayOfWeek`）
  - `monthly`（次を指定する必要があります： `spec.minute`、 `spec.hour`および `spec.dayOfMonth`）
- <DAY_OF_MONTH>：_（オプション）_スケジュールを実行する日にち（1 ~ 31）。このフィールドは、粒度が次の値に設定されている場合は必須です。 `monthly`。
- <DAY_OF_WEEK>：_（オプション）_スケジュールを実行する曜日（0 ~ 7）。0または7の値は日曜日を示します。このフィールドは、粒度が次の値に設定されている場合は必須です。 `weekly`。
- <HOUR_OF_DAY>：_（オプション）_スケジュールを実行する時間（0 ~ 23）。このフィールドは、粒度が次の値に設定されている場合は必須です。 `daily`、 `weekly`または `monthly`。
- <MINUTE_OF_HOUR>：_（オプション）_スケジュールを実行する分（0~59）。このフィールドは、粒度が次の値に設定されている場合は必須です。 `hourly`、 `daily`、 `weekly`または `monthly`。

```
apiVersion: astra.netapp.io/v1
kind: Schedule
metadata:
  namespace: astra-connector
  name: <CR_NAME>
spec:
  applicationRef: <APPLICATION_NAME>
  appVaultRef: <APPVAULT_NAME>
  backupRetention: "<BACKUPS_RETAINED>"
  snapshotRetention: "<SNAPSHOTS_RETAINED>"
  granularity: <GRANULARITY>
  dayOfMonth: "<DAY_OF_MONTH>"
  dayOfWeek: "<DAY_OF_WEEK>"
  hour: "<HOUR_OF_DAY>"
  minute: "<MINUTE_OF_HOUR>"
```

2. データを入力した後、 `astra-control-schedule-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-control-schedule-cr.yaml
```

結果

Astra Control 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。

このタスクについて

Astra Controlでは、次のドライバでサポートされるストレージクラスを使用したSnapshotの作成がサポートされます。

- `ontap-nas`
- `ontap-san`
- `ontap-san-economy`

> アプリケーションがでサポートされるストレージクラスを使用している場合 `ontap-nas-economy` ドライバ、スナップショットを作成できません。スナップショットには代替のストレージクラスを使用します。

Web UIを使用したSnapshotの作成

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、*次へ*を選択します。
4. [技術プレビュー]ストレージバケットのリストからスナップショットのデスティネーションバケットを選択します。
5. Snapshot の概要を確認し、「 * Snapshot * 」を選択します。

[テクニカルプレビュー] CRを使用したスナップショットの作成

手順

1. カスタムリソース（CR）ファイルを作成して名前を付けます。 `astra-control-snapshot-cr.yaml`。カッコ内の値を、Astra Controlの環境とクラスタの構成に合わせて更新します。
   - <CR_NAME>：このカスタムリソースの名前。環境に適した一意の適切な名前を選択します。
   - <APPLICATION_NAME>：Snapshotを作成するアプリケーションのKubernetes名。
   - <APPVAULT_NAME>：スナップショットの内容を格納するAppVaultの名前。
   - <RECLAIM_POLICY>：_（オプション）_スナップショットCRが削除されたときのスナップショットの処理を定義します。有効なオプション：
     - `Retain`
     - `Delete` （デフォルト）

```
apiVersion: astra.netapp.io/v1
kind: Snapshot
metadata:
  namespace: astra-connector
  name: <CR_NAME>
spec:
  applicationRef: <APPLICATION_NAME>
  appVaultRef: <APPVAULT_NAME>
  reclaimPolicy: <RECLAIM_POLICY>
```

2. データを入力した後、 `astra-control-snapshot-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-control-snapshot-cr.yaml
```

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* Healthy である場合に成功し

ます（ Data protection > Snapshots ページの State *列）

バックアップを作成します

アプリはいつでもバックアップできます。

このタスクについて

Astra Controlのバケットで使用可能な容量が報告されません。Astra Controlで管理されるアプリケーションをバックアップまたはクローニングする前に、該当するストレージ管理システムでバケット情報を確認してください。

アプリケーションがでサポートされるストレージクラスを使用している場合 `ontap-nas-economy` 運転手、あなたがする必要があります バックアップとリストアの有効化 機能性：次を定義したことを確認してください： `backendType` のパラメータ "Kubernetesストレージオブジェクト" を使用します `ontap-nas-economy` 保護処理を実行する前に

> Astra Controlでは、次のドライバでサポートされるストレージクラスを使用したバックアップの作成がサポートされます。
>
> - `ontap-nas`
> - `ontap-nas-economy`
> - `ontap-san`
> - `ontap-san-economy`

Web UIを使用したバックアップの作成

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. [技術プレビュー]ストレージバケットのリストからバックアップ先のバケットを選択します。
6. 「 * 次へ * 」を選択します。
7. バックアップの概要を確認し、「バックアップ」を選択します。

[テクニカルプレビュー] CRを使用したバックアップの作成

手順

1. カスタムリソース（CR）ファイルを作成して名前を付けます。 `astra-control-backup-cr.yaml`。カッコ内の値を、Astra Controlの環境とクラスタの構成に合わせて更新します。
   - <CR_NAME>：このカスタムリソースの名前。環境に適した一意の適切な名前を選択します。
   - <APPLICATION_NAME>：バックアップするアプリケーションのKubernetes名。
   - <APPVAULT_NAME>：バックアップコンテンツを格納するAppVaultの名前。

```
apiVersion: astra.netapp.io/v1
kind: Backup
metadata:
  namespace: astra-connector
  name: <CR_NAME>
spec:
  applicationRef: <APPLICATION_NAME>
  appVaultRef: <APPVAULT_NAME>
```

2. データを入力した後、 `astra-control-backup-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-control-backup-cr.yaml
```

結果

Astra Control ：アプリケーションのバックアップを作成

- 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キャンセルする必要がある場合は、の手順に従ってください [バックアップをキャンセルします]。バックアップを削除するには、完了するまで待ってから、の手順を実行します [バックアップを削除します]。
- 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。

ONTAP NAS経済性に優れた運用向けのバックアップとリストアを実現

Astra Control Provisionerは、バックアップとリストアの機能を提供します。この機能は、 `ontap-nas-economy` ストレージクラス。

作業を開始する前に

- これで完了です "Astra Control Provisionerを有効にしました"。
- Astra Controlでアプリケーションを定義しておきます。この手順を完了するまで、このアプリケーションの保護機能は制限されます。
- これで完了です `ontap-nas-economy` ストレージバックエンドのデフォルトのストレージクラスとして選択されています。

手順

1. ONTAPストレージバックエンドで次の手順を実行します。
   a. をホストしているSVMを検索します。 `ontap-nas-economy`-アプリケーションのボリュームベース。
   b. ボリュームを作成するONTAPに接続されている端末にログインします。
   c. SVMのSnapshotディレクトリを非表示にします。

      この変更はSVM全体に影響します。非表示のディレクトリには引き続きアクセスできます。

```
nfs modify -vserver <svm name> -v3-hide-snapshot enabled
```

+

      ONTAPストレージバックエンドのsnapshotディレクトリが非表示になっていることを確認します。このディレクトリを非表示にしないと、アプリケーション（特にNFSv3を使用している場合）へのアクセスが失われる可能性があります。

2. Astra Control Provisionerで次の手順を実行します。
   a. 次のPVごとにSnapshotディレクトリを有効にします。 `ontap-nas-economy` ベースで、アプリケーションに関連付けられています。

```
tridentctl update volume <pv name> --snapshot-dir=true --pool-level
=true -n trident
```

b. 関連付けられている各PVに対してSnapshotディレクトリが有効になっていることを確認します。

```
tridentctl get volume <pv name> -n trident -o yaml | grep snapshotDir
```

対応：

```
snapshotDirectory: "true"
```

3. Astra Controlで、関連付けられているSnapshotディレクトリをすべて有効にしたあとにアプリケーションを更新し、Astra Controlが変更された値を認識するようにします。

結果

Astra Controlを使用して、アプリケーションのバックアップとリストアを実行できるようになります。各PVCは、他のアプリケーションでバックアップおよびリストアに使用することもできます。

変更不可のバックアップの作成

変更不可のバックアップは、バックアップを格納するバケットの保持ポリシーで禁止されているかぎり、変更、削除、上書きすることはできません。保持ポリシーが設定されたバケットにアプリケーションをバックアップすることで、変更不可のバックアップを作成できます。を参照してください "データ保護" を参照してください。

作業を開始する前に

保持ポリシーを使用してデスティネーションバケットを設定する必要があります。その方法は、使用するストレージプロバイダによって異なります。詳細については、ストレージプロバイダのドキュメントを参照してください。

- * Amazon Web Services *： "バケットの作成時にS3オブジェクトロックを有効にし、デフォルトの保持モードを「governance」にデフォルトの保持期間を設定する"。
- * NetApp StorageGRID *： "バケットの作成時にS3オブジェクトロックを有効にし、デフォルトの保持モードを「compliance」にデフォルトの保持期間を設定する"。

Astra Controlのバケットで使用可能な容量が報告されません。Astra Controlで管理されるアプリケーションをバックアップまたはクローニングする前に、該当するストレージ管理システムでバケット情報を確認してください。

アプリケーションがでサポートされるストレージクラスを使用している場合 `ontap-nas-economy` ドライバ。を定義していることを確認してください `backendType` のパラメータ "Kubernetesストレージオブジェクト" を使用します `ontap-nas-economy` 保護処理を実行する前に

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、バックアップのデスティネーションバケットを選択します。Write Once Read Many（WORM）バケット名の横にステータスが「Locked」と表示されます。

   バケットのタイプがサポートされていない場合は、バケットにカーソルを合わせるか選択すると表示されます。

6. 「 * 次へ * 」を選択します。
7. バックアップの概要を確認し、「バックアップ」を選択します。

結果

Astra Controlがアプリケーションの変更不可のバックアップを作成

- 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。
- 同じアプリケーションの書き換え不能な2つのバックアップを同じバケットに同時に作成しようとすると、Astra Controlによって2つ目のバックアップが開始されなくなります。最初のバックアップが完了してから、別のバックアップを開始してください。
- 実行中の変更不可のバックアップはキャンセルできません。
- 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。

変更不可のバックアップのステータスは、使用しているバケットの横に「Locked」と表示されます。

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。

現在レプリケート中のSnapshotは削除できません。

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. [* 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 がスナップショットを削除します。

バックアップをキャンセルします

実行中のバックアップをキャンセルすることができます。

バックアップをキャンセルするには、バックアップが実行されている必要があります `Running` 状態。にあるバックアップはキャンセルできません `Pending` 状態。

実行中の変更不可のバックアップはキャンセルできません。

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. 「 * Backups * 」を選択します。
4. 目的のバックアップの[アクション（* Actions ）]列の**[**オプション（**Options**）]メニューから、**[***キャンセル（* Cancel *）]を選択します。
5. 処理を確認するために「CANCEL」と入力し、「* Yes、cancel backup *」を選択します。

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。バケットの保持ポリシーで変更不可のバケットに作成されたバックアップは削除できません。

保持期間が終了する前に変更不可のバックアップを削除することはできません。

実行中のバックアップをキャンセルする必要がある場合は、の手順に従ってください [バックアップをキャンセルします]。バックアップを削除するには、完了するまで待ってから、次の手順を実行します。

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. 「 * 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 がバックアップを削除する。

[技術プレビュー]クラスタ全体を保護する

クラスタ上の管理対象外のネームスペースの一部またはすべてについて、スケジュールされた自動バックアップを作成できます。これらのワークフローは、NetAppによってKubernetesサービスアカウント、ロールバインド、およびcronジョブとして提供され、Pythonスクリプトを使用してオーケストレーションされます。

動作の仕組み

フルクラスタバックアップワークフローを設定してインストールすると、cronジョブが定期的に実行され、まだ管理されていないネームスペースが保護され、インストール時に選択したスケジュールに基づいて保護ポリシーが自動的に作成されます。

フルクラスタバックアップワークフローでクラスタ上のすべての管理対象外のネームスペースを保護する必要がない場合は、ラベルベースのバックアップワークフローを使用できます。ラベルベースのバックアップのワークフローでもcronタスクを使用しますが、管理対象外のネームスペースをすべて保護する代わりに、指定したラベルでネームスペースを識別して、Bronze、Silver、またはGoldのバックアップポリシーに基づいてネームスペースを保護することもできます。

選択したワークフローのスコープに含まれる新しい名前空間が作成されると、管理者の操作なしで自動的に保護されます。これらのワークフローはクラスタ単位で実装されるため、クラスタの重要度に応じて、それぞれのクラスタで独自の保護レベルを持つワークフローを使用できます。

例：完全なクラスタ保護

たとえば、フルクラスタバックアップワークフローを構成してインストールすると、任意のネームスペース内のすべてのアプリケーションが定期的に管理され、管理者による追加の作業なしに保護されます。ワークフローのインストール時に名前空間が存在している必要はありません。将来追加された名前空間は保護されます。

例：ラベルベースの保護

詳細については、ラベルベースのワークフローを使用できます。たとえば、このワークフローをインストールし、必要な保護レベルに応じて、保護する名前空間に複数のラベルのいずれかを適用するようにユーザに指示できます。これにより、ユーザーはこれらのラベルのいずれかを使用して名前空間を作成でき、管理者に通知する必要はありません。新しいネームスペースとその中のすべてのアプリは自動的に保護されます。

すべてのネームスペースのスケジュールされたバックアップを作成する

フルクラスタバックアップワークフローを使用して、クラスタ上のすべてのネームスペースのスケジュールされたバックアップを作成できます。

手順

1. クラスタにネットワークでアクセスできるマシンに、次のファイルをダウンロードします。
   - "コンポーネント.yaml CRDファイル"
   - "protectCluster.py Pythonスクリプト"
2. ツールキットを設定してインストールするには、次の手順に従います。 "付属の手順に従います。"。

特定のネームスペースのスケジュールされたバックアップを作成する

ラベルベースのバックアップワークフローを使用して、ラベル別に特定のネームスペースのスケジュールされたバックアップを作成できます。

手順

1. クラスタにネットワークでアクセスできるマシンに、次のファイルをダウンロードします。
   - "コンポーネント.yaml CRDファイル"
   - "protectCluster.py Pythonスクリプト"
2. ツールキットを設定してインストールするには、次の手順に従います。 "付属の手順に従います。"。

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 の略" アプリを復元するには、

作業を開始する前に

- 最初にアプリケーションを保護する:アプリケーションを復元する前に、アプリケーションのスナップショットまたはバックアップを作成することを強くお勧めします。これにより、リストアに失敗した場合に、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できます。
- デスティネーションボリュームの確認：別のストレージクラスにリストアする場合は、ストレージクラスで同じ永続ボリュームアクセスモード（ReadWriteManyなど）が使用され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。たとえば、ソースの永続ボリュームがRWXアクセスモードを使用している場合は、Azure Managed Disks 、AWS EBS、Google Persistent Disk、など、RWXを提供できないデスティネーションストレージクラスを選択します `ontap-san`を指定すると、リストア処理は失敗します。原因は失敗します。永続ボリュームのアクセスモードの詳細については、を参照してください "Kubernetes" ドキュメント
- 必要なスペースを確保するための計画：NetApp ONTAP ストレージを使用するアプリケーションのインプレースリストアを実行すると、リストアしたアプリケーションで使用されるスペースが2倍になることがあります。In Placeリストアを実行したあとに、リストアしたアプリケーションから不要なSnapshotを削除して、ストレージスペースを解放します。
- （**Red Hat OpenShift**クラスタのみ）ポリシーの追加：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名前空間）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 wordpress
oc adm policy add-scc-to-group anyuid system:serviceaccounts:wordpress
oc adm policy add-scc-to-user privileged -z default -n wordpress
```

- サポートされるストレージクラスドライバ：Astra Controlでは、次のドライバに基づくストレージクラスを使用したバックアップのリストアがサポートされます。
  - `ontap-nas`
  - `ontap-nas-economy`
  - `ontap-san`
  - `ontap-san-economy`
- （**ontap-nas-economy**ドライバのみ）バックアップとリストア： `ontap-nas-economy` ドライバを使用して、 "ONTAPストレージバックエンドのSnapshotディレクトリが非表示になっている"。このディレクトリを非表示にしないと、アプリケーション（特にNFSv3を使用している場合）へのアクセスが失われる可能性があります。
- * Helmデプロイ済みアプリ*：Helm 3でデプロイされたアプリ（またはHelm 2からHelm 3にアップグレードされたアプリ）は完全にサポートされます。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。

> リソースを共有するアプリケーションでIn Placeリストア処理を実行すると、予期しない結果が生じる可能性があります。アプリケーション間で共有されているリソースは、いずれかのアプリケーションでインプレースリストアが実行されると置き換えられます。詳細については、を参照してください この例です。

リストアするアーカイブのタイプに応じて、次の手順を実行します。

Web UIを使用してバックアップまたは**Snapshot**からデータをリストア

Astra Control Web UIを使用してデータをリストアできます。

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. [オプション]メニューの[操作]列で、*[リストア]*を選択します。
3. リストアタイプを選択します。
   - 元のネームスペースにリストア：この手順 を使用して、アプリケーションを元のクラスタにインプレースでリストアします。

     > アプリケーションがでサポートされるストレージクラスを使用している場合 `ontap-nas-economy` ドライバ。元のストレージクラスを使用してアプリケーションをリストアする必要があります。アプリケーションを同じネームスペースにリストアする場合、別のストレージクラスを指定することはできません。

     i. アプリをインプレースで復元するために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。これにより、アプリは以前のバージョンに戻ります。

     ii. 「 * 次へ * 」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。

- 新しい名前空間に復元：この手順 を使用して、アプリを別のクラスタまたはソースとは異なる名前空間で別のクラスタに復元します。
  i. 復元されたアプリの名前を指定します。
  ii. リストアする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。
  iii. アプリケーションに関連付けられている各ソースネームスペースのデスティネーションネームスペースを入力します。

Astra Controlは、このリストアオプションの一部として新しいデスティネーションネームスペースを作成します。指定するデスティネーションネームスペースがデスティネーションクラスタに存在していないことを確認してください。

  iv. 「 * 次へ * 」を選択します。
  v. アプリの復元に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。
  vi. 「 * 次へ * 」を選択します。
  vii. 次のいずれかを選択します。
    - 元のストレージクラスを使用してリストア：ターゲットクラスタに存在しない場合を除き、元々 関連付けられていたストレージクラスがアプリケーションで使用されます。この場合、クラスタのデフォルトのストレージクラスが使用されます。
    - 別のストレージクラスを使用したリストア：ターゲットクラスタに存在するストレージクラスを選択してください。元 々 関連付けられていたストレージクラスに関係なく、すべてのアプリケーションボリュームが、リストアの一環としてこの別のストレージクラスに移動されます。
  viii. 「 * 次へ * 」を選択します。

4. フィルタするリソースを選択：
   - すべてのリソースを復元：元のアプリケーションに関連付けられているすべてのリソースを復元します。
   - リソースのフィルタ:元のアプリケーションリソースのサブセットを復元するルールを指定します。
     i. リストアされたアプリケーションにリソースを含めるか除外するかを選択します。
     ii. または[除外ルールを追加]*のいずれかを選択し、アプリケーションのリストア時に正しいリソースをフィルタするようにルールを設定します。設定が正しくなるまで、ルールを編集したり削除したり、ルールを再度作成したりすることができます。

includeルールとexcludeルールの設定については、を参照してください [アプリケーションのリストア中にリソースをフィルタリングします]。

5. 「 * 次へ * 」を選択します。
6. リストア処理の詳細をよく確認し、プロンプトが表示されたら「restore」と入力して*[リストア]*を選択します。

[テクニカルプレビュー]カスタムリソース（**CR**）を使用したバックアップからのリストア

カスタムリソース（CR）ファイルを使用して、別のネームスペースまたは元のソースネームスペースにバックアップからデータをリストアできます。

CRを使用したバックアップからのリストア

手順

1. カスタムリソース（CR）ファイルを作成して名前を付けます。 `astra-control-backup-restore-cr.yaml`。カッコ内の値を、Astra Controlの環境とクラスタの構成に合わせて更新します。
   - <CR_NAME>：このCR操作の名前。環境に適した適切な名前を選択します。
   - <APPVAULT_NAME>：バックアップコンテンツが格納されているAppVaultの名前。
   - <BACKUP_PATH>：バックアップコンテンツが格納されているAppVault内のパス。例：

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

   - <SOURCE_NAMESPACE>：リストア処理のソースネームスペース。
   - <DESTINATION_NAMESPACE>：リストア処理のデスティネーションネームスペース。

     ```
     apiVersion: astra.netapp.io/v1
     kind: BackupRestore
     metadata:
       name: <CR_NAME>
       namespace: astra-connector
     spec:
       appVaultRef: <APPVAULT_NAME>
       appArchivePath: <BACKUP_PATH>
       namespaceMapping: [{"source": "<SOURCE_NAMESPACE>",
     "destination": "<DESTINATION_NAMESPACE>"}]
     ```

2. （オプション）リストアするアプリケーションの特定のリソースのみを選択する必要がある場合は、特定のラベルが付いたリソースを含めるか除外するフィルタリングを追加します。
   - 「<INCLUDE-EXCLUDE>」：_（フィルタリングに必要）_使用 `include` または `exclude` resourceMatchersで定義されているリソースを含めるか除外します。次のresourceMatchersパラメータを追加して、追加または除外するリソースを定義します。
     - <GROUP>：_（オプション）_フィルタリングするリソースのグループ。
     - <KIND>：_（オプション）_フィルタリングするリソースの種類。
     - <VERSION>：_（オプション）_フィルタリングするリソースのバージョン。
     - <NAMES>：（オプション） namesをフィルタリングするリソースのKubernetes metadata.nameフィールドに入力します。
     - <NAMESPACES>：_（オプション）_Namespacesフィルタリングするリソースのkubernetes metadata.nameフィールド。
     - <SELECTORS>：_（オプション）_で定義されているリソースのKubernetes metadata.nameフィールドのラベルセレクタ文字列 "Kubernetes のドキュメント"。例

"trident.netapp.io/os=linux"。

例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. データを入力した後、 `astra-control-backup-restore-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-control-backup-restore-cr.yaml
```

CRを使用したバックアップから元のネームスペースへのリストア

手順

1. カスタムリソース（CR）ファイルを作成して名前を付けます。 `astra-control-backup-ipr-cr.yaml`。カッコ内の値を、Astra Controlの環境とクラスタの構成に合わせて更新します。
   - <CR_NAME>：このCR操作の名前。環境に適した適切な名前を選択します。
   - <APPVAULT_NAME>：バックアップコンテンツが格納されているAppVaultの名前。
   - <BACKUP_PATH>：バックアップコンテンツが格納されているAppVault内のパス。例：

```
ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
20231213023800_94347756-9d9b-401d-a0c3
```

