Astra Control Service 문서

Astra Control Service

NetApp
June 04, 2024

목차

Astra Control Service 문서

릴리스 정보

Astra Control Service의 새로운 기능

NetApp은 새로운 기능, 개선 사항 및 버그 수정을 제공하기 위해 Astra Control Service를 정기적으로 업데이트합니다.

2024년 3월 14일

(기술 미리 보기**)** 선언적 **Kubernetes** 워크플로

이 Astra Control Service 릴리즈에는 기본 Kubernetes 맞춤형 리소스(CR)에서 데이터 관리를 수행할 수 있는 선언적 Kubernetes 기능이 포함되어 있습니다.

이 기능은 Astra Control Service EAP(Early Adopter Program) 인스턴스에서만 사용할 수 있습니다. 에 가입하는 방법에 대한 자세한 내용은 NetApp 영업 담당자에게 문의하십시오.

를 설치한 후 "Astra 커넥터" 관리하려는 클러스터에서 UI 또는 CR에서 다음 CR 기반 클러스터 작업을 수행할 수 있습니다.

- "사용자 지정 리소스를 사용하여 응용 프로그램을 정의합니다"
- "버킷을 정의합니다"
- "전체 클러스터를 보호합니다"
- "응용 프로그램을 백업합니다"
- "스냅샷을 생성합니다"
- "스냅샷 또는 백업에 대한 스케줄을 생성합니다"
- "스냅샷 또는 백업에서 애플리케이션을 복원합니다"

2023년 11월 7일

새로운 기능 및 지원

- * ONTAP-nas-이코노미 드라이버 기반 스토리지 백엔드를 사용하는 애플리케이션을 위한 백업 및 복원 기능 *: 에 대한 백업 및 복원 작업을 활성화합니다 `ontap-nas-economy` 몇 가지 "간단한 단계".
- * 온프레미스 Red Hat OpenShift Container Platform 클러스터에 대한 Astra Control Service 지원 *

  "클러스터를 추가합니다"

- * 변경 불가능한 백업 * : Astra Control이 이제 지원합니다 "변경 불가능한 읽기 전용 백업" 멀웨어 및 기타 위협에 대한 추가 보안 레이어로 사용됩니다.
- * Astra Control Provisioner 소개 *

  23.10 릴리스와 함께 Astra Control은 Astra Control Provisioner라는 새로운 소프트웨어 구성 요소를 도입했으며 이 소프트웨어 구성 요소는 라이센스가 있는 모든 Astra Control 사용자가 사용할 수 있습니다. Astra Control Provisioner는 Astra Trident가 제공하는 기능을 능가하는 고급 관리 및 스토리지 프로비저닝 기능을 제공합니다. 이들 기능은 모든 Astra Control 고객에게 추가 비용 없이 제공됩니다.

◦ * Astra Control Provisioner * 를 시작하십시오
가능합니다 "Astra Control Provisioner를 활성화합니다" Astra Trident 23.10을 사용하도록 환경을 설치 및 구성한 경우

◦ * Astra Control Provisioner 기능 *

Astra Control Provisioner 23.10 릴리즈에서는 다음 기능을 사용할 수 있습니다.

- **Kerberos 5** 암호화를 통한 향상된 스토리지 백엔드 보안: 을(를) 사용하여 스토리지 보안을 향상시킬 수 있습니다 "암호화 활성화 중" 관리되는 클러스터와 스토리지 백엔드 사이의 트래픽에 사용됩니다. Astra Control Provisioner는 Red Hat OpenShift 클러스터에서 Azure NetApp Files 및 사내 ONTAP 볼륨으로의 NFSv4.1 연결을 통해 Kerberos 5 암호화를 지원합니다.
- * 스냅샷을 사용하여 데이터 복구 *: Astra Control Provisioner는 를 사용하여 스냅샷에서 신속하게 제자리에서 볼륨을 복원할 수 있습니다 `TridentActionSnapshotRestore` (TASR) CR
- * 를 사용하여 응용 프로그램을 위한 백업 및 복원 기능 `ontap-nas-economy` 드라이버 지원 스토리지 백엔드 *: 설명 참조 있습니다.

- * AWS(ROSA) 클러스터 기반의 Red Hat OpenShift Service에 대한 Astra Control Service 지원 *

  "클러스터를 추가합니다"

- * NVMe/TCP 스토리지를 사용하는 응용 프로그램 관리 지원 *
  Astra Control은 이제 NVMe/TCP를 통해 연결된 영구 볼륨의 지원을 받는 애플리케이션을 관리할 수 있다.
- * 실행 후크는 기본적으로 해제되어 있습니다. * :이 릴리스부터는 실행 후크 기능이 작동할 수 있습니다 "활성화됨" 또는 추가 보안을 위해 사용 안 함(기본적으로 사용 안 함)을 선택합니다. Astra Control에서 사용할 실행 후크를 아직 생성하지 않은 경우, 다음을 수행해야 합니다 "실행 후크 기능을 활성화합니다" 후크를 만들기 시작합니다. 이 릴리스 이전에 실행 후크를 만든 경우 실행 후크 기능은 계속 사용할 수 있으며 평소처럼 후크를 사용할 수 있습니다.

2023년 10월 2일

새로운 기능 및 지원

이것은 사소한 버그 수정 릴리스입니다.

2023년 7월 27일

새로운 기능 및 지원

- 이제 클론 작업은 라이브 클론만 지원합니다(관리되는 애플리케이션의 현재 상태). 스냅샷 또는 백업에서 복제하려면 복원 워크플로우를 사용합니다.

  "앱 복원"

2023년 6월 26일

새로운 기능 및 지원

- Azure Marketplace 구독은 이제 분당 요금이 아닌 시간당 청구됩니다

  "대금 청구를 설정합니다"

2023년 5월 30일

새로운 기능 및 지원

- 전용 Amazon EKS 클러스터 지원

  "Astra Control Service에서 프라이빗 클러스터를 관리합니다"

- 복원 또는 클론 복제 작업 중에 대상 스토리지 클래스를 선택할 수 있도록 지원합니다

  "앱 복원"

2023년 5월 15일

새로운 기능 및 지원

이것은 사소한 버그 수정 릴리스입니다.

2023년 4월 25일

새로운 기능 및 지원

- 전용 Red Hat OpenShift 클러스터 지원

  "Astra Control Service에서 프라이빗 클러스터를 관리합니다"

- 복원 작업 중에 애플리케이션 리소스를 포함 또는 제외하는 지원

  "앱 복원"

- 데이터 전용 애플리케이션 관리 지원

  "앱 관리를 시작합니다"

2023년 1월 17일

새로운 기능 및 지원

- 추가 필터링 옵션이 포함된 향상된 실행 후크 기능

  "앱 실행 후크 관리"

- NetApp Cloud Volumes ONTAP를 스토리지 백엔드로 지원합니다

  "Astra Control에 대해 알아보십시오"

2022년 11월 22일

새로운 기능 및 지원

- 여러 네임스페이스에 걸쳐 있는 응용 프로그램 지원

  "앱 정의"

- 애플리케이션 정의에 클러스터 리소스 포함 지원

  "앱 정의"

- 백업, 복원 및 클론 작업에 대한 향상된 진행률 보고 기능

  "실행 중인 작업을 모니터링합니다"

- 이미 호환되는 버전의 Astra Trident가 설치된 클러스터 관리 지원

  "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다"

- 단일 Astra Control Service 계정으로 여러 클라우드 공급자 서브스크립션 관리 지원

  "클라우드 인스턴스 관리"

- 퍼블릭 클라우드 환경에서 호스팅되는 셀프 관리 Kubernetes 클러스터를 Astra Control Service에 추가할 수 있습니다

  "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다"

- 이제 Astra Control Service에 대한 청구는 애플리케이션별로 계산되지 않고 네임스페이스당 지불됩니다

  "대금 청구를 설정합니다"

- AWS 마켓플레이스를 통해 Astra Control Service 계약 기간 기반 혜택에 대한 구독을 지원합니다

  "대금 청구를 설정합니다"

알려진 문제 및 제한 사항

- "이 릴리스에 대해 알려진 문제입니다"
- "이 릴리스에 대해 알려진 제한 사항입니다"

2022년 9월 7일

이 릴리스에는 Astra Control Service 인프라의 안정성 및 복원성 향상 기능이 포함되어 있습니다.

2022년 8월 10일

이 릴리즈에는 다음과 같은 새로운 기능과 향상된 기능이 포함되어 있습니다.

- 향상된 애플리케이션 관리 워크플로우 향상된 애플리케이션 관리 워크플로우를 통해 Astra Control에서 관리하는 애플리케이션을 정의할 때 유연성을 높일 수 있습니다.

  "앱 관리"

- Amazon Web Services 클러스터 Astra Control Service에 대한 지원을 통해 이제 Amazon Elastic Kubernetes Service에서 호스팅되는 클러스터에서 실행 중인 앱을 관리할 수 있습니다. Amazon Elastic Block Store 또는 NetApp ONTAP용 Amazon FSx를 스토리지 백엔드로 사용하도록 클러스터를 구성할 수 있습니다.

  "Amazon Web Services를 설정합니다"

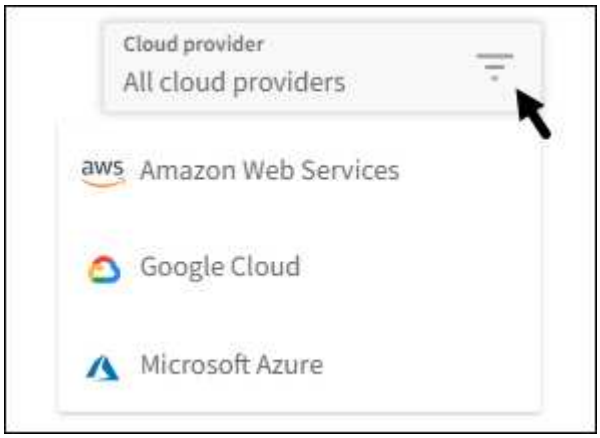
- 향상된 실행 후크 사전 및 사후 스냅샷 실행 후크뿐만 아니라 다음과 같은 유형의 실행 후크를 구성할 수 있습니다.
  - 사전 백업
  - 백업 후
  - 사후 복원

  그 외에도 Astra Control은 이제 여러 실행 후크에 대해 동일한 스크립트를 사용할 수 있도록 지원합니다.

  > 이 릴리즈에서는 특정 애플리케이션에 대해 NetApp에서 제공한 기본 사전/사후 스냅샷 실행 후크가 제거되었습니다. 스냅샷에 대한 실행 후크를 제공하지 않으면 Astra Control Service는 2022년 8월 4일부터 충돌 시에도 정합성이 보장되는 스냅샷을 생성합니다. 를 방문하십시오 "NetApp Verda GitHub 저장소" 사용자 환경에 맞게 수정할 수 있는 샘플 실행 후크 스크립트의 경우

  "앱 실행 후크 관리"

- Azure Marketplace 지원 이제 Azure Marketplace를 통해 Astra Control Service에 등록할 수 있습니다.
- 클라우드 공급자 선택 Astra Control Service 설명서를 읽는 동안 페이지 오른쪽 상단에서 클라우드 공급자를 선택할 수 있습니다. 선택한 클라우드 공급자와 관련된 설명서만 표시됩니다.



2022년 4월 26일

이 릴리즈에는 다음과 같은 새로운 기능과 향상된 기능이 포함되어 있습니다.

- 네임스페이스 역할 기반 액세스 제어(RBAC) Astra Control Service는 이제 구성원 또는 뷰어 사용자에게 네임스페이스 제약 조건을 할당할 수 있도록 지원합니다.

  "네임스페이스 역할 기반 액세스 제어(RBAC)"

- Azure Active Directory 지원 Astra Control Service는 인증 및 ID 관리를 위해 Azure Active Directory를 사용하는 AKS 클러스터를 지원합니다.

  "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다"

- 전용 AKS 클러스터 지원 이제 전용 IP 주소를 사용하는 AKS 클러스터를 관리할 수 있습니다.

  "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다"

- Astra Control에서 버킷 제거 이제 Astra Control Service에서 버킷을 제거할 수 있습니다.

  "버킷을 탈거하십시오"

2021년 12월 14일

이 릴리즈에는 다음과 같은 새로운 기능과 향상된 기능이 포함되어 있습니다.

- 새로운 스토리지 백엔드 옵션
- 데이터 이동 없이 앱 복원을 사용하면 동일한 클러스터 및 네임스페이스로 복원하여 앱의 스냅샷, 클론 복제 또는 백업을 현재 위치에서 복원할 수 있습니다.

  "앱 복원"

- 실행 후크가 있는 스크립트 이벤트 Astra Control은 응용 프로그램의 스냅샷을 생성하기 전이나 후에 실행할 수 있는 사용자 지정 스크립트를 지원합니다. 따라서 데이터베이스 앱의 스냅샷이 일관성을 유지하도록 데이터베이스 트랜잭션을 일시 중지하는 등의 작업을 수행할 수 있습니다.

  "앱 실행 후크 관리"

- 운영자로 구축된 앱 Astra Control은 운영자와 함께 배포할 때 일부 앱을 지원합니다.

  "앱 관리를 시작합니다"

- 리소스 그룹 범위가 Astra Control Service 인 서비스 보안 주체는 이제 리소스 그룹 범위를 사용하는 서비스 보안 주체를 지원합니다.

  "Azure 서비스 보안 주체 만들기"

2021년 8월 5일

이 릴리즈에는 다음과 같은 새로운 기능과 향상된 기능이 포함되어 있습니다.

- Astra 제어 센터
  Astra Control은 이제 새로운 배포 모델로 제공됩니다. _Astra Control Center_는 데이터 센터에 설치하고 운영하여 온프레미스 Kubernetes 클러스터의 Kubernetes 애플리케이션 라이프사이클 관리를 관리할 수 있는 자체 관리형 소프트웨어입니다.

  자세한 내용은 "Astra Control Center 문서로 이동합니다".

- 이제 고유한 버킷을 가져오십시오. Astra가 백업 및 복제에 사용하는 버킷을 관리하려면 다른 버킷을 추가하고 클라우드 공급자의 Kubernetes 클러스터의 기본 버킷을 변경하면 됩니다.

  "버킷을 관리합니다"

2021년 6월 2일

이 릴리스에는 버그 수정 및 Google Cloud 지원에 대한 다음과 같은 개선 사항이 포함되어 있습니다.

- 공유 VPC 지원 이제 공유 VPC 네트워크 구성을 사용하여 GCP 프로젝트의 GKE 클러스터를 관리할 수 있습니다.

- CVS 서비스 유형 Astra Control Service의 영구 볼륨 크기는 이제 CVS 서비스 유형을 사용할 때 최소 300GiB의 영구 볼륨을 생성합니다.

  "Astra Control Service가 Cloud Volumes Service for Google Cloud를 영구 볼륨의 스토리지 백엔드로 사용하는 방법에 대해 알아보십시오".

- 컨테이너 최적화 OS 컨테이너 최적화 OS에 대한 지원은 이제 GKE 작업자 노드에서 지원됩니다. 이것은 Ubuntu에 대한 지원 외에 추가로 제공됩니다.

  "GKE 클러스터 요구 사항에 대해 자세히 알아보십시오".

2021년 4월 15일

이 릴리즈에는 다음과 같은 새로운 기능과 향상된 기능이 포함되어 있습니다.

- AKS 클러스터 Astra Control Service에 대한 지원은 이제 Azure Kubernetes Service(AKS)의 관리되는 Kubernetes 클러스터에서 실행 중인 앱을 관리할 수 있습니다.

  "시작하는 방법을 알아보십시오".

- REST API 이제 Astra Control REST API를 사용할 수 있습니다. 이 API는 최신 기술과 최신 모범 사례를 기반으로 합니다.

  "REST API를 사용하여 애플리케이션 데이터 라이프사이클 관리를 자동화하는 방법에 대해 알아보십시오".

- 연간 서브스크립션 Astra Control Service는 이제 _ Premium Subscription _ 을(를) 제공합니다.

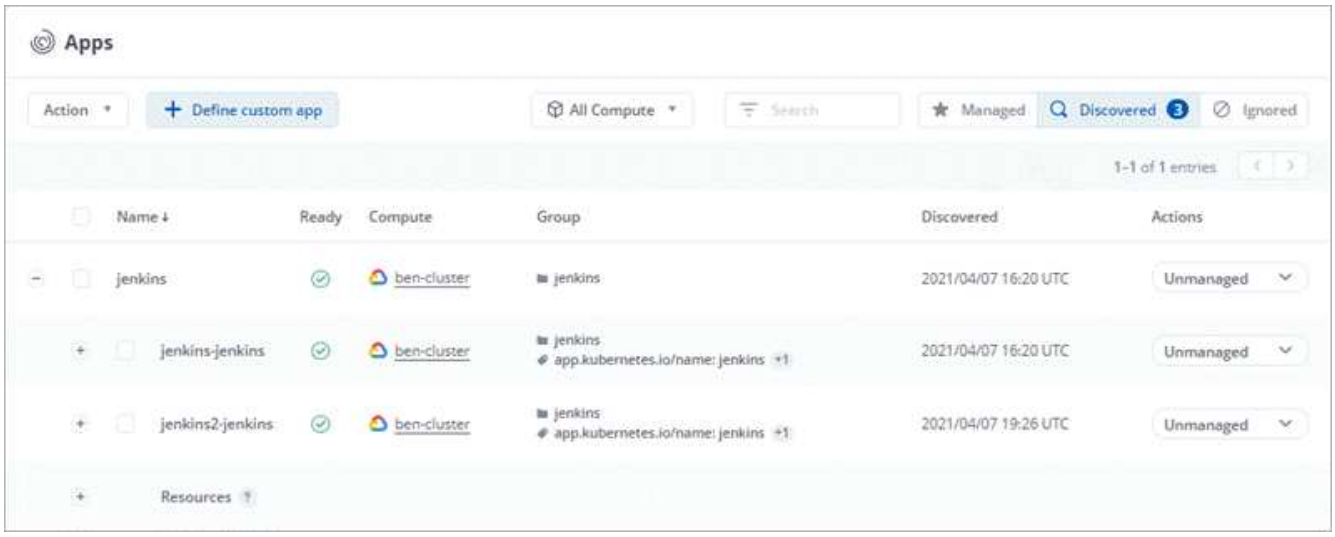
  연 단위 가입으로 할인된 요금으로 선결제하면 _application pack_당 최대 10개의 앱을 관리할 수 있습니다. NetApp 세일즈 팀에 문의하여 조직에 필요한 만큼 팩을 구매하십시오. 예를 들어, Astra Control Service에서 30개의 앱을 관리하려면 3팩 을 구입하십시오.

  연간 구독에서 허용하는 것보다 더 많은 앱을 관리하는 경우 응용 프로그램당 분당 $0.005의 초과 요금(Premium PayGo와 동일)으로 청구됩니다.

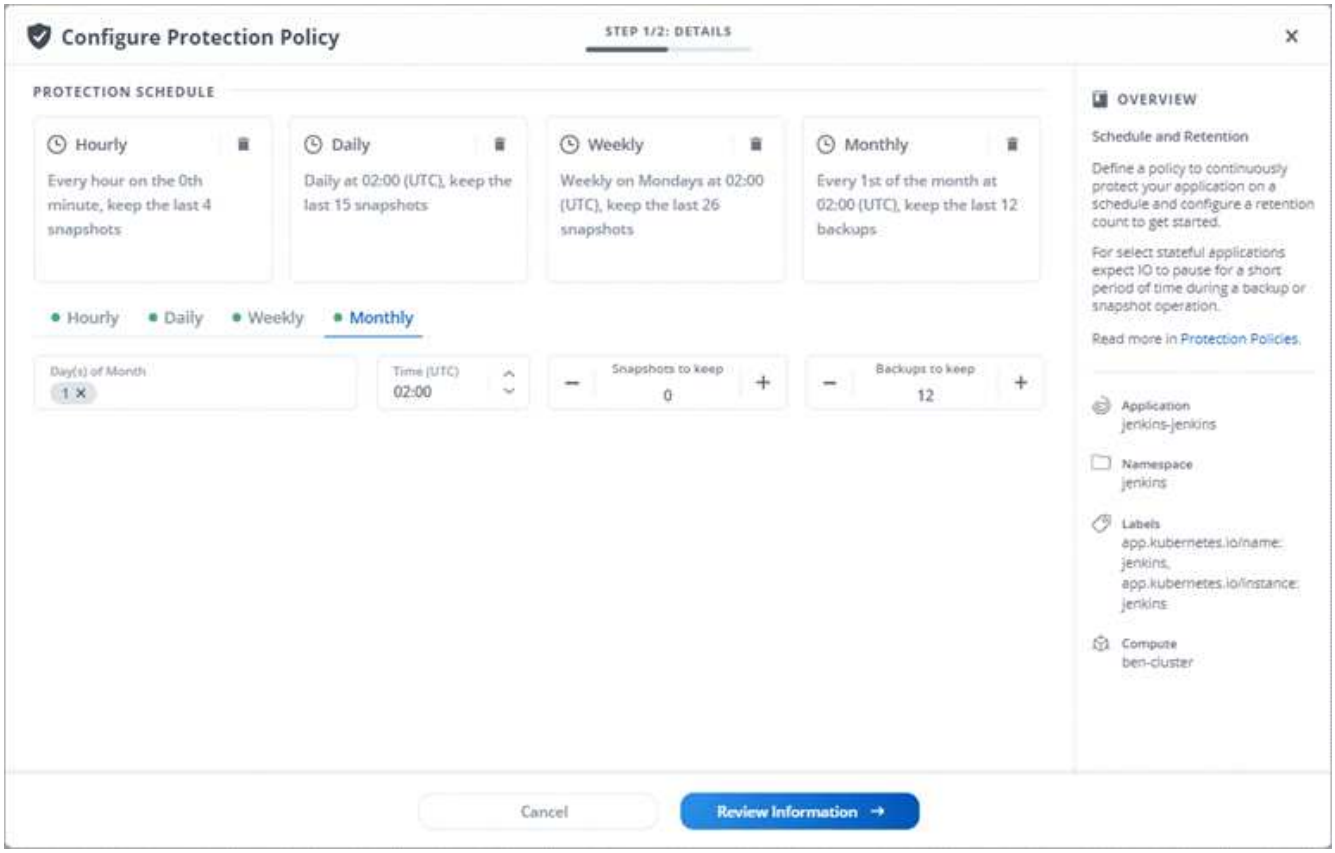
  "Astra Control Service 가격에 대해 자세히 알아보십시오".

- 네임스페이스 및 앱 시각화 우리는 네임스페이스와 앱 간의 계층 구조를 더 잘 표시하기 위해 검색된 앱 페이지를 개선했습니다. 네임스페이스를 확장하여 해당 네임스페이스에 포함된 앱을 확인하기만 하면 됩니다.

  "앱 관리에 대해 자세히 알아보십시오".



- 사용자 인터페이스 향상 데이터 보호 마법사가 향상되어 사용이 간편했습니다. 예를 들어 보호 정책 마법사를 정의하여 보호 스케줄을 보다 쉽게 확인할 수 있습니다.



- 활동 개선 사항 Astra Control 계정의 활동에 대한 세부 정보를 보다 쉽게 확인할 수 있도록 했습니다.
  - 관리 앱, 심각도 수준, 사용자 및 시간 범위를 기준으로 활동 목록을 필터링합니다.
  - Astra Control 계정 활동을 CSV 파일로 다운로드합니다.
  - 클러스터 또는 앱을 선택한 후 클러스터 페이지 또는 앱 페이지에서 직접 활동을 봅니다.

    "계정 활동 보기에 대해 자세히 알아보세요".

2021년 3월 1일

이제 Astra Control Service가 를 지원합니다 "_CVS_서비스 유형입니다" Google Cloud용 Cloud Volumes Service와 함께. 이는 이미 _CVS - Performance_service 유형을 지원하는 것 외에도 가능합니다. Astra Control Service는 Cloud Volumes Service for Google Cloud를 영구 볼륨의 스토리지 백엔드로 사용합니다.

이는 Astra Control Service가 이제 _any_에서 실행 중인 Kubernetes 클러스터의 애플리케이션 데이터를 관리할 수 있다는 것을 의미합니다 "Cloud Volumes Service가 지원되는 Google 클라우드 지역".

Google Cloud 지역 중에서 선택할 수 있는 유연성이 있다면 성능 요구사항에 따라 CVS 또는 CVS 성능을 선택할 수 있습니다. "서비스 유형 선택에 대해 자세히 알아보십시오".

2021년 1월 25일

이제 Astra Control Service가 GA될 예정입니다. 당사는 베타 릴리스로부터 받은 많은 피드백을 통합하여 몇 가지 주목할 만한 개선 사항을 만들었습니다.

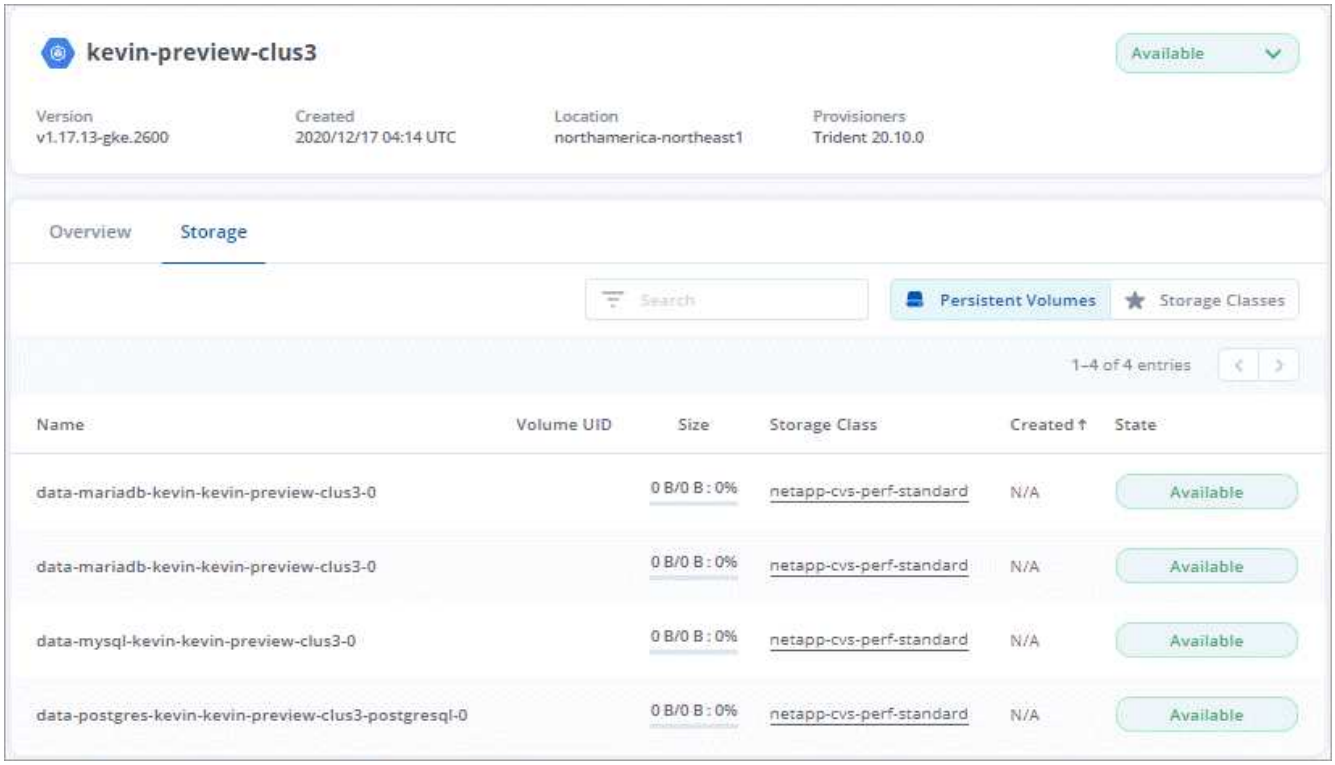
- 무료 요금제로 전환하여 프리미엄 요금제로 전환할 수 있습니다. "청구에 대해 자세히 알아보십시오".
- Astra Control Service는 이제 CVS 성능 서비스 유형을 사용할 때 최소 100GiB의 영구 볼륨을 생성합니다.
- Astra Control Service는 이제 앱을 더 빠르게 검색할 수 있습니다.
- 이제 직접 계정을 만들고 삭제할 수 있습니다.
- Astra Control Service에서 Kubernetes 클러스터에 더 이상 액세스할 수 없을 때 알림 기능이 개선되었습니다.

  Astra Control Service는 연결이 끊긴 클러스터에 대한 앱을 관리할 수 없기 때문에 이러한 알림이 중요합니다.

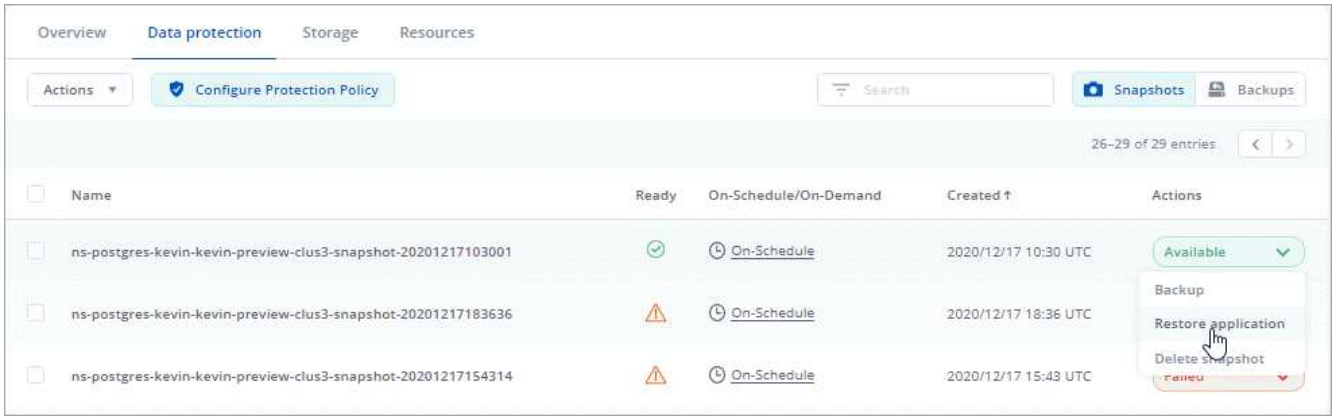
2020년 12월 17일(베타 업데이트)

사용자 경험을 개선하기 위해 주로 버그 픽스에 초점을 맞추었지만 주목할 만한 몇 가지 개선 사항은 다음과 같습니다.