```
apiVersion: astra.netapp.io/v1
kind: BackupInplaceRestore
metadata:
  name: <CR_NAME>
  namespace: astra-connector
spec:
  appVaultRef: <APPVAULT_NAME>
  appArchivePath: <BACKUP_PATH>
```

2. （オプション）リストアするアプリケーションの特定のリソースのみを選択する必要がある場合は、

特定のラベルが付いたリソースを含めるか除外するフィルタリングを追加します。

- 「<INCLUDE-EXCLUDE>」：_（フィルタリングに必要）_使用 `include` または `exclude` resourceMatchersで定義されているリソースを含めるか除外します。次のresourceMatchersパラメータを追加して、追加または除外するリソースを定義します。
  - <GROUP>：_（オプション）_フィルタリングするリソースのグループ。
  - <KIND>：_（オプション）_フィルタリングするリソースの種類。
  - <VERSION>：_（オプション）_フィルタリングするリソースのバージョン。
  - <NAMES>：（オプション） namesをフィルタリングするリソースのKubernetes metadata.nameフィールドに入力します。
  - <NAMESPACES>：_（オプション）_Namespacesフィルタリングするリソースのkubernetes metadata.nameフィールド。
  - <SELECTORS>：_（オプション）_で定義されているリソースのKubernetes metadata.nameフィールドのラベルセレクタ文字列 "Kubernetes のドキュメント"。例 `"trident.netapp.io/os=linux"`。

    例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. データを入力した後、 `astra-control-backup-ipr-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-control-backup-ipr-cr.yaml
```

[テクニカルプレビュー]カスタムリソースを使用した Snapshot**からのリストア（**CR**）**

カスタムリソース（CR）ファイルを使用して、スナップショットから別のネームスペースまたは元のソースネームスペースにデータをリストアできます。

CRを使用した**Snapshot**からのリストア

手順

1. カスタムリソース（CR）ファイルを作成して名前を付けます。 `astra-control-snapshot-restore-cr.yaml`。カッコ内の値を、Astra Controlの環境とクラスタの構成に合わせて更新します。
   - <CR_NAME>：このCR操作の名前。環境に適した適切な名前を選択します。
   - <APPVAULT_NAME>：バックアップコンテンツが格納されているAppVaultの名前。
   - <BACKUP_PATH>：バックアップコンテンツが格納されているAppVault内のパス。例：

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

   - <SOURCE_NAMESPACE>：リストア処理のソースネームスペース。
   - <DESTINATION_NAMESPACE>：リストア処理のデスティネーションネームスペース。

     ```
     apiVersion: astra.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: <CR_NAME>
       namespace: astra-connector
     spec:
       appArchivePath: <BACKUP_PATH>
       appVaultRef: <APPVAULT_NAME>
       namespaceMapping: [{"source": "<SOURCE_NAMESPACE>",
     "destination": "<DESTINATION_NAMESPACE>"}]
     ```

2. （オプション）リストアするアプリケーションの特定のリソースのみを選択する必要がある場合は、特定のラベルが付いたリソースを含めるか除外するフィルタリングを追加します。
   - 「<INCLUDE-EXCLUDE>」：_（フィルタリングに必要）_使用 `include` または `exclude` resourceMatchersで定義されているリソースを含めるか除外します。次のresourceMatchersパラメータを追加して、追加または除外するリソースを定義します。
     - <GROUP>：_（オプション）_フィルタリングするリソースのグループ。
     - <KIND>：_（オプション）_フィルタリングするリソースの種類。
     - <VERSION>：_（オプション）_フィルタリングするリソースのバージョン。
     - <NAMES>：（オプション） namesをフィルタリングするリソースのKubernetes metadata.nameフィールドに入力します。
     - <NAMESPACES>：_（オプション）_Namespacesフィルタリングするリソースのkubernetes metadata.nameフィールド。
     - <SELECTORS>：_（オプション）_で定義されているリソースのKubernetes metadata.nameフィールドのラベルセレクタ文字列 "Kubernetes のドキュメント"。例

"trident.netapp.io/os=linux"。

例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. データを入力した後、 `astra-control-snapshot-restore-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-control-snapshot-restore-cr.yaml
```

CRを使用した**Snapshot**から元のネームスペースへのリストア

手順

1. カスタムリソース（CR）ファイルを作成して名前を付けます。 `astra-control-snapshot-ipr-cr.yaml`。カッコ内の値を、Astra Controlの環境とクラスタの構成に合わせて更新します。
   - <CR_NAME>：このCR操作の名前。環境に適した適切な名前を選択します。
   - <APPVAULT_NAME>：バックアップコンテンツが格納されているAppVaultの名前。
   - <BACKUP_PATH>：バックアップコンテンツが格納されているAppVault内のパス。例：

```
ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
20231213023800_94347756-9d9b-401d-a0c3
```

```
apiVersion: astra.netapp.io/v1
kind: SnapshotInplaceRestore
metadata:
  name: <CR_NAME>
  namespace: astra-connector
spec:
  appArchivePath: <BACKUP_PATH>
  appVaultRef: <APPVAULT_NAME>
```

2. （オプション）リストアするアプリケーションの特定のリソースのみを選択する必要がある場合は、

特定のラベルが付いたリソースを含めるか除外するフィルタリングを追加します。

- 「<INCLUDE-EXCLUDE>」：_（フィルタリングに必要）_使用 `include` または `exclude` resourceMatchersで定義されているリソースを含めるか除外します。次のresourceMatchersパラメータを追加して、追加または除外するリソースを定義します。
  - <GROUP>：_（オプション）_フィルタリングするリソースのグループ。
  - <KIND>：_（オプション）_フィルタリングするリソースの種類。
  - <VERSION>：_（オプション）_フィルタリングするリソースのバージョン。
  - <NAMES>：（オプション） namesをフィルタリングするリソースのKubernetes metadata.nameフィールドに入力します。
  - <NAMESPACES>：_（オプション）_NamespacesフィルタリングするリソースのKubernetes metadata.nameフィールド。
  - <SELECTORS>：_（オプション）_で定義されているリソースのKubernetes metadata.nameフィールドのラベルセレクタ文字列 "Kubernetes のドキュメント"。例 `"trident.netapp.io/os=linux"`。

    例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. データを入力した後、 `astra-control-snapshot-ipr-cr.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-control-snapshot-ipr-cr.yaml
```

結果

Astra Control 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コンテンツが、リストアしたアプリケーションの永続ボリュームのコンテンツ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、またはリストア）が完了して永続ボリュームのサイズを変更したあと、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。

ネームスペースの名前/ IDまたはネームスペースのラベルでネームスペースの制約を受けているメンバーユーザは、同じクラスタの新しいネームスペース、または組織のアカウントに含まれる他のクラスタにアプリケーションをクローニング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。クローン処理またはリストア処理で新しいネームスペースが作成されたあと、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、影響を受けるユーザのロールの制約を更新して、新しいネームスペースへのアクセスを許可できます。

アプリケーションのリストア中にリソースをフィルタリングします

にフィルタルールを追加できます "リストア" リストアされたアプリケーションに含める、またはリストアされたアプリケーションから除外する既存のアプリケーションリソースを指定する処理。指定した名前空間、ラベル、またはGVK（GroupVersionKind）に基づいて、リソースを含めたり除外したりできます。

[Include（含める）]および**[Exclude（除外）]**のシナリオ

- 元のネームスペースを使用する包含ルールを選択した場合（インプレースリストア）：ルールで定義した既存のアプリケーションリソースは削除され、リストアに使用する選択したSnapshotまたはバックアップのリソースで置き換えられます。includeルールで指定しないリソースは変更されません。
- 新しい名前空間を持つ**include**ルールを選択した場合：このルールを使用して、リストアされたアプリケーションで使用する特定のリソースを選択します。対象ルールに指定しないリソースは、リストアされたアプリケーションには含まれません。
- 元のネームスペースを含む除外ルールを選択した場合（インプレースリストア）：除外するように指定したリソースはリストアされず、変更されません。除外するように指定しないリソースは、スナップショットまたはバックアップからリストアされます。対応するStatefulSetがフィルタリングされたリソースに含まれている場合、永続ボリューム上のすべてのデータが削除されて再作成されます。
- 新しい名前空間を持つ除外ルールを選択した場合:このルールを使用して、リストアされたアプリケーションから削除する特定のリソースを選択します。除外するように指定しないリソースは、スナップショットまたはバックアップからリストアされます。

ルールには、includeまたはexcludeタイプがあります。リソースの包含と除外を組み合わせたルールは使用できません。

手順

1. リソースをフィルタするように選択し、[アプリケーションのリストア]ウィザードで[含める]または[除外するルールを追加する]を選択したら、*[除外するルールを追加する]*を選択します。

   Astra Controlで自動的に追加されるクラスタ対象のリソースを除外することはできません。

2. フィルタルールを設定します。

   ネームスペース、ラベル、またはGVKを少なくとも1つ指定する必要があります。フィルタルールを適用したあとに保持するリソースがあれば、リストアしたアプリケーションを正常な状態に保つのに十分であることを確認してください。

   a. ルールの特定のネームスペースを選択します。選択しない場合は、すべての名前空間がフィルタで使用されます。

アプリケーションに複数のネームスペースが含まれていた場合、新しいネームスペースにリストアすると、リソースが含まれていなくてもすべてのネームスペースが作成されます。

b. （オプション）リソース名を入力します。

c. （任意）ラベルセレクタ：を含めます "ラベルセレクタ" をクリックしてルールに追加します。ラベルセレクタは、選択したラベルに一致するリソースのみをフィルタリングするために使用されます。

d. （オプション）[Use GVK (GroupVersionKind) set]を選択してリソースをフィルタリング*し、追加のフィルタリングオプションを指定します。

GVKフィルタを使用する場合は、バージョンと種類を指定する必要があります。

i. （オプション）* Group *：ドロップダウンリストからKubernetes APIグループを選択します。

ii. 種類：ドロップダウンリストから、フィルタで使用するKubernetesリソースタイプのオブジェクトスキーマを選択します。

iii. バージョン：Kubernetes APIのバージョンを選択します。

3. エントリに基づいて作成されたルールを確認します。

4. 「 * 追加」を選択します。

ルールを含むリソースと除外するリソースは必要なだけ作成できます。処理を開始する前に、リストアアプリケーションの概要にルールが表示されます。

リソースを別のアプリケーションと共有するアプリケーションでは、インプレースリストアが複雑になります

リソースを別のアプリケーションと共有し、意図しない結果を生成するアプリケーションに対して、インプレースリストア処理を実行できます。アプリケーション間で共有されているリソースは、いずれかのアプリケーションでインプレースリストアが実行されると置き換えられます。

次に、NetApp SnapMirrorレプリケーションを使用してリストアすると望ましくない状況が発生するシナリオの例を示します。

1. アプリケーションを定義します `app1` ネームスペースを使用する `ns1`。
2. のレプリケーション関係を設定します `app1`。
3. アプリケーションを定義します `app2` （同じクラスタ上）ネームスペースを使用します `ns1` および `ns2`。
4. のレプリケーション関係を設定します `app2`。
5. のレプリケーションを反転した `app2`。これにより、が起動します `app1` 非アクティブ化するソースクラスタ上のアプリケーション。

SnapMirrorテクノロジを使用してストレージバックエンド間でアプリケーションをレプリケート

Astra Controlを使用すると、NetApp SnapMirrorテクノロジの非同期レプリケーション機能を使用して、RPO（目標復旧時点）とRTO（目標復旧時間）の低いアプリケーション

のビジネス継続性を構築できます。設定が完了すると、アプリケーションは、ストレージバックエンド間、同じクラスタ上、または異なるクラスタ間でデータやアプリケーションの変更をレプリケートできるようになります。

バックアップ/リストアとレプリケーションの比較については、を参照してください "データ保護の概念"。

アプリケーションは、オンプレミスのみ、ハイブリッド、マルチクラウドなど、さまざまなシナリオでレプリケートできます。

- オンプレミスサイトAからオンプレミスサイトAへ
- オンプレミスサイトAからオンプレミスサイトBへ
- Cloud Volumes ONTAPでオンプレミスからクラウドへ
- Cloud Volumes ONTAPを使用したクラウドからオンプレミスへの移行
- Cloud Volumes ONTAP を使用したクラウドからクラウドへ（同じクラウドプロバイダ内の異なるリージョン間または異なるクラウドプロバイダ間）

Astra Controlを使用すれば、オンプレミスのクラスタからクラウドへ（Cloud Volumes ONTAP を使用）、またはクラウド間（Cloud Volumes ONTAP からCloud Volumes ONTAP へ）にアプリケーションをレプリケートできます。

> 別のアプリケーションを逆方向に同時に複製できます。たとえば、アプリケーションA、B、Cはデータセンター1からデータセンター2にレプリケートでき、アプリケーションX、Y、Zはデータセンター2からデータセンター1にレプリケートできます。

Astra Controlを使用すると、アプリケーションのレプリケーションに関連する次のタスクを実行できます。

- [レプリケーション関係を設定]
- [デスティネーションクラスタでレプリケートされたアプリケーションをオンラインにする（フェイルオーバー）]
- [フェイルオーバーしたレプリケーションを再同期します]
- [アプリケーションのレプリケーションを反転する]
- [アプリケーションを元のソースクラスタにフェイルバックします]
- [アプリケーションレプリケーション関係を削除します]

レプリケーションの前提条件

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

ONTAP クラスタ

- * Astra Control ProvisionerまたはAstra Trident *：Astra Control ProvisionerまたはAstra Tridentは、ONTAPをバックエンドとして利用するソースとデスティネーションの両方のKubernetesクラスタに存在する必要があります。Astra Controlは、次のドライバでサポートされるストレージクラスを使用して、NetApp SnapMirrorテクノロジによるレプリケーションをサポートします。
  - `ontap-nas`

◦ `ontap-san`

- ライセンス：Data Protection Bundleを使用するONTAP SnapMirror非同期ライセンスが、ソースとデスティネーションの両方のONTAPクラスタで有効になっている必要があります。を参照してください "ONTAPのSnapMirrorライセンスの概要" を参照してください。

ピアリング

- *クラスタとSVM *：ONTAPストレージバックエンドにピア関係が設定されている必要があります。を参照してください "クラスタと SVM のピアリングの概要" を参照してください。

> 2つのONTAPクラスタ間のレプリケーション関係で使用されるSVM名が一意であることを確認してください。

- * Astra Control ProvisionerまたはAstra TridentとSVM *：ピア関係にあるリモートSVMは、デスティネーションクラスタのAstra Control ProvisionerまたはAstra Tridentで使用できる必要があります。

> **Astra Control Center** の略
>
> "Astra Control Centerを導入" シームレスなディザスタリカバリのための第3の障害ドメインまたはセカンダリサイト。

- マネージドバックエンド：レプリケーション関係を作成するには、Astra Control CenterでONTAPストレージバックエンドを追加および管理する必要があります。

> Astra Control Provisionerを有効にしている場合、Astra Control CenterでONTAPストレージバックエンドの追加と管理はオプションです。

- 管理対象クラスタ：Astra Controlを使用して次のクラスタを追加、管理できます。理想的には、異なる障害ドメインまたはサイトに配置されます。
  - ソースKubernetesクラスタ
  - デスティネーションKubernetesクラスタ
  - 関連付けられているONTAPクラスタ
- ユーザアカウント：ONTAPストレージバックエンドをAstra Control Centerに追加する場合は、「admin」ロールのユーザクレデンシャルを適用します。このロールにはアクセス方法があります `http` および `ontapi` ONTAP ソースとデスティネーションの両方のクラスタで有効にします。を参照してください "ONTAP ドキュメントの「ユーザーアカウントの管理」を参照してください" を参照してください。

> Astra Control Provisioner機能では、Astra Control Centerでクラスタを管理するために「admin」ロールを明確に定義する必要はありません。これらのクレデンシャルはAstra Control Centerでは必要ありません。

> Astra Control Centerでは、NVMe over TCPプロトコルを使用するストレージバックエンドのNetApp SnapMirrorレプリケーションはサポートされません。

Astra Trident / ONTAP 構成

Astra Control Centerでは、ソースとデスティネーションの両方のクラスタのレプリケーションをサポートするストレージバックエンドを少なくとも1つ設定する必要があります。ソースクラスタとデスティネーションクラスタが同じである場合は、耐障害性を最大限に高めるために、デスティネーションアプリケーションでソ

ースアプリケーションとは別のストレージバックエンドを使用する必要があります。

> Astra Controlレプリケーションでは、単一のストレージクラスを使用するアプリケーションがサポートされます。ネームスペースにアプリケーションを追加するときは、そのアプリケーションのストレージクラスがネームスペース内の他のアプリケーションと同じであることを確認してください。レプリケートされたアプリケーションにPVCを追加するときは、新しいPVCのストレージクラスがネームスペース内の他のPVCと同じであることを確認してください。

レプリケーション関係を設定

レプリケーション関係の設定には、次の作業が含まれます。

- Astra ControlでアプリケーションSnapshotを作成する頻度を選択します（アプリケーションのKubernetesリソースと、アプリケーションの各ボリュームのボリュームSnapshotが含まれます）。
- レプリケーションスケジュールの選択（Kubernetesリソースと永続ボリュームデータを含む）
- Snapshotの作成時間の設定

手順

1. Astra Controlの左ナビゲーションから、「アプリケーション」を選択します。
2. >[レプリケーション]*タブを選択します。
3. [レプリケーションポリシーの設定]*を選択します。または、[アプリケーション保護]ボックスから[アクション]オプションを選択し、[レプリケーションポリシーの構成]を選択します。
4. 次の情報を入力または選択します。
   - デスティネーションクラスタ：デスティネーションクラスタを入力します（ソースクラスタと同じでもかまいません）。
   - デスティネーションストレージクラス：デスティネーションONTAPクラスタのピアSVMを使用するストレージクラスを選択または入力します。ベストプラクティスとして、デスティネーションストレージクラスでソースストレージクラスとは別のストレージバックエンドを指定することを推奨します。
   - レプリケーションタイプ： `Asynchronous` は、現在使用可能な唯一のレプリケーションタイプです。
   - デスティネーションネームスペース：デスティネーションクラスタの新規または既存のデスティネーションネームスペースを入力します。
   - （任意）[Add namespace]を選択し、ドロップダウンリストからネームスペースを選択して、ネームスペースを追加します。
   - レプリケーション頻度：Astra ControlでSnapshotを作成してデスティネーションにレプリケートする頻度を設定します。
   - オフセット：Astra ControlでSnapshotを作成する時間（分）を設定します。オフセットを使用すると、他のスケジュールされた処理と競合しないようにすることができます。

   > バックアップとレプリケーションのスケジュールをオフセットして、スケジュールの重複を回避します。たとえば、1時間ごとに1時間の最上部にバックアップを実行し、オフセットを5分、間隔を10分に設定してレプリケーションを開始するようにスケジュールを設定します。

5. 「次へ」を選択し、概要を確認して、「保存」を選択します。

最初に、最初のスケジュールが実行される前にステータスに「app_mirror」と表示されます。

Astra Controlが、レプリケーションに使用するアプリケーションSnapshotを作成。

6. アプリケーションのスナップショットステータスを確認するには、**[**アプリケーション**]**>*[スナップショット]*タブを選択します。

   Snapshot名の形式は次のとおりです。 `replication-schedule-<string>`。Astra Controlは、レプリケーションに使用された最後のSnapshotを保持します。古いレプリケーションSnapshotは、レプリケーションが正常に完了すると削除されます。

結果

これにより、レプリケーション関係が作成されます。

Astra Controlは、関係を確立した結果として次のアクションを実行します。

- デスティネーションにネームスペースを作成します（存在しない場合）。
- 送信元アプリケーションのPVCに対応する宛先ネームスペースにPVCを作成します。
- アプリケーションと整合性のある初期スナップショットを作成します。
- 最初のSnapshotを使用して、永続ボリュームのSnapMirror関係を確立します。

[データ保護]*ページには、レプリケーション関係の状態とステータスが表示されます。
<Health status>|<Relationship life cycle state>

たとえば、Normal | Establishedです

レプリケーションの状態とステータスの詳細については、このトピックの最後を参照してください。

デスティネーションクラスタでレプリケートされたアプリケーションをオンラインにする（フェイルオーバー）

Astra Controlを使用すると、レプリケートされたアプリケーションをデスティネーションクラスタにフェイルオーバーできます。この手順 はレプリケーション関係を停止し、デスティネーションクラスタでアプリケーションをオンラインにします。ソースクラスタのアプリケーションが稼働していた場合、この手順 はそのアプリケーションを停止しません。

手順

1. Astra Controlの左ナビゲーションから、「アプリケーション」を選択します。
2. **>**[レプリケーション]*タブを選択します。
3. [アクション]メニューから*[フェイルオーバー]*を選択します。
4. フェイルオーバーページで、情報を確認し、*フェイルオーバー*を選択します。

結果

フェイルオーバー手順が発生すると、次の処理が実行されます。

- デスティネーションアプリケーションは、最新のレプリケートされたSnapshotに基づいて起動されます。

- ソースクラスタとアプリケーション（動作している場合）は停止されず、引き続き実行されます。
- レプリケーションの状態は「フェイルオーバー」に変わり、完了すると「フェイルオーバー」に変わります。
- ソースアプリの保護ポリシーは、フェイルオーバー時にソースアプリに存在するスケジュールに基づいて、デスティネーションアプリにコピーされます。
- ソースアプリで1つ以上のリストア後の実行フックが有効になっている場合、それらの実行フックはデスティネーションアプリに対して実行されます。
- Astra Controlには、ソースクラスタとデスティネーションクラスタの両方のアプリケーションと、それぞれの健全性が表示されます。

フェイルオーバーしたレプリケーションを再同期します

再同期処理によってレプリケーション関係が再確立されます。関係のソースを選択して、ソースクラスタまたはデスティネーションクラスタにデータを保持することができます。この処理は、SnapMirror関係を再確立し、ボリュームのレプリケーションを任意の方向に開始します。

レプリケーションを再確立する前に、新しいデスティネーションクラスタ上のアプリケーションが停止されます。

再同期プロセスの間、ライフサイクルの状態は「Establishing」と表示されます。

手順

1. Astra Controlの左ナビゲーションから、「アプリケーション」を選択します。
2. >[レプリケーション]*タブを選択します。
3. [操作]メニューから*[再同期]*を選択します。
4. 再同期（Resync）ページで、保持するデータを含むソースまたはデスティネーションのアプリケーションインスタンスを選択します。

   デスティネーションのデータが上書きされるため、再同期元は慎重に選択してください。

5. 続行するには、* Resync *を選択します。
6. 「resync」と入力して確定します。
7. 「* Yes、resync *」を選択して終了します。

結果

- Replication（レプリケーション）ページに、レプリケーションステータスとしてEstablishing（確立）が表示されます。
- Astra Controlは、新しいデスティネーションクラスタのアプリケーションを停止します。
- SnapMirror resyncを使用して、指定した方向に永続的ボリュームのレプリケーションを再確立します。
- [レプリケーション]ページに、更新された関係が表示されます。

アプリケーションのレプリケーションを反転する

これは、アプリケーションをデスティネーションストレージバックエンドに移動し、元のソースストレージバックエンドに引き続きレプリケートするという計画的な処理です。Astra Controlは、デスティネーションアプ

リケーションにフェイルオーバーする前に、ソースアプリケーションを停止してデスティネーションにデータをレプリケートします。

この状況では、ソースとデスティネーションを交換しようとしています。

手順

1. Astra Controlの左ナビゲーションから、「アプリケーション」を選択します。
2. >[レプリケーション]*タブを選択します。
3. [操作]メニューから*[逆レプリケーション]*を選択します。
4. リバース・レプリケーションのページで情報を確認し、「リバース・レプリケーション」を選択して続行します。

結果

リバースレプリケーションの結果、次の処理が実行されます。

- 元のソースアプリのKubernetesリソースのスナップショットが作成されます。
- 元のソースアプリケーションのポッドは、アプリケーションのKubernetesリソースを削除することで正常に停止されます（PVCとPVはそのまま維持されます）。
- ポッドがシャットダウンされると、アプリのボリュームのスナップショットが取得され、レプリケートされます。
- SnapMirror関係が解除され、デスティネーションボリュームが読み取り/書き込み可能な状態になります。
- アプリのKubernetesリソースは、元のソースアプリがシャットダウンされた後に複製されたボリュームデータを使用して、シャットダウン前のスナップショットから復元されます。
- 逆方向にレプリケーションが再確立されます。

アプリケーションを元のソースクラスタにフェイルバックします

Astra Controlを使用すると、フェイルオーバー処理後に次の一連の処理を使用して「フェイルバック」を実現できます。このワークフローでは、レプリケーションの方向を元に戻すために、Astra Controlがアプリケーションの変更を元のソースアプリケーションにレプリケート（再同期）してからレプリケーションの方向を反転します。

このプロセスは、デスティネーションへのフェイルオーバーが完了した関係から開始し、次の手順を実行します。

- フェイルオーバー状態から開始します。
- 関係を再同期します。
- レプリケーションを反転する。

手順

1. Astra Controlの左ナビゲーションから、「アプリケーション」を選択します。
2. >[レプリケーション]*タブを選択します。
3. [操作]メニューから*[再同期]*を選択します。
4. フェイルバック処理の場合は、フェイルオーバーしたアプリケーションを再同期処理のソースとして選択します（フェイルオーバー後に書き込まれたデータは保持されます）。

5. 「resync」と入力して確定します。
6. 「* Yes、resync *」を選択して終了します。
7. 再同期が完了したら、[データ保護（Data Protection）]>[レプリケーション（Replication）]タブの[アクション（Actions）]メニューから[*レプリケーションを反転（Reverse replication）]を選択します。
8. リバース・レプリケーションのページで、情報を確認し、*リバース・レプリケーション*を選択します。

結果

このコマンドは、「resync」処理と「reverse relationship」処理の結果を組み合わせて、レプリケーションが再開された元のソースクラスタ上のアプリケーションを元のデスティネーションクラスタにオンラインにします。

アプリケーションレプリケーション関係を削除します

関係を削除すると、2つの異なるアプリケーション間に関係がなくなります。

手順

1. Astra Controlの左ナビゲーションから、「アプリケーション」を選択します。
2. >[レプリケーション]*タブを選択します。
3. [アプリケーションの保護]ボックスまたは関係図で、*[レプリケーション関係の削除]*を選択します。

結果

レプリケーション関係を削除すると、次の処理が実行されます。

- 関係が確立されていても、アプリケーションがデスティネーションクラスタでオンラインになっていない（フェイルオーバーした）場合、Astra Controlは、初期化中に作成されたPVCを保持し、「空」の管理対象アプリケーションをデスティネーションクラスタに残します。また、作成されたバックアップを保持するためにデスティネーションアプリケーションを保持します。
- アプリケーションがデスティネーションクラスタでオンラインになった（フェイルオーバーした）場合、Astra ControlはPVCと宛先アプリケーションを保持します。ソースとデスティネーションのアプリケーションは、独立したアプリケーションとして扱われるようになりました。バックアップスケジュールは、両方のアプリケーションで維持されますが、相互に関連付けられていません。

レプリケーション関係のヘルスステータスと関係のライフサイクル状態

Astra Controlには、関係の健全性と、レプリケーション関係のライフサイクルの状態が表示されます。

レプリケーション関係のヘルスステータス

レプリケーション関係の健常性は、次のステータスで示されます。

- 正常：関係が確立中または確立されており、最新のSnapshotが転送されました。
- 警告：関係がフェイルオーバーされているかフェイルオーバーされています（そのためソースアプリは保護されなくなりました）。
- * 重要 *
  - 関係が確立されているか、フェイルオーバーされていて、前回の調整が失敗しました。
  - 関係が確立され、新しいPVCの追加を最後に調整しようとしても失敗しています。

- 関係は確立されていますが（成功したSnapshotがレプリケートされ、フェイルオーバーが可能です）、最新のSnapshotはレプリケートに失敗したか失敗しました。

レプリケーションのライフサイクル状態

次の状態は、レプリケーションのライフサイクルの各段階を表しています。

- * Establishing *：新しいレプリケーション関係を作成中です。Astra Controlは、必要に応じてネームスペースを作成し、デスティネーションクラスタの新しいボリュームにPersistent Volumeクレーム（PVC；永続ボリューム要求）を作成し、SnapMirror関係を作成します。このステータスは、レプリケーションが再同期中であること、またはレプリケーションを反転中であることを示している可能性もあり
- * established *：レプリケーション関係が存在します。Astra Controlは、PVCが使用可能であることを定期的にチェックし、レプリケーション関係をチェックし、アプリケーションのSnapshotを定期的に作成し、アプリケーション内の新しいソースPVCを特定します。その場合は、レプリケーションに含めるリソースがAstra Controlによって作成されます。
- フェイルオーバー：Astra Controlは、SnapMirror関係を解除し、最後にレプリケートされたアプリケーションのSnapshotからアプリケーションのKubernetesリソースをリストアします。
- フェイルオーバー：Astra Controlは、ソースクラスタからのレプリケーションを停止し、デスティネーションで最新の（成功した）レプリケートされたアプリケーションSnapshotを使用して、Kubernetesリソースをリストアします。
- * resyncing *：Astra Controlは、SnapMirror resyncを使用して、再同期元の新しいデータを再同期先に再同期します。この処理では、同期の方向に基づいて、デスティネーション上の一部のデータが上書きされる可能性があります。Astra Controlは、デスティネーションネームスペースで実行されているアプリケーションを停止し、Kubernetesアプリケーションを削除します。再同期処理の実行中、ステータスは「Establishing」と表示されます。
- リバース：は、元のソースクラスタへのレプリケーションを続行しながらアプリケーションをデスティネーションクラスタに移動する予定の処理です。Astra Controlは、ソースクラスタ上のアプリケーションを停止し、デスティネーションにデータをレプリケートしてから、デスティネーションクラスタにアプリケーションをフェイルオーバーします。リバースレプリケーションの間、ステータスは「Establishing」と表示されます。
- 削除中：
  - レプリケーション関係が確立されたものの、まだフェイルオーバーされていない場合は、レプリケーション中に作成されたPVCがAstra Controlによって削除され、デスティネーションの管理対象アプリケーションが削除されます。
  - レプリケーションがすでにフェイルオーバーされている場合、Astra ControlはPVCと宛先アプリケーションを保持します。

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できます。Astra Control でアプリケーションをクローニングすると、アプリケーション構成と永続的ストレージのクローンが作成されます。

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 Control Center UIまたはを使用できます "Astra Control API の略" アプリケーションのクローン作成と移行を実行します。

作業を開始する前に

- デスティネーションボリュームを確認：別のストレージクラスにクローニングする場合は、ストレージクラスで同じ永続ボリュームアクセスモード（ReadWriteManyなど）が使用され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的ボリュームのアクセスモードが異なると、クローニング処理は失敗します。たとえば、ソースの永続ボリュームがRWXアクセスモードを使用している場合は、Azure Managed Disks、AWS EBS、Google Persistent Disk、など、RWXを提供できないデスティネーションストレージクラスを選択します `ontap-san`を指定すると、クローン処理は失敗します。原因は失敗します。永続ボリュームのアクセスモードの詳細については、を参照してください "Kubernetes" ドキュメント
- 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、ソースクラスタとデスティネーションクラスタを含むクラウドインスタンス（同じでない場合）にデフォルトのバケットを用意する必要があります。クラウドインスタンスごとにデフォルトのバケットを割り当てる必要があります。
- クローン処理中に、IngressClassリソースまたはwebhookを必要とするアプリケーションが正常に機能するためには、これらのリソースがデスティネーションクラスタですでに定義されていない必要があります。

> OpenShift 環境でのアプリケーションのクローニングでは、 Astra Control Center が OpenShift でボリュームをマウントし、ファイルの所有権を変更できるようにする必要があります。そのため、これらの処理を許可するには、 ONTAP ボリュームのエクスポートポリシーを設定する必要があります。次のコマンドを使用して実行できます。
>
> 1. `export-policy rule modify -vserver <storage virtual machine name> -policyname <policy name> -ruleindex 1 -superuser sys`
> 2. `export-policy rule modify -vserver <storage virtual machine name> -policyname <policy name> -ruleindex 1 -anon 65534`

クローンの制限事項

- 明示的なストレージクラス：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導入し、そのアプリケーションのクローンを作成する必要がある場合、ターゲットクラスタには元 々 指定されたストレージクラスが必要です。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。
- * ontap-nas-economy-basedアプリケーション*：アプリケーションのストレージクラスが `ontap-nas-economy` ドライバ。ただし、 "ONTAP NAS経済性に優れた運用向けのバックアップとリストアを実現"。
- クローンとユーザーの制約：名前空間の名前/ IDまたは名前空間のラベルによって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任意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。クローン処理またはリストア処理で新しいネームスペースが作成されたあと、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、影響を受けるユーザのロールの制約を更新して、新しいネームスペースへのアクセスを許可できます。
- クローンはデフォルトバケットを使用：アプリケーションのバックアップまたはアプリケーションのリストア時に、オプションでバケットIDを指定できます。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。
- * Jenkins CI*を使用：オペレータがデプロイした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復元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。
- * S3バケットを使用している場合*：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。

- *特定のバージョンのPostgreSQL *：同じクラスタ内のアプリケーションクローンは、Bitnami PostgreSQL 11.5.0チャートで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。

OpenShift に関する考慮事項

- クラスタおよび**OpenShift**バージョン：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。
- *プロジェクトおよびUID *：OpenShiftクラスタでアプリ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名前空間）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 wordpress
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. クローンの詳細を指定します。
   - 名前を入力します。
   - 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。
   - クローンのデスティネーションネームスペースを入力してください。アプリケーションに関連付けられた各ソースネームスペースは、定義した宛先ネームスペースにマッピングされます。

     Astra Controlでは、クローニング処理の一環として新しいデスティネーションネームスペースが作成されます。指定するデスティネーションネームスペースがデスティネーションクラスタに存在していないことを確認してください。

   - 「 * 次へ * 」を選択します。
   - アプリケーションに関連付けられている元のストレージクラスを保持するか、別のストレージクラスを選択します。

アプリケーションのストレージクラスをネイティブクラウドプロバイダのストレージクラスまたはサポートされているその他のストレージクラスに移行したり、 `ontap-nas-economy` をバックアップされたストレージクラスに追加します `ontap-nas` を使用するか、から作成されたストレージクラスを含む別のクラスタにアプリケーションをコピーします `ontap-nas-economy` ドライバ。

別のストレージクラスを選択し、このストレージクラスがリストア時に存在しない場合は、エラーが返されます。

5. 「 * 次へ * 」を選択します。
6. クローンに関する情報を確認し、* Clone *を選択します。

結果

Astra Controlは、入力した情報に基づいてアプリケーションをクローニングします。新しいアプリケーションクローンがに含まれている場合、クローニング処理は成功します `Healthy` 「アプリケーション」ページで説明します。

クローン処理またはリストア処理で新しいネームスペースが作成されたあと、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、影響を受けるユーザのロールの制約を更新して、新しいネームスペースへのアクセスを許可できます。

データ保護処理（クローン、バックアップ、またはリストア）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データ保護操作と組み合わせて実行するように構成できるカスタムアクションです。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。

実行フックのタイプ

Astra Control Centerでは、実行可能なタイミングに基づいて、次のタイプの実行フックがサポートされます。

- Snapshot前
- Snapshot後
- バックアップ前
- バックアップ後
- リストア後のPOSTコマンドです
- フェイルオーバー後

実行フックフィルタ

アプリケーションに実行フックを追加または編集するときに、実行フックにフィルタを追加して、フックが一致するコンテナを管理できます。フィルタは、すべてのコンテナで同じコンテナイメージを使用し、各イメージを別の目的（Elasticsearchなど）に使用するアプリケーションに便利です。フィルタを使用すると、一部の同一コンテナで実行フックが実行されるシナリオを作成できます。1つの実行フックに対して複数のフィルタを作成すると、それらは論理AND演算子と結合されます。実行フックごとに最大10個のアクティブフィルタを使用できます。

実行フックに追加する各フィルタは、正規表現を使用してクラスタ内のコンテナを照合します。フックがコンテナと一致すると、そのコンテナに関連付けられたスクリプトがフックによって実行されます。フィルタの正規表現では、正規表現2（RE2）構文を使用します。この構文では、一致リストからコンテナを除外するフィルタの作成はサポートされていません。実行フックフィルタの正規表現でAstra Controlがサポートする構文については、を参照してください "正規表現2（RE2）構文のサポート"。

> リストアまたはクローン処理のあとに実行される実行フックにネームスペースフィルタを追加し、リストアまたはクローンのソースとデスティネーションが異なるネームスペースにある場合、ネームスペースフィルタはデスティネーションネームスペースにのみ適用されます。

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。

> 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たり、完全に無効にしたり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。
> 実行フックが関連付けられている状態でバックアップまたはスナップショット操作を開始した後'キャンセルした場合でも'バックアップまたはスナップショット操作がすでに開始されていればフックは実行できますつまり、バックアップ後の実行フックで使用されるロジックは、バックアップが完了したとは見なされません。

- 新しいAstra Control環境では、実行フック機能はデフォルトで無効になっています。
  - 実行フックを使用する前に、実行フック機能を有効にする必要があります。
  - 所有者ユーザまたは管理者ユーザは、現在のAstra Controlアカウントで定義されているすべてのユーザの実行フック機能を有効または無効にできます。を参照してください [実行フック機能を有効にする] および [実行フック機能を無効にする] 手順については、を参照し
  - 機能の有効化ステータスは、Astra Controlのアップグレード中も維持されます。
- 実行フックは、スクリプトを使用してアクションを実行する必要があります。多くの実行フックは、同じスクリプトを参照できます。
- Astra Controlでは、実行フックが実行可能なシェルスクリプトの形式で記述されるようにするスクリプトが必要です。
- スクリプトのサイズは96KB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、バックアップ、または復元操作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'ソフトな障害ですフックが失敗しても'他のフックとデータ保護操作は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。
- オンデマンドのデータ保護処理では、すべてのフックイベントが生成され、*アクティビティ*ページのイベントログに保存されます。ただし、スケジュールされたデータ保護処理については、フック障害イベントだけがイベントログに記録されます（スケジュールされたデータ保護処理自体によって生成されたイベントは記録されたままです）。
- レプリケートされたソースアプリケーションをAstra Control Centerがデスティネーションアプリケーションにフェイルオーバーすると、フェイルオーバーの完了後にソースアプリケーションに対して有効になっているフェイルオーバー後の実行フックがデスティネーションアプリケーションに対して実行されます。

> Astra Control Center 23.04でリストア後のフックを実行していて、Astra Control Centerを23.07以降にアップグレードした場合、フェイルオーバーレプリケーション後にリストア後の実行フックが実行されなくなります。アプリケーションのフェイルオーバー後の実行フックを新しく作成する必要があります。また、フェイルオーバー用の既存のリストア後フックの処理タイプを「リストア後」から「フェイルオーバー後」に変更することもできます。

実行順序

データ保護操作を実行すると、実行フックイベントが次の順序で実行されます。

1. 適用可能なカスタムプリオペレーション実行フックは、適切なコンテナで実行されます。カスタムのプリオペレーションフックは必要なだけ作成して実行できますが、操作前のこれらのフックの実行順序は保証も構成もされていません。
2. データ保護処理が実行されます。
3. 適用可能なカスタムポストオペレーション実行フックは、適切なコンテナで実行されます。必要な数のカスタムポストオペレーションフックを作成して実行できますが、操作後のこれらのフックの実行順序は保証されず、設定もできません。

同じ種類の実行フック（スナップショット前など）を複数作成する場合、これらのフックの実行順序は保証されません。ただし、異なるタイプのフックの実行順序は保証されています。たとえば、すべての異なるタイプのフックを持つ構成の実行順序は次のようになります。

1. 予備フックが実行されます
2. スナップショット前フックが実行されます
3. スナップショット後フックが実行されます
4. バックアップ後のフックが実行されます
5. 復元後のフックが実行されます

シナリオ番号2のこの設定の例は、の表を参照してください [フックが実行されるかどうかを確認します]。

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Snapshotとバックアップをテストして整合性があることを確認します。これを行うには、アプリケーションを一時的なネームスペースにクローニングし、スナップショットまたはバックアップをリストアしてから、アプリケーションをテストします。

フックが実行されるかどうかを確認します

次の表を使用して、アプリケーションでカスタム実行フックが実行されるかどうかを判断します。

アプリケーションの高レベルの処理は、すべてスナップショット、バックアップ、またはリストアの基本的な処理のいずれかを実行することで構成されることに注意してください。シナリオによっては、クローニング処理はこれらの処理のさまざまな組み合わせで構成されるため、クローン処理を実行する実行フックはさまざまです。

In Placeリストア処理では既存のSnapshotまたはバックアップが必要になるため、これらの処理ではSnapshotまたはバックアップフックは実行されません。

開始してスナップショットを含むバックアップをキャンセルし'実行フックが関連付けられている場合は'一部のフックが実行され'ほかのフックが実行されないことがありますつまり、バックアップ後の実行フックでは、バックアップが完了したとは判断できません。キャンセルしたバックアップに関連する実行フックがある場合は、次の点に注意してください。

- バックアップ前およびバックアップ後のフックは常に実行されます。
- バックアップに新しいスナップショットが含まれており'スナップショットが開始されている場合は'スナップショット前フックとスナップショット後フックが実行されます
- スナップショットの開始前にバックアップがキャンセルされた場合は'スナップショット前フックとスナップショット後フックは実行されません

シナリオ（**Scenario**）	操作	既存の**Snapshot**	既存のバックアップ	ネームスペース	クラスタ	スナップショットフックが実行されます	バックアップフックが実行されます	フックを元に戻します	フェールオーバーフックの実行
1.	クローン	N	N	新規	同じ	Y	N	Y	N
2.	クローン	N	N	新規	違う	Y	Y	Y	N
3.	クローンまたはリストア	Y	N	新規	同じ	N	N	Y	N
4.	クローンまたはリストア	N	Y	新規	同じ	N	N	Y	N
5.	クローンまたはリストア	Y	N	新規	違う	N	N	Y	N

シナリオ（**Scenario**）	操作	既存の**Snapshot**	既存のバックアップ	ネームスペース	クラスタ	スナップショットフックが実行されます	バックアップフックが実行されます	フックを元に戻します	フェールオーバーフックの実行
6.	クローンまたはリストア	N	Y	新規	違う	N	N	Y	N
7.	リストア	Y	N	既存	同じ	N	N	Y	N
8.	リストア	N	Y	既存	同じ	N	N	Y	N
9.	スナップショット	該当なし	該当なし	該当なし	該当なし	Y	該当なし	該当なし	N
10.	バックアップ	N	該当なし	該当なし	該当なし	Y	Y	該当なし	N
11.	バックアップ	Y	該当なし	該当なし	該当なし	N	N	該当なし	N
12	フェイルオーバー	Y	該当なし	レプリケーションで作成	違う	N	N	N	Y
13	フェイルオーバー	Y	該当なし	レプリケーションで作成	同じ	N	N	N	Y

実行フックの例

にアクセスします "NetApp Verda GitHubプロジェクト" Apache CassandraやElasticsearchなどの一般的なアプリケーションの実行フックをダウンロードします。また、独自のカスタム実行フックを構築するための例やアイデアを得ることもできます。

実行フック機能を有効にする

所有者または管理者ユーザーの場合は、実行フック機能を有効にできます。この機能を有効にすると、このAstra Controlアカウントで定義されているすべてのユーザが実行フックを使用して、既存の実行フックとフックスクリプトを表示できます。

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。
3. *実行フックを有効にする*を選択します。

   アカウント>*機能設定*タブが表示されます。
4. Execution Hooks*ペインで、設定メニューを選択します。
5. [有効] を選択します。
6. 表示されるセキュリティ警告を確認します。

7. [はい、実行フックを有効にする]*を選択します。

実行フック機能を無効にする

所有者または管理者ユーザは、このAstra Controlアカウントで定義されているすべてのユーザに対して実行フック機能を無効にすることができます。実行フック機能を無効にする前に、既存の実行フックをすべて削除する必要があります。を参照してください [実行フックを削除します] 既存の実行フックを削除する手順については、を参照してください。

手順

1. に移動し、[機能設定]*タブを選択します。
2. [実行フック *] タブを選択します。
3. Execution Hooks*ペインで、設定メニューを選択します。
4. [**Disable**] を選択します。
5. 表示される警告を確認します。
6. を入力します `disable` をクリックして、すべてのユーザに対してこの機能を無効にすることを確認します。
7. [はい、無効にする]*を選択します。

既存の実行フックを表示します

アプリケーションの既存のカスタム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、一致するコンテナの数、作成時間、および実行時間（プリ/ポストオペレーション）を確認できます。を選択できます + アイコンをクリックして、実行するコンテナのリストを展開します。このアプリケーションの実行フックに関連するイベントログを表示するには、*アクティビティ*タブに移動します。

既存のスクリプトを表示します

アップロードされた既存のスクリプトを表示できます。このページでは、使用中のスクリプトと、使用中のフックを確認することもできます。

手順

1. 「アカウント」に移動します。
2. [スクリプト]タブを選択します。

   このページには、アップロードされた既存のスクリプトのリストが表示されます。[使用者*]列には、各スクリプトを使用している実行フックが表示されます。

スクリプトを追加します

各実行フックは、スクリプトを使用してアクションを実行する必要があります。実行フックが参照できるスクリプトを1つ以上追加できます。多くの実行フックは同じスクリプトを参照できます。これにより、1つのスクリプトを変更するだけで多くの実行フックを更新できます。

手順

1. 実行フック機能が 有効。
2. 「アカウント」に移動します。
3. [スクリプト]タブを選択します。
4. 「 * 追加」を選択します。
5. 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。
     v. 「スクリプトを保存」を選択します。
   - 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. [貼り付け（Paste）]または[タイプ（* type）]オプションを選択する
     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。
6. 「スクリプトを保存」を選択します。

結果

新しいスクリプトが、[スクリプト]タブのリストに表示されます。

スクリプトを削除します

不要になって実行フックで使用されなくなったスクリプトは、システムから削除できます。

手順

1. 「アカウント」に移動します。
2. [スクリプト]タブを選択します。
3. 削除するスクリプトを選択し、「アクション」列のメニューを選択します。
4. 「 * 削除」を選択します。