- 첫 번째 Kubernetes 컴퓨팅을 Astra Control Service에 추가하면 클러스터가 있는 지역에 오브젝트 저장소가 생성됩니다.
- 이제 컴퓨팅 레벨에서 스토리지 세부 정보를 볼 때 영구 볼륨에 대한 세부 정보를 확인할 수 있습니다.



- 기존 스냅샷 또는 백업에서 애플리케이션을 복원하는 옵션이 추가되었습니다.



- Astra Control Service에서 관리하는 Kubernetes 클러스터를 삭제하면 클러스터가 * 제거됨 * 상태로 표시됩니다. 그런 다음 Astra Control Service에서 클러스터를 제거할 수 있습니다.
- 이제 계정 소유자는 다른 사용자에 대해 할당된 역할을 수정할 수 있습니다.
- 일반 가용성(GA)을 위해 Astra Control Service가 릴리스될 때 사용할 수 있는 청구 섹션을 추가했습니다.

알려진 문제

알려진 문제점은 이 제품 릴리스를 성공적으로 사용하지 못하게 만들 수 있는 문제를 식별합니다.

현재 릴리즈에는 다음과 같은 알려진 문제가 영향을 줍니다.

인프라

- 삭제되어 다시 생성된 네임스페이스에 앱을 정의할 수 없습니다

백업, 복원, 클론

- 특정 버전의 PostgreSQL을 사용하여 앱 클론이 실패합니다
- 클러스터를 관리하고 볼륨을 추가한 경우 앱 백업 및 스냅숏이 실패합니다
- Kerberos 전송 중 암호화를 사용할 때 백업에서 복원하지 못할 수 있습니다
- 백업 데이터는 보존 정책이 만료된 버킷에 대해 삭제 후에도 버킷에 남아 있습니다

기타 문제

- Astra Trident가 오프라인일 때 내부 서비스 오류(500)와 함께 앱 데이터 관리 작업이 실패했습니다

삭제되어 다시 생성된 네임스페이스에 앱을 정의할 수 없습니다

네임스페이스로 응용 프로그램을 정의하고 네임스페이스를 삭제한 다음 같은 네임스페이스에 응용 프로그램을 다시 설치하면 409 오류 코드와 함께 작업이 실패합니다. 다시 생성된 네임스페이스를 사용하여 앱을 정의하려면 먼저 이전 응용 프로그램 인스턴스를 삭제합니다.

특정 버전의 **PostgreSQL**을 사용하여 앱 클론이 실패합니다

동일한 클러스터 내의 앱 클론은 Bitnami PostgreSQL 11.5.0 차트와 함께 일관되게 실패합니다. 클론을 성공적으로 생성하려면 이전 또는 이후 버전의 차트를 사용하십시오.

클러스터를 관리하고 볼륨을 추가한 경우 앱 백업 및 스냅숏이 실패합니다

이 시나리오에서 백업 및 스냅샷이 실패하고 UI 500 오류가 표시됩니다. 이 문제를 해결하려면 앱 목록을 새로 고치십시오.

Kerberos 전송 중 암호화를 사용할 때 백업에서 복원하지 못할 수 있습니다

백업에서 Kerberos 전송 중 암호화를 사용하는 스토리지 백엔드로 응용 프로그램을 복원하면 복원 작업이 실패할 수 있습니다. 이 문제는 스냅샷에서 복원하거나 NetApp SnapMirror를 사용하여 애플리케이션 데이터를 복제하는 데는 영향을 주지 않습니다.

> NFSv4 볼륨에서 Kerberos 전송 중 암호화를 사용하는 경우 NFSv4 볼륨에서 올바른 설정을 사용하고 있는지 확인하십시오. 의 NetApp NFSv4 도메인 구성 섹션(13페이지)을 참조하십시오 "NetApp NFSv4의 향상된 기능 및 모범 사례 가이드 를 참조하십시오".

백업 데이터는 보존 정책이 만료된 버킷에 대해 삭제 후에도 버킷에 남아 있습니다

버킷의 보존 정책이 만료된 후 앱의 변경 불가능한 백업을 삭제하면 Astra Control에서 백업이 삭제되지만 버킷에서는 삭제되지 않습니다. 이 문제는 다음 릴리스에서 해결될 예정입니다.

Astra Trident가 오프라인일 때 내부 서비스 오류**(500)**와 함께 앱 데이터 관리 작업이 실패했습니다

앱 클러스터의 Astra Trident가 오프라인 상태가 되고 다시 온라인 상태가 되고 앱 데이터 관리를 시도할 때 500 내부 서비스 오류가 발생하는 경우, 앱 클러스터의 모든 Kubernetes 노드를 다시 시작하여 기능을 복원합니다.

알려진 제한 사항

알려진 제한 사항은 이 제품 릴리스에서 지원하지 않거나 올바르게 상호 운용되지 않는 플랫폼, 장치 또는 기능을 식별합니다. 이러한 제한 사항을 주의 깊게 검토하십시오.

일반 제한 사항

지원되는 Kubernetes 구현에서 Astra Control Service의 Kubernetes 클러스터 관리에 영향을 미치는 요소는 다음과 같습니다.

Postgres POD에 대한 기존 연결로 인해 오류가 발생합니다

Postgres Pod에서 작업을 수행할 때 psql 명령을 사용하기 위해 POD 내에서 직접 연결하면 안 됩니다. Astra Control Service는 데이터베이스를 고정 및 고정 해제할 수 있도록 psql 액세스 권한이 필요합니다. 기존 접속이 있는 경우 스냅샷, 백업 또는 클론이 실패합니다.

활동 페이지에는 최대 **100,000**개의 이벤트가 표시됩니다

Astra Control Activity 페이지에는 최대 100,000개의 이벤트가 표시될 수 있습니다. 기록된 이벤트를 모두 보려면 를 사용하여 이벤트를 검색합니다 "Astra Control REST API".

GKE 클러스터 관리에 대한 제한 사항

GKE(Google Kubernetes Engine)에서 Kubernetes 클러스터 관리에 다음과 같은 제한 사항이 적용됩니다.

앱 관리 제한 사항

다음 제한 사항은 Astra Control Service의 애플리케이션 관리에 영향을 줍니다.

ONTAP의 데이터 이동 없이 복원 작업 **- NAS** 이코노미 스토리지 클래스에 장애가 발생했습니다

응용 프로그램의 전체 복원을 수행하고(응용 프로그램을 원래 네임스페이스로 복원) 앱의 저장소 클래스는 을 사용합니다 `ontap-nas-economy` 드라이버, 스냅샷 디렉토리가 숨겨져 있지 않으면 복구 작업이 실패할 수 있습니다. 원래 위치로 복원하기 전에 의 지침을 따릅니다 "ONTAP - NAS - 경제성 작업을 위한 백업 및 복원 지원" 스냅샷 디렉토리를 숨깁니다.

동일한 네임스페이스를 사용하는 여러 응용 프로그램을 집합적으로 다른 네임스페이스로 복원할 수 없습니다

동일한 네임스페이스를 사용하는 여러 응용 프로그램을 관리하는 경우(Astra Control에서 여러 응용 프로그램 정의 생성) 모든 응용 프로그램을 다른 단일 네임스페이스로 복원할 수 없습니다. 각 애플리케이션을 별도의 네임스페이스로 복원해야 합니다.

Astra Control은 클라우드 인스턴스에 대해 기본 버킷을 자동으로 할당하지 않습니다

Astra Control은 클라우드 인스턴스에 대해 기본 버킷을 자동으로 할당하지 않습니다. 클라우드 인스턴스의 기본 버킷을 수동으로 설정해야 합니다. 기본 버킷을 설정하지 않으면 두 클러스터 간에 애플리케이션 클론 작업을 수행할 수 없습니다.

인증서 관리자를 사용하는 앱의 데이터 이동 없는 복원 작업은 지원되지 않습니다

이 Astra Control Service 릴리스는 인증서 관리자와의 응용 프로그램 데이터 이동 없는 복원을 지원하지 않습니다. 복원 작업을 다른 네임스페이스로 복원하고 클론 작업을 지원합니다.

애플리케이션 클론을 세트 스토리지 클래스로 구축한 후에는 애플리케이션 클론이 실패합니다

애플리케이션이 명시적으로 설정된 스토리지 클래스(예: 'helm install…-set global.storageClass=NetApp-cvs-perf-extreme')로 구축된 후 애플리케이션을 복제하려는 이후에 타겟 클러스터에 원래 지정된 스토리지 클래스가 있어야 합니다. 명시적으로 설정된 스토리지 클래스를 가진 애플리케이션을 동일한 스토리지 클래스가 없는 클러스터로 클론 복제하면 실패합니다. 이 시나리오에서는 복구 단계가 없습니다.

PASS by reference operator를 사용하여 설치된 앱의 클론이 실패할 수 있습니다

Astra Control은 네임스페이스 범위 연산자와 함께 설치된 앱을 지원합니다. 이러한 연산자는 일반적으로 "pass-by-reference" 아키텍처가 아니라 "pass-by-value"로 설계되었습니다. 다음은 이러한 패턴을 따르는 일부 운영자 앱에 대한 설명입니다.

- "아파치 K8ssandra"

> K8ssandra 의 경우 현재 위치 복원 작업이 지원됩니다. 새 네임스페이스 또는 클러스터에 대한 복원 작업을 수행하려면 응용 프로그램의 원래 인스턴스를 중단해야 합니다. 이는 이월된 피어 그룹 정보가 인스턴스 간 통신으로 이어지지 않도록 하기 위한 것입니다. 앱 복제는 지원되지 않습니다.

- "젠킨스 CI"
- "Percona XtraDB 클러스터"

Astra Control은 "pass-by-reference" 아키텍처(예: CockroachDB 운영자)로 설계된 운영자를 복제하지 못할 수 있습니다. 이러한 유형의 클론 복제 작업 중에 클론 복제 운영자는 클론 복제 프로세스의 일부로 고유한 새로운 암호가 있음에도 불구하고 소스 운영자의 Kubernetes 암호를 참조하려고 합니다. Astra Control이 소스 운영자의 Kubernetes 암호를 모르기 때문에 클론 작업이 실패할 수 있습니다.

> 클론 작업 중에 IngressClass 리소스 또는 Webhook가 필요한 애플리케이션에는 대상 클러스터에 이미 정의된 리소스가 없어야 합니다.

역할 기반 액세스 제어(RBAC) 제한 사항

Astra Control에서 리소스 또는 기능에 대한 사용자 액세스를 제한하는 방식에 다음과 같은 제한 사항이 적용됩니다.

네임스페이스 **RBAC** 제약 조건이 있는 사용자는 클러스터를 추가 및 관리할 수 없습니다

네임스페이스 RBAC 제약 조건이 있는 사용자는 클러스터를 추가하거나 관리할 수 없습니다. 현재 제한 사항으로 인해 Astra는 이러한 사용자가 클러스터 관리를 해제하는 것을 방지하지 않습니다.

네임스페이스 제약 조건을 가진 멤버 사용자는 관리자 사용자가 제약 조건에 네임스페이스를 추가할 때까지 복제되거나 복원된 앱에 액세스할 수 없습니다

모두 `member` 네임스페이스 이름/ID별 RBAC 제약 조건을 사용하는 사용자는 앱을 동일한 클러스터의 새 네임스페이스 또는 조직 계정의 다른 클러스터로 클론 복제 또는 복원할 수 있습니다. 그러나 동일한 사용자가 새 네임스페이스에서 복제되거나 복원된 앱에 액세스할 수 없습니다. 클론 또는 복구 작업에서 새 네임스페이스를 생성한 후 계정 관리자 /소유자가 을 편집할 수 있습니다 `member` 영향을 받는 사용자가 새 네임스페이스에 대한 액세스 권한을 부여하도록

사용자 계정 및 역할 제약 조건을 업데이트합니다.

특정 스냅샷 컨트롤러 버전을 사용하는 **Kubernetes 1.25** 이상 클러스터의 경우 스냅샷이 실패할 수 있습니다

버전 1.25 이상을 실행하는 Kubernetes 클러스터의 스냅샷은 버전 v1beta1 의 스냅샷 컨트롤러 API가 클러스터에 설치된 경우 실패할 수 있습니다.

이 문제를 해결하려면 기존 Kubernetes 1.25 이상 설치를 업그레이드할 때 다음을 수행하십시오.

1. 기존 스냅샷 CRD 및 기존 스냅샷 컨트롤러를 모두 제거합니다.
2. "Astra Trident를 제거합니다".
3. "스냅샷 CRD 및 스냅샷 컨트롤러를 설치합니다".
4. "최신 Astra Trident 버전을 설치합니다".
5. "VolumeSnapshotClass를 생성합니다".

시작하십시오

Astra Control에 대해 알아보십시오

Astra Control은 Kubernetes 애플리케이션 데이터 라이프사이클 관리 솔루션으로, 상태 저장 애플리케이션의 운영을 단순화합니다. Kubernetes 워크로드를 손쉽게 보호, 백업, 마이그레이션하고 정상 작동하는 애플리케이션 클론을 즉시 생성할 수 있습니다.

피처

Astra Control은 Kubernetes 애플리케이션 데이터 라이프사이클 관리에 중요한 기능을 제공합니다.

- 영구 스토리지를 자동으로 관리합니다
- 애플리케이션 인식 필요 시 스냅샷과 백업을 생성합니다
- 정책 기반 스냅샷 및 백업 작업 자동화
- Kubernetes 클러스터 간에 애플리케이션 및 데이터를 마이그레이션합니다
- NetApp SnapMirror 기술(Astra Control Center)을 사용하여 원격 시스템에 애플리케이션 복제
- 스테이징 환경에서 운영 환경으로 애플리케이션 클론 생성
- 애플리케이션 상태 및 보호 상태를 시각화합니다
- 웹 UI 또는 API를 사용하여 백업 및 마이그레이션 워크플로우를 구현합니다

구축 모델

Astra Control은 두 가지 배포 모델로 제공됩니다.

- * Astra Control Service *: 여러 클라우드 공급자 환경의 Kubernetes 클러스터에 대한 애플리케이션 인식 데이터 관리 기능과 자가 관리 Kubernetes 클러스터를 제공하는 NetApp 관리 서비스입니다.
- * Astra Control Center *: 사내 환경에서 실행되는 Kubernetes 클러스터의 애플리케이션 인식 데이터 관리를 제공하는 자체 관리 소프트웨어입니다. 또한 NetApp Cloud Volumes ONTAP 스토리지 백엔드를 통해 여러 클라우드 공급자 환경에 Astra Control Center를 설치할 수 있습니다.

	Astra 제어 서비스	**Astra** 제어 센터
어떻게 제공됩니까**?**	NetApp에서 제공하는 완전 관리형 클라우드 서비스	소프트웨어를 다운로드, 설치 및 관리할 수 있습니다
어디에 호스팅됩니까**?**	NetApp에서 제공하는 다양한 퍼블릭 클라우드 지원	고유한 Kubernetes 클러스터
어떻게 업데이트됩니까**?**	NetApp에서 관리합니다	모든 업데이트를 관리합니다

	Astra 제어 서비스	**Astra** 제어 센터
지원되는 **Kubernetes** 배포는 무엇입니까**?**	• * 클라우드 공급자 * ◦ Amazon Web Services에서 직접 지원합니다 ▪ Amazon Elastic Kubernetes Service(EKS) ◦ Google 클라우드 ▪ Google Kubernetes Engine(GKE) ◦ Microsoft Azure를 참조하십시오 ▪ Azure Kubernetes 서비스(AKS) • * 자가 관리형 클러스터 * ◦ Kubernetes(업스트림) ◦ RKE(Rancher Kubernetes Engine) ◦ Red Hat OpenShift Container Platform • * 온프레미스 클러스터 * ◦ Red Hat OpenShift Container Platform 온프레미스	• Azure Stack HCI 기반 Azure Kubernetes Service • Google Anthos • Kubernetes(업스트림) • RKE(Rancher Kubernetes Engine) • Red Hat OpenShift Container Platform

	Astra 제어 서비스	Astra 제어 센터
지원되는 스토리지 백엔드는 무엇입니까?	• * 클라우드 공급자 * ◦ Amazon Web Services에서 직접 지원합니다 ▪ Amazon EBS ▪ NetApp ONTAP용 Amazon FSx ▪ "Cloud Volumes ONTAP" ◦ Google 클라우드 ▪ Google 영구 디스크 ▪ NetApp Cloud Volumes Service를 참조하십시오 ▪ "Cloud Volumes ONTAP" ◦ Microsoft Azure를 참조하십시오 ▪ Azure 관리 디스크 ▪ Azure NetApp Files ▪ "Cloud Volumes ONTAP" • * 자가 관리형 클러스터 * ◦ Amazon EBS ◦ Azure 관리 디스크 ◦ Google 영구 디스크 ◦ "Cloud Volumes ONTAP" ◦ NetApp MetroCluster ◦ "롱혼" • * 온프레미스 클러스터 * ◦ NetApp MetroCluster ◦ NetApp ONTAP AFF 및 FAS 시스템 ◦ NetApp ONTAP Select를 참조하십시오 ◦ "Cloud Volumes ONTAP" ◦ "롱혼"	• NetApp ONTAP AFF 및 FAS 시스템 • NetApp ONTAP Select를 참조하십시오 • "Cloud Volumes ONTAP" • "롱혼"

Astra Control Service의 작동 방식

Astra Control Service는 NetApp에서 관리하는 클라우드 서비스로, 항상 최신 기능을 사용하여 업데이트 가능합니다. 이 솔루션은 여러 구성 요소를 활용하여 애플리케이션 데이터 수명 주기 관리를 지원합니다.

높은 수준에서 Astra Control Service는 다음과 같이 작동합니다.

- 클라우드 공급자를 설정하고 Astra 계정에 등록하여 Astra Control Service를 시작할 수 있습니다.

+** GKE 클러스터의 경우 Astra Control Service가 사용합니다 "Google Cloud용 NetApp Cloud Volumes Service" 또는 Google 영구 디스크를 영구 볼륨의 스토리지 백엔드로 사용합니다.

AKS 클러스터의 경우 Astra Control Service가 사용합니다 "Azure NetApp Files" 또는 Azure 관리 디스크를 영구 볼륨의 스토리지 백엔드로 사용합니다.

+** Amazon EKS 클러스터의 경우 Astra Control Service가 사용합니다 "Amazon Elastic Block Store를 클릭합니다" 또는 "NetApp ONTAP용 Amazon FSx" 영구 볼륨의 스토리지 백엔드로 사용됩니다.

- 첫 번째 Kubernetes 컴퓨팅을 Astra Control Service에 추가합니다. 그러면 Astra Control Service에서 다음을 수행합니다.
  - 클라우드 공급자 계정에 백업 복사본이 저장되는 개체 저장소를 만듭니다.

Azure에서 Astra Control Service는 Blob 컨테이너용 리소스 그룹, 스토리지 계정 및 키도 생성합니다.

- 클러스터에 새 관리 역할 및 Kubernetes 서비스 계정을 생성합니다.
- 에서는 해당 새 관리자 역할을 사용하여 클러스터에 link../conceptions/architecture #Astra-control-components[Astra Control Provisioner]를 설치하고 하나 이상의 스토리지 클래스를 생성합니다.
- NetApp 클라우드 서비스 스토리지 오퍼링을 스토리지 백엔드로 사용하는 경우 Astra Control Service는 Astra Control Provisioner를 사용하여 앱에 영구 볼륨을 프로비저닝합니다. Amazon EBS 또는 Azure 관리 디스크를 스토리지 백엔드로 사용하는 경우 공급자별 CSI 드라이버를 설치해야 합니다. 설치 지침은 에 나와 있습니다 "Amazon Web Services를 설정합니다" 및 "Azure 관리 디스크를 사용하여 Microsoft Azure를 설정합니다".
  - 이때 클러스터에서 앱을 정의할 수 있습니다. 영구 볼륨은 새로운 기본 스토리지 클래스를 통해 스토리지 백엔드에서 프로비저닝됩니다.
  - 그런 다음 Astra Control Service를 사용하여 이러한 애플리케이션을 관리하고 스냅샷, 백업 및 클론 생성을 시작합니다.

Astra Control의 무료 플랜을 사용하면 최대 10개의 네임스페이스를 계정에서 관리할 수 있습니다. 10개 이상의 네임스페이스를 관리하려면 Free Plan에서 Premium Plan으로 업그레이드하여 요금 청구를 설정해야 합니다.

Astra Control Center의 작동 방식

Astra Control Center는 프라이빗 클라우드에서 로컬로 실행됩니다.

Astra Control Center는 ONTAP 스토리지 백엔드와 함께 Astra Control Provisioner 구성 스토리지 클래스를 통해 Kubernetes 클러스터를 지원합니다.

Astra Control Center는 AutoSupport 및 Active IQ 에코시스템에 완전히 통합되어 사용자와 NetApp 지원에 문제 해결 및 사용 정보를 제공합니다.

90일 평가판 라이센스를 사용하여 Astra Control Center를 사용해 볼 수 있습니다. 평가판 버전은 전자 메일 및 커뮤니티 옵션을 통해 지원됩니다. 또한 제품 내 지원 대시보드에서 Knowledgebase 문서 및 문서에 액세스할 수 있습니다.

Astra Control Center를 설치하고 사용하려면 반드시 충족해야 합니다 "요구 사항".

Astra Control Center는 다음과 같이 높은 수준에서 작동합니다.

- 현지 환경에 Astra Control Center를 설치합니다. 에 대해 자세히 알아보십시오 "Astra Control Center를 설치합니다".
- 다음과 같은 몇 가지 설정 작업을 완료합니다.
  - 라이센스를 설정합니다.
  - 첫 번째 클러스터를 추가합니다.
  - 클러스터를 추가할 때 검색된 스토리지 백엔드를 추가합니다.
  - 앱 백업을 저장할 오브젝트 저장소 버킷을 추가합니다.

에 대해 자세히 알아보십시오 "Astra Control Center를 설정합니다".

클러스터에 애플리케이션을 추가할 수 있습니다. 클러스터에 이미 관리 중인 일부 애플리케이션이 있는 경우 Astra Control Center를 사용하여 관리할 수 있습니다. 그런 다음 Astra Control Center를 사용하여 스냅샷, 백업, 클론 및 복제 관계를 생성합니다.

를 참조하십시오

- "NetApp Astra 제품군 설명서"
- "Astra Control Center 문서"
- "Astra Control API 설명서"
- "Astra Trident 문서"
- "ONTAP 설명서"

지원되는 **Kubernetes** 구축

Astra Control Service는 EKS(Amazon Elastic Kubernetes Service)의 관리되는 Kubernetes 클러스터에서 실행 중인 앱과 사용자가 직접 관리하는 클러스터를 관리할 수 있습니다.

Astra Control Service는 GKE(Google Kubernetes Engine)의 관리되는 Kubernetes 클러스터에서 실행 중인 앱과 사용자가 직접 관리하는 클러스터를 관리할 수 있습니다.

Astra Control Service는 Azure Kubernetes Service(AKS)의 관리형 Kubernetes 클러스터에서 실행 중인 앱과 사용자가 직접 관리하는 클러스터를 관리할 수 있습니다.

- "Astra Control Service용 Amazon Web Services를 설정하는 방법을 알아보십시오".
- "Google Cloud for Astra Control Service를 설정하는 방법을 알아보십시오".
- "Astra Control Service용 Azure NetApp Files를 사용하여 Microsoft Azure를 설정하는 방법에 대해 알아보십시오".
- "Astra Control Service용 Azure 관리 디스크를 사용하여 Microsoft Azure를 설정하는 방법에 대해 알아보십시오".
- "Astra Control Service에 추가하기 전에 자체 관리 클러스터를 준비하는 방법을 알아보십시오".

Astra Control Service를 빠르게 시작합니다

이 페이지에서는 Astra Control Service를 시작하는 데 필요한 단계에 대한 개괄적인 개요를

제공합니다. 각 단계의 링크를 클릭하면 자세한 내용을 제공하는 페이지로 이동합니다.

[1개] 클라우드 공급자를 설정합니다

1. Google 클라우드:
   - Google Kubernetes Engine 클러스터 요구 사항을 검토합니다.
   - Google Cloud Marketplace에서 Cloud Volumes Service for Google Cloud를 구입합니다.
   - 필요한 API를 사용하도록 설정합니다.
   - 서비스 계정 및 서비스 계정 키를 생성합니다.
   - VPC에서 Google Cloud용 Cloud Volumes Service로 네트워크 피어링을 설정합니다.

     "Google Cloud 요구 사항에 대해 자세히 알아보십시오".
2. Amazon 웹 서비스:
   - Amazon Web Services 클러스터 요구사항을 검토합니다.
   - 아마존 계정을 생성합니다.
   - Amazon Web Services CLI를 설치합니다.
   - IAM 사용자를 생성합니다.
   - 사용 권한 정책을 만들고 첨부합니다.
   - IAM 사용자에 대한 자격 증명을 저장합니다.

     "Amazon Web Services 요구 사항에 대해 자세히 알아보십시오".
3. Microsoft Azure:
   - 사용할 스토리지 백엔드에 대한 Azure Kubernetes Service 클러스터 요구 사항을 검토하십시오.

     "Microsoft Azure 및 Azure NetApp Files 요구 사항에 대해 자세히 알아보십시오".

   "Microsoft Azure 및 Azure 관리 디스크 요구 사항에 대해 자세히 알아보십시오".

자체 클러스터를 관리하고 있지만 클라우드 공급자가 호스팅하지 않는 경우, 자체 관리 클러스터에 대한 요구사항을 검토하십시오.
"자가 관리 클러스터 요구사항에 대해 자세히 알아보십시오".

[2개] Astra Control 등록을 완료합니다

1. 을 생성합니다 "NetApp BlueXP" 계정.
2. Astra Control 계정을 생성할 때 NetApp BlueXP 이메일 ID를 지정합니다 "Astra Control 제품 페이지에서 링크 삽입".

"등록 프로세스에 대해 자세히 알아보십시오".

[세 가지] Astra Control에 클러스터를 추가합니다

로그인한 후 * 클러스터 추가 * 를 선택하여 Astra Control을 사용하여 클러스터 관리를 시작합니다.

"클러스터 추가에 대해 자세히 알아보십시오".

클라우드 공급자를 설정합니다

Amazon Web Services를 설정합니다

Astra Control Service로 Amazon Elastic Kubernetes Service(EKS) 클러스터를 관리하려면 Amazon Web Services 프로젝트를 준비하기 위해 몇 가지 단계가 필요합니다.

Amazon Web Services 설정을 빠르게 시작합니다

다음 단계를 따라 빠르게 시작하거나 나머지 섹션으로 스크롤하여 자세한 내용을 확인하십시오.

[1개] Amazon Web Services에 대한 Astra Control 서비스 요구 사항을 검토합니다

클러스터가 정상 상태이며 지원되는 Kubernetes 버전을 실행 중인지, 작업자 노드가 온라인 상태이고 Linux 또는 Windows가 실행 중인지 등을 확인합니다. 이 단계에 대해 자세히 알아보십시오.

[2개] 아마존 계정을 생성합니다

아마존 계정이 없는 경우 EKS를 사용할 수 있도록 계정을 만들어야 합니다. 이 단계에 대해 자세히 알아보십시오.

[세 가지] Amazon Web Services CLI를 설치합니다

명령줄에서 AWS를 관리할 수 있도록 AWS CLI를 설치합니다. 단계별 지침을 따릅니다.

[네] 선택 사항: IAM 사용자를 생성합니다

IAM(Amazon Identity and Access Management) 사용자를 생성합니다. 또한 이 단계를 건너뛰고 Astra Control Service에서 기존 IAM 사용자를 사용할 수도 있습니다.

단계별 지침을 읽습니다.

[다섯] 사용 권한 정책을 만들고 첨부합니다

Astra Control Service가 AWS 계정과 상호 작용하기 위해 필요한 권한이 있는 정책을 생성합니다.

단계별 지침을 읽습니다.

[6개] IAM 사용자에 대한 자격 증명을 저장합니다

Astra Control Service로 자격 증명을 가져올 수 있도록 IAM 사용자의 자격 증명을 저장합니다.

단계별 지침을 읽습니다.

EKS 클러스터 요구 사항

Kubernetes 클러스터는 Astra Control Service에서 검색 및 관리할 수 있도록 다음 요구사항을 충족해야 합니다.

Kubernetes 버전

클러스터는 1.25~1.28 범위의 Kubernetes 버전을 실행해야 합니다.

이미지 유형

각 작업자 노드의 이미지 유형은 Linux여야 합니다.

클러스터 상태입니다

클러스터가 정상 상태에서 실행되고 있어야 하며 오류가 발생한 상태의 작업자 노드가 없는 온라인 작업자 노드가 하나 이상 있어야 합니다.

Astra Control Provisioner

스토리지 백엔드 작업을 수행하려면 Astra Control Provisioner 및 외부 스냅샷 컨트롤러가 필요합니다. 이러한 작업을 활성화하려면 다음을 수행합니다.

1. "스냅샷 CRD 및 스냅샷 컨트롤러를 설치합니다".
2. "Astra Control Provisioner를 활성화합니다".
3. "VolumeSnapshotClass를 생성합니다".