> スクリプトが1つまたは複数の実行フックに関連付けられている場合、*Delete*アクションは使用できません。スクリプトを削除するには、まず関連する実行フックを編集し、別のスクリプトに関連付けます。

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してAstra Controlに追加できます。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、特定のコマンドを実行するか、実行可能ファイルへの完全パスを指定する場合を除き、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. 実行フック機能が 有効。
2. 「 * 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。
3. [実行フック *] タブを選択します。
4. 「 * 追加」を選択します。
5. [フックの詳細*（Hook Details *）]領域で、次の
   a. *操作*ドロップダウンメニューから操作タイプを選択して、フックをいつ実行するかを決定します。
   b. フックの一意の名前を入力します。
   c. （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。
6. （オプション）フックフィルタの詳細（* Hook Filter Details *）領域で、実行フックが実行されるコンテナを制御するフィルタを追加できます。
   a. [フィルタの追加]を選択します。
   b. [フックフィルタータイプ*]列で、フィルターを適用する属性をドロップダウンメニューから選択します。
   c. [**Regex**]列に、フィルタとして使用する正規表現を入力します。Astra Controlでは、を使用します "正規表現2（RE2）正規表現の正規表現構文"。

      > 正規表現フィールドに他のテキストが含まれていない属性（ポッド名など）の正確な名前でフィルタリングすると、サブストリングの一致が実行されます。正確な名前とその名前だけを照合するには、完全に一致する文字列の一致構文を使用します（例： `^exact_podname$`）。

   d. フィルタをさらに追加するには、*フィルタを追加*を選択します。

      > 実行フックの複数のフィルタは、論理AND演算子と結合されます。実行フックごとに最大10個のアクティブフィルタを使用できます。

7. 完了したら、「次へ」を選択します。
8. [* スクリプト * （ * Script * ）] 領域で、次のいずれかを実行します。
   - 新しいスクリプトを追加します。
     i. 「 * 追加」を選択します。
     ii. 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。
  V. 「スクリプトを保存」を選択します。
- 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. [貼り付け（Paste）]または[タイプ（* type）]オプションを選択する
  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。

◦ リストから既存のスクリプトを選択します。

このスクリプトを使用するように実行フックに指示します。

9. 「 * 次へ * 」を選択します。
10. 実行フックの設定を確認します。
11. 「 * 追加」を選択します。

実行フックの状態を確認します

スナップショット、バックアップ、または復元操作の実行が終了したら、操作の一部として実行された実行フックの状態を確認できます。このステータス情報を使用して、実行フックを保持するか、変更するか、削除するかを決定できます。

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. [データ保護]タブを選択します。
3. 実行中のSnapshotを表示するには「* Snapshots」を選択し、実行中のバックアップを表示するには「* Backups」を選択します。

   フック状態*は、操作完了後の実行フックランのステータスを示します。状態にカーソルを合わせると、詳細を確認できます。たとえば、スナップショット中に実行フック障害が発生した場合、そのスナップショットのフック状態にカーソルを合わせると、失敗した実行フックのリストが表示されます。各失敗の理由を確認するには、左側のナビゲーション領域の*アクティビティ*ページを確認します。

スクリプトの使用状況を表示します

どの実行フックがAstra Control Web UIの特定のスクリプトを使用しているかを確認できます。

手順

1. 「 * アカウント * 」を選択します。

2. [スクリプト]タブを選択します。

   スクリプトのリストにある* Used by *列には、リスト内の各スクリプトを使用しているフックの詳細が表示されます。

3. 目的のスクリプトの[使用者*]列の情報を選択します。

   より詳細なリストが表示され、スクリプトを使用しているフックの名前と、それらが実行されるように構成されている操作のタイプが示されます。

実行フックを編集します

実行フックを編集して、その属性、フィルタ、または使用するスクリプトを変更できます。実行フックを編集するには、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. 編集するフックの*アクション*列のオプションメニューを選択します。
4. 「 * 編集 * 」を選択します。
5. 各セクションを完了したら、「次へ」を選択して、必要な変更を行います。
6. [保存（ Save ）] 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。

5. 表示されたダイアログで、「delete」と入力して確定します。
6. [はい]を選択し、実行フックを削除します。*

を参照してください。

- "NetApp Verda GitHubプロジェクト"

Astra Control Centerを使用したAstra Control Centerの保護

Astra Control Centerが実行されているKubernetesクラスタで致命的なエラーに対する耐障害性を高めるには、Astra Control Centerアプリケーション自体を保護します。セカンダリのAstra Control Centerインスタンスを使用してAstra Control Centerをバックアップおよびリストアするか、基盤となるストレージでONTAPを使用している場合はAstraレプリケーションを使用できます。

これらのシナリオでは、Astra Control Centerの2つ目のインスタンスを別のフォールトドメインに導入して設定し、プライマリAstra Control Centerインスタンスとは別の2つ目のKubernetesクラスタで実行します。2つ目のAstra Controlインスタンスは、プライマリのAstra Control Centerインスタンスのバックアップに使用され、場合によってはリストアにも使用されます。リストアまたはレプリケートされたAstra Control Centerインスタンスは、引き続きアプリケーションクラスタアプリケーションのアプリケーションデータ管理機能を提供し、それらのアプリケーションのバックアップやSnapshotへのアクセスをリストアします。

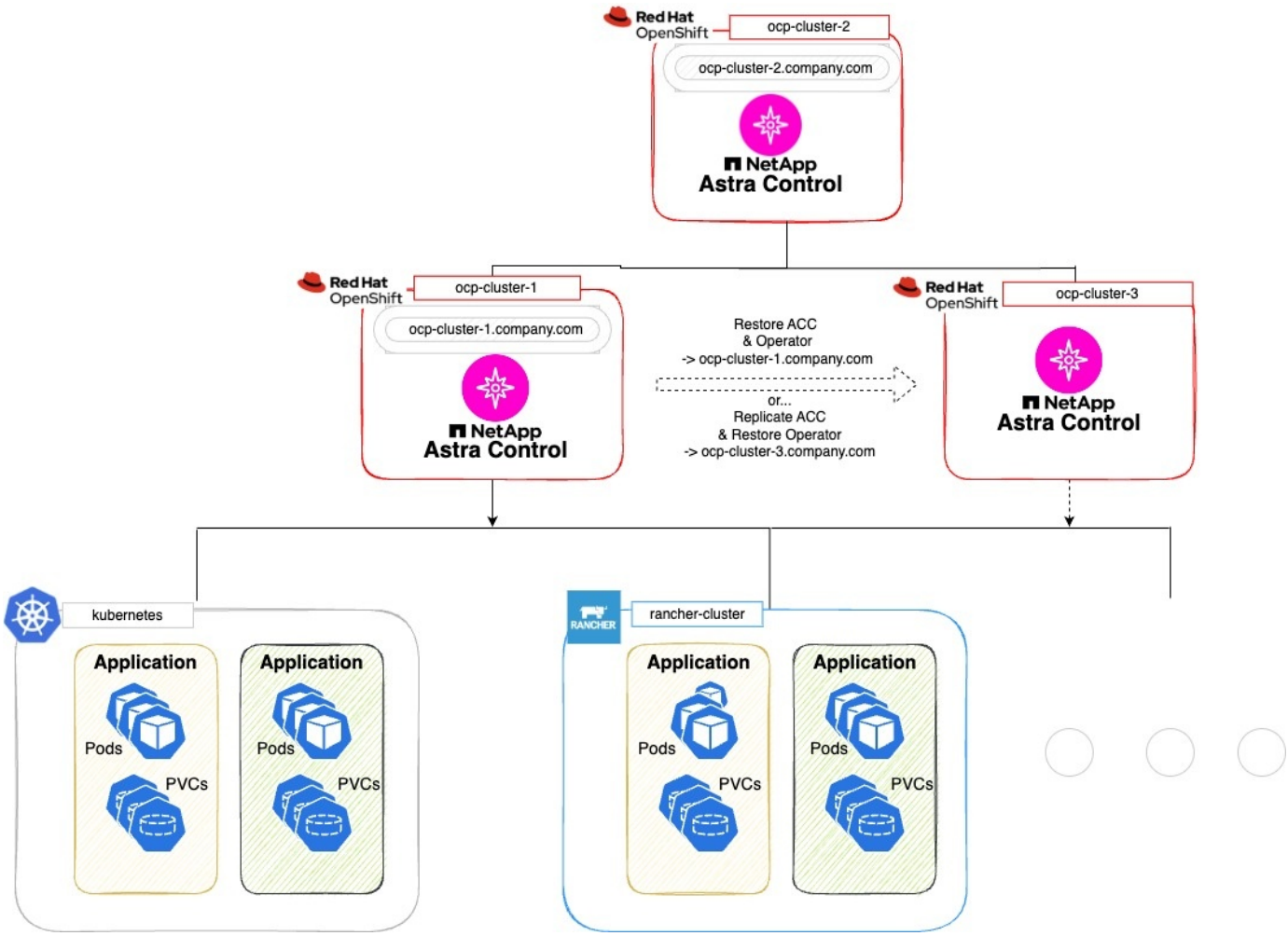
作業を開始する前に

Astra Control Centerの保護シナリオを設定する前に、次の情報を確認してください。

- プライマリ**Astra Control Center**インスタンスを実行する**Kubernetes**クラスタ：このクラスタは、アプリケーションクラスタを管理するプライマリAstra Control Centerインスタンスをホストします。
- セカンダリ**Astra Control Center**インスタンスを実行しているプライマリと同じ**Kubernetes**ディストリビューションタイプの**2**つ目の**Kubernetes**クラスタ：このクラスタは、プライマリAstra Control Centerインスタンスを管理するAstra Control Centerインスタンスをホストします。
- プライマリと同じ**Kubernetes**ディストリビューションタイプの**3**つ目の**Kubernetes**クラスタ：このクラスタは、Astra Control Centerのリストアまたはレプリケートされたインスタンスをホストします。現在プライマリに導入されているものと同じAstra Control Centerネームスペースを使用する必要があります。たとえば、Astra Control Centerがネームスペースに導入されている場合 `netapp-acc` ソースクラスタで、ネームスペース `netapp-acc` デスティネーションKubernetesクラスタ上のどのアプリケーションでも使用できない状態である必要があります。
- * S3互換バケット*：各Astra Control Centerインスタンスには、アクセス可能なS3互換オブジェクトストレージバケットがあります。
- 設定されたロードバランサ：ロードバランサはAstraのIPアドレスを提供し、アプリケーションクラスタと両方のS3バケットへのネットワーク接続を確立する必要があります。
- クラスタは**Astra Control Center**の要件に準拠：Astra Control Center保護で使用される各クラスタは、"Astra Control Centerの一般的な要件"。

このタスクについて

以下の手順では、以下のコマンドを使用してAstra Control Centerを新しいクラスタにリストアするために必要な手順について説明します。 バックアップとリストア または レプリケーション。手順は、ここに示す構成例に基づいています。



この設定例では、次の情報が表示されています。

- プライマリ**Astra Control Center**インスタンスを実行する**Kubernetes**クラスタ：
  - OpenShiftクラスタ： `ocp-cluster-1`
  - Astra Control Centerプライマリインスタンス： `ocp-cluster-1.company.com`
  - このクラスタは、アプリケーションクラスタを管理します。
- セカンダリ**Astra Control Center**インスタンスを実行しているプライマリと同じ**Kubernetes**ディストリビューションタイプの**2**つ目の**Kubernetes**クラスタ：
  - OpenShiftクラスタ： `ocp-cluster-2`
  - Astra Control Centerのセカンダリインスタンス： `ocp-cluster-2.company.com`
  - このクラスタを使用して、プライマリのAstra Control Centerインスタンスをバックアップするか、別のクラスタへのレプリケーションを設定します（この例では、 `ocp-cluster-3` クラスタ）。
- リストア処理に使用されるプライマリと同じ**Kubernetes**ディストリビューションタイプの**3**つ目の**Kubernetes**クラスタ：
  - OpenShiftクラスタ： `ocp-cluster-3`
  - Astra Control Center 3つ目のインスタンス： `ocp-cluster-3.company.com`
  - このクラスタは、Astra Control Centerのリストアまたはレプリケーションのフェイルオーバーに使用

されます。

> アプリケーションクラスタは、上図のKubernetesクラスタとRancherクラスタからわかるように、3つのAstra Control Centerクラスタの外部に配置するのが理想的です。

図には示されていません。

- すべてのクラスタに、Astra TridentまたはAstra Control ProvisionerがインストールされたONTAPバックエンドがあります。
- この構成では、OpenShiftクラスタがMetalLBをロードバランサとして使用しています。
- SnapshotコントローラとVolumeSnapshotClassもすべてのクラスタにインストールされています（を参照）。 "前提条件"。

ステップ**1**オプション：**Astra Control Center**のバックアップとリストア

この手順では、バックアップとリストアを使用して新しいクラスタにAstra Control Centerをリストアするために必要な手順について説明します。

この例では、Astra Control Centerは常に `netapp-acc` 名前空間と演算子は、 `netapp-acc-operator` ネームスペース：

> ここでは説明しませんが、Astra Control Centerのオペレータは、Astra CRと同じネームスペースに導入することもできます。

作業を開始する前に

- プライマリのAstra Control Centerをクラスタにインストールしておきます。
- セカンダリのAstra Control Centerを別のクラスタにインストールしておきます。

手順

1. プライマリAstra Control Centerアプリケーションとデスティネーションクラスタを、（実行中の）セカンダリAstra Control Centerインスタンスから管理 `ocp-cluster-2` クラスタ）：
   a. セカンダリAstra Control Centerインスタンスにログインします。
   b. "プライマリAstra Control Centerクラスタを追加" (`ocp-cluster-1`）。
   c. "デスティネーションの3つ目のクラスタを追加" (`ocp-cluster-3`）をクリックします。
2. セカンダリのAstra Control CenterでAstra Control CenterとAstra Control Centerオペレータを管理します。
   a. [アプリケーション（Applications）]ページで、[定義（ Define）]を選択します
   b. [アプリケーションの定義]ウィンドウで、新しいアプリケーション名を入力します。 (`netapp-acc`）。
   c. プライマリAstra Control Centerを実行しているクラスタを選択 (`ocp-cluster-1`）をクリックします。
   d. を選択します `netapp-acc` Astra Control Centerのネームスペース（*[ネームスペース]*ドロップダウンリスト）。
   e. [クラスタリソース]ページで、*[クラスタを対象とした追加のリソースを含める]*をオンにします。
   f. 「含めるルールを追加」を選択します。

g. 次のエントリを選択し、*[追加]*を選択します。
   - ラベルセレクタ:<label name>
   - グループ：apiextensions.k8s.io
   - バージョン：v1
   - 種類: CustomResourceDefinition

h. アプリケーション情報を確認します。

i. [* 定義（ Define ）] を選択します

「* define *」を選択した後、演算子に対して「アプリケーションの定義」プロセスを繰り返します。 `netapp-acc-operator`）をクリックし、 `netapp-acc-operator` [アプリケーションの定義]ウィザードの名前空間。

3. Astra Control Centerとオペレータのバックアップ：
   a. セカンダリAstra Control Centerで、[Applications]タブを選択して[Applications]ページに移動します。
   b. "バックアップ" Astra Control Centerアプリケーション (`netapp-acc`）。
   c. "バックアップ" 演算子 (`netapp-acc-operator`）。
4. Astra Control Centerとオペレータをバックアップしたら、次のツールでディザスタリカバリ（DR）シナリオをシミュレートします。 "Astra Control Centerのアンインストール" プライマリクラスタから削除します。

> 新しいクラスタ（この手順で説明する3つ目のKubernetesクラスタ）にAstra Control Centerをリストアし、新しくインストールしたAstra Control Centerのプライマリクラスタと同じDNSを使用します。

5. セカンダリAstra Control Centerを使用 "リストア" バックアップから作成したAstra Control Centerアプリケーションのプライマリインスタンス：
   a. [Applications]*を選択し、Astra Control Centerアプリケーションの名前を選択します。
   b. [オプション]メニューの[操作]列で、*[リストア]*を選択します。
   c. リストアタイプとして*[新しいネームスペースにリストアする]*を選択します。
   d. リストア名を入力 (`netapp-acc`）。
   e. デスティネーションの3番目のクラスタを選択 (`ocp-cluster-3`）。
   f. 元のネームスペースと同じネームスペースになるようにデスティネーションネームスペースを更新します。
   g. [Restore Source]ページで、リストア・ソースとして使用するアプリケーション・バックアップを選択します。
   h. [元のストレージクラスを使用してリストアする]*を選択します。
   i. [すべてのリソースをリストア]*を選択します。
   j. リストア情報を確認し、*[Restore]*を選択して、Astra Control Centerをデスティネーションクラスタにリストアするリストアプロセスを開始します。 (`ocp-cluster-3`）。アプリケーションが起動すると、リストアが完了します。 `available` 状態。
6. デスティネーションクラスタでAstra Control Centerを設定します。

a. ターミナルを開き、kubeconfigを使用してデスティネーションクラスタに接続します。（ocp-cluster-3）をクリックします。

b. を確認します ADDRESS Astra Control Center構成の列で参照されるプライマリシステムのDNS名は次のとおりです。

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

対応：

```
NAME  UUID                                   VERSION    ADDRESS
READY
astra 89f4fd47-0cf0-4c7a-a44e-43353dc96ba8 24.02.0-69 ocp-cluster-
1.company.com           True
```

a. 状況に応じて ADDRESS 上記の応答のフィールドにプライマリAstra Control Centerインスタンスのの FQDNがない場合は、Astra Control CenterのDNSを参照するように設定を更新します。

```
kubectl edit acc -n netapp-acc
```

i. を変更します astraAddress の下 spec: FQDNへ (ocp-cluster-1.company.com （この例では）のプライマリAstra Control Centerインスタンス。

ii. 設定を保存します。

iii. アドレスが更新されたことを確認します。

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

b. にアクセスします Astra Control Centerのオペレータのリストア セクションを参照して、リストアプロセスを完了してください。

ステップ**1**オプション：レプリケーションを使用して**Astra Control Center**を保護

この手順では、設定に必要な手順について説明します。 "Astra Control Centerのレプリケーション" を使用して、プライマリAstra Control Centerインスタンスを保護します。

この例では、Astra Control Centerは常に netapp-acc 名前空間と演算子は、 netapp-acc-operator ネームスペース：

作業を開始する前に

- プライマリのAstra Control Centerをクラスタにインストールしておきます。
- セカンダリのAstra Control Centerを別のクラスタにインストールしておきます。

手順

1. セカンダリAstra Control Centerインスタンスからプライマリ Astra Control Centerアプリケーションとデスティネーションクラスタを管理します。
   a. セカンダリAstra Control Centerインスタンスにログインします。
   b. "プライマリAstra Control Centerクラスタを追加" (`ocp-cluster-1`）。
   c. "デスティネーションの3つ目のクラスタを追加" (`ocp-cluster-3`）をクリックします。
2. セカンダリのAstra Control CenterでAstra Control CenterとAstra Control Centerオペレータを管理します。
   a. [Clusters]*を選択し、プライマリAstra Control Centerが含まれるクラスタを選択します。（`ocp-cluster-1`）。
   b. [名前空間]タブを選択します。
   c. 選択するオプション `netapp-acc` および `netapp-acc-operator` 名前空間。
   d. [アクション]メニューを選択し、*[アプリケーションとして定義]*を選択します。
   e. 定義されたアプリケーションを表示するには、*[アプリケーションで表示]*を選択します。
3. レプリケーションのバックエンドを構成します。

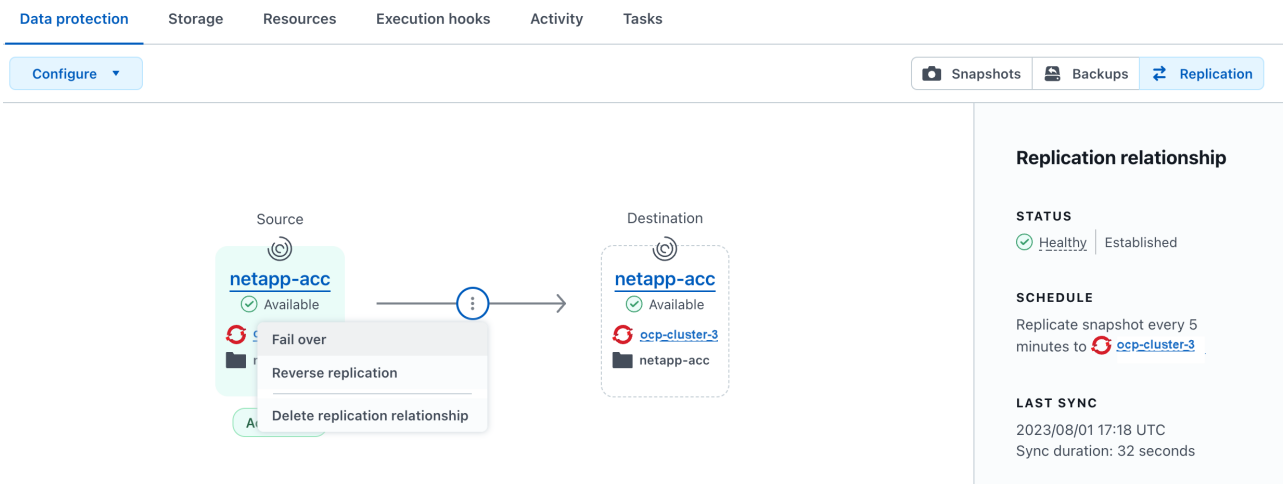
   レプリケーションには、プライマリのAstra Control Centerクラスタとデスティネーションクラスタが必要 (`ocp-cluster-3`）別のピアONTAPストレージバックエンドを使用します。
   各バックエンドがピアリングされてAstra Controlに追加されると、[Backends]ページの*[Discovered]*タブにバックエンドが表示されます。

   a. "ピアバックエンドの追加" をプライマリクラスタのAstra Control Centerに接続します。
   b. "ピアバックエンドの追加" デスティネーションクラスタのAstra Control Centerに接続します。
4. レプリケーションを設定します。
   a. [Applications]画面で、 `netapp-acc` アプリケーション：
   b. [レプリケーションポリシーの設定]*を選択します。
   c. 選択するオプション `ocp-cluster-3` をデスティネーションクラスタとして指定します。
   d. ストレージクラスを選択します。
   e. 入力するコマンド `netapp-acc` をデスティネーションネームスペースとして指定します。
   f. 必要に応じてレプリケーション頻度を変更します。
   g. 「 * 次へ * 」を選択します。
   h. 設定が正しいことを確認し、*[保存]*を選択します。

      レプリケーション関係の移行元 `Establishing` 終了： `Established`。アクティブな場合、このレプリケーションは、レプリケーション設定が削除されるまで5分おきに実行されます。
5. プライマリシステムが破損しているかアクセスできなくなった場合は、レプリケーションをもう一方のクラスタにフェイルオーバーします。