Amazon EBS(Elastic Block Store)용 **CSI** 드라이버

Amazon EBS 스토리지 백엔드를 사용하는 경우 EBS용 컨테이너 스토리지 인터페이스(CSI) 드라이버를 설치해야 합니다(자동으로 설치되지 않음).

CSI 드라이버 설치에 대한 지침은 단계를 참조하십시오.

외부 스냅샷 프로그램을 설치합니다

아직 수행하지 않았다면 "스냅샷 CRD 및 스냅샷 컨트롤러를 설치합니다".

CSI 드라이버를 **Amazon EKS** 애드온으로 설치합니다

1. 서비스 계정에 대한 Amazon EBS CSI 드라이버 IAM 역할을 생성합니다. 지침을 따릅니다 "아마존 문서 "의 지침에 따라 AWS CLI 명령을 사용합니다.
2. 다음 AWS CLI 명령을 사용하여 Amazon EBS CSI 애드온을 추가합니다. 괄호 <>의 정보를 사용자 환경에 맞는 값으로 바꿉니다. driver_role>을 이전 단계에서 생성한 EBS CSI 드라이버 역할의 이름으로 바꿉니다.

```
aws eks create-addon \
  --cluster-name <CLUSTER_NAME> \
  --addon-name aws-ebs-csi-driver \
  --service-account-role-arn
arn:aws:iam::<ACCOUNT_ID>:role/<DRIVER_ROLE>
```

EBS 스토리지 클래스를 구성합니다

1. Amazon EBS CSI 드라이버 GitHub 리포지토리를 시스템에 복제합니다.

```
git clone https://github.com/kubernetes-sigs/aws-ebs-csi-
driver.git
```

2. 동적 프로비저닝 예제 디렉토리로 이동합니다.

```
cd aws-ebs-csi-driver/examples/kubernetes/dynamic-provisioning/
```

3. EBS-SC 스토리지 클래스 및 EBS-Claim 영구 볼륨 클레임을 매니페스트 디렉토리에서 배포합니다.

```
kubectl apply -f manifests/storageclass.yaml
kubectl apply -f manifests/claim.yaml
```

4. EBS-SC 스토리지 클래스를 설명합니다.

```
kubectl describe storageclass ebs-sc
```

스토리지 클래스 속성을 설명하는 출력이 표시됩니다.

아마존 계정을 생성합니다

아마존 계정이 없는 경우 아마존 EKS에 대한 청구를 활성화하려면 계정을 생성해야 합니다.

단계

1. 로 이동합니다 "아마존 홈페이지" 오른쪽 상단에서 * 로그인 * 을 선택하고 * 여기서 시작 * 을 선택합니다.
2. 표시되는 메시지에 따라 계정을 만듭니다.

Amazon Web Services CLI를 설치합니다

명령줄에서 AWS 리소스를 관리할 수 있도록 AWS CLI를 설치합니다.

단계

1. 로 이동합니다 "AWS CLI 시작하기" 지침에 따라 CLI를 설치합니다.

선택 사항**: IAM** 사용자를 생성합니다

IAM 사용자를 생성하여 보안을 강화하고 AWS 서비스 및 리소스를 사용 및 관리할 수 있습니다. 이 단계를 건너뛰고 Astra Control Service에서 기존 IAM 사용자를 사용할 수도 있습니다.

단계

1. 로 이동합니다 "IAM 사용자 생성" 지침에 따라 IAM 사용자를 생성합니다.

사용 권한 정책을 만들고 첨부합니다

Astra Control Service가 AWS 계정과 상호 작용하기 위해 필요한 권한이 있는 정책을 생성합니다.

단계

1. policy.json이라는 새 파일을 만듭니다.
2. 다음 JSON 콘텐츠를 파일에 복사합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetMetricData",
                "fsx:DescribeVolumes",
                "ec2:DescribeRegions",
                "s3:CreateBucket",
                "s3:ListBucket",
                "s3:PutObject",
                "s3:GetObject",
                "iam:SimulatePrincipalPolicy",
                "s3:ListAllMyBuckets",
                "eks:DescribeCluster",
                "eks:ListNodegroups",
                "eks:DescribeNodegroup",
                "eks:ListClusters",
                "iam:GetUser",
                "s3:DeleteObject",
                "s3:DeleteBucket",
                "autoscaling:DescribeAutoScalingGroups"
            ],
            "Resource": "*"
        }
    ]
}
```

3. 정책을 생성합니다.

```
POLICY_ARN=$(aws iam create-policy  --policy-name <policy-name> --policy
-document file://policy.json  --query='Policy.Arn' --output=text)
```

4. 정책을 IAM 사용자에게 연결합니다. '<IAM-user-name>'을(를) 생성한 IAM 사용자의 사용자 이름 또는 기존 IAM 사용자로 대체합니다.

```
aws iam attach-user-policy --user-name <IAM-USER-NAME> --policy-arn
=$POLICY_ARN
```

IAM 사용자에 대한 자격 증명을 저장합니다

Astra Control Service가 사용자를 인식할 수 있도록 IAM 사용자의 자격 증명을 저장합니다.

단계

1. 자격 증명을 다운로드합니다. '<IAM-user-name>'을(를) 사용하려는 IAM 사용자의 사용자 이름으로 바꿉니다.

```
aws iam create-access-key --user-name <IAM-USER-NAME> --output json >
credential.json
```

결과

자격 증명.json 파일이 생성되어 Astra Control Service로 자격 증명을 가져올 수 있습니다.

Google Cloud를 설정합니다

Astra Control Service로 Google Kubernetes Engine 클러스터를 관리하려면 Google Cloud 프로젝트를 준비하기 위해 몇 가지 단계가 필요합니다.

> Google Cloud용 Google Cloud Volumes Service를 스토리지 백엔드로 사용하지 않고 나중에 사용할 계획이라면 Google Cloud용 Google Cloud Volumes Service를 지금 구성하는 데 필요한 단계를 완료해야 합니다. 나중에 서비스 계정을 생성하면 기존 스토리지 버킷에 대한 액세스가 손실될 수 있습니다.

Google Cloud 설정을 빠르게 시작합니다

다음 단계를 따라 빠르게 시작하거나 나머지 섹션으로 스크롤하여 자세한 내용을 확인하십시오.

[1개] Google Kubernetes Engine에 대한 Astra Control Service 요구 사항을 검토합니다

클러스터가 정상 상태이며 지원되는 Kubernetes 버전을 실행 중인지, 작업자 노드가 온라인 상태이고 지원되는 이미지 유형 등을 실행 중인지 확인합니다. 이 단계에 대해 자세히 알아보십시오.

[2개] (선택 사항): Google Cloud용 Cloud Volumes Service를 구입합니다

Cloud Volumes Service for Google Cloud를 스토리지 백엔드로 사용하려면 Google Cloud 마켓플레이스의 NetApp Cloud Volumes Service 페이지로 이동하여 구매 를 선택합니다. 이 단계에 대해 자세히 알아보십시오.

[세 가지] Google Cloud 프로젝트에서 API를 활성화합니다

다음 Google Cloud API를 활성화합니다.

- Google Kubernetes 엔진
- 클라우드 스토리지
- 클라우드 스토리지 JSON API
- 서비스 사용
- Cloud Resource Manager API를 참조하십시오

- NetApp Cloud Volumes Service를 참조하십시오
  - Cloud Volumes Service for Google Cloud에 필요합니다
  - Google 영구 디스크의 경우 선택 사항(권장)입니다
- 서비스 소비자 관리 API
- 서비스 네트워킹 API
- 서비스 관리 API

단계별 지침을 따릅니다.

[네] 필요한 권한이 있는 서비스 계정을 만듭니다

다음 권한이 있는 Google Cloud 서비스 계정을 만듭니다.

- Kubernetes 엔진 관리자
- NetApp Cloud Volumes 관리자
  - Cloud Volumes Service for Google Cloud에 필요합니다
  - Google 영구 디스크의 경우 선택 사항(권장)입니다
- 스토리지 관리자
- 서비스 사용 뷰어
- 네트워크 뷰어 계산

단계별 지침을 읽습니다.

[다섯] 서비스 계정 키를 생성합니다

서비스 계정의 키를 생성하고 키 파일을 안전한 위치에 저장합니다. 단계별 지침을 따릅니다.

[6개] (선택 사항): VPC용 네트워크 피어링을 설정합니다

Cloud Volumes Service for Google Cloud를 스토리지 백엔드로 사용하려는 경우 VPC에서 Cloud Volumes Service for Google Cloud로 네트워크 피어링을 설정합니다. 단계별 지침을 따릅니다.

GKE 클러스터 요구 사항

Kubernetes 클러스터는 Astra Control Service에서 검색 및 관리할 수 있도록 다음 요구사항을 충족해야 합니다. 이러한 요구사항 중 일부는 Cloud Volumes Service for Google Cloud를 스토리지 백엔드로 사용하려는 경우에만 적용됩니다.

Kubernetes 버전
: 클러스터는 1.26~1.28 범위의 Kubernetes 버전을 실행해야 합니다.

이미지 유형
: 각 작업자 노드의 이미지 유형은 이어야 합니다 `COS_CONTAINERD`.

클러스터 상태입니다

클러스터가 정상 상태에서 실행되고 있어야 하며 오류가 발생한 상태의 작업자 노드가 없는 온라인 작업자 노드가 하나 이상 있어야 합니다.

Google Cloud 지역

Google Cloud용 Cloud Volumes Service를 스토리지 백엔드로 사용하려는 경우 클러스터가 에서 실행 중이어야 합니다 "Google Cloud용 Cloud Volumes Service가 지원되는 Google Cloud 지역" Astra Control Service는 CVS 및 CVS 성능 두 가지 서비스 유형을 모두 지원합니다. 모범 사례로서, 스토리지 백엔드로 사용하지 않더라도 Cloud Volumes Service for Google Cloud를 지원하는 영역을 선택해야 합니다. 따라서 성능 요구사항이 변경될 경우 향후에 Google Cloud용 Cloud Volumes Service를 스토리지 백엔드로 손쉽게 사용할 수 있습니다.

네트워킹

Google Cloud용 Cloud Volumes Service를 스토리지 백엔드로 사용하려는 경우, 클러스터가 Cloud Volumes Service for Google Cloud에서 내다보는 VPC에 상주해야 합니다. 이 단계는 아래에 설명되어 있습니다.

프라이빗 클러스터

클러스터가 프라이빗 인 경우, 를 참조하십시오 "인증된 네트워크" Astra Control Service IP 주소를 허용해야 합니다.

52.188.218.166/32

GKE 클러스터의 작동 모드입니다

표준 작동 모드를 사용해야 합니다. 현재 Autopilot 모드가 테스트되지 않았습니다. "작동 모드에 대해 자세히 알아보십시오".

지원합니다

CVS 서비스 유형을 사용하여 NetApp Cloud Volumes Service를 스토리지 백엔드로 사용하는 경우 볼륨을 프로비저닝하기 전에 스토리지 풀을 구성해야 합니다. 을 참조하십시오 "서비스 유형, 스토리지 클래스 및 GKE 클러스터의 PV 크기입니다" 를 참조하십시오.

선택 사항**: Google Cloud**용 **Cloud Volumes Service**를 구입합니다

Astra Control Service는 Cloud Volumes Service for Google Cloud를 영구 볼륨의 스토리지 백엔드로 사용할 수 있습니다. 이 서비스를 사용하려면 영구 볼륨에 대한 청구를 활성화하려면 Google Cloud Marketplace에서 Cloud Volumes Service for Google Cloud를 구입해야 합니다.

단계

1. 로 이동합니다 "NetApp Cloud Volumes Service 페이지를 참조하십시오" Google Cloud Marketplace에서 * 구매 * 를 선택하고 화면의 지시를 따릅니다.

   "Google Cloud 설명서의 단계별 지침에 따라 서비스를 구매하고 활성화합니다".

프로젝트에서 **API**를 사용하도록 설정합니다

프로젝트에 특정 Google Cloud API에 액세스할 수 있는 권한이 필요합니다. API는 GKE(Google Kubernetes Engine) 클러스터 및 NetApp Cloud Volumes Service 스토리지와 같은 Google Cloud 리소스와 상호 작용하는 데 사용됩니다.

단계

1. "Google Cloud 콘솔 또는 gcloud CLI를 사용하여 다음 API를 활성화합니다":

- Google Kubernetes 엔진
- 클라우드 스토리지
- 클라우드 스토리지 JSON API
- 서비스 사용
- Cloud Resource Manager API를 참조하십시오
- NetApp Cloud Volumes Service(Google Cloud용 Cloud Volumes Service에 필요)
- 서비스 소비자 관리 API
- 서비스 네트워킹 API
- 서비스 관리 API

다음 비디오에서는 Google Cloud 콘솔에서 API를 활성화하는 방법을 보여줍니다.

► https://docs.netapp.com/ko-kr/astra-control-service/media/get-started/video-enable-gcp-apis.mp4 *(video)*

서비스 계정을 생성합니다

Astra Control Service는 Google Cloud 서비스 계정을 사용하여 Kubernetes 애플리케이션 데이터를 사용자 대신 관리합니다.

단계

1. Google Cloud로 이동하고 "콘솔, gcloud 명령 또는 다른 기본 설정 방법을 사용하여 서비스 계정을 만듭니다".
2. 서비스 계정에 다음 역할을 부여합니다.
   - * Kubernetes Engine Admin * - 클러스터를 나열하고 앱 관리를 위한 관리자 액세스를 생성하는 데 사용됩니다.
   - * NetApp Cloud Volumes Admin * - 앱의 영구 스토리지를 관리하는 데 사용됩니다.
   - * 스토리지 관리자 * - 애플리케이션 백업을 위한 버킷 및 객체를 관리하는 데 사용됩니다.
   - * 서비스 사용 뷰어 * - 필요한 Cloud Volumes Service for Google Cloud API가 활성화되어 있는지 확인하는 데 사용됩니다.
   - * 컴퓨팅 네트워크 뷰어 * - Kubernetes VPC가 Google Cloud용 Cloud Volumes Service에 연결할 수 있는지 확인하는 데 사용됩니다.

gcloud를 사용하려면 Astra Control 인터페이스 내의 단계를 따르십시오. 계정 > 자격 증명 > 자격 증명 추가 * 를 선택한 다음 * 지침 * 을 선택합니다.

Google Cloud 콘솔을 사용하려는 경우 다음 비디오에서 콘솔에서 서비스 계정을 만드는 방법을 확인할 수 있습니다.

► https://docs.netapp.com/ko-kr/astra-control-service/media/get-started/video-create-gcp-service-account.mp4

(video)

공유 **VPC**에 대한 서비스 계정을 구성합니다

하나의 프로젝트에 상주하지만 다른 프로젝트(공유 VPC)의 VPC를 사용하는 GKE 클러스터를 관리하려면 * Compute Network Viewer * 역할이 있는 호스트 프로젝트의 구성원으로 Astra 서비스 계정을 지정해야 합니다.

단계

1. Google Cloud 콘솔에서 * IAM & Admin * 으로 이동하여 * Service Accounts * 를 선택합니다.
2. 이(가) 있는 Astra 서비스 계정을 찾습니다 "필요한 권한" 그런 다음 전자 메일 주소를 복사합니다.
3. 호스트 프로젝트로 이동한 다음 * IAM & Admin * > * IAM * 을 선택합니다.
4. 추가 * 를 선택하고 서비스 계정에 대한 항목을 추가합니다.
    a. * 새 회원 *: 서비스 계정의 이메일 주소를 입력합니다.
    b. * 역할 *: * Compute Network Viewer * 를 선택합니다.
    c. 저장 * 을 선택합니다.

결과

공유 VPC를 사용하여 GKE 클러스터를 추가하면 Astra와 완전히 연동됩니다.

서비스 계정 키를 생성합니다

Astra Control Service에 사용자 이름과 암호를 제공하는 대신 첫 번째 클러스터를 추가할 때 서비스 계정 키를 제공합니다. Astra Control Service는 서비스 계정 키를 사용하여 방금 설정한 서비스 계정의 ID를 설정합니다.

서비스 계정 키는 JSON(JavaScript Object Notation) 형식으로 저장된 일반 텍스트입니다. 액세스 권한이 있는 GCP 리소스에 대한 정보가 포함되어 있습니다.

키를 생성할 때만 JSON 파일을 보거나 다운로드할 수 있습니다. 그러나 언제든지 새 키를 만들 수 있습니다.

단계

1. Google Cloud로 이동하고 "콘솔, gcloud 명령 또는 다른 기본 설정 방법을 사용하여 서비스 계정 키를 생성합니다".
2. 메시지가 표시되면 서비스 계정 키 파일을 안전한 위치에 저장합니다.

다음 비디오에서는 Google Cloud 콘솔에서 서비스 계정 키를 생성하는 방법을 보여줍니다.

► https://docs.netapp.com/ko-kr/astra-control-service/media/get-started/video-create-gcp-service-account-

key.mp4 *(video)*

선택 사항**: VPC**용 네트워크 피어링을 설정합니다

Google Cloud용 Cloud Volumes Service를 스토리지 백엔드 서비스로 사용하려는 경우 마지막 단계는 VPC에서 Cloud Volumes Service for Google Cloud로 네트워킹 피어링을 설정하는 것입니다.

네트워크 피어링을 설정하는 가장 쉬운 방법은 Cloud Volumes Service에서 gcloud 명령을 직접 가져오는 것입니다. 새 파일 시스템을 생성할 때 Cloud Volumes Service에서 명령을 사용할 수 있습니다.

단계

1. "NetApp BlueXP 글로벌 지역 맵 으로 이동합니다" 클러스터가 있는 Google Cloud 영역에서 사용할 서비스 유형을 식별하십시오.

   Cloud Volumes Service는 CVS와 CVS - 성능이라는 두 가지 서비스 유형을 제공합니다. "이러한 서비스 유형에 대해 자세히 알아보십시오".

2. "Google Cloud Platform에서 Cloud Volumes로 이동합니다".
3. 볼륨 * 페이지에서 * 생성 * 을 선택합니다.
4. 서비스 유형 * 에서 * CVS * 또는 * CVS - 성능 * 을 선택합니다.

   Google Cloud 지역에 맞는 서비스 유형을 선택해야 합니다. 1단계에서 확인한 서비스 유형입니다. 서비스 유형을 선택하면 페이지의 영역 목록이 해당 서비스 유형이 지원되는 지역으로 업데이트됩니다.

   이 단계를 수행한 후에는 네트워킹 정보만 입력하면 명령을 얻을 수 있습니다.

5. 지역 * 에서 지역 및 구역을 선택합니다.
6. Network Details * 에서 VPC를 선택합니다.

   네트워크 피어링을 설정하지 않은 경우 다음 알림이 표시됩니다.

7. 네트워크 피어링 설정 명령을 보려면 버튼을 선택합니다.
8. 명령을 복사하여 Cloud Shell에서 실행합니다.

   이러한 명령 사용에 대한 자세한 내용은 를 참조하십시오 "Cloud Volumes Service for GCP용 QuickStart".

"개인 서비스 액세스 구성 및 네트워크 피어링 설정에 대해 자세히 알아보십시오".

9. 완료되면 * 파일 시스템 생성 * 페이지에서 취소를 선택할 수 있습니다.

   이 볼륨은 네트워크 피어링을 위한 명령만 얻기 위해 만들어지기 시작했습니다.

Azure NetApp Files를 사용하여 Microsoft Azure를 설정합니다

Astra Control Service로 Azure Kubernetes Service 클러스터를 관리하려면 Microsoft Azure 구독을 준비하기 위해 몇 가지 단계가 필요합니다. Azure NetApp Files를 스토리지 백엔드로 사용하려는 경우 다음 지침을 따르십시오.

Azure 설정을 위한 빠른 시작

다음 단계를 따라 빠르게 시작하거나 나머지 섹션으로 스크롤하여 자세한 내용을 확인하십시오.

[1개] Azure Kubernetes Service에 대한 Astra Control Service 요구 사항을 검토합니다

클러스터가 정상 상태이며 지원되는 Kubernetes 버전을 실행 중인지, 노드 풀이 온라인 상태이고 Linux가 실행 중인지 등을 확인합니다. 이 단계에 대해 자세히 알아보십시오.

[2개] Microsoft Azure에 등록하십시오

Microsoft Azure 계정을 만듭니다. 이 단계에 대해 자세히 알아보십시오.

[세 가지] Azure NetApp Files에 등록하십시오

NetApp 리소스 공급자를 등록합니다. 이 단계에 대해 자세히 알아보십시오.

[네] NetApp 계정을 만듭니다

Azure 포털에서 Azure NetApp Files로 이동하여 NetApp 계정을 생성합니다. 이 단계에 대해 자세히 알아보십시오.

[다섯] 용량 풀 설정

영구 볼륨에 대해 하나 이상의 용량 풀을 설정합니다. 이 단계에 대해 자세히 알아보십시오.

[6개] Azure NetApp Files에 서브넷 위임

Astra Control Service가 해당 서브넷에 영구 볼륨을 생성할 수 있도록 Azure NetApp Files에 서브넷을 위임합니다. 이 단계에 대해 자세히 알아보십시오.

[7번] Azure 서비스 보안 주체 만들기

Contributor 역할이 있는 Azure 서비스 보안 주체를 만듭니다. 이 단계에 대해 자세히 알아보십시오.

[8개] 선택 사항: Azure 백업 버킷의 중복성을 구성합니다

기본적으로 Astra Control Service 버킷은 Azure Kubernetes Service 백업을 저장하는 데 사용되며 LRS(Locally Redundant Storage) 이중화 옵션을 사용합니다. 선택적 단계로서 Azure 버킷을 위한 더욱 내구성 있는 이중화 수준을 구성할 수 있습니다. 이 단계에 대해 자세히 알아보십시오.

Azure Kubernetes Service 클러스터 요구사항

Kubernetes 클러스터는 Astra Control Service에서 검색 및 관리할 수 있도록 다음 요구사항을 충족해야 합니다.

Kubernetes 버전

클러스터에서 Kubernetes 버전 1.26~1.28을 실행해야 합니다.

이미지 유형

모든 노드 풀의 이미지 유형은 Linux여야 합니다.

클러스터 상태입니다

클러스터가 정상 상태에서 실행되고 있어야 하며 오류가 발생한 상태의 작업자 노드가 없는 온라인 작업자 노드가 하나 이상 있어야 합니다.

Azure 지역

클러스터는 Azure NetApp Files를 사용할 수 있는 지역에 상주해야 합니다. "지역별 Azure 제품 보기".

구독

클러스터는 Azure NetApp Files가 설정된 구독에 상주해야 합니다. 구독은 필요할 때 선택할 수 있습니다 Azure NetApp Files에 등록하십시오.

VNET

다음 VNET 요구 사항을 고려하십시오.

- 클러스터는 Azure NetApp Files 위임 서브넷에 직접 액세스할 수 있는 VNET에 상주해야 합니다. 위임된 서브넷을 설정하는 방법에 대해 알아봅니다.
- Kubernetes 클러스터가 다른 VNET에 있는 Azure NetApp Files 위임 서브넷을 통해 피어링된 VNET에 있는 경우 피어링 연결의 양측이 온라인 상태여야 합니다.
- Azure NetApp Files가 있는 VNET(즉시 피어링된 VNets 포함)에서 사용되는 IP 수의 기본 제한은 1,000입니다. "Azure NetApp Files 리소스 제한을 봅니다".

  제한에 근접하면 다음 두 가지 옵션을 사용할 수 있습니다.

  - 가능합니다 "한도 증수에 대한 요청을 제출합니다". 도움이 필요하면 NetApp 담당자에게 문의하십시오.
  - 새 AKS(Amazon Kubernetes Service) 클러스터를 생성할 때 클러스터에 대한 새 네트워크를 지정합니다. 새 네트워크가 생성되면 새 서브넷을 프로비저닝하고 Azure NetApp Files에 서브넷을 위임합니다.

Microsoft Azure에 등록하십시오

Microsoft Azure 계정이 없는 경우 먼저 Microsoft Azure에 가입합니다.

단계

1. 로 이동합니다 "Azure 구독 페이지입니다" Azure 서비스에 가입하려면
2. 계획을 선택하고 지침에 따라 구독을 완료합니다.

Azure NetApp Files에 등록하십시오

NetApp 리소스 공급자를 등록하여 Azure NetApp Files에 액세스하십시오.

단계

1. Azure 포털에 로그인합니다.
2. "Azure NetApp Files 설명서에 따라 NetApp 리소스 공급자를 등록하십시오".

NetApp 계정을 만듭니다

Azure NetApp Files에서 NetApp 계정을 만듭니다.

단계

1. "Azure NetApp Files 설명서에 따라 Azure 포털에서 NetApp 계정을 만드십시오".

용량 풀을 설정합니다

Astra Control Service가 용량 풀에서 영구 볼륨을 프로비저닝할 수 있도록 하나 이상의 용량 풀이 필요합니다. Astra Control Service는 사용자를 위한 용량 풀을 생성하지 않습니다.

Kubernetes 앱의 용량 풀을 설정할 때는 다음 사항을 고려하십시오.

- AKS 클러스터를 Astra Control Service로 관리할 Azure 지역에서 용량 풀을 생성해야 합니다.
- 용량 풀에는 Ultra, Premium 또는 Standard 서비스 수준이 있을 수 있습니다. 각 서비스 수준은 서로 다른 성능 요구 사항을 충족하도록 설계되었습니다. Astra Control Service는 이 세 가지를 모두 지원합니다.

  Kubernetes 클러스터와 함께 사용할 각 서비스 수준에 대해 용량 풀을 설정해야 합니다.

  "Azure NetApp Files의 서비스 수준에 대해 자세히 알아보십시오".

- Astra Control Service로 보호할 앱의 용량 풀을 생성하기 전에 해당 애플리케이션에 필요한 성능과 용량을 선택하십시오.

  용량을 적절하게 프로비저닝하면 사용자가 필요에 따라 영구 볼륨을 생성할 수 있습니다. 용량을 사용할 수 없는 경우 영구 볼륨을 프로비저닝할 수 없습니다.

- Azure NetApp Files 용량 풀은 수동 또는 자동 QoS 유형을 사용할 수 있습니다. Astra Control Service는 자동 QoS 용량 풀을 지원합니다. 수동 QoS 용량 풀은 지원되지 않습니다.

단계

1. "Azure NetApp Files 설명서에 따라 자동 QoS 용량 풀을 설정합니다".

Azure NetApp Files에 서브넷 위임

Astra Control Service가 해당 서브넷에 영구 볼륨을 생성할 수 있도록 Azure NetApp Files에 서브넷을 위임해야 합니다. Azure NetApp Files를 사용하면 VNET에 하나의 위임된 서브넷만 가질 수 있습니다.

피어링된 VNets를 사용하는 경우 피어링 연결의 양쪽이 모두 온라인 상태여야 합니다. 즉, Kubernetes 클러스터가 있는 VNET와 Azure NetApp Files에서 위임한 서브넷이 있는 VNET입니다.

단계

1. "Azure NetApp Files 설명서에 따라 Azure NetApp Files에 서브넷을 위임합니다".

모두 끝냈군요

위임된 서브넷에서 실행 중인 클러스터를 검색하기 전에 약 10분 정도 기다립니다.

Azure 서비스 보안 주체 만들기

Astra Control Service에는 Contributor 역할이 할당된 Azure 서비스 보안 주체가 필요합니다. Astra Control Service는 이 서비스 보안 주체를 사용하여 Kubernetes 애플리케이션 데이터를 사용자 대신 관리합니다.

서비스 보안 주체는 응용 프로그램, 서비스 및 도구와 함께 사용하기 위해 특별히 만들어진 ID입니다. 서비스 보안 주체에 역할을 할당하면 특정 Azure 리소스에 대한 액세스가 제한됩니다.

Azure CLI를 사용하여 서비스 보안 주체를 만들려면 다음 단계를 수행하십시오. 출력 내용을 JSON 파일에 저장하고 나중에 Astra Control Service에 제공해야 합니다. "CLI 사용에 대한 자세한 내용은 Azure 설명서를 참조하십시오".

다음 단계에서는 서비스 보안 주체를 만들 수 있는 권한이 있고 Microsoft Azure SDK(az 명령)가 컴퓨터에 설치되어 있다고 가정합니다.

요구 사항

- 서비스 보안 주체는 일반 인증을 사용해야 합니다. 인증서가 지원되지 않습니다.
- 서비스 보안 주체는 Azure 구독에 대한 Contributor 또는 Owner 액세스 권한을 부여해야 합니다.
- 범위에 대해 선택하는 구독 또는 리소스 그룹에는 AKS 클러스터와 Azure NetApp Files 계정이 포함되어야 합니다.

단계

1. AKS 클러스터가 있는 가입 및 테넌트 ID(Astra Control Service에서 관리하려는 클러스터)를 식별합니다.

```
az configure --list-defaults
az account list --output table
```

2. 전체 구독 또는 리소스 그룹을 사용하는 경우에 따라 다음 중 하나를 실행합니다.
   - 서비스 보안 주체를 만들고 Contributor 역할을 할당하고 클러스터가 상주하는 전체 구독에 범위를 지정합니다.

```
az ad sp create-for-rbac --name service-principal-name --role
contributor --scopes /subscriptions/SUBSCRIPTION-ID
```

   - 서비스 보안 주체를 만들고 Contributor 역할을 할당하고 클러스터가 있는 리소스 그룹을 지정합니다.

```
az ad sp create-for-rbac --name service-principal-name --role
contributor --scopes /subscriptions/SUBSCRIPTION-
ID/resourceGroups/RESOURCE-GROUP-ID
```

3. 생성된 Azure CLI 출력을 JSON 파일로 저장합니다.

   Astra Control Service가 AKS 클러스터를 검색하고 Kubernetes 데이터 관리 작업을 관리할 수 있도록 이 파일을 제공해야 합니다. "Astra Control Service에서 자격 증명 관리에 대해 자세히 알아보십시오".

4. 선택 사항: JSON 파일에 가입 ID를 추가하면 파일을 선택할 때 Astra Control Service가 자동으로 ID를 채웁니다.

그렇지 않으면 메시지가 표시되면 Astra Control Service에 구독 ID를 입력해야 합니다.

◦ 예 *

```
{
  "appId": "0db3929a-bfb0-4c93-baee-aaf8",
  "displayName": "sp-example-dev-sandbox",
  "name": "http://sp-example-dev-sandbox",
  "password": "mypassword",
  "tenant": "011cdf6c-7512-4805-aaf8-7721afd8ca37",
  "subscriptionId": "99ce999a-8c99-99d9-a9d9-99cce99f99ad"
}
```

5. 선택 사항: 서비스 보안 주체를 테스트합니다. 서비스 보안 주체가 사용하는 범위에 따라 다음 예제 명령 중에서 선택합니다.

구독 범위

```
az login --service-principal --username APP-ID-SERVICEPRINCIPAL
--password PASSWORD --tenant TENANT-ID
az group list --subscription SUBSCRIPTION-ID
az aks list --subscription SUBSCRIPTION-ID
az storage container list --account-name STORAGE-ACCOUNT-NAME
```

리소스 그룹 범위

```
az login --service-principal --username APP-ID-SERVICEPRINCIPAL
--password PASSWORD --tenant TENANT-ID
az aks list --subscription SUBSCRIPTION-ID --resource-group RESOURCE-
GROUP-ID
```

선택 사항**: Azure** 백업 버킷의 중복성을 구성합니다

Azure 백업 버킷에 대해 보다 내구성이 뛰어난 이중화 수준을 구성할 수 있습니다. 기본적으로 Astra Control Service 버킷은 Azure Kubernetes Service 백업을 저장하는 데 사용되며 LRS(Locally Redundant Storage) 이중화 옵션을 사용합니다. Azure 버킷에 보다 내구성이 뛰어난 이중화 옵션을 사용하려면 다음을 수행해야 합니다.

단계

1. 필요한 중복 수준을 사용하는 Azure 저장소 계정을 만듭니다 "참조하십시오".
2. 를 사용하여 새 저장소 계정에 Azure 컨테이너를 생성합니다 "참조하십시오".
3. 컨테이너를 Astra Control Service에 버킷으로 추가합니다. 을 참조하십시오 "추가 버킷을 추가합니다".
4. (선택 사항) 새로 생성한 버킷을 Azure 백업의 기본 버킷으로 사용하려면 이 버킷을 Azure의 기본 버킷으로 설정합니다. 을 참조하십시오 "기본 버킷을 변경합니다".

Azure 관리 디스크를 사용하여 Microsoft Azure를 설정합니다

Astra Control Service로 Azure Kubernetes Service 클러스터를 관리하려면 Microsoft Azure 구독을 준비하기 위해 몇 가지 단계가 필요합니다. Azure 관리 디스크를 스토리지 백엔드로 사용하려는 경우 다음 지침을 따르십시오.

Azure 설정을 위한 빠른 시작

다음 단계를 따라 빠르게 시작하거나 나머지 섹션으로 스크롤하여 자세한 내용을 확인하십시오.

[1개] Azure Kubernetes Service에 대한 Astra Control Service 요구 사항을 검토합니다

클러스터가 정상 상태이며 지원되는 Kubernetes 버전을 실행 중인지, 노드 풀이 온라인 상태이고 Linux가 실행 중인지 등을 확인합니다. 이 단계에 대해 자세히 알아보십시오.

[2개] Microsoft Azure에 등록하십시오

Microsoft Azure 계정을 만듭니다. 이 단계에 대해 자세히 알아보십시오.

[세 가지] Azure 서비스 보안 주체 만들기

Contributor 역할이 있는 Azure 서비스 보안 주체를 만듭니다. 이 단계에 대해 자세히 알아보십시오.

[네] 컨테이너 스토리지 인터페이스(CSI) 드라이버 세부 정보를 구성합니다

CSI 드라이버와 함께 작동하도록 Azure 가입 및 클러스터를 구성해야 합니다. 이 단계에 대해 자세히 알아보십시오.

[다섯] 선택 사항: Azure 백업 버킷의 중복성을 구성합니다

기본적으로 Astra Control Service 버킷은 Azure Kubernetes Service 백업을 저장하는 데 사용되며 LRS(Locally Redundant Storage) 이중화 옵션을 사용합니다. 선택적 단계로서 Azure 버킷을 위한 더욱 내구성 있는 이중화 수준을 구성할 수 있습니다. 이 단계에 대해 자세히 알아보십시오.

Azure Kubernetes Service 클러스터 요구사항

Kubernetes 클러스터는 Astra Control Service에서 검색 및 관리할 수 있도록 다음 요구사항을 충족해야 합니다.

Kubernetes 버전
: 클러스터에서 Kubernetes 버전 1.26~1.28을 실행해야 합니다.

이미지 유형
: 모든 노드 풀의 이미지 유형은 Linux여야 합니다.

클러스터 상태입니다
: 클러스터가 정상 상태에서 실행되고 있어야 하며 오류가 발생한 상태의 작업자 노드가 없는 온라인 작업자 노드가 하나 이상 있어야 합니다.

Azure 지역
: 모범 사례로서, 스토리지 백엔드로 사용하지 않더라도 Azure NetApp Files를 지원하는 영역을 선택해야 합니다. 따라서 성능 요구 사항이 변경될 경우 향후에 Azure NetApp Files를 스토리지 백엔드로 사용할 수 있습니다. "지역별 Azure 제품 보기".

CSI 드라이버

클러스터에는 적절한 CSI 드라이버가 설치되어 있어야 합니다.

Microsoft Azure에 등록하십시오

Microsoft Azure 계정이 없는 경우 먼저 Microsoft Azure에 가입합니다.

단계

1. 로 이동합니다 "Azure 구독 페이지입니다" Azure 서비스에 가입하려면
2. 계획을 선택하고 지침에 따라 구독을 완료합니다.

Azure 서비스 보안 주체 만들기

Astra Control Service에는 Contributor 역할이 할당된 Azure 서비스 보안 주체가 필요합니다. Astra Control Service는 이 서비스 보안 주체를 사용하여 Kubernetes 애플리케이션 데이터를 사용자 대신 관리합니다.

서비스 보안 주체는 응용 프로그램, 서비스 및 도구와 함께 사용하기 위해 특별히 만들어진 ID입니다. 서비스 보안 주체에 역할을 할당하면 특정 Azure 리소스에 대한 액세스가 제한됩니다.

Azure CLI를 사용하여 서비스 보안 주체를 만들려면 다음 단계를 수행하십시오. 출력 내용을 JSON 파일에 저장하고 나중에 Astra Control Service에 제공해야 합니다. "CLI 사용에 대한 자세한 내용은 Azure 설명서를 참조하십시오".

다음 단계에서는 서비스 보안 주체를 만들 수 있는 권한이 있고 Microsoft Azure SDK(az 명령)가 컴퓨터에 설치되어 있다고 가정합니다.

요구 사항

- 서비스 보안 주체는 일반 인증을 사용해야 합니다. 인증서가 지원되지 않습니다.
- 서비스 보안 주체는 Azure 구독에 대한 Contributor 또는 Owner 액세스 권한을 부여해야 합니다.
- 범위에 대해 선택하는 구독 또는 리소스 그룹에는 AKS 클러스터와 Azure NetApp Files 계정이 포함되어야 합니다.

단계

1. AKS 클러스터가 있는 가입 및 테넌트 ID(Astra Control Service에서 관리하려는 클러스터)를 식별합니다.

```
az configure --list-defaults
az account list --output table
```

2. 전체 구독 또는 리소스 그룹을 사용하는 경우에 따라 다음 중 하나를 실행합니다.
   - 서비스 보안 주체를 만들고 Contributor 역할을 할당하고 클러스터가 상주하는 전체 구독에 범위를 지정합니다.

```
az ad sp create-for-rbac --name service-principal-name --role
contributor --scopes /subscriptions/SUBSCRIPTION-ID
```

   - 서비스 보안 주체를 만들고 Contributor 역할을 할당하고 클러스터가 있는 리소스 그룹을 지정합니다.

```
az ad sp create-for-rbac --name service-principal-name --role
contributor --scopes /subscriptions/SUBSCRIPTION-
ID/resourceGroups/RESOURCE-GROUP-ID
```

3. 생성된 Azure CLI 출력을 JSON 파일로 저장합니다.

   Astra Control Service가 AKS 클러스터를 검색하고 Kubernetes 데이터 관리 작업을 관리할 수 있도록 이 파일을 제공해야 합니다. "Astra Control Service에서 자격 증명 관리에 대해 자세히 알아보십시오".

4. 선택 사항: JSON 파일에 가입 ID를 추가하면 파일을 선택할 때 Astra Control Service가 자동으로 ID를 채웁니다.

   그렇지 않으면 메시지가 표시되면 Astra Control Service에 구독 ID를 입력해야 합니다.

   ◦ 예 *

```
{
  "appId": "0db3929a-bfb0-4c93-baee-aaf8",
  "displayName": "sp-example-dev-sandbox",
  "name": "http://sp-example-dev-sandbox",
  "password": "mypassword",
  "tenant": "011cdf6c-7512-4805-aaf8-7721afd8ca37",
  "subscriptionId": "99ce999a-8c99-99d9-a9d9-99cce99f99ad"
}
```

5. 선택 사항: 서비스 보안 주체를 테스트합니다. 서비스 보안 주체가 사용하는 범위에 따라 다음 예제 명령 중에서 선택합니다.

   구독 범위

```
az login --service-principal --username APP-ID-SERVICEPRINCIPAL
--password PASSWORD --tenant TENANT-ID
az group list --subscription SUBSCRIPTION-ID
az aks list --subscription SUBSCRIPTION-ID
az storage container list --account-name STORAGE-ACCOUNT-NAME
```

   리소스 그룹 범위

```
az login --service-principal --username APP-ID-SERVICEPRINCIPAL
--password PASSWORD --tenant TENANT-ID
az aks list --subscription SUBSCRIPTION-ID --resource-group RESOURCE-
GROUP-ID
```

컨테이너 스토리지 인터페이스(CSI) 드라이버 세부 정보를 구성합니다

Azure 관리 디스크를 Astra Control Service와 함께 사용하려면 필요한 CSI 드라이버를 설치해야 합니다.

Azure 구독에서 CSI 드라이버 기능을 활성화합니다

CSI 드라이버를 설치하기 전에 Azure 구독에서 CSI 드라이버 기능을 활성화해야 합니다.

단계

1. Azure 명령줄 인터페이스를 엽니다.
2. 다음 명령을 실행하여 드라이버를 등록합니다.

```
az feature register --namespace "Microsoft.ContainerService" --name
"EnableAzureDiskFileCSIDriver"
```

3. 다음 명령을 실행하여 변경 내용이 전파되었는지 확인합니다.

```
az provider register -n Microsoft.ContainerService
```

다음과 유사한 출력이 표시됩니다.

```
{
"id": "/subscriptions/b200155f-001a-43be-87be-
3edde83acef4/providers/Microsoft.Features/providers/Microsoft.ContainerSer
vice/features/EnableAzureDiskFileCSIDriver",
"name": "Microsoft.ContainerService/EnableAzureDiskFileCSIDriver",
"properties": {
   "state": "Registering"
},
"type": "Microsoft.Features/providers/features"
}
```

Azure Kubernetes Service 클러스터에 Azure Managed Disk CSI 드라이버를 설치합니다

Azure CSI 드라이버를 설치하여 준비를 완료할 수 있습니다.

단계

1. 로 이동합니다 "Microsoft CSI 드라이버 문서".
2. 지침에 따라 필요한 CSI 드라이버를 설치합니다.

선택 사항: Azure 백업 버킷의 중복성을 구성합니다

Azure 백업 버킷에 대해 보다 내구성이 뛰어난 이중화 수준을 구성할 수 있습니다. 기본적으로 Astra Control Service 버킷은 Azure Kubernetes Service 백업을 저장하는 데 사용되며 LRS(Locally Redundant Storage) 이중화 옵션을 사용합니다. Azure 버킷에 보다 내구성이 뛰어난 이중화 옵션을 사용하려면 다음을 수행해야 합니다.

단계

1. 필요한 중복 수준을 사용하는 Azure 저장소 계정을 만듭니다 "참조하십시오".

2. 를 사용하여 새 저장소 계정에 Azure 컨테이너를 생성합니다 "참조하십시오".
3. 컨테이너를 Astra Control Service에 버킷으로 추가합니다. 을 참조하십시오 "추가 버킷을 추가합니다".
4. (선택 사항) 새로 생성한 버킷을 Azure 백업의 기본 버킷으로 사용하려면 이 버킷을 Azure의 기본 버킷으로 설정합니다. 을 참조하십시오 "기본 버킷을 변경합니다".

Astra Control Service 계정에 등록하십시오

Astra Control Service를 사용하려면 NetApp BlueXP 계정과 연결된 Astra Control Service 계정이 필요합니다. Astra Control Service 등록 프로세스를 완료한 다음, BlueXP 계정이 없는 경우 BlueXP에 등록하여 Astra Control Service에 액세스합니다.

Astra Control 계정에 등록하십시오

Astra Control Service에 로그인하기 전에 등록 프로세스를 완료하여 Astra Control Service 계정을 얻어야 합니다.

Astra Control Service를 사용하면 계정 내에서 앱을 관리할 수 있습니다. 계정에는 계정 내에서 앱을 보고 관리할 수 있는 사용자와 청구 세부 정보가 포함됩니다.

단계

1. "BlueXP에서 Astra Control 페이지로 이동합니다".
2. 무료 요금제에 가입하기 * 를 선택합니다.
3. 양식에 필요한 정보를 입력합니다.

   양식을 작성할 때 주의해야 할 몇 가지 중요한 사항은 다음과 같습니다.

   - 글로벌 무역 규정 준수 요구 사항을 충족하는지 확인하기 때문에 귀하의 상호와 주소는 정확해야 합니다.
   - Astra 계정 이름 * 은 귀사의 Astra Control Service 계정 이름입니다. 이 이름은 Astra Control Service 사용자 인터페이스에서 확인할 수 있습니다. 필요한 경우 추가 계정(최대 5개)을 만들 수 있습니다.
   - NetApp BlueXP 계정이 있는 경우 * 회사 이메일 주소 * 필드에 해당 계정에 사용하는 이메일을 여기에 입력하십시오. NetApp BlueXP 계정이 아직 없는 경우 BlueXP에 등록할 때 여기에 입력한 이메일 주소를 사용하십시오.
4. 계정 만들기 * 를 선택합니다.

BlueXP에 등록하십시오

Astra Control Service는 NetApp BlueXP의 인증 서비스에 통합되어 있습니다. BlueXP 또는 NetApp Support 사이트 자격 증명을 사용하여 NetApp BlueXP에 로그인할 수 있습니다. NetApp BlueXP 또는 NetApp Support 사이트 계정이 아직 없는 경우 BlueXP에 등록 하여 Astra Control Service 및 NetApp의 기타 클라우드 서비스에 액세스할 수 있습니다. BlueXP 또는 NetApp Support 사이트 계정이 있고 등록을 완료한 경우 에 액세스할 수 있습니다 "Astra 제어 서비스" BlueXP 또는 NetApp Support 사이트 자격 증명을 직접 사용합니다.

또한 회사 디렉토리(통합 ID)의 자격 증명을 사용하여 Single Sign-On을 사용하여 BlueXP에 로그인할 수 있습니다. 자세한 내용은 로 이동하십시오 "Help Center(도움말 센터" 그런 다음 * Cloud Central 로그인 옵션 * 을 선택합니다.

단계

1. 로 이동합니다 "NetApp BlueXP".
2. 오른쪽 상단에서 * 시작하기 * 를 선택합니다.
3. 등록 * 을 선택합니다.
4. 양식을 작성합니다.

   여기에 입력한 전화 번호 및 이메일 주소가 이전 Astra Control 등록 양식에서 사용한 주소와 동일한지 확인하십시오.

5. 등록 * 을 선택합니다.

   양식에 입력한 이메일 주소는 NetApp BlueXP 사용자 ID를 위한 것입니다. 새 Astra Control 계정에 등록하거나 Astra Control 관리자가 기존 Astra Control 계정에 초대할 때 이 BlueXP 사용자 ID를 사용합니다.

6. NetApp BlueXP의 이메일을 기다려 주십시오. 이메일 주소는 saas.support@netapp.com 이며 도착하는 데 몇 분 정도 걸릴 수 있습니다. 스팸 폴더를 확인하십시오.
7. 전자 메일이 도착하면 전자 메일의 링크를 선택하여 전자 메일 주소를 확인합니다.

결과

이제 활성 BlueXP 사용자 로그인이 있습니다.

이제 등록하셨으므로 에서 BlueXP 자격 증명을 사용하여 Astra Control에 직접 액세스할 수 있습니다 https://astra.netapp.io.

Astra Control Service에 클러스터를 추가합니다

환경을 설정한 후에는 Kubernetes 클러스터를 생성하고 Astra Control Service에 추가할 준비가 된 것입니다. 이를 통해 Astra Control Service를 사용하여 클러스터의 애플리케이션을 보호할 수 있습니다.

Astra Control Service에 추가해야 하는 클러스터의 유형에 따라 클러스터를 추가하려면 다른 단계를 사용해야 합니다.

- "공용 공급자 관리 클러스터를 Astra Control Service에 추가합니다": 다음 단계를 수행하여 공용 IP 주소가 있고 클라우드 공급자가 관리하는 클러스터를 추가할 수 있습니다. 클라우드 공급자의 서비스 주 계정, 서비스 계정 또는 사용자 계정이 필요합니다.
- "Astra Control Service에 프라이빗 공급자 관리 클러스터를 추가합니다": 다음 단계를 수행하여 프라이빗 IP 주소가 있고 클라우드 공급자가 관리하는 클러스터를 추가할 수 있습니다. 클라우드 공급자의 서비스 주 계정, 서비스 계정 또는 사용자 계정이 필요합니다.
- "공용 자가 관리 클러스터를 Astra Control Service에 추가합니다": 다음 단계를 사용하여 공용 IP 주소가 있고 조직에서 관리하는 클러스터를 추가할 수 있습니다. 추가하려는 클러스터에 대해 kubecononfig 파일을 생성해야 합니다.
- "Astra Control Service에 프라이빗 자체 관리 클러스터를 추가합니다": 다음 단계를 사용하여 개인 IP 주소가 있고 조직에서 관리하는 클러스터를 추가할 수 있습니다. 추가하려는 클러스터에 대해 kubecononfig 파일을 생성해야 합니다.

Astra Connector를 설치하여 클러스터를 관리한다

Astra Connector는 관리형 클러스터에 상주하는 소프트웨어로, 관리형 클러스터와 Astra Control 간의 통신을 지원합니다. Astra Control Service를 사용하여 관리되는 클러스터의 경우 Astra Connector에 사용할 수 있는 2가지 버전이 있습니다.

- * 이전 버전의 Astra Connector *: "이전 버전의 Astra Connector를 설치합니다" Kubernetes 네이티브가 아닌 워크플로우를 통해 클러스터를 관리하려는 경우, 클러스터에 위치한다.
- [기술 미리보기] * 선언 Kubernetes Astra Connector *: "선언적 Kubernetes 워크플로로 관리되는 클러스터용 Astra Connector를 설치합니다" 선언적 Kubernetes 워크플로우를 사용하여 클러스터를 관리하려는 경우 클러스터에 두어야 합니다. 클러스터에 Astra Connector를 설치하면 클러스터가 Astra Control에 자동으로 추가됩니다.

선언적 Kubernetes Astra Connector는 Astra Control EAP(Early Adopter Program)의 일부로만 제공됩니다. 에 가입하는 방법에 대한 자세한 내용은 NetApp 영업 담당자에게 문의하십시오.

이전 버전의 Astra Connector를 설치합니다

Astra Control Service는 이전 버전의 Astra Connector를 사용하여 Kubernetes가 아닌 네이티브 워크플로우로 관리되는 프라이빗 클러스터 와 Astra Control Service 간의 통신을 지원합니다. Kubernetes가 아닌 네이티브 워크플로우로 관리하려는 프라이빗 클러스터에 Astra Connector를 설치해야 합니다.

이전 버전의 Astra Connector는 Kubernetes가 아닌 네이티브 워크플로로 관리되는 다음 유형의 프라이빗 클러스터를 지원합니다.

- Amazon Elastic Kubernetes Service(EKS)
- Azure Kubernetes 서비스(AKS)
- Google Kubernetes Engine(GKE)
- AWS 기반 Red Hat OpenShift Service(ROSA)
- AWS PrivateLink와 함께 Rosa
- Red Hat OpenShift Container Platform 온-프레미스

이 작업에 대해

- 이러한 단계를 수행할 때 Astra Control Service로 관리할 프라이빗 클러스터에 대해 다음 명령을 실행합니다.
- 방호 호스트를 사용하는 경우 방호 호스트의 명령줄에서 이러한 명령을 실행하십시오.

시작하기 전에

- Astra Control Service를 사용하여 관리할 프라이빗 클러스터에 액세스해야 합니다.
- 클러스터에 Astra Connector 연산자를 설치하려면 Kubernetes 관리자 권한이 필요합니다.

단계

1. Kubernetes가 아닌 네이티브 워크플로로 관리할 프라이빗 클러스터에 이전 Astra Connector 연산자를 설치합니다. 이 명령을 실행하면 네임스페이스가 생성됩니다 `astra-connector-operator` 이 만들어지고 구성이 네임스페이스에 적용됩니다.

```
kubectl apply -f https://github.com/NetApp/astra-connector-
operator/releases/download/23.07.0-
202310251519/astraconnector_operator.yaml
```

2. 작업자가 설치되어 있고 준비가 되었는지 확인합니다.

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

3. Astra Control에서 API 토큰을 받습니다. 을 참조하십시오 "Astra 자동화 문서" 를 참조하십시오.
4. Astra-Connector 네임스페이스를 생성합니다.

```
kubectl create ns astra-connector
```

5. Astra Connector CR 파일을 생성하고 이름을 지정합니다 `astra-connector-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - * <ASTRA_CONTROL_SERVICE_URL> * : Astra Control Service의 웹 UI URL. 예를 들면 다음과 같습니다.

     ```
     https://astra.netapp.io
     ```

   - * <ASTRA_CONTROL_SERVICE_API_TOKEN> *: 이전 단계에서 얻은 Astra Control API 토큰.
   - * <PRIVATE_AKS_CLUSTER_NAME> *: (AKS 클러스터만 해당) - 전용 Azure Kubernetes Service 클러스터의 클러스터 이름입니다. 전용 AKS 클러스터를 추가하는 경우에만 주석을 해제하고 이 줄을 채웁니다.
   - * <ASTRA_CONTROL_ACCOUNT_ID> *: Astra Control 웹 UI에서 가져옵니다. 페이지 오른쪽 상단의 그림 아이콘을 선택하고 * API 액세스 * 를 선택합니다.

```
apiVersion: netapp.astraconnector.com/v1
kind: AstraConnector
metadata:
  name: astra-connector
  namespace: astra-connector
spec:
  natssync-client:
    cloud-bridge-url: <ASTRA_CONTROL_SERVICE_URL>
  imageRegistry:
    name: theotw
    secret: ""
  astra:
    token: <ASTRA_CONTROL_SERVICE_API_TOKEN>
    #clusterName: <PRIVATE_AKS_CLUSTER_NAME>
    accountId: <ASTRA_CONTROL_ACCOUNT_ID>
    acceptEULA: yes
```

6. 를 채운 후 `astra-connector-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

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

7. Astra Connector가 완전히 구축되었는지 확인:

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

8. 클러스터가 Astra Control에 등록되었는지 확인:

```
kubectl get astraconnector -n astra-connector
```

다음과 유사한 출력이 표시됩니다.

```
NAME              REGISTERED   ASTRACONNECTORID
STATUS
astra-connector   true         be475ae5-1511-4eaa-9b9e-712f09b0d065
Registered with Astra
```

ASTRACONNECTORID를 기록해 둡니다. Astra Control에 클러스터를 추가할 때 필요합니다.

다음 단계

Astra Connector를 설치했으므로 이제 프라이빗 클러스터를 Astra Control Service에 추가할 준비가 되었습니다.

- "Astra Control Service에 프라이빗 공급자 관리 클러스터를 추가합니다": 다음 단계를 수행하여 프라이빗 IP 주소가 있고 클라우드 공급자가 관리하는 클러스터를 추가할 수 있습니다. 클라우드 공급자의 서비스 주 계정, 서비스 계정 또는 사용자 계정이 필요합니다.
- "Astra Control Service에 프라이빗 자체 관리 클러스터를 추가합니다": 다음 단계를 사용하여 개인 IP 주소가 있고 조직에서 관리하는 클러스터를 추가할 수 있습니다. 추가하려는 클러스터에 대해 kubecononfig 파일을 생성해야 합니다.

를 참조하십시오

- "클러스터를 추가합니다"

(기술 미리 보기) 선언적 **Kubernetes Astra Connector**를 설치합니다

선언적 Kubernetes 워크플로우를 사용하여 관리되는 클러스터는 Astra Connector를 사용하여 관리형 클러스터와 Astra Control 간의 통신을 지원합니다. 선언적 Kubernetes 워크플로로 관리할 모든 클러스터에 Astra Connector를 설치해야 합니다.

Kubernetes 명령 및 CR(Custom Resource) 파일을 사용하여 선언적 Kubernetes Astra Connector를 설치합니다.

이 작업에 대해

- 이러한 단계를 수행할 때 Astra Control으로 관리하려는 클러스터에서 다음 명령을 실행합니다.
- 방호 호스트를 사용하는 경우 방호 호스트의 명령줄에서 이러한 명령을 실행하십시오.

시작하기 전에

- Astra Control을 사용하여 관리할 클러스터에 액세스해야 합니다.
- 클러스터에 Astra Connector 연산자를 설치하려면 Kubernetes 관리자 권한이 필요합니다.

> Kubernetes 1.25 이상 클러스터의 기본값인 Pod 보안 승인 적용을 사용하여 클러스터를 구성한 경우 해당 네임스페이스에 PSA 제한을 활성화해야 합니다. 을 참조하십시오 "Astra Control을 사용하여 클러스터 관리를 위한 환경을 준비합니다" 를 참조하십시오.

단계

1. 선언적 Kubernetes 워크플로로 관리할 클러스터에 Astra Connector 연산자를 설치합니다. 이 명령을 실행하면 네임스페이스가 생성됩니다 `astra-connector-operator` 이 만들어지고 구성이 네임스페이스에 적용됩니다.

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

2. 작업자가 설치되어 있고 준비가 되었는지 확인합니다.

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

3. Astra Control에서 API 토큰을 받습니다. 을 참조하십시오 "Astra 자동화 문서" 를 참조하십시오.
4. 토큰을 사용하여 암호를 생성합니다. <API_TOKEN>를 Astra Control에서 받은 토큰으로 대체:

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

5. Astra Connector 이미지를 가져오는 데 사용할 Docker 암호를 생성합니다. 괄호 안의 값을 사용자 환경의 정보로 대체:

<ASTRA_CONTROL_ACCOUNT_ID>는 Astra Control 웹 UI에서 찾을 수 있습니다. 웹 UI에서 페이지 오른쪽 상단의 그림 아이콘을 선택하고 * API access * 를 선택합니다.