   フェイルオーバーが正常に実行されるように、デスティネーションクラスタにAstra Control Centerがインストールされていないことを確認してください。

a. 縦の楕円アイコンを選択し、*フェイルオーバー*を選択します。



b. 詳細を確認し、*[フェイルオーバー]*を選択してフェイルオーバープロセスを開始します。

レプリケーション関係のステータスがに変わります。 `Failing over` 次に `Failed over` 完了したら、

6. フェイルオーバーの設定を完了します。

a. ターミナルを開き、3番目のクラスタのkubeconfigを使用して接続します。（`ocp-cluster-3`）。このクラスタにはAstra Control Centerがインストールされています。

b. 3つ目のクラスタのAstra Control Center FQDNを確認（`ocp-cluster-3`）。

c. Astra Control CenterのDNSを参照するように設定を更新します。

```
kubectl edit acc -n netapp-acc
```

i. を変更します `astraAddress` の下 `spec:` FQDNを使用（`ocp-cluster-3.company.com`）をクリックします。

ii. 設定を保存します。

iii. アドレスが更新されたことを確認します。

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

d. 必要なすべてのtraefik CRDが存在することを確認します。

```
kubectl get crds | grep traefik
```

必要なtraefik CRD：

```
ingressroutes.traefik.containo.us
ingressroutes.traefik.io
ingressroutetcps.traefik.containo.us
ingressroutetcps.traefik.io
ingressrouteudps.traefik.containo.us
ingressrouteudps.traefik.io
middlewares.traefik.containo.us
middlewares.traefik.io
middlewaretcps.traefik.containo.us
middlewaretcps.traefik.io
serverstransports.traefik.containo.us
serverstransports.traefik.io
tlsoptions.traefik.containo.us
tlsoptions.traefik.io
tIsstores.traefik.containo.us
tIsstores.traefik.io
traefikservices.traefik.containo.us
traefikservices.traefik.io
```

a. 上記のCRDの一部がない場合は、次の手順を実行します。

   i. に進みます "traefikドキュメント"。

   ii. 「定義」領域をファイルにコピーします。

   iii. 変更を適用：

   ```
   kubectl apply -f <file name>
   ```

   iv. traefikを再起動します。

   ```
   kubectl get pods -n netapp-acc | grep -e "traefik" | awk '{print
   $1}' | xargs kubectl delete pod -n netapp-acc
   ```

b. にアクセスします Astra Control Centerのオペレータのリストア セクションを参照して、リストアプロセスを完了してください。

ステップ**2**：**Astra Control Center**のオペレータをリストア

セカンダリのAstra Control Centerを使用して、プライマリのAstra Control Centerオペレータをバックアップからリストアデスティネーションネームスペースは、ソースネームスペースと同じである必要があります。Astra Control Centerをプライマリソースクラスタから削除しても、同じリストア手順を実行するためのバックアップは引き続き存在します。

手順

1. *アプリケーション*を選択し、オペレータアプリの名前を選択します。（`netapp-acc-operator`）。

2. [操作]列の[オプション]メニューから*[リストア]*を選択します。
3. リストアタイプとして*[新しいネームスペースにリストアする]*を選択します。
4. デスティネーションの3番目のクラスタを選択 (`ocp-cluster-3`）。
5. ネームスペースをプライマリソースクラスタに関連付けられているネームスペースと同じに変更する (`netapp-acc-operator`）。
6. リストア・ソースとして以前に作成されたバックアップを選択します。
7. [元のストレージクラスを使用してリストアする]*を選択します。
8. [すべてのリソースをリストア]*を選択します。
9. 詳細を確認し、*[リストア]*をクリックしてリストアプロセスを開始します。

   [Applications]ページには、Astra Control Centerオペレータがデスティネーションの第3のクラスタにリストアされていることが表示される (`ocp-cluster-3`）。プロセスが完了すると、状態はとして表示されます。 `Available`。10分以内に、ページでDNSアドレスが解決されます。

結果

Astra Control Centerとその登録済みクラスタ、Snapshotとバックアップを使用した管理対象アプリケーションを、デスティネーションの第3のクラスタで利用できるようになりました。(`ocp-cluster-3`）。元のインスタンスに対して使用していた保護ポリシーは、新しいインスタンスにも適用されます。スケジュールされたバックアップやオンデマンドのバックアップやスナップショットを引き続き作成できます。

トラブルシューティング

システムの健全性と保護プロセスが成功したかどうかを確認します。

- ポッドが実行されていません：すべてのポッドが実行中であることを確認します。

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

  一部のポッドが `CrashLookBackOff` 状態、再起動し、次の状態に移行する必要があります。 `Running` 状態。

- システムステータスの確認：Astra Control Centerシステムが `ready` 都道府県：

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

  対応：

  ```
  NAME  UUID                                   VERSION    ADDRESS
  READY
  astra 89f4fd47-0cf0-4c7a-a44e-43353dc96ba8 24.02.0-69 ocp-cluster-
  1.company.com           True
  ```

- 導入ステータスの確認：Astra Control Centerの導入情報を表示して `Deployment State` はです `Deployed`。

```
kubectl describe acc astra -n netapp-acc
```

- リストアした**Astra Control Center UI**で**404**エラーが返される： `AccTraefik` 入力オプションとして、traefik CRD すべてインストールされていることを確認します。

アプリケーションとクラスタの健常性を監視

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。

クラスタの健全性を表示してストレージクラスを管理します

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。管理対象クラスタのデフォルトのストレージクラスを変更することもできます。

クラスタの健常性と詳細を表示します

クラスタの場所、ワーカーノード、永続ボリューム、ストレージクラスなどの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。

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。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しないでください。代わりに、この手順 を使用してください。kubectlを使用して変更を行った場合、Astra Controlはその変更を元に戻します。

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。

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、そのアプリケーションの詳細がAstra Controlに表示されます。これにより、アプリケーションの通信ステータス（Astra Controlがアプリケーションと通信できるかどうか）、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特定できます。

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. 情報を確認します。

   アプリステータス

   Astra Controlがアプリケーションと通信できるかどうかを示すステータスが表示されます。

   - アプリ保護ステータス：アプリの保護状態を表示します。
     - * 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、既存のスナップショット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的ボリュームを表示します。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されている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

ローカルユーザとロールを管理します

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 の略" ユーザを管理するには、を実行

LDAPを使用して、選択したユーザの認証を実行することもできます。

LDAP を使用する

LDAPは、分散ディレクトリ情報にアクセスするための業界標準プロトコルであり、エンタープライズ認証に広く使用されています。Astra Control CenterをLDAPサーバーに接続して、選択したAstra Controlユーザーの認証を実行できます。大まかには、AstraとLDAPを統合し、Astra ControlユーザおよびLDAP定義に対応するグループを定義することです。Astra Control APIまたはWeb UIを使用して、LDAP認証とLDAPユーザおよびグループを設定できます。詳細については、次のドキュメントを参照してください。

- "リモート認証とユーザーの管理には、Astra Control APIを使用します"
- "リモートユーザとリモートグループの管理には、Astra Control UIを使用します"
- "リモート認証を管理するには、Astra Control UIを使用します"

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。

拘束の追加の詳細については、を参照してください "ローカルユーザとロールを管理します"。

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 の略" 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

管理者または所有者ユーザーは、メンバーまたはビューアーの役割に名前空間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"

リモート認証を管理する

LDAPは、分散ディレクトリ情報にアクセスするための業界標準プロトコルであり、エンタープライズ認証に広く使用されています。Astra Control CenterをLDAPサーバーに接続して、選択したAstra Controlユーザーの認証を実行できます。

大まかには、AstraとLDAPを統合し、Astra ControlユーザおよびLDAP定義に対応するグループを定義することです。Astra Control APIまたはWeb UIを使用して、LDAP認証とLDAPユーザおよびグループを設定できます。

Astra Control Centerでは、リモート認証を有効にしたときに設定されるユーザログイン属性を使用して、リモートユーザを検索して追跡します。Astra Control Centerに表示するリモートユーザのこのフィールドには、Eメールアドレス（「mail」）またはユーザプリンシパル名（「userPrincipalName」）の属性が存在している必要があります。この属性は、Astra Control Centerで認証およびリモートユーザの検索に使用されるユーザ名です。

LDAPS認証用の証明書を追加します

LDAPサーバのプライベートTLS証明書を追加して、LDAPS接続を使用する際にAstra Control CenterがLDAPサーバで認証できるようにします。この処理は、1回だけ、またはインストールした証明書の有効期限が切れたときにのみ実行してください。

手順

1. 「アカウント」に移動します。

2. ［証明書］タブを選択します。
3. 「 * 追加」を選択します。
4. をアップロードします `.pem` クリップボードからファイルの内容をファイルまたは貼り付けます。
5. [**Trusted**]チェックボックスをオンにします。
6. [証明書の追加]を選択します。

リモート認証を有効にします

LDAP認証を有効にして、Astra ControlとリモートLDAPサーバ間の接続を設定できます。

作業を開始する前に

LDAPSを使用する場合は、Astra Control CenterがLDAPサーバに対して認証できるように、Astra Control CenterにLDAPサーバのプライベートTLS証明書がインストールされていることを確認してください。を参照してください LDAPS認証用の証明書を追加します 手順については、を参照し

手順

1. 「*アカウント」>「接続」に移動します。
2. [* Remote Authentication （リモート認証）]ペインで、設定メニューを選択します。
3. 「 * 接続」を選択します。
4. サーバのIPアドレス、ポート、および優先接続プロトコル（LDAPまたはLDAPS）を入力します。

> ベストプラクティスとして、LDAPサーバに接続するときはLDAPSを使用してください。LDAPSに接続する前に、LDAPサーバのプライベートTLS証明書をAstra Control Centerにインストールする必要があります。

5. サービスアカウントのクレデンシャルをEメール形式で入力します（administrator@example.com ）。Astra Controlは、LDAPサーバとの接続時にこれらのクレデンシャルを使用します。
6. [* User Match]セクションで、次の手順を実行します。
   a. LDAPサーバからユーザ情報を取得するときに使用するベースDNと適切なユーザ検索フィルタを入力します。
   b. （オプション）ディレクトリでuser login属性が使用されている場合 `userPrincipalName` ではなく `mail`と入力します `userPrincipalName` [ユーザーログイン属性（User login attribute）]フィールドの正しい属性に入力します。
7. ［グループ一致］セクションで、グループ検索ベースDNと適切なカスタムグループ検索フィルタを入力します。

> 正しいベース識別名（DN）と、* User Match および Group Match *の適切な検索フィルタを使用してください。ベースDNは、検索を開始するディレクトリツリーのレベルをAstra Controlに指示し、検索フィルタは、Astra Controlが検索するディレクトリツリーの部分を制限します。

8. [送信] を選択します。

結果

[リモート認証]ペインのステータスは、LDAPサーバーへの接続が確立されると、[保留中]になり、次に[接続済

み]になります。

リモート認証を無効にします

LDAPサーバへのアクティブな接続を一時的に無効にすることができます。

> LDAPサーバへの接続を無効にすると、すべての設定が保存され、Astra Controlに追加されたすべてのリモートユーザとリモートグループがそのLDAPサーバから保持されます。このLDAPサーバにいつでも再接続できます。

手順

1. 「*アカウント」>「接続」に移動します。
2. [* Remote Authentication （リモート認証）]ペインで、設定メニューを選択します。
3. [**Disable**] を選択します。

結果

[* Remote Authentication （リモート認証）]ペインのステータスが[* Disabled（無効）]に変わります。すべてのリモート認証設定、リモートユーザ、およびリモートグループが維持され、いつでも接続を再度有効にすることができます。

リモート認証の設定を編集します

LDAPサーバーへの接続を無効にした場合、または*リモート認証*ペインが「接続エラー」状態にある場合は、設定を編集できます。

> 「リモート認証」ペインが「無効」状態の場合、LDAPサーバのURLまたはIPアドレスを編集することはできません。必要です [リモート認証を切断します] 最初に。

手順

1. 「*アカウント」>「接続」に移動します。
2. [* Remote Authentication （リモート認証）]ペインで、設定メニューを選択します。
3. 「 * 編集 * 」を選択します。
4. 必要な変更を行い、* Edit *を選択します。

リモート認証を切断します

LDAPサーバから切断して、Astra Controlから構成設定を削除できます。

> LDAPユーザが切断した場合、セッションはすぐに終了します。LDAPサーバから切断すると、そのLDAPサーバのすべての構成設定がAstra Controlから削除されるだけでなく、そのLDAPサーバから追加されたすべてのリモートユーザとリモートグループも削除されます。

手順

1. 「*アカウント」>「接続」に移動します。
2. [* Remote Authentication （リモート認証）]ペインで、設定メニューを選択します。
3. 「切断」を選択します。

結果

「リモート認証」パネルのステータスが「切断済み」に変わります。リモート認証設定、リモートユーザ、およびリモートグループがAstra Controlから削除される。

リモートユーザとリモートグループを管理します

Astra ControlシステムでLDAP認証を有効にしている場合は、LDAPユーザおよびグループを検索して、承認されたシステムのユーザに含めることができます。

リモートユーザを追加します

アカウント所有者と管理者は、リモートユーザをAstra Controlに追加できます。Astra Control Centerは最大10、000人のLDAPリモートユーザをサポートします。

> Astra Control Centerでは、リモート認証を有効にしたときに設定されるユーザログイン属性を使用して、リモートユーザを検索して追跡します。Astra Control Centerに表示するリモートユーザのこのフィールドには、Eメールアドレス（「mail」）またはユーザプリンシパル名（「userPrincipalName」）の属性が存在している必要があります。この属性は、Astra Control Centerで認証およびリモートユーザの検索に使用されるユーザ名です。

> 同じEメールアドレス（「mail」または「user principal name」属性に基づく）を持つローカルユーザがシステムにすでに存在する場合は、リモートユーザを追加できません。ユーザをリモートユーザとして追加するには、最初にローカルユーザをシステムから削除してください。

手順

1. [**Account**（アカウント*）]領域に移動します。
2. [*Users & groups]タブを選択します。
3. ページの右端で、*リモートユーザー*を選択します。
4. 「 * 追加」を選択します。
5. 必要に応じて、ユーザのEメールアドレスを* Filter by email *フィールドに入力して、LDAPユーザを検索します。
6. リストから1人以上のユーザを選択します。
7. ユーザにロールを割り当てます。

> ユーザとユーザのグループに異なるロールを割り当てると、より権限の高いロールが優先されます。

8. 必要に応じて、このユーザに1つ以上のネームスペースの制約を割り当て、*ロールを制約に制限*を選択して適用します。新しい名前空間制約を追加するには、*制約の追加*を選択します。

> ユーザにLDAPグループメンバーシップを使用して複数のロールを割り当てると、最も権限の高いロールの制約だけが有効になります。たとえば'ローカルビューアロールを持つユーザーがメンバーロールにバインドされた3つのグループを結合すると'メンバーロールからの制約の合計が有効になり'ビューアロールからの制約はすべて無視されます

9. 「 * 追加」を選択します。

結果

新しいユーザがリモートユーザのリストに表示されます。このリストでは、ユーザーに対するアクティブな拘束を表示したり、*アクション*メニューからユーザーを管理したりできます。

リモートグループを追加します

複数のリモートユーザを一度に追加するには、アカウント所有者と管理者がリモートグループをAstra Controlに追加します。リモートグループを追加すると、そのグループ内のすべてのリモートユーザがAstra Controlにログインできるようになり、グループと同じロールが継承されます。

Astra Control Centerでは、最大5、000のLDAPリモートグループがサポートされます。

手順

1. [**Account**（アカウント*）]領域に移動します。
2. [*Users & groups]タブを選択します。
3. ページの右端で、*リモートグループ*を選択します。
4. 「 * 追加」を選択します。

   このウィンドウには、Astra Controlがディレクトリから取得したLDAPグループの共通名と識別名のリストが表示されます。

5. 必要に応じて、「共通名でフィルタ」フィールドにグループの共通名を入力してLDAPグループを検索します。
6. リストから1つ以上のグループを選択します。
7. グループにロールを割り当てます。

   選択したロールは、このグループのすべてのユーザに割り当てられます。ユーザとユーザのグループに異なるロールを割り当てると、より権限の高いロールが優先されます。

8. 必要に応じて、このグループに1つ以上の名前空間制約を割り当て、*制約にロールを制限*を選択して適用します。新しい名前空間制約を追加するには、*制約の追加*を選択します。

   - アクセス対象のリソースが最新の**Astra Connector**がインストールされているクラスタに属している場合：LDAPグループメンバーシップを通じてユーザに複数のロールが割り当てられると、ロールの制約が結合されます。たとえば、ローカルのビューアロールを持つユーザーが、メンバーロールにバインドされている3つのグループに参加した場合、ユーザーは元のリソースへのビューアロールアクセス権と、グループメンバーシップによって取得されたリソースへのメンバーロールアクセス権を持つようになります。
   - アクセス対象のリソースが**Astra Connector**がインストールされていないクラスタに属している場合：ユーザにLDAPグループメンバーシップを通じて複数のロールが割り当てられている場合、最も権限の厳しいロールの制約のみが有効になります。

9. 「 * 追加」を選択します。

結果

新しいグループがリモートグループのリストに表示されます。このグループのリモートユーザは、各リモートユーザがログインするまで、リモートユーザのリストに表示されません。このリストでは、*アクション*メニューからグループの詳細を表示したり、グループを管理したりできます。

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方法については、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、その中に* 1つの*コンテキスト要素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 Control Center UIまたはを使用できます "Astra Control API の略" 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 "NetApp Support Site"。
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"

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
- Microsoft Azure
- 汎用 S3

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

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

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]
- [デフォルトバケットを設定する]
- [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]
- "[Tech previewカスタムリソースを使用したバケットの管理"]

> 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 ）]列のメニューから、[*編集（ Edit）]を選択します。
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。

デフォルトバケットを設定する

クラスタ間でクローニングを実行する場合、Astra Controlにはデフォルトバケットが必要です。すべてのクラ

スタにデフォルトバケットを設定するには、次の手順を実行します。

手順

1. 「* Cloud Instances *」に移動します。
2. リスト内のクラウドインスタンスの*アクション*列でメニューを選択します。
3. 「 * 編集 * 」を選択します。
4. [* Bucket*]リストで、デフォルトにするバケットを選択します。
5. [保存（ Save ）] 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。
- バケットのクラウドプロバイダの保持期間が終了する前にWrite Once Read Many（WORM）バケットを削除することはできません。WORMバケットは、バケット名の横に「Locked」と表示されます。

- 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。

作業を開始する前に

- 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。

あれば続けることはできません。

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。

[Tech preview]カスタムリソースを使用したバケットの管理

アプリケーションクラスタでAstra Controlのカスタムリソース（CR）を使用してバケットを追加できます。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。カスタムリソースメソッドを使用している場合、アプリケーションのスナップショット機能にはバケットが必要です。

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

Astra ControlのバケットカスタムリソースはAppVaultと呼ばれます。このCRには、保護処理でバケットを使用するために必要な設定が含まれています。

作業を開始する前に

- Astra Control Centerで管理されているクラスタから到達できるバケットを用意します。
- バケットのクレデンシャルがあることを確認します。
- バケットが次のいずれかのタイプであることを確認します。
  - NetApp ONTAP S3
  - NetApp StorageGRID S3 の略

- Microsoft Azure
- 汎用 S3

Amazon Web Services（AWS）では、Generic S3バケットタイプが使用されます。

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

手順

1. カスタムリソース（CR）ファイルを作成し、という名前を付けます（例： `astra-appvault.yaml`）。
2. 次の属性を設定します。
   - * metadata.name*:_（必須）_ AppVaultカスタムリソースの名前。
   - * spec.prefix *：（オプション） AppVaultに保存されているすべてのエンティティの名前のプレフィックスが付いたパス。
   - * spec.providerConfig*:_（必須）_指定されたプロバイダを使用してAppVaultにアクセスするために必要な設定を保存します。
   - * spec.providerCredentials*:_(必須)_指定されたプロバイダを使用してAppVaultにアクセスするために必要なすべての資格情報への参照を保存します。
     - * spec.providerCredentials.valueFromSecret*:_（オプション）_は、クレデンシャル値がシークレットから取得される必要があることを示します。
       - * key *：_（valueFromSecretを使用する場合は必須）_選択するシークレットの有効なキー。
       - * name *：_（valueFromSecretを使用する場合は必須）_このフィールドの値を含むシークレットの名前。同じネームスペースになければなりません。
   - * spec.providerType*:_（必須）_バックアップの提供元を決定します（例：NetApp ONTAP S3、Microsoft Azure）。

     YAMLの例：

```
apiVersion: astra.netapp.io/v1
kind: AppVault
metadata:
  name: astra-appvault
spec:
  providerType: generic-s3
  providerConfig:
    path: testpath
    endpoint: 192.168.1.100:80
    bucketName: bucket1
    secure: "false"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        name: s3-creds
        key: accessKeyID
    secretAccessKey:
      valueFromSecret:
        name: s3-creds
        key: secretAccessKey
```

3. データを入力した後、 `astra-appvault.yaml` 正しい値を持つファイルを作成し、CRを適用します。

```
kubectl apply -f astra-appvault.yaml -n astra-connector
```

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

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。

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認証の詳細を編集します]
- [検出された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の管理を解除します]
- [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。

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. ダッシュボードのストレージバックエンドパネルで状態を確認します。
   - * 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。

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、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。

ストレージバックエンド認証の詳細を編集します

Astra Control Centerには、ONTAP バックエンドの認証に2つのモードがあります。

- クレデンシャルベースの認証：必要な権限を持つONTAP ユーザのユーザ名とパスワード。ONTAP のバージョンとの互換性を最大限に高めるには、adminなどの事前定義されたセキュリティログインロールを使用する必要があります。
- 証明書ベースの認証：Astra Control Centerは、バックエンドにインストールされている証明書を使用してONTAP クラスタと通信することもできます。クライアント証明書、キー、および信頼されたCA証明書を使用する（推奨）。

既存のバックエンドを更新して、あるタイプの認証から別の方法に移行できます。一度にサポートされる認証

方式は1つだけです。

証明書ベースの認証の有効化の詳細については、を参照してください "ONTAP ストレージバックエンドで認証を有効にします"。

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. [Credentials]フィールドで、*[Edit]*アイコンを選択します。
4. [Edit]ページで、次のいずれかを選択します。
   - 管理者のクレデンシャルを使用：ONTAP クラスタ管理IPアドレスと管理者のクレデンシャルを入力します。クレデンシャルはクラスタ全体のクレデンシャルである必要があります。

     ここで入力するクレデンシャルのユーザは、を持っている必要があります `ontapi` ONTAP クラスタのONTAP System Managerで有効になっているユーザログインアクセス方法。SnapMirrorレプリケーションを使用する場合は、アクセス方法が指定された「admin」ロールのユーザクレデンシャルを適用します `ontapi` および `http`、ソースとデスティネーションの両方のONTAP クラスタ。を参照してください "ONTAP ドキュメントの「ユーザーアカウントの管理」を参照してください" を参照してください。
   - 証明書を使用：証明書をアップロードします `.pem` ファイル、証明書キー `.key` ファイルを指定し、必要に応じて認証局ファイルを指定します。
5. [保存（ Save ）] を選択します。

検出されたストレージバックエンドを管理します

管理対象外で検出されたストレージバックエンドを管理するように選択できます。ストレージバックエンドを管理する際に、Astra Controlに認証用の証明書の有効期限が切れているかどうかが表示されます。

手順

1. 左のナビゲーションから、 * Backends * を選択します。
2. [Discovered]*オプションを選択します。
3. ストレージバックエンドを選択します。
4. [オプション]メニューの*列から、[管理]*を選択します。
5. 変更を行います。
6. [保存（ Save ）] 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。

作業を開始する前に

- 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。

詳細については、こちらをご覧ください

- "Astra Control API を使用"

実行中のタスクを監視します

Astra Controlの過去24時間に完了、失敗、またはキャンセルされた実行中のタスクとタスクの詳細を表示できます。たとえば、実行中のバックアップ、リストア、またはクローン処理のステータスを表示して、完了した割合や推定残り時間などの詳細を確認できます。実行済みのスケジュール済み処理または手動で開始した処理のステータスを表示できます。

実行中または完了済みのタスクを表示している間に、タスクの詳細を展開して、各サブタスクのステータスを確認できます。進行中のタスクまたは完了したタスクの場合はタスクの進捗状況バーが緑色で表示され、キャンセルされたタスクの場合は青色で表示され、エラーのために失敗したタスクの場合は赤色で表示されます。

クローン処理の場合、タスクサブタスクはSnapshotとSnapshotのリストア処理で構成されます。

失敗したタスクの詳細については、を参照してください "アカウントのアクティビティを監視"。

手順

1. タスクの実行中に、「アプリケーション」に移動します。
2. リストからアプリケーションの名前を選択します。
3. アプリケーションの詳細で、[タスク]タブを選択します。

   現在または過去のタスクの詳細を表示したり、タスクの状態でフィルタリングしたりできます。

タスクは最大24時間、*タスク*リストに保存されます。を使用して、この制限とその他のタスクモニタの設定を行うことができます "Astra Control API の略"。

[技術プレビュー] CRSを使用したAstra Controlアプリケーションの管理

Kubernetesのカスタムリソース（CR）を使用してAstra Controlアプリケーションを管理次のオプションを使用できます。

- "Kubernetesのカスタムリソースを使用してアプリケーションを定義する"
- "カスタムリソースを使用したバケットの管理"

Prometheus接続またはFluentd接続でインフラを監視

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。インフラ全体を監視してインサイトを取得するには、Prometheusを設定するか、Fluentd接続を追加します。

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（サポートバンドルをNetApp Support Siteにアップロードするための）プロキシが必要な場合は、Astra Control Centerでプロキシサーバを設定する必要があります。

- Prometheusに接続
- Fluentd に接続します

NetApp Support Siteへの接続用プロキシサーバを追加する

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（サポートバンドルをNetApp Support Siteにアップロードするための）プロキシが必要な場合は、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。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*[編集]*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。無効にする前に、他の接続が中断される可能性があることを警告するメッセージが表示されます。

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。

Prometheusに接続

Prometheusを使用して、Astra Control Centerのデータを監視できます。Kubernetesクラスタの指標エンドポイントから指標を収集するようにPrometheusを設定したり、Prometheusを使用して指標データを表示したりすることもできます。

Prometheusの使用の詳細については、でそれぞれのドキュメントを参照してください "Prometheusでの作業の開始"。

必要なもの

PrometheusパッケージがAstra Control Centerクラスタ、またはAstra Control Centerクラスタと通信可能な別のクラスタにダウンロードしてインストールされていることを確認します。

の公式ドキュメントに記載されている手順に従ってください "Prometheus をインストールする"。

Prometheusは、Astra Control Center Kubernetesクラスタと通信できる必要があります。PrometheusがAstra Control Centerクラスタにインストールされていない場合は、Astra Control Centerクラスタで実行されている指標サービスと通信できることを確認する必要があります。

Prometheus を設定する

Astra Control Centerは、KubernetesクラスタのTCPポート9090で指標サービスを公開します。このサービスから指標を収集するには、 Prometheus を設定する必要があります。

手順

1. Prometheusサーバにログインします。
2. にクラスタエントリを追加します `prometheus.yml` ファイル。を参照してください `yml` ファイルで、クラスタに関する次のようなエントリをに追加します `scrape_configs section`：

```
job_name: '<Add your cluster name here. You can abbreviate. It just
needs to be a unique name>'
  metrics_path: /accounts/<replace with your account ID>/metrics
  authorization:
     credentials: <replace with your API token>
  tls_config:
     insecure_skip_verify: true
  static_configs:
    - targets: ['<replace with your astraAddress. If using FQDN, the
prometheus server has to be able to resolve it>']
```

を設定した場合は `tls_config insecure_skip_verify` 終了： `true`では、TLS暗号化プロトコルは必要ありません。

3. Prometheusサービスを再起動します。

```
sudo systemctl restart prometheus
```

Prometheusにアクセスする

PrometheusのURLにアクセスします。

手順

1. ブラウザで、Prometheus URLをポート9090と入力します。
2. * Status **>** Targets *を選択して、接続を確認します。

Prometheusでデータを表示する

Prometheusを使用してAstra Control Centerのデータを表示できます。

手順

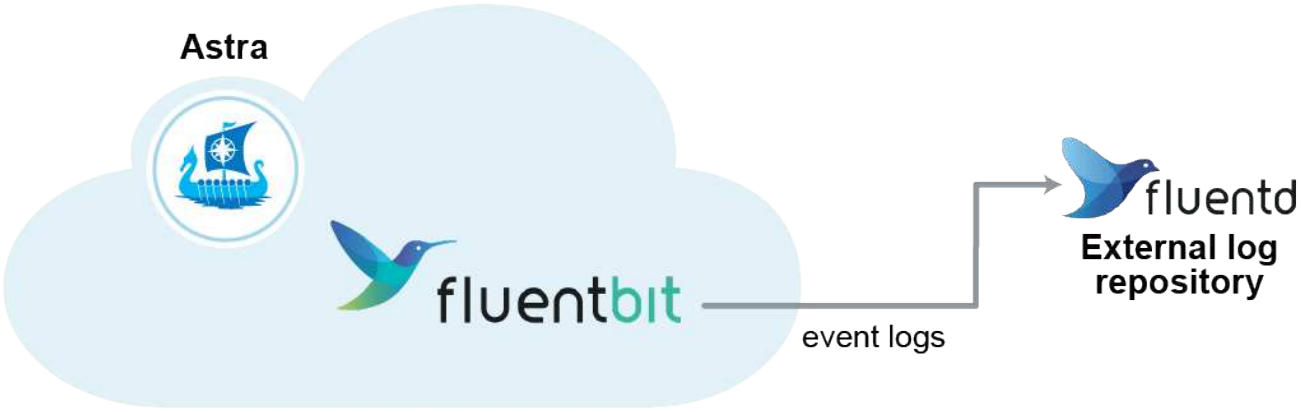
1. ブラウザで、PrometheusのURLを入力します。
2. Prometheusメニューで* Graph *を選択します。
3. メトリクスエクスプローラを使用するには、[**Execute**]の横にあるアイコンを選択します。
4. 選択するオプション `scrape_samples_scraped` をクリックし、* Execute *を選択します。
5. 時間の経過に伴うサンプルのスクレイピングを確認するには、* Graph *を選択します。

複数のクラスタデータが収集された場合、各クラスタの指標は異なる色で表示されます。

Fluentd に接続します

Astra Control Centerで監視されているシステムから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。

Fluentd接続は、宣言型Kubernetesワークフローで管理されるクラスタではサポートされません。Fluentdは、非Kubernetesネイティブワークフローで管理されるクラスタにのみ接続できます。



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。

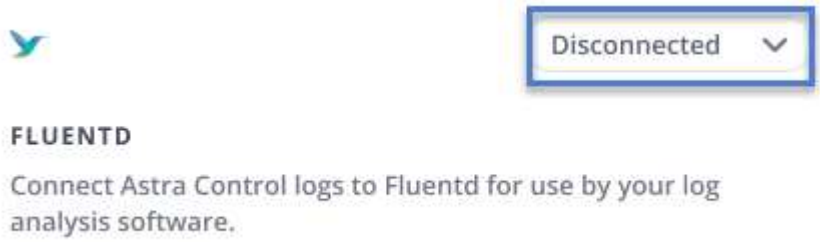
作業を開始する前に

- 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て、ログがにあることを確認する必要があります `/var/log/containers/`。

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*[編集]*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。

アプリケーションの管理を解除すると、次のようになります。

- 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. アプリケーションを選択します。

3. [アクション]列の[オプション]メニューから、*Unmanage*を選択します。
4. 情報を確認します。
5. 「 unmanage 」と入力して確定します。
6. 「はい、アプリケーションの管理を解除」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をAstra Control Centerから管理しないようにします。

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。

クラスタの管理を解除する場合：

- 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。
- Astra Control ProvisionerまたはAstra Tridentはクラスタからアンインストールされません。 "Astra Trident のアンインストール方法をご確認ください"。

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. 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「Removing *」に変わります。その後、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。

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ていたすべてのリソースが削除されます。

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、インストールイメージをダウンロードし、以下の手順を実行します。この手順を使用して、インターネット接続環境またはエアギャップ環境の Astra コントロールセンターをアップグレードできます。

以下の手順では、2番目に新しいリリースから今回の最新リリースへのAstra Control Centerのアップグレードプロセスについて説明します。現在のリリースより2つ以上遅れているバージョンから直接アップグレードすることはできません。インストールされているAstra Control Centerのバージョンが最新リリースの背後に多数のバージョンがある場合は、インストールされているAstra Control Centerのバージョンが最新リリースの

背後にあるまで、より新しいバージョンへのチェーンアップグレードが必要になることがあります。リリースされたバージョンの完全なリストについては、 "リリースノート"。

作業を開始する前に

アップグレードする前に、環境が "Astra Control Center環境の最小要件"。環境に次の要素が必要です。

- 有効 **"Astra Controlプロビジョニングツール" Astra Trident**を使用
  a. 実行しているAstra Tridentのバージョンを確認します。

```
kubectl get tridentversion -n trident
```

> Astra Trident 23.01以前を実行している場合は、以下を使用 "手順" Astra Control Provisionerにアップグレードする前に、Astra Tridentの最新バージョンにアップグレードすること。Astra Tridentがバージョン24.02の4リリース期間内にある場合は、Astra Control Provisioner 24.02への直接アップグレードを実行できます。たとえば、Astra Trident 23.04からAstra Control Provisioner 24.02に直接アップグレードできます。

  b. Astra Control Provisionerが "有効"。Astra Control Provisionerは、23.10より前のリリースのAstra Control Centerでは機能しません。最新の機能にアクセスするには、アップグレードするAstra Control Centerと同じバージョンのAstra Control Provisionerをアップグレードしてください。

- サポートされている**Kubernetes**ディストリビューション

  実行しているKubernetesのバージョンを確認します。

```
kubectl get nodes -o wide
```

- 十分なクラスタリソース

  使用可能なクラスタリソースを確認します。

```
kubectl describe node <node name>
```

- デフォルトのストレージクラス

  デフォルトのストレージクラスを確認します。

```
kubectl get storageclass
```

- 正常で利用可能な**API**サービス

  すべての API サービスが正常な状態であり、使用可能であることを確認します。

```
kubectl get apiservices
```

- （ローカルレジストリのみ）**Astra Control Center**イメージのプッシュとアップロードに使用できるローカルレジストリ
- （**OpenShift**のみ）正常で利用可能なクラスタオペレータ

  すべてのクラスタオペレータが正常な状態であり、使用可能であることを確認します。

  ```
  kubectl get clusteroperators
  ```

また、次の点も考慮する必要があります。

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。

- * NetApp Astra Controlイメージレジストリへのアクセス*：
  Astra Control Provisionerなど、Astra Controlのインストールイメージや機能強化された機能をNetAppイメージレジストリから取得することができます。
  a. レジストリへのログインに必要なAstra ControlアカウントIDを記録します。

     アカウントIDはAstra Control Service Web UIで確認できます。ページ右上の図アイコンを選択し、* APIアクセス*を選択して、アカウントIDを書き留めます。

  b. 同じページから* APIトークンの生成*を選択し、APIトークン文字列をクリップボードにコピーしてエディターに保存します。

  c. Astra Controlレジストリにログインします。

     ```
     docker login cr.astra.netapp.io -u <account-id> -p <api-token>
     ```

- * Istioサービスメッシュの導入*
  Astra Control Centerのインストール時にIstioサービスメッシュをインストールした場合、このAstra Control Centerのアップグレードには Istioサービスメッシュが含まれます。サービスメッシュをまだ持っていない場合は、 "初期導入" Astra Control Centerの略。

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。

アップグレードを開始する前に、Astra Control Center UIからログアウトします。

- Astra Control Centerをダウンロードして展開します
- [ローカルレジストリを使用する場合は、追加の手順を実行します。]
- 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
- [システムステータスを確認します]

Astra Control Centerオペレータ（たとえば、 `kubectl delete -f astra_control_center_operator_deploy.yaml`）Astra Control Centerのアップグレードまたは操作中はいつでもポッドを削除しないようにします。

Astra Control Centerをダウンロードして展開します

次のいずれかの場所からAstra Control Centerのイメージをダウンロードします。

- * Astra Controlサービスのイメージレジストリ*：Astra Control Centerのイメージでローカルレジストリを使用しない場合や、NetApp Support Siteからバンドルをダウンロードするよりもこの方法を使用する場合は、このオプションを使用します。
- * NetApp Support Site *：このオプションは、Astra Control Centerのイメージでローカルレジストリを使用する場合に使用します。

Astra Controlイメージレジストリ

1. Astra Control Serviceにログインします。
2. ダッシュボードで、*[Deploy a self-managed instance of Astra Control]*を選択します。
3. 手順に従ってAstra Controlイメージのレジストリにログインし、Astra Control Centerのインストールイメージを取得してイメージを展開します。

NetApp Support Site

1. Astra Control Centerを含むバンドルをダウンロードします (`astra-control-center-[version].tar.gz`）をクリックします "Astra Control Centerのダウンロードページ"。
2. （推奨ですがオプション）Astra Control Centerの証明書と署名のバンドルをダウンロードします (`astra-control-center-certs-[version].tar.gz`）をクリックして、バンドルのシグネチャを確認します。

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

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

出力にはと表示されます `Verified OK` 検証が成功したあとに、

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

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

ローカルレジストリを使用する場合は、追加の手順を実行します。

Astra Control Centerバンドルをローカルのレジストリにプッシュする場合は、NetApp Astra kubectlコマンドラインプラグインを使用する必要があります。

NetApp Astra kubectlプラグインを削除して、再度インストールします

イメージをローカルのDockerリポジトリにプッシュするには、最新バージョンのNetApp Astra kubectlコマンドラインプラグインを使用する必要があります。

1. プラグインがインストールされているかどうかを確認します。

```
kubectl astra
```

2. 次のいずれかを実行します。
   - プラグインがインストールされている場合は、コマンドによってkubectlプラグインのヘルプが返され、既存のバージョンのkubectl-Astraを削除できます。 `delete /usr/local/bin/kubectl-astra`。
   - コマンドからエラーが返された場合は、プラグインがインストールされていません。次の手順に進んでインストールしてください。
3. プラグインをインストールします。
   a. 使用可能なNetApp Astra kubectlプラグインのバイナリを表示し、オペレーティングシステムとCPUアーキテクチャに必要なファイルの名前をメモします。

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

```
ls kubectl-astra/
```

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

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

イメージをレジストリに追加する

1. Astra Control Centerバンドルをローカルのレジストリにプッシュする場合は、コンテナエンジンに応じた手順を実行します。

Docker です

a. tarballのルートディレクトリに移動します。次のように表示されます。 `acc.manifest.bundle.yaml` ファイルと次のディレクトリ：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. Astra Control Centerのイメージディレクトリにあるパッケージイメージをローカルレジストリにプッシュします。を実行する前に、次の置換を行ってください `push-images` コマンドを実行します

- <BUNDLE_FILE> をAstra Controlバンドルファイルの名前に置き換えます（`acc.manifest.bundle.yaml`）。
- <MY_FULL_REGISTRY_PATH> をDockerリポジトリのURLに置き換えます。次に例を示します。 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
- <MY_REGISTRY_USER> をユーザ名に置き換えます。
- <MY_REGISTRY_TOKEN> をレジストリの認証済みトークンに置き換えます。

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

ポドマン

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

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. レジストリにログインします。

```
podman login <YOUR_REGISTRY>
```

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

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

レジストリ設定に応じて、スクリプトが作成するイメージパスは次のようになります。

```
https://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version
```

2. ディレクトリを変更します。

```
cd manifests
```

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

1. （ローカルレジストリのみ）ローカルレジストリを使用している場合は、次の手順を実行します。

   a. Astra Control Centerオペレータによる導入YAMLを開きます。

   ```
   vim astra_control_center_operator_deploy.yaml
   ```

   注釈付きサンプルYAMLは以下の手順に従います。

   b. 認証が必要なレジストリを使用する場合は、のデフォルト行を置換または編集します `imagePullSecrets: []` 次の条件を満たす場合：

   ```
   imagePullSecrets: [{name: astra-registry-cred}]
   ```

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

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

   e. に次の値を追加します `env` セクション。

   ```
   - 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
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        control-plane: controller-manager
```

```
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: ASTRA_IMAGE_REGISTRY/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_UPGRADETIMEOUT
          value: 300m
        image: ASTRA_IMAGE_REGISTRY/acc-operator:24.02.68
        imagePullPolicy: IfNotPresent
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8081
          initialDelaySeconds: 15
          periodSeconds: 20
        name: manager
        readinessProbe:
          httpGet:
            path: /readyz
            port: 8081
          initialDelaySeconds: 5
          periodSeconds: 10
        resources:
          limits:
            cpu: 300m
            memory: 750Mi
          requests:
            cpu: 100m
            memory: 75Mi
        securityContext:
          allowPrivilegeEscalation: false
      imagePullSecrets: []
```

```
      securityContext:
        runAsUser: 65532
      terminationGracePeriodSeconds: 10
```

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.as
tra.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
```

3. ポッドが実行中であることを確認します

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

Astra Control Center をアップグレードします

1. Astra Control Centerカスタムリソース（CR）を編集します。

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

注釈付きサンプルYAMLは以下の手順に従います。

2. Astraのバージョン番号を変更します (`astraVersion` の内部 `spec`）から `23.10.0` 終了： `24.02.0`：

現在のリリースより2つ以上遅れているバージョンから直接アップグレードすることはできません。リリースされたバージョンの完全なリストについては、 "リリースノート"。

```
spec:
  accountName: "Example"
  astraVersion: "[Version number]"
```

3. イメージレジストリを変更します。
   - （ローカルレジストリのみ）ローカルレジストリを使用している場合は、イメージのレジストリパスがイメージをプッシュしたレジストリパスと一致していることを確認します。 前の手順。更新 `imageRegistry` の内部 `spec` 前回のインストール後にローカルレジストリが変更された場合。
   - （Astra Controlイメージレジストリ）Astra Controlイメージレジストリを使用 (`cr.astra.netapp.io`）を使用して、更新されたAstra Controlバンドルをダウンロードしました。

```
  imageRegistry:
    name: "[cr.astra.netapp.io or your_registry_path]"
```

4. に次の項目を追加します `crds` の内部の設定 `spec`：

```
crds:
  shouldUpgrade: true
```

5. 内に次の行を追加します `additionalValues` の内部 `spec` Astra Control Center CRで、次の手順を実行します。

```
additionalValues:
    nautilus:
      startupProbe:
        periodSeconds: 30
        failureThreshold: 600
    keycloak-operator:
      livenessProbe:
        initialDelaySeconds: 180
      readinessProbe:
        initialDelaySeconds: 180
```

6. ファイルエディタを保存して終了します。変更が適用され、アップグレードが開始されます。

7. （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 pods -n [netapp-acc or custom namespace]
```

8. アップグレードが完了して準備ができたことを示すため、Astra Controlのステータス状態が表示されるまで待ちます (`True`）：

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

対応：

```
NAME    UUID                                   VERSION     ADDRESS
READY
astra   9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f   24.02.0-69
10.111.111.111  True
```

処理中のアップグレードステータスを監視するには、次のコマンドを実行します。`kubectl get AstraControlCenter -o yaml -n [netapp-acc or custom namespace]`

Astra Control Centerのオペレータログを調べるには、次のコマンドを実行します。`kubectl logs deploy/acc-operator-controller-manager -n netapp-acc-operator -c manager -f`

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. バージョンがアップグレードされたことを確認します。UIの* Support *ページを参照してください。
3. 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。

OpenShift OperatorHubを使用したAstra Control Centerのアップグレード

Red Hat認定オペレータを使用してAstra Control Centerをインストールした場合は、OperatorHubから更新されたオペレータを使用してAstra Control Centerをアップグレードできます。この手順を使用して、 "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。

作業を開始する前に

- 環境の前提条件を満たしている：アップグレードする前に、環境が "Astra Control Center環境の最小要件"。

- 有効になっていることを確認します。 **"Astra Controlプロビジョニングツール" Astra Trident**を使用
  a. 実行しているAstra Tridentのバージョンを確認します。

```
kubectl get tridentversion -n trident
```

> Astra Trident 23.01以前を実行している場合は、以下を使用 "手順" Astra Control Provisionerにアップグレードする前に、Astra Tridentの最新バージョンにアップグレードすること。Astra Tridentがバージョン24.02の4リリース期間内にある場合は、Astra Control Provisioner 24.02への直接アップグレードを実行できます。たとえば、Astra Trident 23.04からAstra Control Provisioner 24.02に直接アップグレードできます。

  b. Astra Control Provisionerが "有効"。Astra Control Provisionerは、23.10より前のリリースのAstra Control Centerでは機能しません。最新の機能にアクセスするには、アップグレードするAstra Control Centerと同じバージョンのAstra Control Provisionerをアップグレードしてください。

- 正常なクラスタオペレータと**API**サービスを確保：
  - OpenShiftクラスタから、すべてのクラスタオペレータが正常な状態にあることを確認します。

```
oc get clusteroperators
```

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

```
oc get apiservices
```

- * OpenShiftの権限*：ここに記載されているアップグレード手順を実行するために必要なすべての権限とRed Hat OpenShift Container Platformへのアクセス権が必要です。
- （**ONTAP SAN**ドライバのみ）マルチパスの有効化：ONTAP SANドライバを使用している場合は、すべてのKubernetesクラスタでマルチパスが有効になっていることを確認してください。

また、次の点も考慮する必要があります。

- * NetApp Astra Controlイメージレジストリへのアクセス*：

  Astra Control Provisionerなど、Astra Controlのインストールイメージや機能強化された機能をNetAppイメージレジストリから取得することができます。

  a. レジストリへのログインに必要なAstra ControlアカウントIDを記録します。

     アカウントIDはAstra Control Service Web UIで確認できます。ページ右上の図アイコンを選択し、* APIアクセス*を選択して、アカウントIDを書き留めます。

  b. 同じページから* APIトークンの生成*を選択し、APIトークン文字列をクリップボードにコピーしてエディターに保存します。

  c. Astra Controlレジストリにログインします。

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

手順

- [オペレータインストールページへのアクセス]
- [既存のオペレータのアンインストール]
- [最新のオペレータのインストール]
- Astra Control Center をアップグレードします