```
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 파일을 생성하고 이름을 지정합니다 `astra-connector-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - <ASTRA_CONTROL_ACCOUNT_ID>: 이전 단계에서 Astra Control 웹 UI에서 구함.
   - <CLUSTER_NAME>: Astra Control에서 이 클러스터를 할당해야 하는 이름입니다.
   - <ASTRA_CONTROL_URL>: Astra Control의 웹 UI URL입니다. 예를 들면 다음과 같습니다.

```
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. 를 채운 후 `astra-connector-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

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

8. Astra Connector가 완전히 구축되었는지 확인:

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

9. 클러스터가 Astra Control에 등록되었는지 확인:

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

다음과 유사한 출력이 표시됩니다.

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

10. Astra Control 웹 UI의 * 클러스터 * 페이지에서 관리되는 클러스터 목록에 클러스터가 나타나는지 확인합니다.

공급자 관리 클러스터를 추가합니다

공용 공급자 관리 클러스터를 **Astra Control Service**에 추가합니다

클라우드 환경을 설정한 후에는 Kubernetes 클러스터를 생성하고 Astra Control Service에 추가할 준비가 된 것입니다.

- Kubernetes 클러스터를 생성합니다
- 클러스터를 Astra Control Service에 추가합니다
- 기본 스토리지 클래스를 변경합니다

Kubernetes 클러스터를 생성합니다

클러스터가 아직 없는 경우 요건에 맞는 클러스터를 생성할 수 있습니다 "Amazon Elastic Kubernetes Service(EKS)에 대한 Astra Control 서비스 요구사항". 클러스터가 아직 없는 경우 요건에 맞는 클러스터를 생성할 수 있습니다 "Google Kubernetes Engine(GKE)에 대한 Astra Control Service 요구 사항". 클러스터가 아직 없는 경우 요건에 맞는 클러스터를 생성할 수 있습니다 "Azure NetApp Files를 사용하는 Azure Kubernetes Service(AKS)에 대한 Astra Control 서비스 요구사항" 또는 "Azure 관리 디스크를 사용하는 Azure Kubernetes Service(AKS)에 대한 Astra Control Service 요구 사항".

> Astra Control Service는 인증 및 ID 관리를 위해 Azure Active Directory(Azure AD)를 사용하는 AKS 클러스터를 지원합니다. 클러스터를 생성할 때 의 지침을 따릅니다 "공식 문서" Azure AD를 사용하도록 클러스터를 구성합니다. 클러스터가 AKS로 관리되는 Azure AD 통합에 대한 요구 사항을 충족하는지 확인해야 합니다.

클러스터를 **Astra Control Service**에 추가합니다

Astra Control Service에 로그인한 후 첫 번째 단계는 클러스터 관리를 시작하는 것입니다. Astra Control Service에 클러스터를 추가하려면 먼저 특정 작업을 수행하고 클러스터가 특정 요구 사항을 충족하는지 확인해야 합니다.

Azure Kubernetes Service 및 Google Kubernetes Engine 클러스터를 관리할 때 Astra Control Provisioner 설치 및 라이프사이클 관리에 대한 두 가지 옵션이 있습니다.

- Astra Control Service를 사용하여 Astra Control Provisioner의 라이프사이클을 자동으로 관리할 수 있습니다. 이렇게 하려면 Astra Trident가 설치되어 있지 않고 Astra Control Service를 사용하여 관리하려는 클러스터에 Astra Control Provisioner가 활성화되어 있지 않아야 합니다. 이 경우 클러스터 관리를 시작할 때 Astra Control Service가 Astra Control Provisioner를 자동으로 활성화하며 Astra Control Provisioner 업그레이드는 자동으로 처리됩니다.
- Astra Control Provisioner의 라이프사이클을 직접 관리할 수 있습니다. 이렇게 하려면 Astra Control Service로 클러스터를 관리하기 전에 클러스터에서 Astra Control Provisioner를 활성화합니다. 이 경우 Astra Control Service는 Astra Control Provisioner가 이미 활성화되었음을 감지하여 Astra Control Provisioner 업그레이드를 다시 설치하거나 관리하지 않습니다. 을 참조하십시오 "Astra Control Provisioner를 활성화합니다" 단계를 위해 Astra Control Provisioner를 활성화합니다.

Astra Control Service를 사용하여 Amazon Web Services 클러스터를 관리할 때 Astra Control Provisioner에서만 사용할 수 있는 스토리지 백엔드가 필요한 경우 Astra Control Service를 통해 클러스터를 관리하기 전에 클러스터에서 Astra Control Provisioner를 수동으로 활성화해야 합니다. 을 참조하십시오 "Astra Control Provisioner를 활성화합니다" Astra Control Provisioner를 활성화하는 단계에 대해 알아보십시오.

시작하기 전에

Amazon Web Services에서 직접 지원합니다

- 클러스터를 생성한 IAM 사용자의 자격 증명이 포함된 JSON 파일이 있어야 합니다. "IAM 사용자를 생성하는 방법을 알아봅니다".
- Amazon FSx for NetApp ONTAP에는 Astra Control Provisioner가 필요합니다. Amazon FSx for NetApp ONTAP를 EKS 클러스터의 스토리지 백엔드로 사용할 계획인 경우 에서 Astra Control Provisioner 정보를 참조하십시오 "EKS 클러스터 요구 사항".
- (선택 사항) 제공해야 하는 경우 `kubectl` 클러스터 생성자가 아닌 다른 IAM 사용자에 대한 클러스터에 대한 명령 액세스는 의 지침을 참조하십시오 "Amazon EKS에서 클러스터를 생성한 후 다른 IAM 사용자 및 역할에 대한 액세스를 제공하려면 어떻게 해야 합니까?".
- NetApp Cloud Volumes ONTAP를 스토리지 백엔드로 사용하려는 경우 Amazon Web Services와 연동되도록 Cloud Volumes ONTAP를 구성해야 합니다. Cloud Volumes ONTAP를 참조하십시오 "설치 설명서".

Microsoft Azure를 참조하십시오

- 서비스 보안 주체를 생성할 때 Azure CLI의 출력이 포함된 JSON 파일이 있어야 합니다. "서비스 보안 주체를 설정하는 방법에 대해 알아봅니다".

  JSON 파일에 추가하지 않은 경우 Azure 구독 ID도 필요합니다.

- NetApp Cloud Volumes ONTAP를 스토리지 백엔드로 사용하려는 경우 Microsoft Azure와 연동하도록 Cloud Volumes ONTAP를 구성해야 합니다. Cloud Volumes ONTAP를 참조하십시오 "설치 설명서".

Google 클라우드

- 필요한 권한이 있는 서비스 계정에 대한 서비스 계정 키 파일이 있어야 합니다. "서비스 계정 설정 방법에 대해 알아보십시오".
- NetApp Cloud Volumes ONTAP를 스토리지 백엔드로 사용하려는 경우 Cloud Volumes ONTAP이 Google Cloud와 연동되도록 구성해야 합니다. Cloud Volumes ONTAP를 참조하십시오 "설치 설명서".

단계

1. (선택 사항) Amazon EKS 클러스터를 추가하거나 Astra Control Provisioner의 설치 및 업그레이드를 직접 관리하려는 경우 클러스터에서 Astra Control Provisioner를 활성화합니다. 을 참조하십시오 "Astra Control Provisioner를 활성화합니다" 참조하십시오.
2. 브라우저에서 Astra Control Service 웹 UI를 엽니다.
3. 대시보드에서 * Kubernetes 클러스터 관리 * 를 선택합니다.

   표시되는 메시지에 따라 클러스터를 추가합니다.

4. * 공급자 *: 클라우드 공급자를 선택한 다음 새 클라우드 인스턴스를 생성하는 데 필요한 자격 증명을 제공하거나 사용할 기존 클라우드 인스턴스를 선택하십시오.
5. * Amazon Web Services *: JSON 파일을 업로드하거나 클립보드에서 해당 JSON 파일의 콘텐츠를 붙여넣어 Amazon Web Services IAM 사용자 계정에 대한 세부 정보를 제공합니다.

   JSON 파일에는 클러스터를 생성한 IAM 사용자의 자격 증명이 포함되어야 합니다.

6. * Microsoft Azure *: JSON 파일을 업로드하거나 클립보드에서 해당 JSON 파일의 내용을 붙여넣어 Azure 서비스

보안 주체에 대한 세부 정보를 제공합니다.

JSON 파일에는 서비스 보안 주체를 생성할 때 Azure CLI의 출력이 포함되어야 합니다. 또한 구독 ID를 포함할 수 있으므로 Astra에 자동으로 추가됩니다. 그렇지 않으면 JSON을 제공한 후 ID를 수동으로 입력해야 합니다.

7. * Google Cloud Platform *: 파일을 업로드하거나 클립보드의 콘텐츠를 붙여 넣어 서비스 계정 키 파일을 제공합니다.

   Astra Control Service는 서비스 계정을 사용하여 Google Kubernetes Engine에서 실행 중인 클러스터를 검색합니다.

8. * 기타 *: 이 탭은 자체 관리형 클러스터에만 사용됩니다.

   a. * 클라우드 인스턴스 이름 *: 이 클러스터를 추가할 때 생성될 새 클라우드 인스턴스의 이름을 입력합니다. 에 대해 자세히 알아보십시오 "클라우드 인스턴스".

   b. 다음 * 을 선택합니다.

      Astra Control Service에는 선택할 수 있는 클러스터 목록이 표시됩니다.

   c. * 클러스터 *: 목록에서 Astra Control Service에 추가할 클러스터를 선택합니다.

      클러스터 목록에서 선택할 때는 * 자격 조건 * 열에 주의해야 합니다. 클러스터가 "부적격" 또는 "부분적으로 적격"인 경우 상태 위로 마우스를 가져가면 클러스터에 문제가 있는지 확인할 수 있습니다. 예를 들어 클러스터에 작업자 노드가 없는 것을 식별할 수 있습니다.

   d. 다음 * 을 선택합니다.

   e. (선택 사항) * 스토리지 *: 선택적으로 이 클러스터에 Kubernetes 애플리케이션을 배포할 스토리지 클래스를 선택하여 기본적으로 사용하도록 합니다.

9. 클러스터에 대한 새 기본 스토리지 클래스를 선택하려면 * 새 기본 스토리지 클래스 할당 * 확인란을 설정합니다.

10. 목록에서 새 기본 스토리지 클래스를 선택합니다.

    각 클라우드 공급자의 스토리지 서비스에는 다음과 같은 가격, 성능 및 복원력 정보가 표시됩니다.

    - Google Cloud용 Cloud Volumes Service: 가격, 성능 및 복원력 정보
    - Google 영구 디스크: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
    - Azure NetApp Files: 성능 및 복원력 정보
    - Azure 관리 디스크: 사용 가능한 가격, 성능 또는 복원력 정보가 없습니다
    - Amazon Elastic Block Store: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
    - NetApp ONTAP용 Amazon FSx: 가격, 성능 또는 복원력 정보 없음
    - NetApp Cloud Volumes ONTAP: 가격, 성능 또는 복원력 정보를 제공할 수 없습니다

    각 스토리지 클래스는 다음 서비스 중 하나를 활용할 수 있습니다.

    - "Google Cloud용 Cloud Volumes Service"
    - "Google 영구 디스크"
      - "Azure NetApp Files"

- "Azure로 관리되는 디스크"
- "Amazon Elastic Block Store를 클릭합니다"
- "NetApp ONTAP용 Amazon FSx"
- "NetApp Cloud Volumes ONTAP를 참조하십시오"

  에 대해 자세히 알아보십시오 "Amazon Web Services 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "AKS 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "GKE 클러스터용 저장소 클래스".

  a. 다음 * 을 선택합니다.

  b. * 검토 및 승인 *: 구성 세부 정보를 검토합니다.

  c. 클러스터를 Astra Control Service에 추가하려면 * 추가 * 를 선택합니다.

결과

이 클라우드 공급자를 위해 추가한 첫 번째 클러스터인 경우 Astra Control Service는 해당 클러스터에서 실행되는 애플리케이션 백업을 위해 클라우드 공급자용 오브젝트 저장소를 생성합니다. (이 클라우드 공급자에 후속 클러스터를 추가할 경우 더 이상 오브젝트 저장소가 생성되지 않습니다.) 기본 스토리지 클래스를 지정한 경우 Astra Control Service는 사용자가 지정한 기본 스토리지 클래스를 설정합니다. Amazon Web Services 또는 Google Cloud Platform에서 관리되는 클러스터의 경우 Astra Control Service는 클러스터에 관리자 계정도 생성합니다. 이 작업은 몇 분 정도 걸릴 수 있습니다.

기본 스토리지 클래스를 변경합니다

클러스터의 기본 스토리지 클래스를 변경할 수 있습니다.

Astra Control을 사용하여 기본 스토리지 클래스를 변경합니다

Astra Control 내에서 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 클러스터에서 이전에 설치된 스토리지 백엔드 서비스를 사용하는 경우 이 방법을 사용하여 기본 스토리지 클래스를 변경하지 못할 수 있습니다(* 기본값으로 설정* 작업은 선택할 수 없음). 이 경우 를 사용할 수 있습니다 명령줄을 사용하여 기본 스토리지 클래스를 변경합니다.

단계

1. Astra Control Service UI에서 * Clusters * 를 선택합니다.
2. 클러스터 * 페이지에서 변경할 클러스터를 선택합니다.
3. Storage * 탭을 선택합니다.
4. 스토리지 클래스 * 범주를 선택합니다.
5. 기본값으로 설정할 스토리지 클래스에 대해 * Actions * 메뉴를 선택합니다.
6. Set as default * 를 선택합니다.

명령줄을 사용하여 기본 스토리지 클래스를 변경합니다

Kubernetes 명령을 사용하여 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 이 방법은 클러스터의 구성에 관계없이 작동합니다.

단계

1. Kubernetes 클러스터에 로그인합니다.

2. 클러스터의 스토리지 클래스를 나열합니다.

```
kubectl get storageclass
```

3. 기본 스토리지 클래스에서 기본 지정을 제거합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-
class":"false"}}}'
```

4. 다른 스토리지 클래스를 기본값으로 표시합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 새 기본 스토리지 클래스를 확인합니다.

```
kubectl get storageclass
```

Astra Control Service에 프라이빗 공급자 관리 클러스터를 추가합니다

Astra Control Service를 사용하여 개인 GKE(Google Kubernetes Engine) 클러스터를 관리할 수 있습니다. 이 지침에서는 이미 전용 AKS 또는 OpenShift 클러스터를 만들고 원격으로 액세스할 수 있는 안전한 방법을 준비했다고 가정합니다. 전용 AKS 또는 OpenShift 클러스터를 만들고 액세스하는 방법에 대한 자세한 내용은 다음 문서를 참조하십시오.

- "프라이빗 AKS 클러스터를 위한 Azure 설명서"
- "전용 OpenShift 클러스터를 위한 Azure 문서"

Astra Control Service를 사용하여 AKS(프라이빗 Azure Kubernetes Service) 클러스터뿐 아니라 AKS의 프라이빗 Red Hat OpenShift 클러스터를 관리할 수 있습니다. 이 지침에서는 이미 전용 AKS 또는 OpenShift 클러스터를 만들고 원격으로 액세스할 수 있는 안전한 방법을 준비했다고 가정합니다. 전용 AKS 또는 OpenShift 클러스터를 만들고 액세스하는 방법에 대한 자세한 내용은 다음 문서를 참조하십시오.

- "프라이빗 AKS 클러스터를 위한 Azure 설명서"
- "전용 OpenShift 클러스터를 위한 Azure 문서"

Astra Control Service를 사용하여 전용 EKS(Amazon Elastic Kubernetes Service) 클러스터를 관리할 수 있습니다. 이 지침에서는 이미 전용 EKS 클러스터를 생성하고 원격으로 액세스할 수 있는 안전한 방법을 준비했다고 가정합니다. 전용 EKS 클러스터 생성 및 액세스에 대한 자세한 내용은 을 참조하십시오 "아마존 EKS 문서".

Astra Control Service에 프라이빗 클러스터를 추가하려면 다음 작업을 수행해야 합니다.

1. Astra Connector를 설치합니다
2. 영구 스토리지를 설정합니다
3. Astra Control Service에 프라이빗 공급자 관리 클러스터를 추가합니다

Astra Connector를 설치합니다

프라이빗 클러스터를 추가하기 전에 Astra Control이 클러스터와 통신할 수 있도록 클러스터에 Astra Connector를 설치해야 합니다. 을 참조하십시오 "Kubernetes가 아닌 네이티브 워크플로우로 관리되는 프라이빗 클러스터용 Astra Connector의 이전 버전을 설치합니다" 를 참조하십시오.

영구 스토리지를 설정합니다

클러스터의 영구 스토리지를 구성합니다. 영구 스토리지 구성에 대한 자세한 내용은 시작 설명서를 참조하십시오.

- "Azure NetApp Files를 사용하여 Microsoft Azure를 설정합니다"
- "Azure 관리 디스크를 사용하여 Microsoft Azure를 설정합니다"
- "Amazon Web Services를 설정합니다"
- "Google Cloud를 설정합니다"

Astra Control Service에 프라이빗 공급자 관리 클러스터를 추가합니다

이제 Astra Control Service에 프라이빗 클러스터를 추가할 수 있습니다.

Azure Kubernetes Service 및 Google Kubernetes Engine 클러스터를 관리할 때 Astra Control Provisioner 설치 및 라이프사이클 관리에 대한 두 가지 옵션이 있습니다.

- Astra Control Service를 사용하여 Astra Control Provisioner의 라이프사이클을 자동으로 관리할 수 있습니다. 이렇게 하려면 Astra Trident가 설치되어 있지 않고 Astra Control Service를 사용하여 관리하려는 클러스터에 Astra Control Provisioner가 활성화되어 있지 않아야 합니다. 이 경우 클러스터 관리를 시작할 때 Astra Control Service가 Astra Control Provisioner를 자동으로 활성화하며 Astra Control Provisioner 업그레이드는 자동으로 처리됩니다.
- Astra Control Provisioner의 라이프사이클을 직접 관리할 수 있습니다. 이렇게 하려면 Astra Control Service로 클러스터를 관리하기 전에 클러스터에서 Astra Control Provisioner를 활성화합니다. 이 경우 Astra Control Service는 Astra Control Provisioner가 이미 활성화되었음을 감지하여 Astra Control Provisioner 업그레이드를 다시 설치하거나 관리하지 않습니다. 을 참조하십시오 "Astra Control Provisioner를 활성화합니다" 단계를 위해 Astra Control Provisioner를 활성화합니다.

Astra Control Service를 사용하여 Amazon Web Services 클러스터를 관리할 때 Astra Control Provisioner에서만 사용할 수 있는 스토리지 백엔드가 필요한 경우 Astra Control Service를 통해 클러스터를 관리하기 전에 클러스터에서 Astra Control Provisioner를 수동으로 활성화해야 합니다. 을 참조하십시오 "Astra Control Provisioner를 활성화합니다" Astra Control Provisioner를 활성화하는 단계에 대해 알아보십시오.

시작하기 전에

Amazon Web Services에서 직접 지원합니다

- 클러스터를 생성한 IAM 사용자의 자격 증명이 포함된 JSON 파일이 있어야 합니다. "IAM 사용자를 생성하는 방법을 알아봅니다".
- Amazon FSx for NetApp ONTAP에는 Astra Control Provisioner가 필요합니다. Amazon FSx for NetApp ONTAP를 EKS 클러스터의 스토리지 백엔드로 사용할 계획인 경우 에서 Astra Control Provisioner 정보를 참조하십시오 "EKS 클러스터 요구 사항".
- (선택 사항) 제공해야 하는 경우 `kubectl` 클러스터 생성자가 아닌 다른 IAM 사용자에 대한 클러스터에 대한 명령 액세스는 의 지침을 참조하십시오 "Amazon EKS에서 클러스터를 생성한 후 다른 IAM 사용자 및 역할에 대한 액세스를 제공하려면 어떻게 해야 합니까?".
- NetApp Cloud Volumes ONTAP를 스토리지 백엔드로 사용하려는 경우 Amazon Web Services와 연동되도록 Cloud Volumes ONTAP를 구성해야 합니다. Cloud Volumes ONTAP를 참조하십시오 "설치 설명서".

Microsoft Azure를 참조하십시오

- 서비스 보안 주체를 생성할 때 Azure CLI의 출력이 포함된 JSON 파일이 있어야 합니다. "서비스 보안 주체를 설정하는 방법에 대해 알아봅니다".

  JSON 파일에 추가하지 않은 경우 Azure 구독 ID도 필요합니다.

- NetApp Cloud Volumes ONTAP를 스토리지 백엔드로 사용하려는 경우 Microsoft Azure와 연동하도록 Cloud Volumes ONTAP를 구성해야 합니다. Cloud Volumes ONTAP를 참조하십시오 "설치 설명서".

Google 클라우드

- 필요한 권한이 있는 서비스 계정에 대한 서비스 계정 키 파일이 있어야 합니다. "서비스 계정 설정 방법에 대해 알아보십시오".
- 클러스터가 프라이빗 인 경우, 를 참조하십시오 "인증된 네트워크" Astra Control Service IP 주소를 허용해야 합니다.

  52.188.218.166/32

- NetApp Cloud Volumes ONTAP를 스토리지 백엔드로 사용하려는 경우 Cloud Volumes ONTAP이 Google Cloud와 연동되도록 구성해야 합니다. Cloud Volumes ONTAP를 참조하십시오 "설치 설명서".

단계

1. (선택 사항) Amazon EKS 클러스터를 추가하거나 Astra Control Provisioner의 설치 및 업그레이드를 직접 관리하려는 경우 클러스터에서 Astra Control Provisioner를 활성화합니다. 을 참조하십시오 "Astra Control Provisioner를 활성화합니다" 참조하십시오.
2. 브라우저에서 Astra Control Service 웹 UI를 엽니다.
3. 대시보드에서 * Kubernetes 클러스터 관리 * 를 선택합니다.

   표시되는 메시지에 따라 클러스터를 추가합니다.

4. * 공급자 *: 클라우드 공급자를 선택한 다음 새 클라우드 인스턴스를 생성하는 데 필요한 자격 증명을 제공하거나 사용할 기존 클라우드 인스턴스를 선택하십시오.
5. * Amazon Web Services *: JSON 파일을 업로드하거나 클립보드에서 해당 JSON 파일의 콘텐츠를 붙여넣어

Amazon Web Services IAM 사용자 계정에 대한 세부 정보를 제공합니다.

JSON 파일에는 클러스터를 생성한 IAM 사용자의 자격 증명이 포함되어야 합니다.

6. * Microsoft Azure *: JSON 파일을 업로드하거나 클립보드에서 해당 JSON 파일의 내용을 붙여넣어 Azure 서비스 보안 주체에 대한 세부 정보를 제공합니다.

   JSON 파일에는 서비스 보안 주체를 생성할 때 Azure CLI의 출력이 포함되어야 합니다. 또한 구독 ID를 포함할 수 있으므로 Astra에 자동으로 추가됩니다. 그렇지 않으면 JSON을 제공한 후 ID를 수동으로 입력해야 합니다.

7. * Google Cloud Platform *: 파일을 업로드하거나 클립보드의 콘텐츠를 붙여 넣어 서비스 계정 키 파일을 제공합니다.

   Astra Control Service는 서비스 계정을 사용하여 Google Kubernetes Engine에서 실행 중인 클러스터를 검색합니다.

8. * 기타 *: 이 탭은 자체 관리형 클러스터에만 사용됩니다.

   a. * 클라우드 인스턴스 이름 *: 이 클러스터를 추가할 때 생성될 새 클라우드 인스턴스의 이름을 입력합니다. 에 대해 자세히 알아보십시오 "클라우드 인스턴스".

   b. 다음 * 을 선택합니다.

      Astra Control Service에는 선택할 수 있는 클러스터 목록이 표시됩니다.

   c. * 클러스터 *: 목록에서 Astra Control Service에 추가할 클러스터를 선택합니다.

      클러스터 목록에서 선택할 때는 * 자격 조건 * 열에 주의해야 합니다. 클러스터가 "부적격" 또는 "부분적으로 적격"인 경우 상태 위로 마우스를 가져가면 클러스터에 문제가 있는지 확인할 수 있습니다. 예를 들어 클러스터에 작업자 노드가 없는 것을 식별할 수 있습니다.

9. 다음 * 을 선택합니다.

10. (선택 사항) * 스토리지 *: 선택적으로 이 클러스터에 Kubernetes 애플리케이션을 배포할 스토리지 클래스를 선택하여 기본적으로 사용하도록 합니다.

    a. 클러스터에 대한 새 기본 스토리지 클래스를 선택하려면 * 새 기본 스토리지 클래스 할당 * 확인란을 설정합니다.

    b. 목록에서 새 기본 스토리지 클래스를 선택합니다.

       각 클라우드 공급자의 스토리지 서비스에는 다음과 같은 가격, 성능 및 복원력 정보가 표시됩니다.

       - Google Cloud용 Cloud Volumes Service: 가격, 성능 및 복원력 정보
       - Google 영구 디스크: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
       - Azure NetApp Files: 성능 및 복원력 정보
       - Azure 관리 디스크: 사용 가능한 가격, 성능 또는 복원력 정보가 없습니다
       - Amazon Elastic Block Store: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
       - NetApp ONTAP용 Amazon FSx: 가격, 성능 또는 복원력 정보 없음
       - NetApp Cloud Volumes ONTAP: 가격, 성능 또는 복원력 정보를 제공할 수 없습니다

각 스토리지 클래스는 다음 서비스 중 하나를 활용할 수 있습니다.

- "Google Cloud용 Cloud Volumes Service"
- "Google 영구 디스크"
- "Azure NetApp Files"
- "Azure로 관리되는 디스크"
- "Amazon Elastic Block Store를 클릭합니다"
- "NetApp ONTAP용 Amazon FSx"
- "NetApp Cloud Volumes ONTAP를 참조하십시오"

에 대해 자세히 알아보십시오 "Amazon Web Services 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "AKS 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "GKE 클러스터용 저장소 클래스".

c. 다음 * 을 선택합니다.

d. * 검토 및 승인 *: 구성 세부 정보를 검토합니다.

e. 클러스터를 Astra Control Service에 추가하려면 * 추가 * 를 선택합니다.

결과

이 클라우드 공급자를 위해 추가한 첫 번째 클러스터인 경우 Astra Control Service는 해당 클러스터에서 실행되는 애플리케이션 백업을 위해 클라우드 공급자용 오브젝트 저장소를 생성합니다. (이 클라우드 공급자에 후속 클러스터를 추가할 경우 더 이상 오브젝트 저장소가 생성되지 않습니다.) 기본 스토리지 클래스를 지정한 경우 Astra Control Service는 사용자가 지정한 기본 스토리지 클래스를 설정합니다. Amazon Web Services 또는 Google Cloud Platform에서 관리되는 클러스터의 경우 Astra Control Service는 클러스터에 관리자 계정도 생성합니다. 이 작업은 몇 분 정도 걸릴 수 있습니다.

기본 스토리지 클래스를 변경합니다

클러스터의 기본 스토리지 클래스를 변경할 수 있습니다.

Astra Control을 사용하여 기본 스토리지 클래스를 변경합니다

Astra Control 내에서 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 클러스터에서 이전에 설치된 스토리지 백엔드 서비스를 사용하는 경우 이 방법을 사용하여 기본 스토리지 클래스를 변경하지 못할 수 있습니다(* 기본값으로 설정* 작업은 선택할 수 없음). 이 경우 를 사용할 수 있습니다 명령줄을 사용하여 기본 스토리지 클래스를 변경합니다.

단계

1. Astra Control Service UI에서 * Clusters * 를 선택합니다.
2. 클러스터 * 페이지에서 변경할 클러스터를 선택합니다.
3. Storage * 탭을 선택합니다.
4. 스토리지 클래스 * 범주를 선택합니다.
5. 기본값으로 설정할 스토리지 클래스에 대해 * Actions * 메뉴를 선택합니다.
6. Set as default * 를 선택합니다.

명령줄을 사용하여 기본 스토리지 클래스를 변경합니다

Kubernetes 명령을 사용하여 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 이 방법은 클러스터의 구성에 관계없이 작동합니다.

단계

1. Kubernetes 클러스터에 로그인합니다.
2. 클러스터의 스토리지 클래스를 나열합니다.

```
kubectl get storageclass
```

3. 기본 스토리지 클래스에서 기본 지정을 제거합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-
class":"false"}}}'
```

4. 다른 스토리지 클래스를 기본값으로 표시합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 새 기본 스토리지 클래스를 확인합니다.