オペレータインストールページへのアクセス

1. OpenShift Container Platformまたはエコシステムカタログに対応する手順を完成させます。

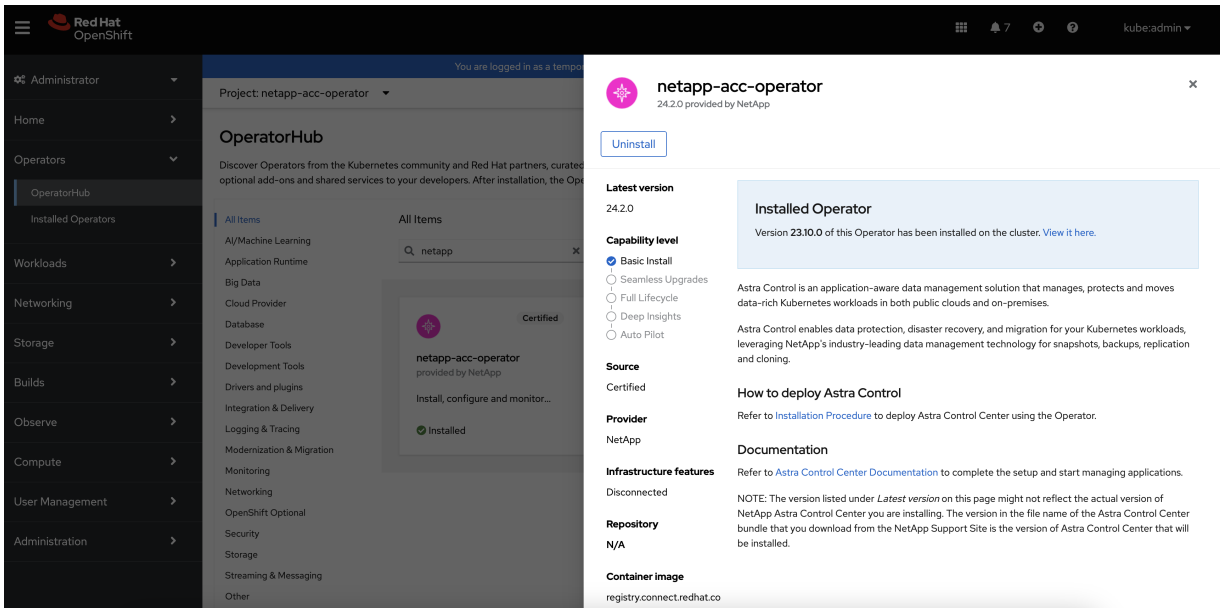
Red Hat OpenShift Webコンソール

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

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

> このオペレータを使用している場合は、Astra Control Centerの最新バージョンにのみアップグレードできます。

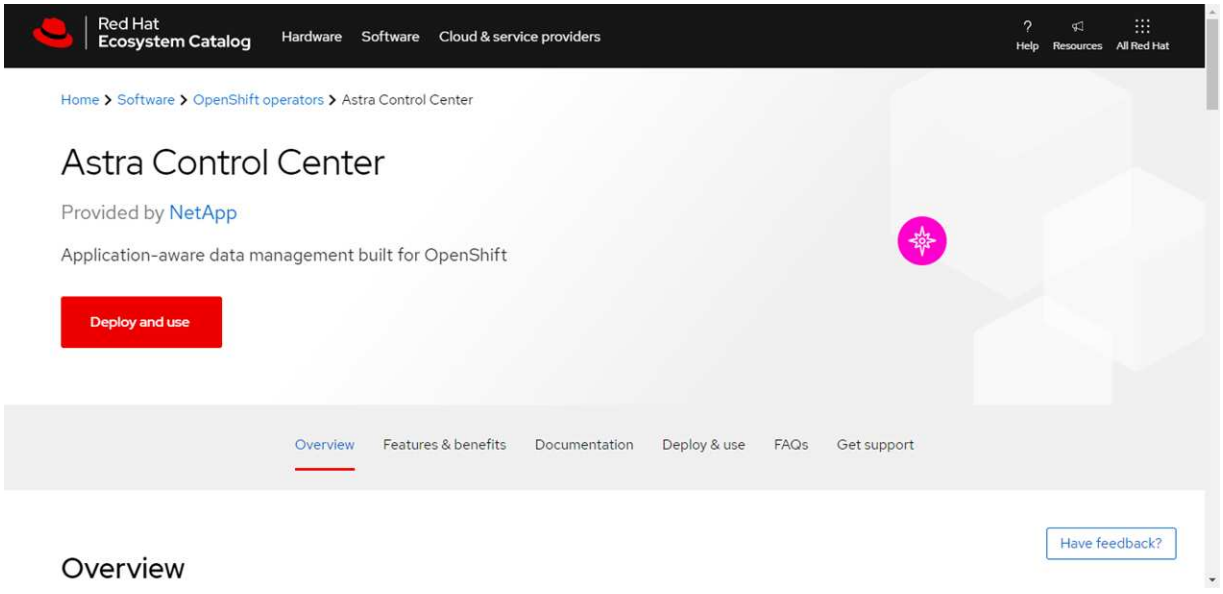
c. を検索します `netapp-acc` にアクセスし、NetApp Astra Control Centerオペレータを選択します。



Red Hat エコシステムカタログ

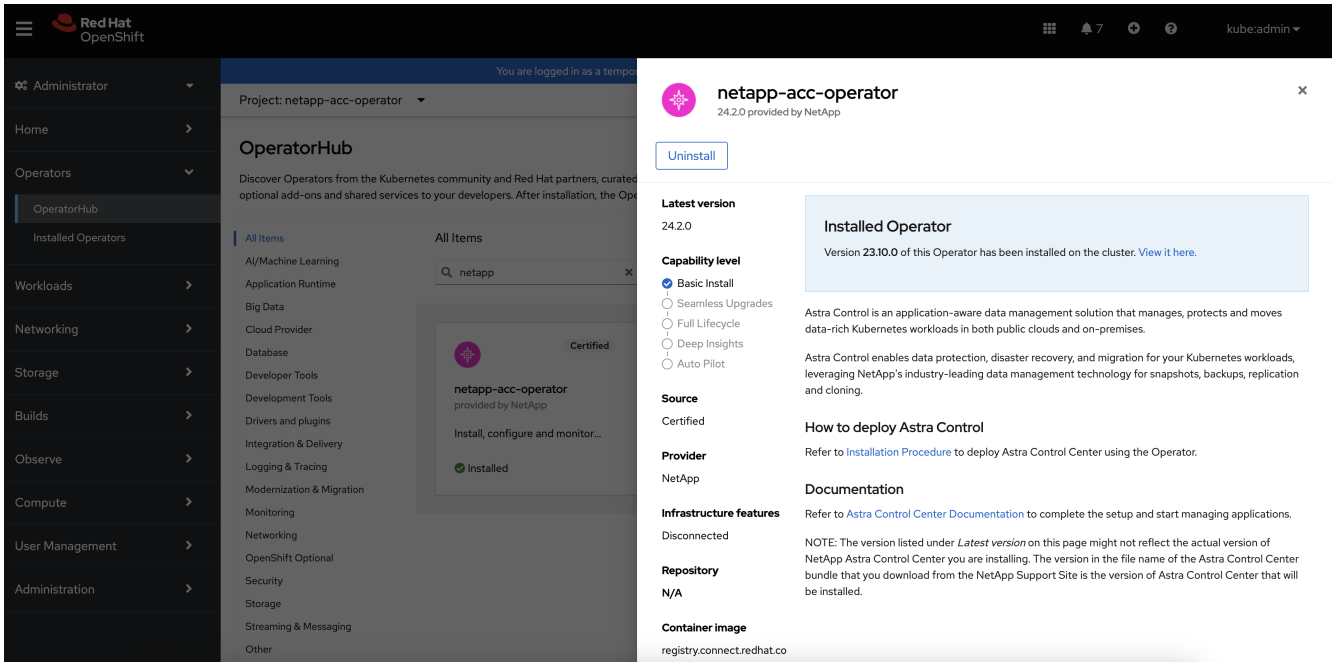
a. NetApp Astra Control Center を選択します "演算子"。

b. [Deploy and Use]*を選択します。



既存のオペレータのアンインストール

1. [netapp-acc-operator]ページで、*[アンインストール]*を選択して既存のオペレータを削除します。

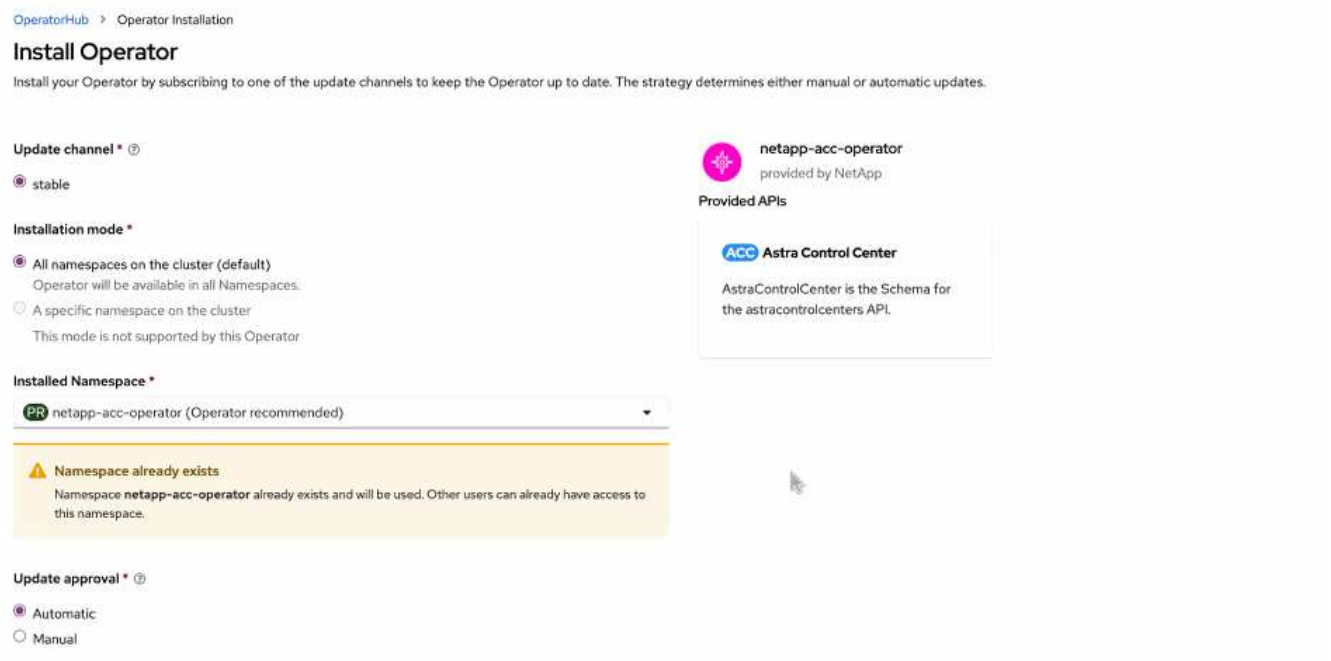


2. 操作を確定します。

> この処理では、netapp-acc-operatorが削除されますが、関連付けられている元のネームスペースとリソース（シークレットなど）は保持されます。

最新のオペレータのインストール

1. に移動します `netapp-acc` オペレータページを再度表示します。
2. [Install Operator]ページに入力し、最新の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. オペレータの `netapp-acc-operator` 削除されたオペレータの以前のインストールから残っている名前空間(またはカスタム名前空間)。

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。

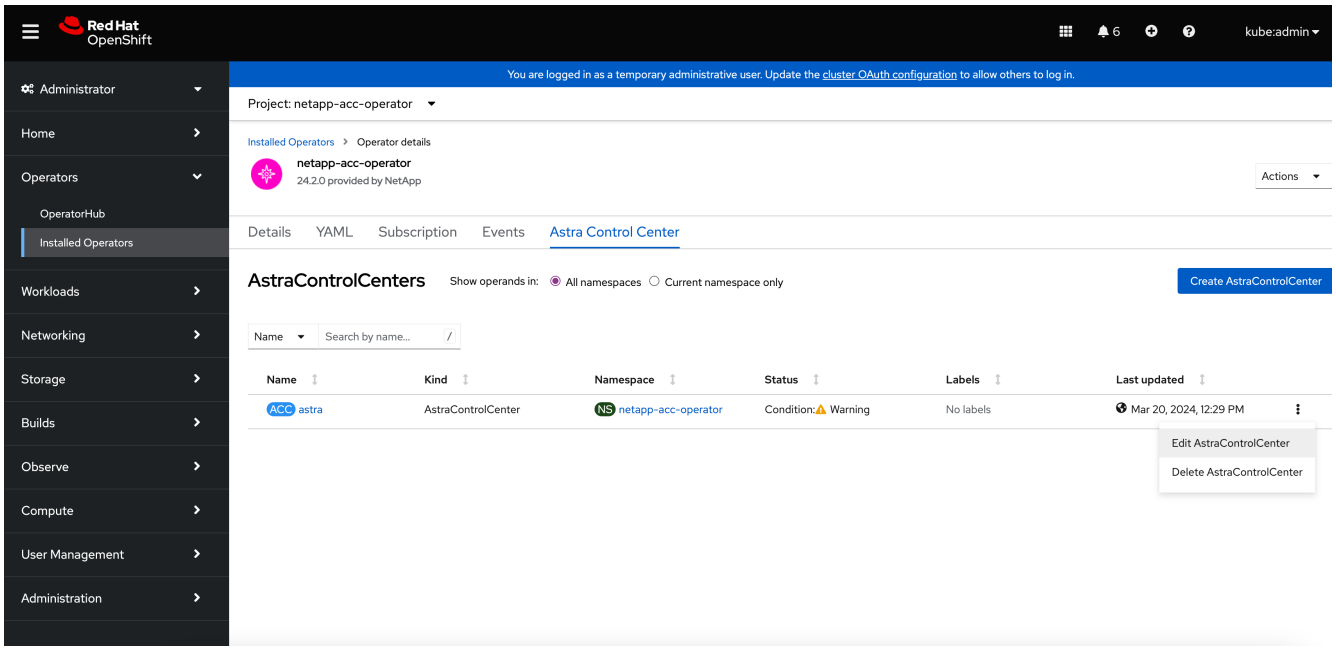
c. 「 * Install * 」を選択します。

手動承認ストラテジーを選択した場合は、このオペレーター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。

3. 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 をアップグレードします

1. [Astra Control Center]の[Operator]タブで、前のインストールで使用していたAstra Control Centerを選択し、*[Edit AstraControlCenter]*を選択します。



2. を更新します `AstraControlCenter` YAML：

   a. Astra Control Centerの最新バージョン（24.02.0-69など）を入力します。

   b. インチ `imageRegistry.name`必要に応じて、イメージレジストリパスを更新します。

      - Astra Controlレジストリオプションを使用している場合は、パスをに変更します。 `cr.astra.netapp.io`。
      - ローカルレジストリを設定した場合は、前の手順でイメージをプッシュしたローカルイメージレジストリパスを変更または保持します。

      入らないでください `http://` または `https://` をクリックします。

   c. を更新します `imageRegistry.secret` 必要に応じて。

      オペレータによるアンインストールプロセスでは、既存のシークレットは削除されません。このフィールドを更新する必要があるのは、既存のシークレットとは異なる名前で新しいシークレットを作成する場合だけです。

   d. に次の項目を追加します `crds` 構成：

```
crds:
  shouldUpgrade: true
```

3. 変更を保存します。
4. アップグレードが正常に完了したことを示す画面が表示されます。

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]。

作業を開始する前に

1. "すべてのアプリケーションの管理を解除します" クラスタ。
2. "すべてのクラスタの管理を解除します"。

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.yaml -n netapp-acc
```

結果

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

2. を削除するには、次のコマンドを使用します `netapp-acc` （またはカスタム名）ネームスペース：

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

結果の例：

```
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. 削除 `acc-monitoring` エージェント：

```
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"

Astra Controlプロビジョニングツールを使用

ストレージバックエンドの暗号化の設定

Astra Control Provisionerを使用すると、管理対象クラスタとストレージバックエンドの間のトラフィックの暗号化を有効にすることで、データアクセスセキュリティを強化できます。

Astra Control Provisionerは、次の2種類のストレージバックエンドでKerberos暗号化をサポートします。

- *オンプレミスのONTAP *- Astra Control Provisionerは、Red Hat OpenShiftおよびアップストリームのKubernetesクラスタからオンプレミスのONTAPボリュームへのNFSv3およびNFSv4接続でKerberos暗号化をサポートします。
- * Azure NetApp Files *- Astra Control Provisionerは、アップストリームのKubernetesクラスタからAzure NetApp Filesボリュームへの NFSv4.1接続でKerberos暗号化をサポートします。

作成、削除、サイズ変更、スナップショット、クローン、 読み取り専用のクローンを作成し、NFS暗号化を使用するボリュームをインポートします。

オンプレミスの**ONTAP**ボリュームで転送中の**Kerberos**暗号化を設定

管理対象クラスタとオンプレミスのONTAPストレージバックエンドの間のストレージトラフィックに対してKerberos暗号化を有効にすることができます。

> オンプレミスのONTAPストレージバックエンドを使用するNFSトラフィックに対するKerberos暗号化は、 `ontap-nas` ストレージドライバ。

作業を開始する前に

- 次のことを確認します。 "Astra Control Provisionerを有効にしました" 管理対象クラスタ。
- にアクセスできることを確認します。 `tridentctl` ユーティリティ。
- ONTAPストレージバックエンドへの管理者アクセス権があることを確認します。
- ONTAPストレージバックエンドから共有するボリュームの名前を確認しておきます。
- NFSボリュームのKerberos暗号化をサポートするようにONTAP Storage VMを準備しておく必要があります。を参照してください "データ LIF で Kerberos を有効にします" 手順については、を参照し
- Kerberos暗号化で使用するNFSv4ボリュームが正しく設定されていることを確認します。『NetApp NFSv4ドメイン設定』のセクション（13ページ）を参照してください。 "『NetApp NFSv4 Enhancements and Best Practices Guide』"。

ONTAPエクスポートポリシーを追加または変更する

既存のONTAPエクスポートポリシーにルールを追加するか、ONTAP Storage VMのルートボリュームおよびアップストリームのKubernetesクラスタと共有するONTAPボリュームに対してKerberos暗号化をサポートする新しいエクスポートポリシーを作成する必要があります。追加するエクスポートポリシールールまたは新規に作成するエクスポートポリシーでは、次のアクセスプロトコルとアクセス権限がサポートされている必要があります。

アクセスプロトコル

NFS、NFSv3、およびNFSv4の各アクセスプロトコルを使用してエクスポートポリシーを設定します。

詳細を確認

ボリュームのニーズに応じて、次の3つのバージョンのいずれかを設定できます。

- * Kerberos 5 *-（認証と暗号化）
- * Kerberos 5i *-（ID保護による認証と暗号化）
- * Kerberos 5p *-（IDおよびプライバシー保護による認証および暗号化）

適切なアクセス権限を指定してONTAPエクスポートポリシールールを設定します。たとえば、Kerberos 5i暗号化とKerberos 5p暗号化が混在しているNFSボリュームをクラスタにマウントする場合は、次のアクセス設定を使用します。

を入力します	読み取り専用アクセス	読み取り/書き込みアクセス	スーパーユーザアクセス
「 UNIX 」	有効	有効	有効
Kerberos 5i	有効	有効	有効
Kerberos 5p	有効	有効	有効

ONTAPエクスポートポリシーおよびエクスポートポリシールールの作成方法については、次のドキュメントを参照してください。

- "エクスポートポリシーを作成する"
- "エクスポートポリシーにルールを追加する"

ストレージバックエンドの作成

Kerberos暗号化機能を含むAstra Control Provisionerストレージバックエンド構成を作成できます。

このタスクについて

Kerberos暗号化を設定するストレージバックエンド構成ファイルを作成するときに、`spec.nfsMountOptions` パラメータ：

- `spec.nfsMountOptions: sec=krb5` （認証と暗号化）
- `spec.nfsMountOptions: sec=krb5i` （ID保護による認証と暗号化）
- `spec.nfsMountOptions: sec=krb5p` （IDおよびプライバシー保護による認証および暗号化）

Kerberosレベルを1つだけ指定してください。パラメータリストで複数のKerberos暗号化レベルを指定した場合は、最初のオプションのみが使用されます。

手順

1. 管理対象クラスタで、次の例を使用してストレージバックエンド構成ファイルを作成します。括弧<>の値は、環境の情報で置き換えます。

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret
```

2. 前の手順で作成した構成ファイルを使用して、バックエンドを作成します。

```
tridentctl create backend -f <backend-configuration-file>
```

バックエンドの作成に失敗した場合は、バックエンドの設定に何か問題があります。次のコマンドを実行すると、ログを表示して原因を特定できます。

```
tridentctl logs
```

構成ファイルで問題を特定して修正したら、 create コマンドを再度実行できます。

ストレージクラスを作成する。

ストレージクラスを作成して、Kerberos暗号化を使用してボリュームをプロビジョニングできます。

このタスクについて

ストレージクラスオブジェクトを作成するときに、を使用して3つの異なるバージョンのKerberos暗号化のいずれかを指定できます。 `mountOptions` パラメータ：

- `mountOptions: sec=krb5` （認証と暗号化）
- `mountOptions: sec=krb5i` （ID保護による認証と暗号化）
- `mountOptions: sec=krb5p` （IDおよびプライバシー保護による認証および暗号化）

Kerberosレベルを1つだけ指定してください。パラメータリストで複数のKerberos暗号化レベルを指定した場合は、最初のオプションのみが使用されます。ストレージバックエンド構成で指定した暗号化レベルがストレージクラスオブジェクトで指定したレベルと異なる場合は、ストレージクラスオブジェクトが優先されます。

手順

1. 次の例を使用して、StorageClass Kubernetesオブジェクトを作成します。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
parameters:
  backendType: "ontap-nas"
  storagePools: "ontapnas_pool"
  trident.netapp.io/nasType: "nfs"
allowVolumeExpansion: True
```

2. ストレージクラスを作成します。

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. ストレージクラスが作成されていることを確認します。

```
kubectl get sc ontap-nas-sc
```

次のような出力が表示されます。

```
NAME            PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

ボリュームのプロビジョニング

ストレージバックエンドとストレージクラスを作成したら、ボリュームをプロビジョニングできるようになり

ました。以下の手順を参照してください。 "ボリュームのプロビジョニング"。

Azure NetApp Filesボリュームでの転送中Kerberos暗号化の設定

管理対象クラスタと単一のAzure NetApp Filesストレージバックエンドまたは Azure NetApp Filesストレージバックエンドの仮想プールの間のストレージトラフィックに対してKerberos暗号化を有効にすることができます。