```
kubectl get storageclass
```

자가 관리 클러스터를 추가합니다

공용 자가 관리 클러스터를 **Astra Control Service**에 추가합니다

환경을 설정한 후에는 Kubernetes 클러스터를 생성하고 Astra Control Service에 추가할 준비가 된 것입니다.

자가 관리형 클러스터는 직접 프로비저닝하고 관리하는 클러스터입니다. Astra Control Service는 퍼블릭 클라우드 환경에서 실행되는 자체 관리형 클러스터를 지원합니다. A를 업로드하여 자가 관리 클러스터를 Astra Control Service에 추가할 수 있습니다 `kubeconfig.yaml` 파일. 클러스터가 여기에 설명된 요구 사항을 충족하는지 확인해야 합니다.

지원되는 **Kubernetes** 배포

Astra Control Service를 사용하여 다음과 같은 유형의 퍼블릭, 자가 관리 클러스터를 관리할 수 있습니다.

Kubernetes 배포	지원되는 버전
Kubernetes(업스트림)	1.27 ~ 1.29
RKE(Rancher Kubernetes Engine)	RKE 1: Rancher Manager 2.7.9 포함 버전 1.24.17, 1.25.13, 1.26.8 RKE 2: Rancher Manager 2.6.13이 있는 버전 1.23.16 및 1.24.13 RKE 2: Rancher Manager 2.7.9 포함 버전 1.24.17, 1.25.14, 1.26.9
Red Hat OpenShift Container Platform	4.12에서 4.14까지

이 지침에서는 이미 자체 관리형 클러스터를 생성했다고 가정합니다.

- 클러스터를 Astra Control Service에 추가합니다
- 기본 스토리지 클래스를 변경합니다

클러스터를 **Astra Control Service**에 추가합니다

Astra Control Service에 로그인한 후 첫 번째 단계는 클러스터 관리를 시작하는 것입니다. Astra Control Service에 클러스터를 추가하려면 먼저 특정 작업을 수행하고 클러스터가 특정 요구 사항을 충족하는지 확인해야 합니다.

자가 관리형 클러스터는 직접 프로비저닝하고 관리하는 클러스터입니다. Astra Control Service는 퍼블릭 클라우드 환경에서 실행되는 자체 관리형 클러스터를 지원합니다. 자체 관리형 클러스터는 Astra Control Provisioner를 사용하여 NetApp 스토리지 서비스와 상호 연결하거나 컨테이너 스토리지 인터페이스(CSI) 드라이버를 사용하여 Amazon EBS(Elastic Block Store), Azure Managed Disks 및 Google Persistent Disk와 상호 작용할 수 있습니다.

Astra Control Service는 다음과 같은 Kubernetes 배포를 사용하는 자체 관리 클러스터를 지원합니다.

- Red Hat OpenShift Container Platform
- Rancher Kubernetes 엔진
- 업스트림 Kubernetes

자가 관리형 클러스터는 다음 요구사항을 충족해야 합니다.

- 클러스터를 인터넷을 통해 액세스할 수 있어야 합니다.
- CSI 드라이버로 활성화된 스토리지를 사용 중이거나 사용할 계획이면 해당 CSI 드라이버가 클러스터에 설치되어 있어야 합니다. CSI 드라이버를 사용하여 스토리지를 통합하는 방법에 대한 자세한 내용은 스토리지 서비스 설명서를 참조하십시오.
- 하나의 컨텍스트 요소만 포함된 클러스터 kubecononfig 파일에 액세스할 수 있습니다. 를 따릅니다 "참조하십시오" kubecononfig 파일을 생성합니다.
- 개인 CA(인증 기관)를 참조하는 kubeconfig 파일을 사용하여 클러스터를 추가하는 경우 에 다음 줄을 추가합니다 `cluster` kubeconfig 파일의 섹션. 이를 통해 Astra Control이 클러스터를 추가할 수 있습니다.

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

- * Rancher 전용 *: Rancher 환경에서 애플리케이션 클러스터를 관리할 때 Rancher가 제공하는 kubecon무화과 파일에서 애플리케이션 클러스터의 기본 컨텍스트를 수정하여 Rancher API 서버 컨텍스트 대신 컨트롤 플레인 컨텍스트를 사용합니다. 따라서 Rancher API 서버의 부하가 줄어들고 성능이 향상됩니다.
- * Astra Control Provisioner 요구 사항 *: 클러스터를 관리하려면 Astra Trident 구성 요소를 포함하여 올바르게 구성된 Astra Control Provisioner가 있어야 합니다.
  - * Astra Trident 환경 요구 사항 검토 *: Astra Control Provisioner를 설치 또는 업그레이드하기 전에 를 검토하십시오 "지원되는 프런트엔드, 백엔드 및 호스트 구성".
  - * Astra Control Provisioner 기능 활성화 *: Astra Trident 23.10 이상을 설치하고 활성화하는 것이 좋습니다 "Astra Control Provisioner 고급 스토리지 기능". 향후 릴리즈에서 Astra Control Provisioner가 활성화되어 있지 않으면 Astra Control이 Astra Trident를 지원하지 않습니다.
  - * 스토리지 백엔드 구성 *: 최소 하나의 스토리지 백엔드가 있어야 합니다 "Astra Trident에서 구성됨" 클러스터에서.
  - * 스토리지 클래스 구성 *: 최소 하나의 스토리지 클래스가 있어야 합니다 "Astra Trident에서 구성됨" 클러스터에서. 기본 저장소 클래스가 구성된 경우 기본 주석이 있는 * 전용 * 저장소 클래스인지 확인합니다.
  - * 볼륨 스냅샷 컨트롤러 구성 및 볼륨 스냅샷 클래스 설치 *: "볼륨 스냅샷 컨트롤러를 설치합니다" 따라서 Astra Control에서 스냅샷을 생성할 수 있습니다. "생성" 하나 이상 `VolumeSnapshotClass` Astra

Trident 사용:

단계

1. 대시보드에서 * Kubernetes 클러스터 관리 * 를 선택합니다.

   표시되는 메시지에 따라 클러스터를 추가합니다.

2. * 공급자 *: * 기타 * 탭을 선택하여 자체 관리 클러스터에 대한 세부 정보를 추가합니다.

   a. * 기타 *: 을 업로드하여 자체 관리되는 클러스터에 대한 세부 정보를 제공합니다 `kubeconfig.yaml` 파일을 클릭하거나 의 내용을 붙여 넣습니다 `kubeconfig.yaml` 파일을 클립보드에 저장합니다.

   > 직접 만드는 경우 `kubeconfig` 파일에서 * 하나의 * 컨텍스트 요소만 정의해야 합니다. 을 참조하십시오 "Kubernetes 문서" 을 참조하십시오 `kubeconfig` 파일.

3. * 자격 증명 이름 *: Astra Control에 업로드하는 자가 관리 클러스터 자격 증명의 이름을 입력합니다. 기본적으로 자격 증명 이름은 클러스터 이름으로 자동 채워집니다.

4. * Private route identifier *: 이 필드는 전용 클러스터에서만 사용할 수 있습니다.

5. 다음 * 을 선택합니다.

6. (선택 사항) * 스토리지 *: 선택적으로 이 클러스터에 Kubernetes 애플리케이션을 배포할 스토리지 클래스를 선택하여 기본적으로 사용하도록 합니다.

   a. 클러스터에 대한 새 기본 스토리지 클래스를 선택하려면 * 새 기본 스토리지 클래스 할당 * 확인란을 설정합니다.

   b. 목록에서 새 기본 스토리지 클래스를 선택합니다.

   > 각 클라우드 공급자의 스토리지 서비스에는 다음과 같은 가격, 성능 및 복원력 정보가 표시됩니다.
   >
   > - Google Cloud용 Cloud Volumes Service: 가격, 성능 및 복원력 정보
   > - Google 영구 디스크: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
   > - Azure NetApp Files: 성능 및 복원력 정보
   > - Azure 관리 디스크: 사용 가능한 가격, 성능 또는 복원력 정보가 없습니다
   > - Amazon Elastic Block Store: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
   > - NetApp ONTAP용 Amazon FSx: 가격, 성능 또는 복원력 정보 없음
   > - NetApp Cloud Volumes ONTAP: 가격, 성능 또는 복원력 정보를 제공할 수 없습니다

   각 스토리지 클래스는 다음 서비스 중 하나를 활용할 수 있습니다.

   - "Google Cloud용 Cloud Volumes Service"
   - "Google 영구 디스크"
     - "Azure NetApp Files"
     - "Azure로 관리되는 디스크"
     - "Amazon Elastic Block Store를 클릭합니다"

- "NetApp ONTAP용 Amazon FSx"
- "NetApp Cloud Volumes ONTAP를 참조하십시오"

  에 대해 자세히 알아보십시오 "Amazon Web Services 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "AKS 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "GKE 클러스터용 저장소 클래스".

c. 다음 * 을 선택합니다.

d. * 검토 및 승인 *: 구성 세부 정보를 검토합니다.

e. 클러스터를 Astra Control Service에 추가하려면 * 추가 * 를 선택합니다.

기본 스토리지 클래스를 변경합니다

클러스터의 기본 스토리지 클래스를 변경할 수 있습니다.

Astra Control을 사용하여 기본 스토리지 클래스를 변경합니다

Astra Control 내에서 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 클러스터에서 이전에 설치된 스토리지 백엔드 서비스를 사용하는 경우 이 방법을 사용하여 기본 스토리지 클래스를 변경하지 못할 수 있습니다(* 기본값으로 설정* 작업은 선택할 수 없음). 이 경우 를 사용할 수 있습니다 명령줄을 사용하여 기본 스토리지 클래스를 변경합니다.

단계

1. Astra Control Service UI에서 * Clusters * 를 선택합니다.
2. 클러스터 * 페이지에서 변경할 클러스터를 선택합니다.
3. Storage * 탭을 선택합니다.
4. 스토리지 클래스 * 범주를 선택합니다.
5. 기본값으로 설정할 스토리지 클래스에 대해 * Actions * 메뉴를 선택합니다.
6. Set as default * 를 선택합니다.

명령줄을 사용하여 기본 스토리지 클래스를 변경합니다

Kubernetes 명령을 사용하여 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 이 방법은 클러스터의 구성에 관계없이 작동합니다.

단계

1. Kubernetes 클러스터에 로그인합니다.
2. 클러스터의 스토리지 클래스를 나열합니다.

```
kubectl get storageclass
```

3. 기본 스토리지 클래스에서 기본 지정을 제거합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-
class":"false"}}}'
```

4. 다른 스토리지 클래스를 기본값으로 표시합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 새 기본 스토리지 클래스를 확인합니다.

```
kubectl get storageclass
```

Astra Control Service에 프라이빗 자체 관리 클러스터를 추가합니다

환경을 설정한 후에는 Kubernetes 클러스터를 생성하고 Astra Control Service에 추가할 준비가 된 것입니다.

자가 관리형 클러스터는 직접 프로비저닝하고 관리하는 클러스터입니다. Astra Control Service는 퍼블릭 클라우드 환경에서 실행되는 자체 관리형 클러스터를 지원합니다. A를 업로드하여 자가 관리 클러스터를 Astra Control Service에 추가할 수 있습니다 `kubeconfig.yaml` 파일. 클러스터가 여기에 설명된 요구 사항을 충족하는지 확인해야 합니다.

지원되는 **Kubernetes** 배포

Astra Control Service를 사용하여 다음과 같은 유형의 프라이빗, 자가 관리 클러스터를 관리할 수 있습니다.

Kubernetes 배포	지원되는 버전
Kubernetes(업스트림)	1.27 ~ 1.29
RKE(Rancher Kubernetes Engine)	RKE 1: Rancher Manager 2.7.9 포함 버전 1.24.17, 1.25.13, 1.26.8 RKE 2: Rancher Manager 2.6.13이 있는 버전 1.23.16 및 1.24.13 RKE 2: Rancher Manager 2.7.9 포함 버전 1.24.17, 1.25.14, 1.26.9
Red Hat OpenShift Container Platform	4.12에서 4.14까지

이 지침은 이미 프라이빗 클러스터를 생성하고 원격 액세스를 위한 보안 방법을 준비했다고 가정합니다.

Astra Control Service에 프라이빗 클러스터를 추가하려면 다음 작업을 수행해야 합니다.

1. Astra Connector를 설치합니다
2. 영구 스토리지를 설정합니다

3. 프라이빗 자체 관리 클러스터를 Astra Control Service에 추가합니다

Astra Connector를 설치합니다

프라이빗 클러스터를 추가하기 전에 Astra Control이 클러스터와 통신할 수 있도록 클러스터에 Astra Connector를 설치해야 합니다. 을 참조하십시오 "Kubernetes가 아닌 네이티브 워크플로우로 관리되는 프라이빗 클러스터용 Astra Connector의 이전 버전을 설치합니다" 를 참조하십시오.

영구 스토리지를 설정합니다

클러스터의 영구 스토리지를 구성합니다. 영구 스토리지 구성에 대한 자세한 내용은 시작 설명서를 참조하십시오.

- "Azure NetApp Files를 사용하여 Microsoft Azure를 설정합니다"
- "Azure 관리 디스크를 사용하여 Microsoft Azure를 설정합니다"
- "Amazon Web Services를 설정합니다"
- "Google Cloud를 설정합니다"

프라이빗 자체 관리 클러스터를 **Astra Control Service**에 추가합니다

이제 Astra Control Service에 프라이빗 클러스터를 추가할 수 있습니다.

자가 관리형 클러스터는 직접 프로비저닝하고 관리하는 클러스터입니다. Astra Control Service는 퍼블릭 클라우드 환경에서 실행되는 자체 관리형 클러스터를 지원합니다. 자체 관리형 클러스터는 Astra Control Provisioner를 사용하여 NetApp 스토리지 서비스와 상호 연결하거나 컨테이너 스토리지 인터페이스(CSI) 드라이버를 사용하여 Amazon EBS(Elastic Block Store), Azure Managed Disks 및 Google Persistent Disk와 상호 작용할 수 있습니다.

Astra Control Service는 다음과 같은 Kubernetes 배포를 사용하는 자체 관리 클러스터를 지원합니다.

- Red Hat OpenShift Container Platform
- Rancher Kubernetes 엔진
- 업스트림 Kubernetes

자가 관리형 클러스터는 다음 요구사항을 충족해야 합니다.

- 클러스터를 인터넷을 통해 액세스할 수 있어야 합니다.
- CSI 드라이버로 활성화된 스토리지를 사용 중이거나 사용할 계획이면 해당 CSI 드라이버가 클러스터에 설치되어 있어야 합니다. CSI 드라이버를 사용하여 스토리지를 통합하는 방법에 대한 자세한 내용은 스토리지 서비스 설명서를 참조하십시오.
- 하나의 컨텍스트 요소만 포함된 클러스터 kubecononfig 파일에 액세스할 수 있습니다. 를 따릅니다 "참조하십시오" kubecononfig 파일을 생성합니다.
- 개인 CA(인증 기관)를 참조하는 kubeconfig 파일을 사용하여 클러스터를 추가하는 경우 에 다음 줄을 추가합니다 `cluster` kubeconfig 파일의 섹션. 이를 통해 Astra Control이 클러스터를 추가할 수 있습니다.

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

- * Rancher 전용 *: Rancher 환경에서 애플리케이션 클러스터를 관리할 때 Rancher가 제공하는 kubecon무화과 파일에서 애플리케이션 클러스터의 기본 컨텍스트를 수정하여 Rancher API 서버 컨텍스트 대신 컨트롤 플레인 컨텍스트를 사용합니다. 따라서 Rancher API 서버의 부하가 줄어들고 성능이 향상됩니다.
- * Astra Control Provisioner 요구 사항 *: 클러스터를 관리하려면 Astra Trident 구성 요소를 포함하여 올바르게 구성된 Astra Control Provisioner가 있어야 합니다.
  - * Astra Trident 환경 요구 사항 검토 *: Astra Control Provisioner를 설치 또는 업그레이드하기 전에 를 검토하십시오 "지원되는 프런트엔드, 백엔드 및 호스트 구성".
  - * Astra Control Provisioner 기능 활성화 *: Astra Trident 23.10 이상을 설치하고 활성화하는 것이 좋습니다 "Astra Control Provisioner 고급 스토리지 기능". 향후 릴리즈에서 Astra Control Provisioner가 활성화되어 있지 않으면 Astra Control이 Astra Trident를 지원하지 않습니다.
  - * 스토리지 백엔드 구성 *: 최소 하나의 스토리지 백엔드가 있어야 합니다 "Astra Trident에서 구성됨" 클러스터에서.
  - * 스토리지 클래스 구성 *: 최소 하나의 스토리지 클래스가 있어야 합니다 "Astra Trident에서 구성됨" 클러스터에서. 기본 저장소 클래스가 구성된 경우 기본 주석이 있는 * 전용 * 저장소 클래스인지 확인합니다.
  - * 볼륨 스냅샷 컨트롤러 구성 및 볼륨 스냅샷 클래스 설치 *: "볼륨 스냅샷 컨트롤러를 설치합니다" 따라서 Astra Control에서 스냅샷을 생성할 수 있습니다. "생성" 하나 이상 `VolumeSnapshotClass` Astra

Trident 사용:

단계

1. 대시보드에서 * Kubernetes 클러스터 관리 * 를 선택합니다.

   표시되는 메시지에 따라 클러스터를 추가합니다.

2. * 공급자 *: * 기타 * 탭을 선택하여 자체 관리 클러스터에 대한 세부 정보를 추가합니다.
3. * 기타 *: 을 업로드하여 자체 관리되는 클러스터에 대한 세부 정보를 제공합니다 `kubeconfig.yaml` 파일을 클릭하거나 의 내용을 붙여 넣습니다 `kubeconfig.yaml` 파일을 클립보드에 저장합니다.

   직접 만드는 경우 `kubeconfig` 파일에서 * 하나의 * 컨텍스트 요소만 정의해야 합니다. 을 참조하십시오 "참조하십시오" 을 참조하십시오 `kubeconfig` 파일.

4. * 자격 증명 이름 *: Astra Control에 업로드하는 자가 관리 클러스터 자격 증명의 이름을 입력합니다. 기본적으로 자격 증명 이름은 클러스터 이름으로 자동 채워집니다.
5. * Private route identifier *: Astra Connector로부터 얻을 수 있는 private route identifier를 입력합니다. 을 통해 Astra Connector에 문의하면 `kubectl get astraconnector -n astra-connector` 명령, 전용 라우트 식별자를 라고 합니다 `ASTRACONNECTORID`.

   프라이빗 경로 식별자는 Astra Connector와 연결된 이름으로, 프라이빗 Kubernetes 클러스터를 Astra에서 관리할 수 있도록 합니다. 이런 맥락에서 프라이빗 클러스터는 API 서버를 인터넷에 노출하지 않는 Kubernetes 클러스터입니다.

6. 다음 * 을 선택합니다.
7. (선택 사항) * 스토리지 *: 선택적으로 이 클러스터에 Kubernetes 애플리케이션을 배포할 스토리지 클래스를 선택하여 기본적으로 사용하도록 합니다.
   a. 클러스터에 대한 새 기본 스토리지 클래스를 선택하려면 * 새 기본 스토리지 클래스 할당 * 확인란을 설정합니다.
   b. 목록에서 새 기본 스토리지 클래스를 선택합니다.

      각 클라우드 공급자의 스토리지 서비스에는 다음과 같은 가격, 성능 및 복원력 정보가 표시됩니다.

      - Google Cloud용 Cloud Volumes Service: 가격, 성능 및 복원력 정보
      - Google 영구 디스크: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
      - Azure NetApp Files: 성능 및 복원력 정보
      - Azure 관리 디스크: 사용 가능한 가격, 성능 또는 복원력 정보가 없습니다
      - Amazon Elastic Block Store: 가격, 성능 또는 복원력 정보를 사용할 수 없습니다
      - NetApp ONTAP용 Amazon FSx: 가격, 성능 또는 복원력 정보 없음
      - NetApp Cloud Volumes ONTAP: 가격, 성능 또는 복원력 정보를 제공할 수 없습니다

      각 스토리지 클래스는 다음 서비스 중 하나를 활용할 수 있습니다.

- "Google Cloud용 Cloud Volumes Service"
- "Google 영구 디스크"
- "Azure NetApp Files"
- "Azure로 관리되는 디스크"
- "Amazon Elastic Block Store를 클릭합니다"
- "NetApp ONTAP용 Amazon FSx"
- "NetApp Cloud Volumes ONTAP를 참조하십시오"

  에 대해 자세히 알아보십시오 "Amazon Web Services 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "AKS 클러스터용 스토리지 클래스입니다". 에 대해 자세히 알아보십시오 "GKE 클러스터용 저장소 클래스".

c. 다음 * 을 선택합니다.

d. * 검토 및 승인 *: 구성 세부 정보를 검토합니다.

e. 클러스터를 Astra Control Service에 추가하려면 * 추가 * 를 선택합니다.

기본 스토리지 클래스를 변경합니다

클러스터의 기본 스토리지 클래스를 변경할 수 있습니다.

Astra Control을 사용하여 기본 스토리지 클래스를 변경합니다

Astra Control 내에서 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 클러스터에서 이전에 설치된 스토리지 백엔드 서비스를 사용하는 경우 이 방법을 사용하여 기본 스토리지 클래스를 변경하지 못할 수 있습니다(* 기본값으로 설정* 작업은 선택할 수 없음). 이 경우 를 사용할 수 있습니다 명령줄을 사용하여 기본 스토리지 클래스를 변경합니다.

단계

1. Astra Control Service UI에서 * Clusters * 를 선택합니다.
2. 클러스터 * 페이지에서 변경할 클러스터를 선택합니다.
3. Storage * 탭을 선택합니다.
4. 스토리지 클래스 * 범주를 선택합니다.
5. 기본값으로 설정할 스토리지 클래스에 대해 * Actions * 메뉴를 선택합니다.
6. Set as default * 를 선택합니다.

명령줄을 사용하여 기본 스토리지 클래스를 변경합니다

Kubernetes 명령을 사용하여 클러스터의 기본 스토리지 클래스를 변경할 수 있습니다. 이 방법은 클러스터의 구성에 관계없이 작동합니다.

단계

1. Kubernetes 클러스터에 로그인합니다.
2. 클러스터의 스토리지 클래스를 나열합니다.

```
kubectl get storageclass
```

3. 기본 스토리지 클래스에서 기본 지정을 제거합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-
class":"false"}}}'
```

4. 다른 스토리지 클래스를 기본값으로 표시합니다. <SC_NAME>를 스토리지 클래스 이름으로 바꿉니다.

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 새 기본 스토리지 클래스를 확인합니다.

```
kubectl get storageclass
```

Astra Trident 버전을 확인합니다

스토리지 서비스에 Astra Control Provisioner 또는 Astra Trident를 사용하는 자체 관리형 클러스터를 추가하려면 Astra Trident의 설치된 버전이 23.10 이상이어야 합니다.

단계

1. 실행 중인 Astra Trident 버전을 확인합니다.

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

Astra Trident가 설치된 경우 다음과 유사한 출력이 표시됩니다.

```
NAME      VERSION
trident   24.02.0
```

Astra Trident가 설치되지 않은 경우 다음과 유사한 출력이 표시됩니다.

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

2. 다음 중 하나를 수행합니다.
   - Astra Trident 23.01 이하를 실행 중인 경우 다음을 사용합니다 "지침" Astra Control Provisioner로

업그레이드하기 전에 Astra Trident의 최신 버전으로 업그레이드하십시오. 가능합니다 "직접 업그레이드를 수행합니다" Astra Trident가 버전 24.02의 4개 릴리즈 윈도우 내에 있는 경우 Astra Control Provisioner 24.02에 등록됩니다. 예를 들어, Astra Trident 23.04에서 Astra Control Provisioner 24.02로 직접 업그레이드할 수 있습니다.

  - Astra Trident 23.10 이상을 실행 중인 경우 Astra Control Provisioner가 설치되었는지 확인합니다 "활성화됨". Astra Control Provisioner는 23.10 이전 Astra Control Center 릴리즈에서 작동하지 않습니다. "Astra Control Provisioner를 업그레이드합니다" 최신 기능에 액세스하기 위해 업그레이드하는 Astra Control Center와 동일한 버전을 사용합니다.

3. Pod가 실행 중인지 확인합니다.

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

4. 스토리지 클래스가 지원되는 Astra Trident 드라이버를 사용하고 있는지 확인합니다. 공급자 이름은 이어야 합니다 `csi.trident.netapp.io`. 다음 예를 참조하십시오.

```
kubectl get sc
```

샘플 반응:

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

kubecononfig 파일을 생성합니다

kubecononfig 파일을 사용하여 Astra Control Service에 클러스터를 추가할 수 있습니다. 추가하려는 클러스터 유형에 따라 특정 단계를 사용하여 클러스터에 대한 kubeconfig 파일을 수동으로 생성해야 할 수 있습니다.

- Amazon EKS 클러스터용 kubbeconfig 파일을 생성합니다
- AWS(ROSA) 클러스터에서 Red Hat OpenShift Service에 대한 kubeconfig 파일을 생성합니다
- 다른 유형의 클러스터에 사용할 kubecononfig 파일을 생성합니다

Amazon EKS 클러스터용 kubbeconfig 파일을 생성합니다

다음 지침에 따라 아마존 EKS 클러스터에 대한 kubbeconfig 파일과 영구 토큰 암호를 생성하십시오. EKS에서 호스팅되는 클러스터에 영구 토큰 암호가 필요합니다.

단계

1. 아마존 문서의 지침에 따라 kubecononfig 파일을 생성합니다.

   "Amazon EKS 클러스터에 대한 kubbeconfig 파일을 생성하거나 업데이트합니다"

2. 다음과 같이 서비스 계정을 생성합니다.

   a. 라는 서비스 계정 파일을 생성합니다 `astracontrol-service-account.yaml`.

      필요에 따라 서비스 계정 이름을 조정합니다. 네임스페이스 `kube-system` 은(는) 이러한 단계에 필요합니다. 여기서 서비스 계정 이름을 변경하는 경우 다음 단계에서 동일한 변경 사항을 적용해야 합니다.

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

+

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: astra-admin-account
  namespace: kube-system
```

3. 서비스 계정 적용:

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

4. 을 생성합니다 `ClusterRoleBinding` 파일을 호출했습니다 `astracontrol-clusterrolebinding.yaml`.

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

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: astra-admin-binding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: astra-admin-account
  namespace: kube-system
```

5. 클러스터 역할 바인딩을 적용합니다.

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

6. 라는 서비스 계정 토큰 암호 파일을 생성합니다 `astracontrol-secret.yaml`.

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

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: astra-admin-account
  name: astra-admin-account
  namespace: kube-system
type: kubernetes.io/service-account-token
```

7. 토큰 암호 적용:

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

8. 토큰 암호 검색:

```
kubectl get secret astra-admin-account -n kube-system -o
jsonpath='{.data.token}' | base64 -d
```

9. 를 교체합니다 `user` 다음 예제에서와 같이 토큰이 있는 AWS EKS kubecononfig 파일의 섹션입니다.

```
user:
    token: k8s-aws-
v1.aHR0cHM6Ly9zdHMudXMtd2VzdC0yLmFtYXpvbmF3cy5jb20vP0FjdGlvbj1HZXRDYWxsZ
XJJZGVudGl0eSZWZXJzaW9uPTIwMTEtMDYtMTUmWC1BbXotQWxnb3JpdGhtPUFXUzQtSE1BQ
y1TSEEyNTYmWC1BbXotQ3JlZGVudGlhbD1BS0lBM1JEWDdKU0haWU9LSEQ2SyUyRjIwMjMwN
DAzJTJGdXMtd2VzdC0yJTJGc3RzJTJGYXdzNF9yZXF1ZXN0JlgtQW16LURhdGU9MjAyMzA0M
DNUMjA0MzQwWiZYLUFtei1FeHBpcmVzPTYwJlgtQW16LVNpZ25lZEhlYWRlcnM9aG9zdCUzQ
ngtazhzLWF3cy1pZCZYLUFtei1TaWduYXR1cmU9YjU4ZWM0NzdiM2NkZGYxNGRhNzU4MGI2Z
WQ2zY2NzI2YWIwM2UyNThjMjRhNTJjNmVhNjc4MTRlNjJkOTg2Mg
```

AWS(ROSA) 클러스터에서 **Red Hat OpenShift Service**에 대한 **kubeconfig** 파일을 생성합니다

다음 지침에 따라 AWS(ROSA) 클러스터에서 Red Hat OpenShift Service에 대한 kubeconfig 파일을 생성합니다.

단계

1. Rosa 클러스터에 로그인합니다.
2. 서비스 계정 생성:

```
oc create sa astracontrol-service-account
```

3. 클러스터 역할 추가:

```
oc adm policy add-cluster-role-to-user cluster-admin -z astracontrol-service-account
```

4. 다음 예제를 사용하여 서비스 계정 암호 구성 파일을 만듭니다.

```
<strong>secret-astra-sa.yaml</strong>
```

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

5. 비밀 만들기:

```
oc create -f secret-astra-sa.yaml
```

6. 생성한 서비스 계정을 편집하고 Astra Control 서비스 계정 암호 이름을 에 추가합니다 `secrets` 섹션:

```
oc edit sa astracontrol-service-account
```

```
apiVersion: v1
imagePullSecrets:
- name: astracontrol-service-account-dockercfg-dvfcd
kind: ServiceAccount
metadata:
  creationTimestamp: "2023-08-04T04:18:30Z"
  name: astracontrol-service-account
  namespace: default
  resourceVersion: "169770"
  uid: 965fa151-923f-4fbd-9289-30cad15998ac
secrets:
- name: astracontrol-service-account-dockercfg-dvfcd
- name: secret-astracontrol-service-account ####ADD THIS ONLY####
```

7. 교체 서비스 계정 암호를 나열합니다 `<CONTEXT>` 올바른 설치 상황:

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

출력의 끝은 다음과 유사합니다.

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

의 각 요소에 대한 인덱스입니다 `secrets` 어레이는 0으로 시작합니다. 위의 예에서 의 인덱스입니다 `astracontrol-service-account-dockercfg-dvfcd` 는 0이고 의 인덱스입니다 `secret-astracontrol-service-account` 1입니다. 출력에서 서비스 계정의 인덱스 번호를 기록해 둡니다. 다음 단계에서는 이 인덱스 번호가 필요합니다.

8. 다음과 같이 kubecononfig를 생성합니다.
   a. 을 생성합니다 `create-kubeconfig.sh` 파일. 대치 `TOKEN_INDEX` 다음 스크립트의 시작 부분에 올바른 값이 있습니다.

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

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

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

CONTEXT=$(kubectl config current-context)

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

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

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

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

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

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

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

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

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

```
  set-context ${NEW_CONTEXT} --namespace ${NAMESPACE}

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

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

b. Kubernetes 클러스터에 적용할 명령을 소스 하십시오.

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

9. (선택 사항) kubeconfig의 이름을 클러스터의 의미 있는 이름으로 바꿉니다.

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

다른 유형의 클러스터에 사용할 **kubecononfig** 파일을 생성합니다

다음 지침에 따라 Rancher, Upstream Kubernetes 및 Red Hat OpenShift 클러스터에 대한 제한적이거나 확장된 역할 kubeconfig 파일을 생성합니다.

kubeconfig를 사용하여 관리되는 클러스터의 경우 Astra Control Service에 대해 제한된 권한 또는 확장된 권한 관리자 역할을 선택적으로 생성할 수 있습니다.

다음 시나리오 중 하나가 사용자 환경에 적용되는 경우 이 절차를 통해 별도의 kubecononfig를 생성할 수 있습니다.

- 관리하는 클러스터에 대한 Astra Control 권한을 제한하려고 합니다
- 여러 개의 컨텍스트를 사용하며 설치 중에 구성된 기본 Astra Control kubecononfig를 사용할 수 없거나, 단일 컨텍스트의 제한된 역할은 사용자 환경에서 작동하지 않습니다

시작하기 전에

절차 단계를 완료하기 전에 관리하려는 클러스터에 대해 다음 사항을 확인해야 합니다.

- A "지원되는 버전입니다" kubectl이 설치되어 있습니다.
- Astra Control Service를 사용하여 추가하고 관리하려는 클러스터에 대한 kubectl 액세스

  이 절차를 수행하려면 Astra Control Service를 실행하는 클러스터에 액세스할 필요가 없습니다.

- 활성 컨텍스트에 대한 클러스터 관리자 권한으로 관리하려는 클러스터에 대한 활성 kubecononfig입니다

단계

1. 서비스 계정 생성:

a. 라는 서비스 계정 파일을 생성합니다 `astracontrol-service-account.yaml`.

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

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

b. 서비스 계정 적용:

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

2. Astra Control에서 클러스터를 관리할 수 있는 충분한 권한을 가진 다음 클러스터 역할 중 하나를 생성합니다.

제한된 클러스터 역할

이 역할에는 Astra Control에서 관리할 클러스터를 관리하는 데 필요한 최소 권한이 포함되어 있습니다.

a. 을 생성합니다 `ClusterRole` 호출되는 파일(예: `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 클러스터에만 해당) 의 끝에 다음을 추가합니다 `astra-admin-account.yaml` 파일:

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

c. 클러스터 역할 적용:

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

클러스터 역할이 확장되었습니다

이 역할에는 Astra Control에서 관리할 클러스터에 대한 확장된 권한이 포함됩니다. 여러 컨텍스트를 사용하고 설치 중에 구성된 기본 Astra Control kubecononfig를 사용할 수 없거나 단일 컨텍스트의 제한된 역할을 사용할 수 없는 경우 이 역할을 사용할 수 있습니다.

다음 사항을 참조하십시오 `ClusterRole` 일반 Kubernetes의 예는 단계입니다. 사용자 환경에 대한 지침은 Kubernetes 배포 문서를 참조하십시오.

a. 을 생성합니다 `ClusterRole` 호출되는 파일(예: `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. 클러스터 역할 적용:

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

3. 클러스터 역할에 대한 클러스터 역할 바인딩을 서비스 계정에 생성합니다.

a. 을 생성합니다 `ClusterRoleBinding` 파일을 호출했습니다 `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. 클러스터 역할 바인딩을 적용합니다.

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

4. 토큰 암호 생성 및 적용:
   a. 라는 토큰 비밀 파일을 만듭니다 `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. 토큰 암호 적용:

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

5. 토큰 암호를 에 추가하여 서비스 계정에 추가합니다 `secrets` 배열(다음 예제의 마지막 줄):

```
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. 교체 서비스 계정 암호를 나열합니다 `<context>` 올바른 설치 상황:

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

출력의 끝은 다음과 유사합니다.

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

의 각 요소에 대한 인덱스입니다 `secrets` 어레이는 0으로 시작합니다. 위의 예에서 의 인덱스입니다 `astracontrol-service-account-dockercfg-48xhx` 는 0이고 의 인덱스입니다 `secret-astracontrol-service-account` 1입니다. 출력에서 서비스 계정의 인덱스 번호를 기록해 둡니다. 다음 단계에서는 이 인덱스 번호가 필요합니다.

7. 다음과 같이 kubecononfig를 생성합니다.
   a. 을 생성합니다 `create-kubeconfig.sh` 파일.
   b. 대치 `TOKEN_INDEX` 다음 스크립트의 시작 부분에 올바른 값이 있습니다.

```
<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. Kubernetes 클러스터에 적용할 명령을 소스 하십시오.

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

8. (선택 사항) kubeconfig의 이름을 클러스터의 의미 있는 이름으로 바꿉니다.

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

다음 단계

이제 Astra Control에 로그인하여 클러스터를 추가하면 Astra Control의 애플리케이션 데이터 관리 기능을 사용할 수 있습니다.

- "앱 관리를 시작합니다"
- "앱 보호"
- "앱 클론 복제"
- "대금 청구를 설정합니다"
- "사용자를 초대하고 관리합니다"
- "클라우드 공급자 자격 증명을 관리합니다"
- "알림을 관리합니다"
- "Astra Control의 자체 관리형 인스턴스를 구축"

Astra Control Service 비디오

Astra Control Service의 최신 비디오 콘텐츠를 보려면 NetApp TV를 확인하십시오. NetApp

TV에는 Astra Control Service의 특정 기능을 시연하거나 특정 일반 작업을 완료하는 방법을 보여 주는 비디오가 포함되어 있습니다.

"Astra Control Service 비디오"

개념

아키텍처 및 구성 요소

Astra Control은 상태 저장 애플리케이션의 작업을 간소화하고 하이브리드 및 멀티 클라우드 환경에서 Kubernetes 워크로드를 저장, 보호, 이동하는 데 도움이 되는 Kubernetes 애플리케이션 데이터 라이프사이클 관리 솔루션입니다.

제공합니다

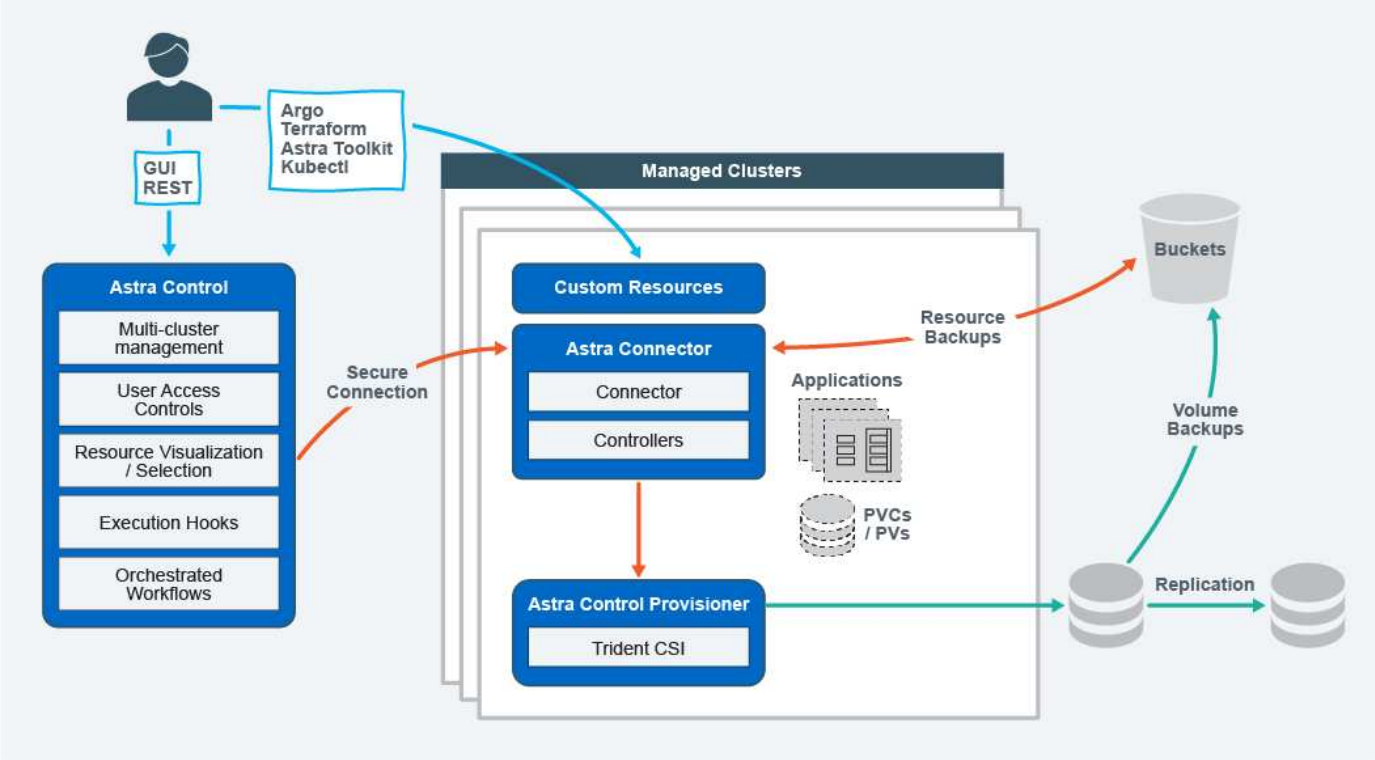
Astra Control은 Kubernetes 애플리케이션 데이터 라이프사이클 관리에 중요한 기능을 제공합니다.

- 매장 *:
- 컨테이너식 워크로드를 위한 동적 스토리지 프로비저닝
- 컨테이너에서 영구 볼륨으로 전송 중인 데이터를 암호화
- 교차 지역, 교차 영역 복제
- 보호 *:
- 전체 애플리케이션 및 해당 데이터의 자동화된 검색 및 애플리케이션 인식 보호
- 조직의 요구에 따라 스냅샷 버전에서 즉시 응용 프로그램 복구
- 여러 영역, 지역 및 클라우드 공급자 간에 신속하게 페일오버 수행
- 이동 *:
- Kubernetes 클러스터와 클라우드 내부 및 클라우드 간에 완벽한 애플리케이션 및 데이터 이동성
- 전체 애플리케이션 및 데이터의 즉각적인 클론 복제
- 일관된 웹 UI 및 API를 통해 한 번의 클릭으로 애플리케이션 마이그레이션

있습니다

Astra Control의 아키텍처는 IT 부서가 Kubernetes 애플리케이션의 기능과 가용성을 향상하고 퍼블릭 클라우드와 온프레미스 환경 전반에서 컨테이너식 워크로드의 관리, 보호 및 이동을 간소화하는 고급 데이터 관리 기능을 제공할 수 있도록 지원합니다. 또한 REST API 및 SDK를 통해 자동화 기능을 제공하므로 프로그래밍 방식으로 액세스하여 기존 워크플로우와 원활하게 통합할 수 있습니다.

Astra Control은 Kubernetes 네이티브로, 사용자 지정 리소스를 활용하는 데이터 보호 워크플로우를 지원하면서 기존 API 및 SDK와 역호환성을 유지할 수 있습니다. Kubernetes 네이티브 데이터 보호는 중요한 이점을 제공합니다. Kubernetes API 및 리소스와 원활하게 통합되어 조직의 기존 CI/CD 및/또는 GitOps 툴을 통해 데이터 보호를 애플리케이션 라이프사이클의 고유한 부분으로 활용할 수 있습니다.



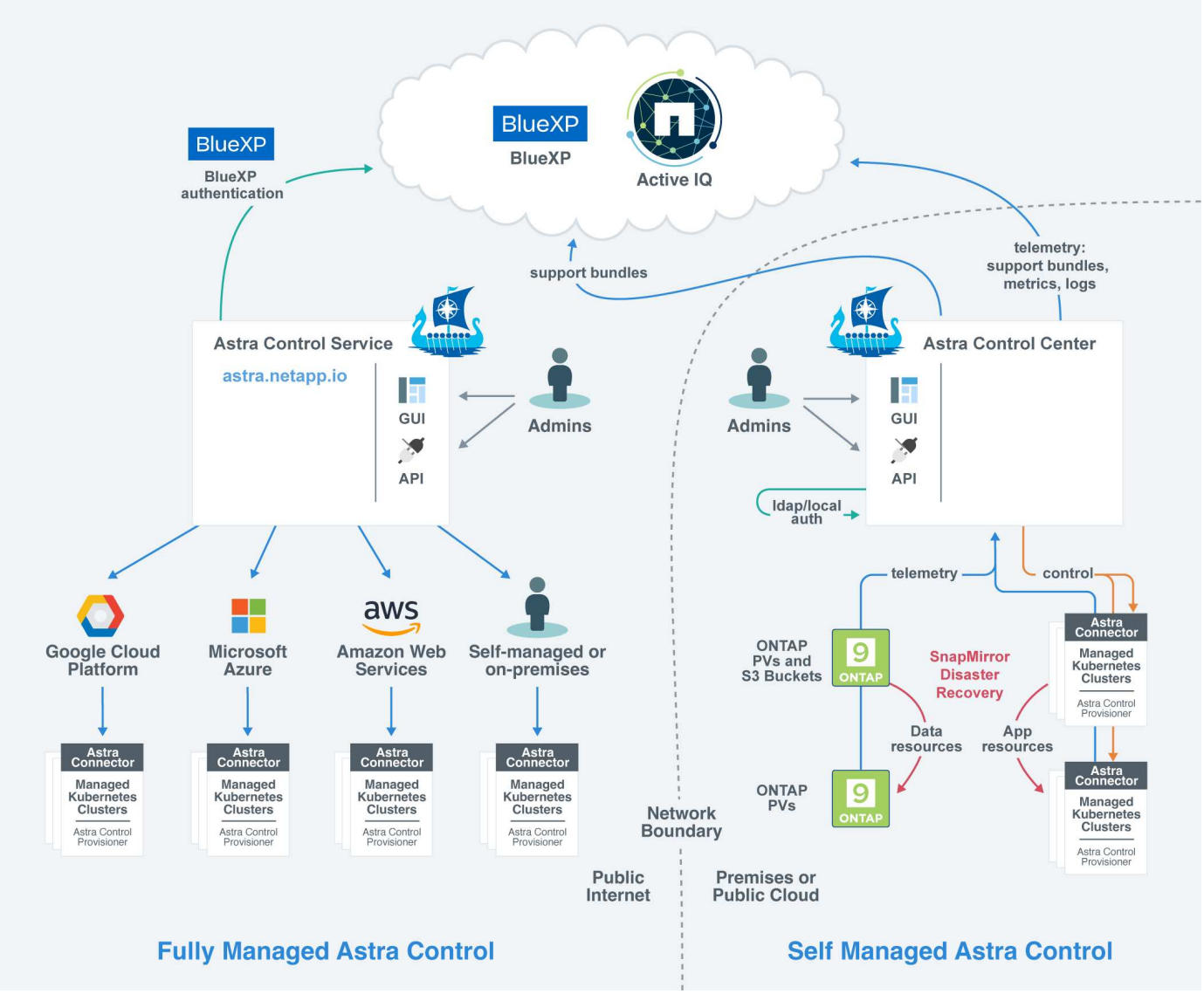
Astra Control은 다음과 같은 4가지 보완 구성 요소를 기반으로 구축되었습니다.

- * Astra Control *: Astra Control은 모든 관리형 클러스터를 위한 중앙 집중식 관리 서비스로, 클라우드 및 온프레미스에서의 애플리케이션 보호 및 이동성을 위해 오케스트레이션된 워크로드와 다음과 같은 기능을 제공합니다.
  - 여러 클러스터와 클라우드의 통합 보기
  - 오케스트레이션된 워크플로 보호
  - 세분화된 리소스 시각화 및 선택
- * Astra Connector *: Astra Connector는 Astra Control과 협력하여 각 관리형 클러스터에 대한 보안 연결을 제공하여 연결 상태와 상관 없이 예약된 작업을 로컬에서 실행할 수 있도록 지원합니다.
  - 연결 상태에 관계없이 예약된 작업을 로컬로 실행합니다
  - 클러스터 간에 Astra의 시스템 리소스 사용을 배포하고 최적화하는 로컬 운영
  - 보안을 강화하기 위해 클러스터에 대한 최소 권한 액세스를 허용하는 로컬 설치
- * Astra Control Provisioner *: Astra Control Provisioner는 핵심 CSI 프로비저닝 기능과 고급 스토리지 관리 기능을 제공하여 보안 및 재해 복구 구성을 강화하고 다음과 같은 기능을 제공합니다.
  - 컨테이너식 워크로드를 위한 동적 스토리지 프로비저닝
  - 고급 스토리지 관리:
    - 컨테이너에서 PV로 전송 중인 데이터 암호화
    - SnapMirror Cloud 기능: 지역 간, 교차 영역 복제
- * Astra Custom 리소스 *: 각 클러스터에 사용되는 맞춤형 리소스는 Kubernetes 네이티브의 방식으로 로컬에서 실행할 수 있으므로 Kubernetes 네이티브의 다른 툴링 및 자동화와의 통합을 단순화하고 다음과 같은 기능을 제공합니다.
  - 직접 에코시스템 툴 통합 및 자동화 워크플로우

◦ 사용자 지정 워크플로를 지원하는 하위 수준의 기본 형식

구축 모델

Astra Control은 2가지 구축 모델로 제공됩니다.



- * Astra Control Service *: NetApp 관리 서비스로, 여러 클라우드 공급자 환경 및 셀프 관리 Kubernetes 클러스터에서 Kubernetes 클러스터의 애플리케이션 인식 데이터 관리를 제공합니다.

  "Astra Control Service 문서"

- * Astra Control Center *: 사내 환경에서 실행되는 Kubernetes 클러스터의 애플리케이션 인식 데이터 관리를 제공하는 자체 관리 소프트웨어입니다. 또한 NetApp Cloud Volumes ONTAP 스토리지 백엔드를 통해 여러 클라우드 공급자 환경에 Astra Control Center를 설치할 수 있습니다.

  "Astra Control Center 문서"

	Astra 제어 서비스	**Astra** 제어 센터
어떻게 제공됩니까**?**	NetApp에서 제공하는 완전 관리형 클라우드 서비스	소프트웨어를 다운로드, 설치 및 관리할 수 있습니다
어디에 호스팅됩니까**?**	NetApp에서 제공하는 다양한 퍼블릭 클라우드 지원	고유한 Kubernetes 클러스터
어떻게 업데이트됩니까**?**	NetApp에서 관리합니다	모든 업데이트를 관리합니다
지원되는 **Kubernetes** 배포는 무엇입니까**?**	• * 클라우드 공급자 * ◦ Amazon Web Services에서 직접 지원합니다 ▪ Amazon Elastic Kubernetes Service(EKS) ◦ Google 클라우드 ▪ Google Kubernetes Engine(GKE) ◦ Microsoft Azure를 참조하십시오 ▪ Azure Kubernetes 서비스(AKS) • * 자가 관리형 클러스터 * ◦ Kubernetes(업스트림) ◦ RKE(Rancher Kubernetes Engine) ◦ Red Hat OpenShift Container Platform • * 온프레미스 클러스터 * ◦ Red Hat OpenShift Container Platform 온프레미스	• Azure Stack HCI 기반 Azure Kubernetes Service • Google Anthos • Kubernetes(업스트림) • RKE(Rancher Kubernetes Engine) • Red Hat OpenShift Container Platform

	Astra 제어 서비스	Astra 제어 센터
지원되는 스토리지 백엔드는 무엇입니까?	• * 클라우드 공급자 * ◦ Amazon Web Services에서 직접 지원합니다 ▪ Amazon EBS ▪ NetApp ONTAP용 Amazon FSx ▪ "Cloud Volumes ONTAP" ◦ Google 클라우드 ▪ Google 영구 디스크 ▪ NetApp Cloud Volumes Service를 참조하십시오 ▪ "Cloud Volumes ONTAP" ◦ Microsoft Azure를 참조하십시오 ▪ Azure 관리 디스크 ▪ Azure NetApp Files ▪ "Cloud Volumes ONTAP" • * 자가 관리형 클러스터 * ◦ Amazon EBS ◦ Azure 관리 디스크 ◦ Google 영구 디스크 ◦ "Cloud Volumes ONTAP" ◦ NetApp MetroCluster ◦ "롱혼" • * 온프레미스 클러스터 * ◦ NetApp MetroCluster ◦ NetApp ONTAP AFF 및 FAS 시스템 ◦ NetApp ONTAP Select를 참조하십시오 ◦ "Cloud Volumes ONTAP" ◦ "롱혼"	• NetApp ONTAP AFF 및 FAS 시스템 • NetApp ONTAP Select를 참조하십시오 • "Cloud Volumes ONTAP" • "롱혼"

를 참조하십시오

- "Astra Control Service 문서"
- "Astra Control Center 문서"
- "Astra Trident 문서"
- "Astra Control API를 참조하십시오"

- "Cloud Insights 설명서"
- "ONTAP 설명서"

데이터 보호

Astra Control Service에서 사용 가능한 데이터 보호 유형과 이를 사용하여 앱을 보호하는 최선의 방법에 대해 알아보십시오.

스냅샷, 백업 및 보호 정책

스냅샷과 백업 모두 다음과 같은 유형의 데이터를 보호합니다.

- 애플리케이션 자체입니다
- 애플리케이션과 연결된 모든 영구 데이터 볼륨
- 응용 프로그램에 속하는 모든 리소스 아티팩트가 있습니다

snapshot_은 앱과 동일한 프로비저닝된 볼륨에 저장된 앱의 시점 복사본입니다. 대개 빠릅니다. 로컬 스냅샷을 사용하여 애플리케이션을 이전 시점으로 복원할 수 있습니다. 스냅샷은 빠른 클론에 유용합니다. 스냅샷에는 구성 파일을 포함하여 앱의 모든 Kubernetes 객체가 포함됩니다. 스냅샷은 동일한 클러스터 내에서 앱을 클론 생성하거나 복원하는 데 유용합니다.

백업 _ 은(는) 스냅샷을 기반으로 합니다. 외부 오브젝트 저장소에 저장되며, 이로 인해 로컬 스냅샷과 비교하여 촬영 속도가 느려질 수 있습니다. 앱 백업을 동일한 클러스터에 복원하거나 백업을 다른 클러스터에 복원하여 앱을 마이그레이션할 수 있습니다. 또한 백업의 보존 기간을 더 길게 선택할 수도 있습니다. 이러한 백업은 외부 개체 저장소에 저장되므로 일반적으로 서버 장애 또는 데이터 손실 시 스냅샷보다 더 뛰어난 보호 기능을 제공합니다.

보호 정책 _ 은(는) 해당 앱에 대해 정의한 일정에 따라 스냅샷, 백업 또는 둘 모두를 자동으로 생성하여 앱을 보호하는 방법입니다. 또한 보호 정책을 사용하여 스케줄에 보관할 스냅샷과 백업의 수를 선택하고 스케줄 세분화 수준을 다르게 설정할 수 있습니다. 보호 정책을 통해 백업 및 스냅샷을 자동화하는 것이 각 애플리케이션이 조직의 요구 사항과 SLA(서비스 수준 계약) 요구 사항에 따라 보호되도록 하는 가장 좋은 방법입니다.

_최근 백업 _ 이(가) 있을 때까지 완전히 보호할 수 없습니다. 백업은 영구 볼륨으로부터 멀리 떨어진 개체 저장소에 저장되기 때문에 이 작업이 중요합니다. 장애 또는 사고로 인해 클러스터와 관련 영구 스토리지가 삭제되면 복구할 백업이 필요합니다. 스냅샷을 사용하면 복구할 수 없습니다.

스냅샷 또는 백업을 수행하지만 "내부 서버 문제로 인해 리소스가 생성되지 않았습니다"라는 오류와 함께 작업이 실패하는 경우 사용 중인 스토리지 백엔드에 올바른 드라이버가 설치되어 있는지 확인하십시오. 일부 스토리지 백엔드는 CSI(Container Storage Interface) 드라이버를 필요로 하지만, 다른 스토리지 백엔드는 외부 스냅샷 컨트롤러를 필요로 합니다.

변경 불가능한 백업

변경 불가능한 백업은 지정된 기간 동안 변경하거나 삭제할 수 없는 백업입니다. 변경 불가능한 백업을 생성할 때 Astra Control은 사용 중인 버킷이 WORM(Write Once, Read Many) 버킷인지 확인하고, 그럴 경우 Astra Control에서 백업을 변경 가능한지 확인합니다.
Astra Control Service는 다음 플랫폼 및 버킷 유형에 따라 변경 불가능한 백업 생성을 지원합니다.

- S3 오브젝트 잠금이 구성된 Amazon S3 버킷을 사용하는 Amazon Web Services

- 보존 정책이 구성된 Azure 버킷을 사용하는 Microsoft Azure
- 보존 정책이 구성된 Google Cloud 스토리지 버킷을 사용하는 GKE(Google Kubernetes Engine
- S3 오브젝트 잠금이 구성된 S3 버킷을 사용하는 NetApp StorageGRID

변경 불가능한 백업 작업 시 다음 사항에 유의하십시오.

- 지원되지 않는 플랫폼에서 WORM 버킷에 백업하거나 지원되지 않는 버킷 유형에 백업하는 경우 보존 시간이 경과해도 백업 삭제 실패와 같은 예상치 못한 결과가 발생할 수 있습니다.
- Astra Control은 데이터 라이프사이클 관리 정책 또는 변경 불가능한 백업으로 사용하는 버킷의 오브젝트 수동 삭제를 지원하지 않습니다. 스토리지 백엔드가 Astra Control 스냅샷 또는 백업 데이터의 라이프사이클을 관리하도록 구성되지 않았는지 확인하십시오.

복제

clone_은 앱, 해당 구성 및 영구 데이터 볼륨의 정확한 복제입니다. 동일한 Kubernetes 클러스터 또는 다른 클러스터에 클론을 수동으로 생성할 수 있습니다. 애플리케이션 및 스토리지를 Kubernetes 클러스터 간에 이동해야 하는 경우 앱 클론을 생성하는 것이 유용할 수 있습니다.

AWS 클러스터를 위한 스토리지 클래스 및 성능

Astra Control Service는 Amazon EBS(Elastic Block Store), NetApp ONTAP용 Amazon FSx 또는 NetApp Cloud Volumes ONTAP를 EKS(Elastic Kubernetes Service) 클러스터의 스토리지 백엔드로 사용할 수 있습니다.

Amazon EBS(Elastic Block Store)

클러스터는 컨테이너 스토리지 인터페이스(CSI) 드라이버를 사용하여 EBS와 인터페이스할 수 있습니다. EBS를 EKS 클러스터의 스토리지 백엔드로 사용하는 경우 일부 스토리지 클래스 매개 변수를 구성할 수 있습니다. 매개 변수의 의미와 매개 변수를 구성하는 방법에 대한 자세한 내용은 을 참조하십시오 "Kubernetes 문서".

EBS에 여러 가지 유형의 볼륨을 사용할 수 있습니다.

- 솔리드 스테이트 드라이브(SSD)
- 하드 디스크 드라이브(HDD)
- 이전 세대

각 볼륨 유형 및 성능에 대한 자세한 내용은 을 참조하십시오 "Amazon EBS 문서". 가격 정보는 을 참조하십시오 "Amazon EBS 가격".

NetApp ONTAP용 Amazon FSx

NetApp ONTAP용 FSx를 AWS 클러스터의 스토리지 백엔드로 사용하는 경우, I/O 성능은 파일 시스템의 구성과 워크로드의 특성에 따라 달라집니다. NetApp ONTAP 성능을 위한 FSx에 대한 자세한 내용은 을 참조하십시오 "NetApp ONTAP 성능을 위한 Amazon FSx". 가격 정보는 을 참조하십시오 "NetApp ONTAP용 Amazon FSx 가격".

NetApp Cloud Volumes ONTAP를 참조하십시오

성능 권장사항을 비롯한 NetApp Cloud Volumes ONTAP 구성에 대한 자세한 내용은 를 참조하십시오 "NetApp Cloud Volumes ONTAP 문서".

AKS 클러스터의 스토리지 클래스 및 PV 크기입니다

Astra Control Service는 Azure Kubernetes Service(AKS) 클러스터를 위한 스토리지 백엔드로 Azure NetApp Files, Azure 관리 디스크 또는 NetApp Cloud Volumes ONTAP를 지원합니다.

Azure NetApp Files

Astra Control Service는 Azure Kubernetes Service(AKS) 클러스터를 위한 스토리지 백엔드로 Azure NetApp Files를 지원합니다. 스토리지 클래스 및 영구 볼륨 크기를 선택하면 성능 목표를 달성하는 데 어떤 도움이 되는지 알아야 합니다.

서비스 수준 및 스토리지 클래스

Azure NetApp Files는 Ultra 스토리지, Premium 스토리지 및 Standard 스토리지의 세 가지 서비스 수준을 지원합니다. 각 서비스 수준은 성능 요구 사항에 따라 다르게 설계되었습니다.

최고의 스토리지
: 1TiB당 최대 128MiB/s의 처리량을 제공합니다.

Premium 스토리지
: 1TiB당 최대 64MiB/s의 처리량을 제공합니다.

Standard 스토리지
: 1TiB당 최대 16개의 MIB/s 처리량을 제공합니다.

이러한 서비스 수준은 용량 풀의 특성입니다. Kubernetes 클러스터와 함께 사용할 각 서비스 수준에 대해 용량 풀을 설정해야 합니다. "용량 풀을 설정하는 방법에 대해 알아봅니다".

Astra Control Service는 이러한 서비스 수준을 영구 볼륨의 스토리지 클래스로 사용합니다. Kubernetes 클러스터를 Astra Control Service에 추가하면 기본 스토리지 클래스로 Ultra, Premium 또는 Standard를 선택하라는 메시지가 표시됩니다. 스토리지 클래스의 이름은 *NetApp-anf-perf-ultra*,*netapp-anf-perf-premium*,_netapp-anf-perf-standard_입니다.

"Azure NetApp Files 문서에서 이러한 서비스 수준에 대해 자세히 알아보십시오".

영구 볼륨 크기 및 성능

위에서 설명한 것처럼 각 서비스 수준의 처리량은 프로비저닝된 용량의 1TiB입니다. 다시 말해, 볼륨이 클수록 성능이 향상됩니다. 따라서 볼륨을 프로비저닝할 때 용량 및 성능 요구사항을 모두 고려해야 합니다.

최소 볼륨 크기입니다

Astra Control Service는 PVC가 더 작은 볼륨 크기를 요구하더라도 최소 볼륨 크기 100GiB를 사용하여 영구 볼륨을 프로비저닝합니다. 예를 들어, Hrom 차트의 PVC가 6GiB를 요청하는 경우 Astra Control Service는 자동으로 100GiB

볼륨을 프로비저닝합니다.

애플리케이션 백업

Azure NetApp Files 스토리지에 상주하는 애플리케이션을 백업하는 경우 Astra Control Service가 자동으로 용량 풀을 확장합니다. 백업이 완료된 후 Astra Control Service는 용량 풀을 이전 크기로 축소합니다. Azure 구독에 따라 이 경우 저장소 요금이 부과될 수 있습니다. 용량 풀 크기 조정 이벤트 기록은 * Activity * 페이지 이벤트 로그에서 확인할 수 있습니다.

크기 조정 작업 중에 용량 풀이 Azure 구독에서 허용하는 최대 크기를 초과하면 백업 작업이 실패하고 Azure API에서 경고가 트리거됩니다.

Azure로 관리되는 디스크

Astra Control Service는 CSI(Container Storage Interface) 드라이버를 사용하여 Azure 관리 디스크와 스토리지 백엔드로 인터페이스할 수 있습니다. 이 서비스는 Azure에서 관리하는 블록 레벨 스토리지를 제공합니다.

"Azure 관리 디스크에 대해 자세히 알아보십시오".

NetApp Cloud Volumes ONTAP를 참조하십시오

성능 권장사항을 비롯한 NetApp Cloud Volumes ONTAP 구성에 대한 자세한 내용은 를 참조하십시오 "NetApp Cloud Volumes ONTAP 문서".

서비스 유형, 스토리지 클래스 및 GKE 클러스터의 PV 크기입니다

Astra Control Service는 영구 볼륨의 스토리지 백엔드 옵션으로 NetApp Cloud Volumes Service for Google Cloud, Google Persistent Disk 또는 NetApp Cloud Volumes ONTAP를 지원합니다.