作業を開始する前に

- 管理対象のRed Hat OpenShiftクラスタでAstra Control Provisionerが有効になっていることを確認します。を参照してください "Astra Control Provisionerを有効にする" 手順については、を参照し
- にアクセスできることを確認します。 `tridentctl` ユーティリティ。
- 要件を確認し、次の手順に従って、Kerberos暗号化用のAzure NetApp Filesストレージバックエンドの準備が完了していることを確認します。 "Azure NetApp Files のドキュメント"。
- Kerberos暗号化で使用するNFSv4ボリュームが正しく設定されていることを確認します。『NetApp NFSv4ドメイン設定』のセクション（13ページ）を参照してください。 "『NetApp NFSv4 Enhancements and Best Practices Guide』"。

ストレージバックエンドの作成

Kerberos暗号化機能を含むAzure NetApp Filesストレージバックエンド構成を作成できます。

このタスクについて

Kerberos暗号化を設定するストレージバックエンド構成ファイルを作成する場合は、次の2つのレベルのいずれかで適用するように定義できます。

- ストレージバックエンドレベル*を使用して `spec.kerberos` フィールド
- 仮想プールレベル*を使用して `spec.storage.kerberos` フィールド

仮想プールレベルで構成を定義する場合、ストレージクラスのラベルを使用してプールが選択されます。

どちらのレベルでも、次の3つのバージョンのKerberos暗号化のいずれかを指定できます。

- `kerberos: sec=krb5` （認証と暗号化）
- `kerberos: sec=krb5i` （ID保護による認証と暗号化）
- `kerberos: sec=krb5p` （IDおよびプライバシー保護による認証および暗号化）

手順

1. 管理対象クラスタで、ストレージバックエンドを定義する必要がある場所（ストレージバックエンドレベルまたは仮想プールレベル）に応じて、次のいずれかの例を使用してストレージバックエンド構成ファイルを作成します。括弧<>の値は、環境の情報で置き換えます。

ストレージバックエンドレベルの例

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-anf-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-anf-secret
```

仮想プールレベルの例

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-anf-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
        type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-anf-secret
```

2. 前の手順で作成した構成ファイルを使用して、バックエンドを作成します。

```
tridentctl create backend -f <backend-configuration-file>
```

バックエンドの作成に失敗した場合は、バックエンドの設定に何か問題があります。次のコマンドを実行すると、ログを表示して原因を特定できます。

```
tridentctl logs
```

構成ファイルで問題を特定して修正したら、 create コマンドを再度実行できます。

ストレージクラスを作成する。

ストレージクラスを作成して、Kerberos暗号化を使用してボリュームをプロビジョニングできます。

手順

1. 次の例を使用して、StorageClass Kubernetesオブジェクトを作成します。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "nfs"
  selector: "type=encryption"
```

2. ストレージクラスを作成します。

```
kubectl create -f sample-input/storage-class-anf-sc-nfs.yaml
```

3. ストレージクラスが作成されていることを確認します。

```
kubectl get sc anf-sc-nfs
```

次のような出力が表示されます。

```
NAME          PROVISIONER             AGE
anf-sc-nfs    csi.trident.netapp.io   15h
```

ボリュームのプロビジョニング

ストレージバックエンドとストレージクラスを作成したら、ボリュームをプロビジョニングできるようになりました。以下の手順を参照してください。 "ボリュームのプロビジョニング"。

Snapshotを使用したボリュームデータのリカバリ

Astra Control Provisionerを使用すると、Snapshotからボリュームをインプレースですばやくリストアできます。 `TridentActionSnapshotRestore` （TASR）CR。このCRはKubernetesの必須アクションとして機能し、処理の完了後も維持されません。

Astra Control Provisionerは、上でのSnapshotのリストアをサポートします。 `ontap-san`、 `ontap-san-economy`、 `ontap-nas`、 `ontap-nas-flexgroup`、 `azure-netapp-files`、 `gcp-cvs`および `solidfire-san` ドライバ。

作業を開始する前に

バインドされたPVCと使用可能なボリュームSnapshotが必要です。

- PVCステータスがバインドされていることを確認します。

```
kubectl get pvc
```

- ボリュームSnapshotを使用する準備が完了していることを確認します。

```
kubectl get vs
```

手順

1. TASR CRを作成します。次に、PVC用のCRを作成する例を示します。 `pvc1` ボリュームSnapshot `pvc1-snapshot`。

```
cat tasr-pvc1-snapshot.yaml

apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: this-doesnt-matter
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. スナップショットからリストアするにはCRを適用します。この例では、Snapshotからリストアします。 `pvc1`。

```
kubectl create -f tasr-pvc1-snapshot.yaml

tridentactionsnapshotrestore.trident.netapp.io/this-doesnt-matter
created
```

結果

Astra Control ProvisionerがSnapshotからデータをリストアします。Snapshotのリストアステータスを確認できます。

```
kubectl get tasr -o yaml

apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: this-doesnt-matter
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

- ほとんどの場合、障害が発生したときにAstra Control Provisionerで処理が自動的に再試行されません。操作を再度実行する必要があります。
- 管理者アクセス権を持たないKubernetesユーザは、アプリケーションネームスペースにTASR CRを作成するために、管理者から権限を付与されなければならない場合があります。

SnapMirrorによるボリュームのレプリケート

Astra Control Provisionerを使用すると、ディザスタリカバリ用にデータをレプリケートするために、一方のクラスタのソースボリュームとピアクラスタのデスティネーションボリュームの間にミラー関係を作成できます。名前空間カスタムリソース定義（CRD）を使用して、次の操作を実行できます。

- ボリューム（PVC）間のミラー関係を作成する
- ボリューム間のミラー関係の削除
- ミラー関係を解除する
- 災害時（フェイルオーバー）にセカンダリボリュームを昇格する
- クラスタからクラスタへのアプリケーションのロスレス移行の実行（計画的なフェイルオーバーまたは移行時）

レプリケーションの前提条件

作業を開始する前に、次の前提条件を満たしていることを確認してください。

ONTAP クラスタ

- * Astra Control Provisioner *：Astra Control Provisionerバージョン23.10以降 "Astra Tridentのサポート" ONTAPをバックエンドとして利用するソースとデスティネーションの両方のKubernetesクラスタに存在する必要があります。
- ライセンス：Data Protection Bundleを使用するONTAP SnapMirror非同期ライセンスが、ソースとデスティネーションの両方のONTAPクラスタで有効になっている必要があります。を参照してください "ONTAPのSnapMirrorライセンスの概要" を参照してください。

ピアリング

- *クラスタとSVM *：ONTAPストレージバックエンドにピア関係が設定されている必要があります。を参照してください "クラスタと SVM のピアリングの概要" を参照してください。

> 2つのONTAPクラスタ間のレプリケーション関係で使用されるSVM名が一意であることを確認してください。

- * Astra Control ProvisionerとSVM *：ピア関係にあるリモートSVMがデスティネーションクラスタのAstra Control Provisionerで使用可能である必要があります。

サポートされるドライバ

- ボリュームレプリケーションは、ONTAP-NASドライバとONTAP-SANドライバでサポートされます。

ミラー**PVC**の作成

以下の手順に従って、CRDの例を使用してプライマリボリュームとセカンダリボリュームの間にミラー関係を作成します。

手順

1. プライマリKubernetesクラスタで次の手順を実行します。
   a. を使用してStorageClassオブジェクトを作成します。 `trident.netapp.io/replication: true` パラメータ

      例

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  fsType: "nfs"
  trident.netapp.io/replication: "true"
```

b. 以前に作成したStorageClassを使用してPVCを作成します。

例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

c. ローカル情報を含むMirrorRelationship CRを作成します。

例

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

Astra Control Provisionerは、ボリュームの内部情報とボリュームの現在のデータ保護（DP）の状態をフェッチし、MirrorRelationshipのstatusフィールドに値を入力します。

d. TridentMirrorRelationship CRを取得して、PVCの内部名とSVMを取得します。

```
kubectl get tmr csi-nas
```

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
status:
  conditions:
  - state: promoted
    localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
    localPVCName: csi-nas
    observedGeneration: 1
```

2. セカンダリKubernetesクラスタで次の手順を実行します。
   a. trident.netapp.io/replication: trueパラメータを使用してStorageClassを作成します。

      例

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        trident.netapp.io/replication: true
      ```

   b. デスティネーションとソースの情報を含むMirrorRelationship CRを作成します。

      例

      ```
      kind: TridentMirrorRelationship
      apiVersion: trident.netapp.io/v1
      metadata:
        name: csi-nas
      spec:
        state: established
        volumeMappings:
        - localPVCName: csi-nas
          remoteVolumeHandle:
      "datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
      ```

Astra Control Provisionerは、設定された関係ポリシー名（ONTAPの場合はデフォルト）を使用してSnapMirror関係を作成し、初期化します。

c. セカンダリ（SnapMirrorデスティネーション）として機能するStorageClassを作成してPVCを作成します。

例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
  - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

Astra Control ProvisionerはTridentMirrorRelationship CRDを確認し、関係が存在しない場合はボリュームの作成に失敗します。関係が存在する場合は、Astra Control Provisionerによって、新しいFlexVolがMirrorRelationshipで定義されているリモートSVMとピア関係にあるSVMに配置されます。

ボリュームレプリケーションの状態

Trident Mirror Relationship（TMR）は、PVC間のレプリケーション関係の一端を表すCRDです。デスティネーションTMRの状態はで、Astra Control Provisionerに必要な状態が示されます。宛先TMRの状態は次のとおりです。

- 確立済み：ローカルPVCはミラー関係のデスティネーションボリュームであり、これは新しい関係です。
- 昇格：ローカルPVCはReadWriteでマウント可能であり、ミラー関係は現在有効ではありません。
- * reestablished *：ローカルPVCはミラー関係のデスティネーションボリュームであり、以前はそのミラー関係に含まれていました。
  - デスティネーションボリュームはデスティネーションボリュームの内容を上書きするため、ソースボリュームとの関係が確立されたことがある場合は、reestablished状態を使用する必要があります。
  - ボリュームが以前にソースとの関係になかった場合、再確立状態は失敗します。

計画外フェールオーバー時にセカンダリ**PVC**を昇格する

セカンダリKubernetesクラスタで次の手順を実行します。

- TridentMirrorRelationshipの_spec.state_フィールド を次のように更新します。 `promoted`。

計画的フェイルオーバー中にセカンダリ**PVC**を昇格

計画的フェイルオーバー（移行）中に、次の手順を実行してセカンダリPVCをプロモートします。

手順

1. プライマリKubernetesクラスタでPVCのSnapshotを作成し、Snapshotが作成されるまで待ちます。
2. プライマリKubernetesクラスタで、SnapshotInfo CRを作成して内部の詳細を取得します。

   例

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. セカンダリKubernetesクラスタで、_TridentMirrorRelationship_CRの_spec.state_フィールドを_promoted_に更新し、_spec.promotedSnapshotHandle_をSnapshotのinternalNameにします。
4. セカンダリKubernetesクラスタで、TridentMirrorRelationshipのステータス（status.stateフィールド）がPromotedになっていることを確認します。

フェイルオーバー後にミラー関係をリストアする

ミラー関係をリストアする前に、新しいプライマリとして作成する側を選択します。

手順

1. セカンダリKubernetesクラスタで、TridentMirrorRelationshipの_spec.remoteVolumeHandle_fieldの値が更新されていることを確認します。
2. セカンダリKubernetesクラスタで、TridentMirrorRelationshipの_spec.mirror_fieldを `reestablished`。

その他の処理

Astra Control Provisionerでは、プライマリボリュームとセカンダリボリュームに対する次の処理がサポートされます。

新しいセカンダリ**PVC**へのプライマリ**PVC**の複製

プライマリPVCとセカンダリPVCがすでに存在していることを確認します。

手順

1. PersistentVolumeClaim CRDとTridentMirrorRelationship CRDを、確立されたセカンダリ（デスティネーション）クラスタから削除します。
2. プライマリ（ソース）クラスタからTridentMirrorRelationship CRDを削除します。
3. 確立する新しいセカンダリ（デスティネーション）PVC用に、プライマリ（ソース）クラスタに新しいTridentMirrorRelationship CRDを作成します。

ミラー、プライマリ、またはセカンダリ**PVC**のサイズ変更

PVCは通常どおりサイズ変更できます。データ量が現在のサイズを超えると、ONTAPは自動的に宛先フレフxolを拡張します。

PVCからのレプリケーションの削除

レプリケーションを削除するには、現在のセカンダリボリュームで次のいずれかの操作を実行します。

- セカンダリPVCのMirrorRelationshipを削除します。これにより、レプリケーション関係が解除されます。
- または、spec.stateフィールドを_promoted_に更新します。

（以前にミラーリングされていた）**PVC**の削除

Astra Control Provisionerは、ボリュームの削除を試行する前に、レプリケートされたPVCをチェックし、レプリケーション関係を解放します。

TMRの削除

ミラー関係の一方のTMRを削除すると、Astra Control Provisionerが削除を完了する前に、残りのTMRが_promoted_stateに移行します。削除対象として選択したTMRがすでに_promoted_stateである場合は、既存のミラー関係は存在せず、TMRが削除され、Astra Control ProvisionerがローカルPVCを_ReadWrite_に昇格します。この削除により、ONTAP内のローカルボリュームのSnapMirrorメタデータが解放されます。このボリュームを今後ミラー関係で使用する場合は、新しいミラー関係を作成するときに、レプリケーション状態が_established_volumeである新しいTMRを使用する必要があります。

ONTAPがオンラインのときにミラー関係を更新

ミラー関係は、確立後にいつでも更新できます。を使用できます `state: promoted` または `state: reestablished` 関係を更新するフィールド。
デスティネーションボリュームを通常のReadWriteボリュームに昇格する場合は、_promotedSnapshotHandle_を使用して、現在のボリュームのリストア先となる特定のSnapshotを指定できます。

ONTAPがオフラインの場合にミラー関係を更新

CRDを使用すると、Astra ControlからONTAPクラスタに直接接続せずにSnapMirror更新を実行できます。次のTridentActionMirrorUpdateの形式例を参照してください。

例

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` TridentActionMirrorUpdate CRDの状態を反映します。*Succeeded*、*In Progress*、_Failed_のいずれかの値を指定できます。

Astra Control 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 Control Center UIまたはAstra Control APIのいずれかを使用できます。

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"。

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。

"Astra Controlに関するNetAppのナレッジベース"

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Discordチャネルなど、24時間365日利用可能な無料のセルフサポートオプションをご用意しています。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、NetApp Support Site （ 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・オプションを使用してサポートに連絡するか、Discordチャネルを使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、NetApp Support Site （ 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。

- "ナレッジベースを使用（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。
- 製品ドキュメントを参照してください：現在表示しているドキュメントサイトです。
- "* Discord *で助けを得なさい"：thePubカテゴリのAstraに移動して、他のユーザ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のインストール中に、を指定した場合 `enrolled: true` の場合 `autoSupport` Astra Control Centerカスタムリソース（CR）ファイル (`astra_control_center.yaml`）を指定すると、日次サポートバンドルがに自動的にアップロードされます "NetApp Support Site"。

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、NetApp Support Site（ 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NetApp Support Site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。
   - Digital Advisorを使用します。

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 23.10ドキュメント"
- "Astra Control Center 23.07ドキュメント"
- "Astra Control Center 23.04のドキュメント"
- "Astra Control Center 22.11ドキュメント"
- "Astra Control Center 22.08ドキュメント"
- "Astra Control Center 22.04 のドキュメント"
- "Astra Control Center 21.12 ドキュメント"
- "Astra Control Center 21.08 ドキュメント"

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**とは何ですか？

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

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

ライセンス

評価ライセンスを使用しています。フルライセンスに変更するにはどうすればよいですか。

フルライセンスに変更するには、ネットアップからネットアップライセンスファイル（NLF）を入手します。

- 手順 *
  1. 左側のナビゲーションから、 * アカウント * > * ライセンス * を選択します。
  2. ライセンスの概要で、ライセンス情報の右側にある[Options]メニューを選択します。
  3. [置換]*を選択します。
  4. 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。アプリを管理することはできますか**?**

はい。評価ライセンス（デフォルトでインストールされている組み込み評価ライセンスを含む）を使用して、アプリケーションの管理機能をテストできます。評価用ライセンスとフルライセンスでは、機能や機能に違いはありません。評価用ライセンスは、単純に寿命が短くなります。を参照してください "ライセンス" を参照してください。

Kubernetes クラスタを登録しています

Astra Controlに追加したあと、**Kubernetes**クラスタにワーカーノードを追加する必要があります。どうすればいいですか**?**

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

クラスタの管理を適切に解除するにはどうすればよいですか？

1. "Astra Control からアプリケーションの管理を解除"。
2. "Astra Control からクラスタの管理を解除"。

Astra Controlから**Kubernetes**クラスタを削除したあと、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。

管理を解除すると、**Astra Control Provisioner**（**Astra Trident**）はクラスタから自動的にアンインストールされますか。

Astra Control Centerでクラスタの管理を解除しても、Astra Control ProvisionerまたはAstra Tridentはクラスタから自動的にアンインストールされません。Astra Control Provisionerとそのコンポーネントまたは Astra Tridentをアンインストールするには、次の手順を実行する必要があります： "次の手順に従って、Astra Control Provisionerサービスが含まれているAstra Tridentインスタンスをアンインストールします。"。

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。

Astra Controlからの管理を停止したアプリケーションはどうなりますか？

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。

Astra Controlは、ネットアップ以外のストレージ上にあるアプリケーションを管理できますか。

いいえAstra Controlは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を検出できますが、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を管理することはできません。

Astra Control自体を管理する必要はありますか？

デフォルトではAstra Control Centerは管理可能なアプリケーションとして表示されていませんが、 "バックアップとリストア" 別のAstra Control Centerインスタンスを使用するAstra Control Centerインスタンス。

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？

いいえ、ポッドの健全性はアプリ管理には影響しません。

データ管理の操作

私のアプリケーションは複数の**PVS**を使用しています。**Astra Control**はこれらの**PV**の**Snapshot**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で作成された**Snapshot**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で作成されたSnapshotとバックアップは、Astra Controlでのみ管理できます。

Astra Controlプロビジョニングツール

Astra Control Provisionerのストレージプロビジョニング機能と**Astra Trident**のストレージプロビジョニング機能の違いは何ですか。

Astra Control Provisionerは、Astra Controlの一部として、オープンソースのAstra Tridentでは利用できないストレージプロビジョニング機能のスーパーセットをサポートします。これらの機能は、オープンソースのTridentで利用できるすべての機能に加えて提供されます。

Astra Control Provisionerは**Astra Trident**の後継ですか。

Astra Control Provisionerは、Astra Controlアーキテクチャ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としてAstra Tridentに代わるものです。Astra Controlを使用する場合は、 "Astra Control Provisionerを有効にする" Astra Controlを使用する場合。Astra Tridentはこのリリースでも引き続きサポートされますが、今後のリリースではサポートされません。Astra Tridentは引き続きオープンソースであり、NetAppの新しいCSIやその他の機能でリリース、保守、サポート、更新されます。ただし、Astra Controlの今後のリリースで使用できるのは、Astra TridentのCSI機能と拡張されたストレージ管理機能を備えたAstra Control Provisionerだけです。

Astra Tridentの料金は発生しますか？

いいえAstra Tridentは引き続きオープンソースであり、無償でダウンロードできます。Astra Control Provisioner機能を使用するには、Astra Controlライセンスが必要です。

Astra Controlをすべてインストールして使用しなくても、**Astra Control**でストレージ管理機能とプロビジョニング機能を使用できますか。

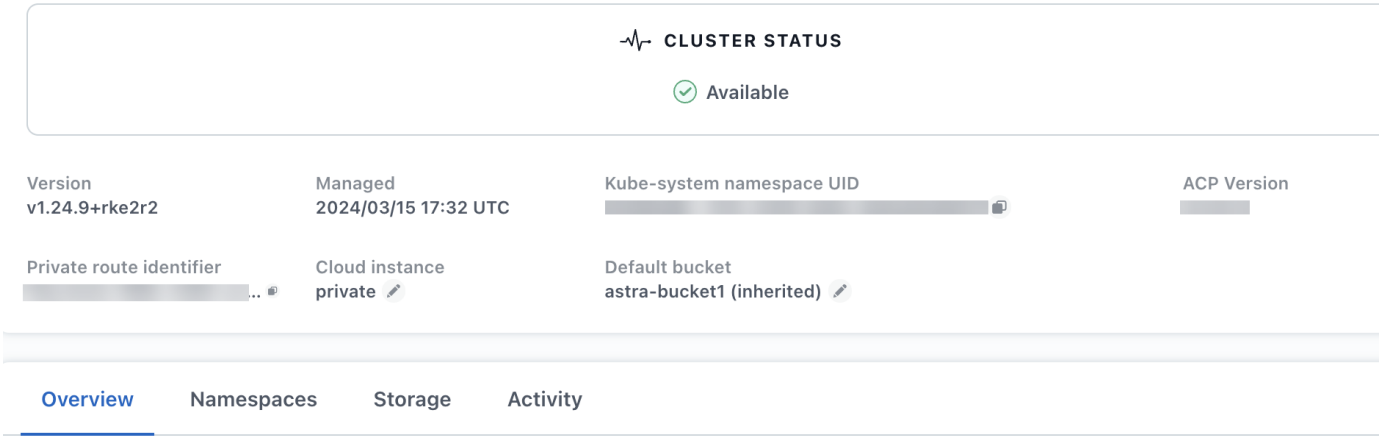
はい。Astra Control Provisionerにアップグレードして、Astra Controlのすべてのデータ管理機能を使用する必要がなくても、その機能を使用できます。

既存の**Astra Trident**ユーザから**Astra Control**に移行して、高度なストレージ管理とプロビジョニングの機能を使用するにはどうすればよいですか？

既存のAstra Tridentユーザ（パブリッククラウドのAstra Tridentのユーザを含む）の場合は、まずAstra Controlライセンスを取得する必要があります。インストールが完了したら、Astra Control Provisionerバンドルをダウンロードし、Astra Tridentをアップグレードし、 "Astra Control Provisioner機能を有効にする"。

クラスタの**Astra Trident**に代わって**Astra Control Provisioner**が使用されているかどうかを確認するにはどうすればよいですか。

Astra Control Provisionerをインストールすると、Astra Control UIのホストクラスタに `ACP version` 代わりに `Trident version` フィールドと現在インストールされているバージョン番号。



UIにアクセスできない場合は、次の方法でインストールが正常に完了したことを確認できます。

Astra Trident運用者

を確認します `trident-acp` コンテナが実行中で、 `acpVersion` はです `23.10.0` またはそれ以降（最小バージョンは23.10）でステータスが `Installed`：

```
kubectl get torc -o yaml
```

対応：

```
status:
  acpVersion: 24.10.0
  currentInstallationParams:
    ...
    acpImage: <my_custom_registry>/trident-acp:24.10.0
    enableACP: "true"
    ...
  ...
  status: Installed
```

Tridentctl

Astra Control Provisionerが有効になっていることを確認します。

```
./tridentctl -n trident version
```

対応：

```
+----------------+----------------+-------------+ | SERVER VERSION |
CLIENT VERSION | ACP VERSION | +----------------+----------------
+-------------+ | 24.10.0 | 24.10.0 | 24.10.0. | +----------------
+----------------+-------------+
```

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 Control API ライセンス

https://docs.netapp.com/us-en/astra-automation/media/astra-api-license.pdf