Google Cloud용 Cloud Volumes Service

Astra Control Service는 Cloud Volumes Service for Google Cloud를 영구 볼륨의 스토리지 백엔드로 사용할 수 있습니다. 서비스 유형, 스토리지 클래스 및 영구 볼륨 크기를 선택하면 성능 목표를 달성하는 데 어떤 도움이 되는지 알아야 합니다.

개요

Google Cloud용 Cloud Volumes Service는 두 가지 서비스 유형, 즉 _CVS_와 _CVS - 성능 _ 을(를) 제공합니다. 이러한 서비스 유형은 특정 Google Cloud 지역에서 지원됩니다. "NetApp BlueXP 글로벌 지역 맵 으로 이동합니다" 클러스터가 상주하는 Google Cloud 영역에서 지원되는 서비스 유형을 식별합니다.

Kubernetes 클러스터가 특정 지역에 있어야 하는 경우 해당 지역에서 지원되는 서비스 유형을 사용하게 됩니다.

그러나 Google Cloud 지역 중에서 선택할 수 있는 유연성이 있다면 성능 요구 사항에 따라 다음 사항을 권장합니다.

- 중간~고성능 스토리지가 필요한 K8s 애플리케이션의 경우 CVS 성능을 지원하는 Google Cloud 영역을 선택하고 프리미엄 또는 익스트림 스토리지 클래스를 사용합니다. 이러한 워크로드에는 AI/ML 파이프라인, CI/CD 파이프라인, 미디어 처리, 관계형, NoSQL, 시계열 등을 포함한 데이터베이스 등이 있습니다
- 중간 규모의 스토리지 성능 요구(웹 애플리케이션, 범용 파일 스토리지 등)가 필요한 K8s 애플리케이션의 경우 표준

스토리지 클래스를 사용하여 CVS 또는 CVS 성능을 지원하는 Google Cloud 영역을 선택하십시오.

> CVS 서비스 유형을 Astra Control Provisioner와 함께 사용하는 경우 볼륨을 프로비저닝하기 전에 스토리지 풀을 구성해야 합니다. 스토리지 풀이 구성되지 않은 볼륨을 프로비저닝하는 경우 볼륨 프로비저닝이 실패합니다. 을 참조하십시오 "Cloud Volumes Service 설명서" 볼륨 생성에 대한 자세한 내용은 를 참조하십시오.

다음 표에서는 이 페이지에 설명된 정보를 빠르게 비교합니다.

서비스 유형입니다	사용 사례	지원 지역	스토리지 클래스	최소 볼륨 크기
CVS - 성능	중간 이상의 스토리지 성능 요구사항이 있는 애플리케이션	"지원되는 Google Cloud 지역을 봅니다"	• NetApp-cvs-perf-standard • NetApp-cvs-perf-프리미엄 • NetApp-cvs-perf-extreme	100GiB
CV	중간 규모의 스토리지 성능 요구사항을 가진 애플리케이션	"지원되는 Google Cloud 지역을 봅니다"	NetApp-CV-표준	300GiB

CVS - 성능 서비스 유형입니다

스토리지 클래스를 선택하고 영구 볼륨을 생성하기 전에 CVS - 성능 서비스 유형에 대해 자세히 알아보십시오.

스토리지 클래스

CVS 성능 서비스 유형에는 Standard, Premium, Extreme의 세 가지 서비스 레벨이 지원됩니다. Astra Control Service에 클러스터를 추가하면 영구 볼륨의 기본 스토리지 클래스로 Standard, Premium 또는 Extreme을 선택하라는 메시지가 표시됩니다. 이러한 각 서비스 수준은 용량 및 대역폭 요구 사항에 따라 다르게 설계되었습니다.

스토리지 클래스의 이름은 *NetApp-cvs-perf-standard*,*NetApp-cvs-perf-premium* 및 _NetApp-cvs-perf-extreme_입니다.

"Cloud Volumes Service for Google Cloud 문서에서 이러한 서비스 수준에 대해 자세히 알아보십시오".

영구 볼륨 크기 및 성능

"Google Cloud 문서가 설명하는 대로"각 서비스 수준에 허용되는 대역폭은 프로비저닝된 용량 GiB 단위로 표시됩니다. 즉, 볼륨이 클수록 성능이 향상됩니다.

위에 링크된 Google Cloud 페이지를 읽어보시기 바랍니다. 또한 비용 비교와 예제를 통해 서비스 수준을 볼륨 크기와 연계하여 성능 목표를 달성하는 방법을 보다 잘 이해할 수 있습니다.

최소 볼륨 크기입니다

Astra Control Service는 PVC가 더 작은 볼륨 크기를 요청하더라도 CVS-Performance 서비스 유형의 최소 볼륨 크기 100GiB를 사용하여 영구 볼륨을 프로비저닝합니다. 예를 들어, Hrom 차트의 PVC가 6GiB를 요청하는 경우 Astra Control Service는 자동으로 100GiB 볼륨을 프로비저닝합니다.

CVS 서비스 유형입니다

스토리지 클래스를 선택하고 영구 볼륨을 생성하기 전에 CVS 서비스 유형에 대해 자세히 알아보십시오.

스토리지 클래스

CVS 서비스 유형인 Standard에서는 하나의 서비스 수준이 지원됩니다. CVS 서비스 유형이 지원되는 지역에서 클러스터를 관리하는 경우 Astra Control Service는 표준 서비스 수준을 영구 볼륨의 기본 스토리지 클래스로 사용합니다. 스토리지 클래스 이름은 _NetApp-CV-standard_입니다.

"Cloud Volumes Service for Google Cloud 문서에서 표준 서비스 수준에 대해 자세히 알아보십시오".

영구 볼륨 크기 및 성능

CVS 서비스 유형에 허용되는 대역폭은 프로비저닝된 용량 GiB입니다. 즉, 볼륨이 클수록 성능이 향상됩니다.

최소 볼륨 크기입니다

Astra Control Service는 PVC가 더 작은 볼륨 크기를 요청하더라도 CVS 서비스 유형의 최소 볼륨 크기 300GiB를 사용하여 영구 볼륨을 프로비저닝합니다. 예를 들어 20GiB가 요청되면 Astra Control Service는 300GiB 볼륨을 자동으로 프로비저닝합니다.

제한 사항으로 인해 PVC가 700-999 GiB의 볼륨을 요청하는 경우 Astra Control Service는 자동으로 1000GiB의 볼륨 크기를 프로비저닝합니다.

Google 영구 디스크

Astra Control Service는 CSI(Container Storage Interface) 드라이버를 사용하여 Google 영구 디스크와 스토리지 백엔드로 인터페이스할 수 있습니다. 이 서비스는 Google에서 관리하는 블록 레벨 스토리지를 제공합니다.

"Google 영구 디스크에 대해 자세히 알아보십시오".

"Google 영구 디스크의 다양한 성능 수준에 대해 자세히 알아보십시오".

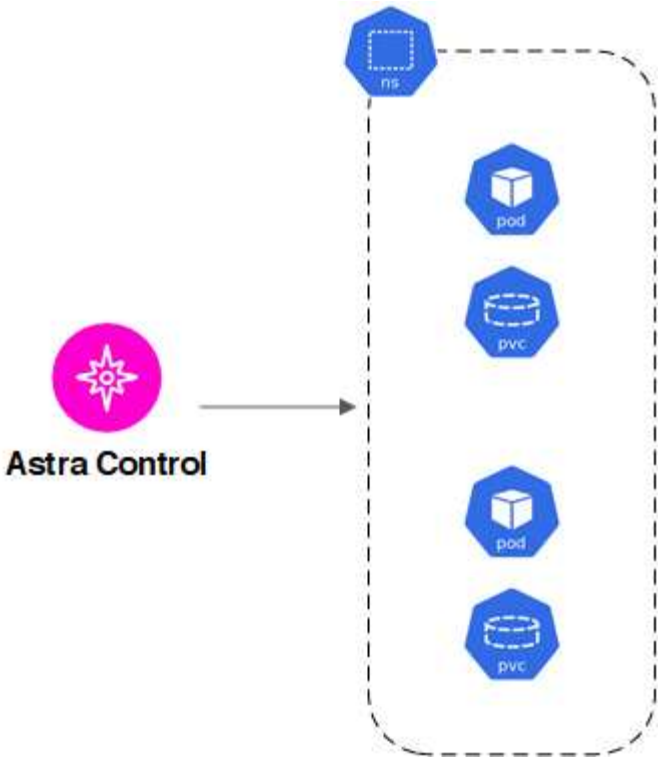
NetApp Cloud Volumes ONTAP를 참조하십시오

성능 권장사항을 비롯한 NetApp Cloud Volumes ONTAP 구성에 대한 자세한 내용은 를 참조하십시오 "NetApp Cloud Volumes ONTAP 문서".

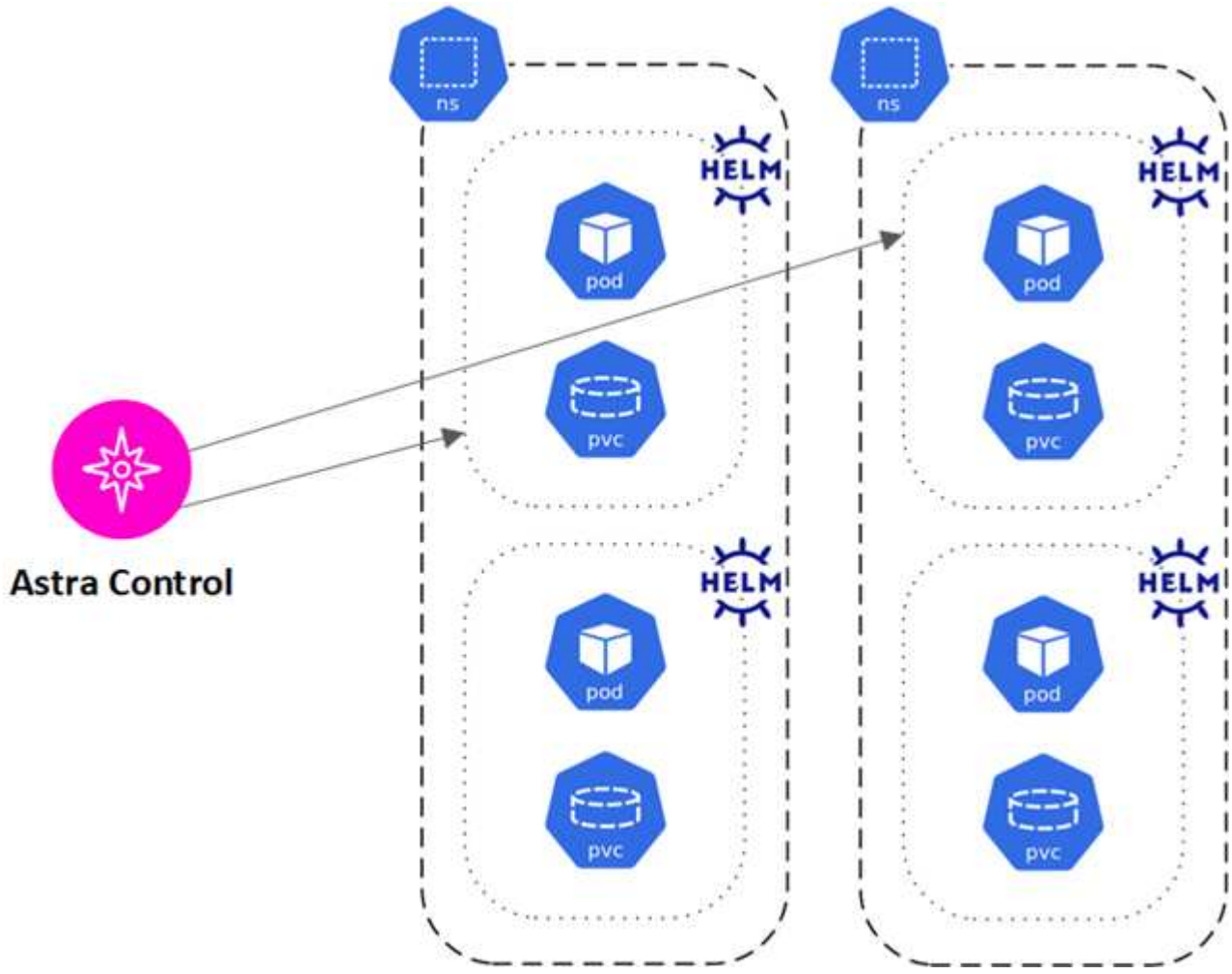
애플리케이션 관리

Astra Control이 클러스터를 검색할 때 해당 클러스터의 앱은 관리 방법을 선택할 때까지 관리되지 않습니다. Astra Control에서 관리되는 응용 프로그램은 다음 중 하나일 수 있습니다.

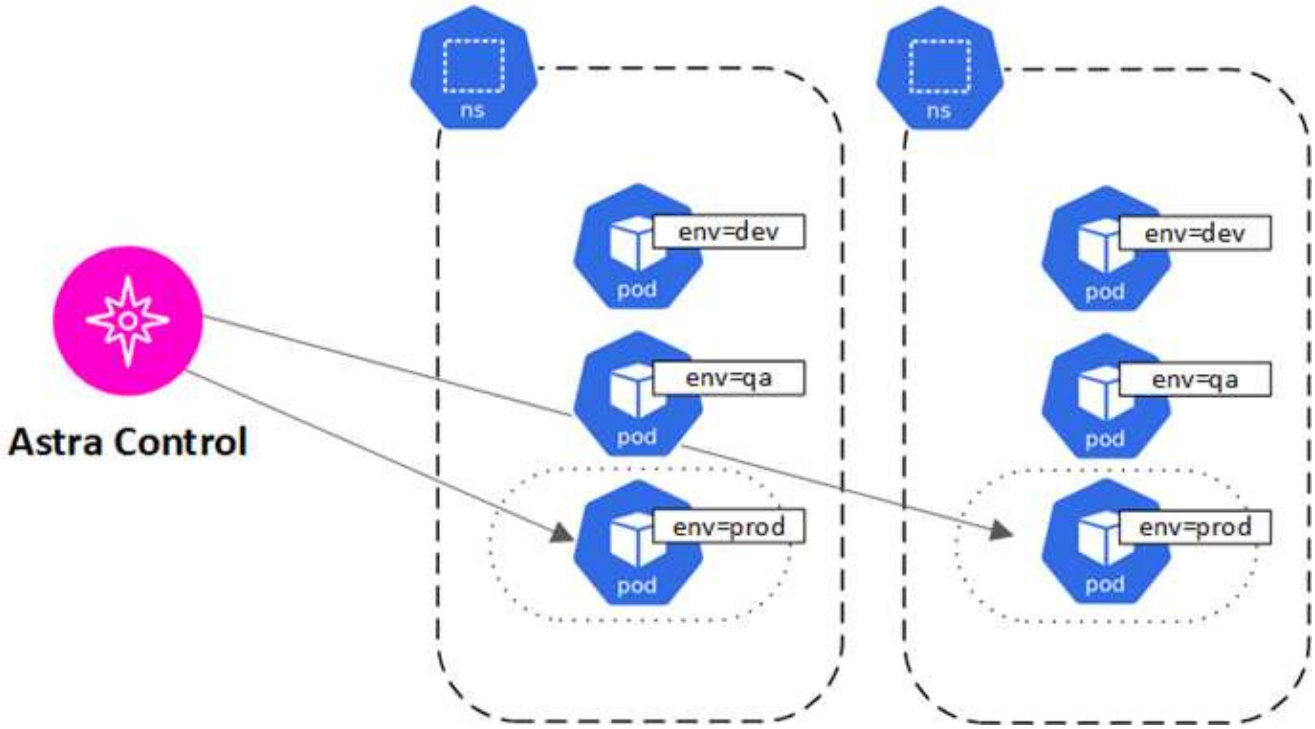
- 네임스페이스에서 모든 리소스를 포함하는 네임스페이스입니다



- 하나 이상의 네임스페이스 내에 배포된 개별 응용 프로그램(이 예제에서는 H제어 3이 사용됨)



- 하나 이상의 네임스페이스 내에서 Kubernetes 레이블로 식별되는 리소스 그룹입니다



사용자 역할 및 네임스페이스

Astra Control의 사용자 역할 및 네임스페이스, 그리고 이를 사용하여 조직의 리소스에 대한 액세스를 제어하는 방법에 대해 알아봅니다.

사용자 역할

역할을 사용하여 Astra Control의 리소스 또는 기능에 대한 사용자의 액세스를 제어할 수 있습니다. Astra Control의 사용자 역할은 다음과 같습니다.

- 소유자 * 는 관리자 권한을 가지며 계정을 삭제할 수 있습니다.
- Admin * 은 구성원 권한을 가지며 다른 사용자를 초대할 수 있습니다.
- 회원 * 은 앱과 클러스터를 완벽하게 관리할 수 있습니다.
- Viewer * 는 리소스를 볼 수 있습니다.

멤버 또는 뷰어 사용자에게 제약 조건을 추가하여 사용자를 하나 이상의 사용자로 제한할 수 있습니다 [네임스페이스].

네임스페이스

네임스페이스는 Astra Control에서 관리하는 클러스터 내의 특정 리소스에 할당할 수 있는 범위입니다. Astra Control은 클러스터를 Astra Control에 추가할 때 클러스터의 네임스페이스를 검색합니다. 네임스페이스가 검색되면 사용자에게 제약 조건으로 할당할 수 있습니다. 해당 네임스페이스에 대한 액세스 권한이 있는 멤버만 해당 리소스를 사용할 수 있습니다. 회사 내의 물리적 영역이나 부서 등 조직에 적합한 패러다임을 사용하여 네임스페이스에 대한 액세스를 제어할 수 있습니다. 사용자에게 제약 조건을 추가하면 해당 사용자가 모든 네임스페이스에 액세스하거나 특정 네임스페이스 집합만 액세스하도록 구성할 수 있습니다. 네임스페이스 레이블을 사용하여 네임스페이스 제약 조건을

할당할 수도 있습니다.

자세한 내용을 확인하십시오

- "역할을 관리합니다"

Astra Control Service를 사용합니다

Astra Control Service에 로그인합니다

Astra Control Service는 로 이동하여 SaaS 기반 사용자 인터페이스를 통해 액세스할 수 있습니다 https://astra.netapp.io.

Single Sign-On을 사용하여 회사 디렉터리(통합 ID)의 자격 증명을 사용하여 로그인할 수 있습니다. 자세한 내용은 로 이동하십시오 "Help Center(도움말 센터" 그런 다음 * Cloud Central 로그인 옵션 * 을 선택합니다.

시작하기 전에

- "BlueXP 사용자 ID입니다".
- "새로운 Astra Control 계정" 또는 "기존 계정에 대한 초대".
- 지원되는 웹 브라우저입니다.

  Astra Control Service는 최신 버전의 Firefox, Safari 및 Chrome을 최소 1280 x 720의 해상도로 지원합니다.

단계

1. 웹 브라우저를 열고 로 이동합니다 https://astra.netapp.io.
2. NetApp BlueXP 자격 증명을 사용하여 로그인합니다.

애플리케이션 관리 및 보호

앱 관리를 시작합니다

먼저 해 "Astra Control에 Kubernetes 클러스터를 추가합니다", 클러스터(Astra Control 외부)에 앱을 설치한 다음 Astra Control의 애플리케이션 페이지로 이동하여 앱을 정의할 수 있습니다.

실행 중인 Pod가 있는 스토리지 리소스가 포함된 앱 또는 실행 중인 Pod 없이 스토리지 리소스가 포함된 앱을 정의하고 관리할 수 있습니다. 실행 중인 Pod가 없는 앱을 데이터 전용 애플리케이션이라고 합니다.

설명합니다

Astra Control에는 다음과 같은 앱 관리 요구 사항이 있습니다.

- * Licensing *: 10개 이상의 네임스페이스를 관리하려면 Astra Control 가입이 필요합니다.
- * 네임스페이스 *: Astra Control을 사용하여 단일 클러스터에서 하나 이상의 지정된 네임스페이스 내에서 응용 프로그램을 정의할 수 있습니다. 앱은 동일한 클러스터 내에서 여러 네임스페이스에 걸쳐 있는 리소스를 포함할 수 있습니다. Astra Control은 여러 클러스터에서 앱을 정의하는 기능을 지원하지 않습니다.
- * 스토리지 클래스 *: 스토리지 클래스가 명시적으로 설정된 앱을 설치하고 앱을 복제해야 하는 경우 클론 작업의 타겟 클러스터에 원래 지정된 스토리지 클래스가 있어야 합니다. 명시적으로 설정된 스토리지 클래스를 가진 애플리케이션을 동일한 스토리지 클래스가 없는 클러스터로 클론 복제하면 실패합니다.

- * Kubernetes 리소스 *: Astra Control에서 수집하지 않은 Kubernetes 리소스를 사용하는 앱에는 전체 앱 데이터 관리 기능이 없을 수 있습니다. Astra Control은 다음과 같은 Kubernetes 리소스를 수집합니다.

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		

지원되는 앱 설치 방법

Astra Control은 다음과 같은 응용 프로그램 설치 방법을 지원합니다.

- * 매니페스트 파일 *: Astra Control은 kubctl을 사용하여 매니페스트 파일에서 설치된 앱을 지원합니다. 예를 들면 다음과 같습니다.

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

- * Helm 3 *: Helm을 사용하여 앱을 설치하는 경우 Astra Control에 Helm 버전 3이 필요합니다. Helm 3(또는 Helm 2에서 Helm 3으로 업그레이드)과 함께 설치된 앱의 관리 및 클론 생성이 완벽하게 지원됩니다. Helm 2가 설치된 앱 관리는 지원되지 않습니다.
- * 운용자 구축 앱 *: Astra Control은 네임스페이스 범위 연산자로 설치된 앱을 지원합니다. 일반적으로 "pass-by-reference" 아키텍처가 아니라 "pass-by-value"로 설계되었습니다. 운영자와 설치하는 앱은 동일한 네임스페이스를 사용해야 합니다. 운영자가 배포 .YAML 파일을 수정해야 할 수도 있습니다.

  다음은 이러한 패턴을 따르는 일부 운영자 앱에 대한 설명입니다.

  - "아파치 K8ssandra"

    > K8ssandra 의 경우 현재 위치 복원 작업이 지원됩니다. 새 네임스페이스 또는 클러스터에 대한 복원 작업을 수행하려면 응용 프로그램의 원래 인스턴스를 중단해야 합니다. 이는 이월된 피어 그룹 정보가 인스턴스 간 통신으로 이어지지 않도록 하기 위한 것입니다. 앱 복제는 지원되지 않습니다.

  - "젠킨스 CI"
  - "Percona XtraDB 클러스터"

  Astra Control은 "pass-by-reference" 아키텍처(예: CockroachDB 운영자)로 설계된 운영자를 복제하지 못할 수 있습니다. 이러한 유형의 클론 복제 작업 중에 클론 복제 운영자는 클론 복제 프로세스의 일부로 고유한 새로운 암호가 있음에도 불구하고 소스 운영자의 Kubernetes 암호를 참조하려고 합니다. Astra Control이 소스 운영자의

Kubernetes 암호를 모르기 때문에 클론 작업이 실패할 수 있습니다.

클러스터에 앱을 설치합니다

먼저 해 "클러스터가 추가되었습니다" Astra Control은 클러스터에서 앱을 설치하거나 기존 앱을 관리할 수 있습니다. 하나 이상의 네임스페이스로 범위가 지정된 모든 앱을 관리할 수 있습니다.

Astra Control은 스토리지가 Astra Control에서 지원하는 스토리지 클래스에 있는 경우에만 상태 저장 앱을 관리합니다. Astra Control Service는 Astra Control Provisioner 또는 일반 CSI 드라이버에서 지원하는 모든 스토리지 클래스를 지원합니다.

- "GKE 클러스터용 저장소 클래스에 대해 알아봅니다"
- "AKS 클러스터용 스토리지 클래스에 대해 알아보십시오"
- "AWS 클러스터를 위한 스토리지 클래스에 대해 알아보십시오"

앱 정의

Astra Control이 클러스터에서 네임스페이스를 검색한 후 관리할 애플리케이션을 정의할 수 있습니다. 선택할 수 있습니다 하나 이상의 네임스페이스를 포괄하는 응용 프로그램을 관리합니다 또는 전체 네임스페이스를 단일 애플리케이션으로 관리합니다. 데이터 보호 작업에 필요한 세분화 수준으로 세분화됩니다.

Astra Control을 사용하면 계층 구조의 수준(네임스페이스 및 해당 네임스페이스 또는 스패닝 네임스페이스의 응용 프로그램)을 별도로 관리할 수 있지만 가장 좋은 방법은 하나 또는 다른 수준을 선택하는 것입니다. 작업이 네임스페이스 및 앱 수준에서 동시에 발생하면 Astra Control에서 수행하는 작업이 실패할 수 있습니다.

예를 들어, "Maria"에 대해 주간 백업 주기를 갖는 백업 정책을 설정할 수 있지만 "MariaDB"(동일한 네임스페이스)를 더 자주 백업해야 할 수 있습니다. 이러한 요구사항에 따라 단일 네임스페이스 앱이 아니라 앱을 별도로 관리해야 합니다.

시작하기 전에

- Astra Control에 Kubernetes 클러스터가 추가되었습니다.
- 클러스터에 설치된 애플리케이션 하나 이상 지원되는 앱 설치 방법에 대해 자세히 알아보십시오.
- Astra Control에 추가한 Kubernetes 클러스터의 기존 네임스페이스
- (선택 사항) Any의 Kubernetes 레이블 "지원되는 Kubernetes 리소스".

레이블은 식별을 위해 Kubernetes 객체에 할당할 수 있는 키/값 쌍입니다. 레이블을 사용하면 Kubernetes 오브젝트를 더 쉽게 정렬, 구성 및 찾을 수 있습니다. Kubernetes 레이블에 대해 자세히 알아보려면 "공식 Kubernetes 설명서를 참조하십시오".

이 작업에 대해

- 시작하기 전에, 또한 이해해야 합니다 "표준 및 시스템 네임스페이스 관리".
- Astra Control에서 앱과 여러 네임스페이스를 사용하려는 경우 고려해 보십시오 "네임스페이스 제약 조건을 사용하여 사용자 역할 수정" 앱을 정의하기 전에
- Astra Control API를 사용하여 앱을 관리하는 방법에 대한 지침은 을 참조하십시오 "Astra 자동화 및 API 정보".

애플리케이션 관리 옵션

- 앱으로 관리할 리소스를 정의합니다
- 앱으로 관리할 네임스페이스를 정의합니다

앱으로 관리할 리소스를 정의합니다

를 지정할 수 있습니다 "앱을 구성하는 Kubernetes 리소스" Astra Control을 통해 관리하고자 하는 것입니다. 앱을 정의하면 Kubernetes 클러스터의 요소를 단일 애플리케이션으로 그룹화할 수 있습니다. 이 Kubernetes 리소스 모음은 네임스페이스 및 레이블 선택기 기준에 따라 구성됩니다.

앱을 정의하면 클론, 스냅샷, 백업을 비롯한 Astra Control 작업에 포함할 항목을 보다 세부적으로 제어할 수 있습니다.

앱을 정의할 때 보호 정책이 있는 여러 앱에 Kubernetes 리소스를 포함하지 않아야 합니다. Kubernetes 리소스의 보호 정책이 중복되어 데이터 충돌이 발생할 수 있습니다.

앱 네임스페이스에 클러스터 범위 리소스를 추가하는 방법에 대해 자세히 알아보십시오.

Namespace 리소스와 연결된 클러스터 리소스 및 자동으로 포함된 Astra Control을 가져올 수 있습니다. 특정 그룹, 종류, 버전 및 레이블(선택 사항)의 리소스를 포함할 규칙을 추가할 수 있습니다. Astra Control에 자동으로 포함되지 않는 리소스가 있는 경우 이 작업을 수행할 수 있습니다.

Astra Control에 의해 자동으로 포함되는 클러스터 범위 리소스는 제외할 수 없습니다.

다음을 추가할 수 있습니다 `apiVersions` (API 버전과 결합된 그룹):

자원 종류	**apiVersions**(그룹 + 버전)
`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`	Admissions registration.k8s.io/v1
`ValidatingWebhookConfiguration`	Admissions registration.k8s.io/v1

단계

1. 응용 프로그램 페이지에서 * 정의 * 를 선택합니다.
2. 응용 프로그램 정의 * 창에서 응용 프로그램 이름을 입력합니다.
3. 응용 프로그램이 실행되는 클러스터를 * 클러스터 * 드롭다운 목록에서 선택합니다.
4. Namespace* 드롭다운 목록에서 응용 프로그램의 네임스페이스를 선택합니다.

Astra Control을 사용하여 단일 클러스터에서 하나 이상의 지정된 네임스페이스 내에서 앱을 정의할 수 있습니다. 앱은 동일한 클러스터 내에서 여러 네임스페이스에 걸쳐 있는 리소스를 포함할 수 있습니다. Astra Control은 여러 클러스터에서 앱을 정의하는 기능을 지원하지 않습니다.

5. (선택 사항) 각 네임스페이스에서 Kubernetes 리소스에 대한 레이블을 입력합니다. 단일 레이블 또는 레이블 선택 조건(쿼리)을 지정할 수 있습니다.

   Kubernetes 레이블에 대해 자세히 알아보려면 "공식 Kubernetes 설명서를 참조하십시오".

6. (선택 사항) * 네임스페이스 추가 * 를 선택하고 드롭다운 목록에서 네임스페이스를 선택하여 앱에 대한 네임스페이스를 추가합니다.
7. (선택 사항) 추가하는 모든 추가 네임스페이스에 대한 단일 레이블 또는 레이블 선택기 조건을 입력합니다.
8. (선택 사항) Astra Control에 자동으로 포함되는 리소스 외에 클러스터 범위 리소스를 포함하려면 * 추가 클러스터 범위 리소스 포함 * 을 선택하여 다음을 완료합니다.
   a. 포함 규칙 추가 * 를 선택합니다.
   b. * Group *: 드롭다운 목록에서 리소스의 API 그룹을 선택합니다.
   c. * Kind *: 드롭다운 목록에서 개체 스키마의 이름을 선택합니다.
   d. * 버전 *: API 버전을 입력합니다.
   e. * 라벨 선택기 *: 규칙에 추가할 라벨을 선택적으로 포함합니다. 이 레이블은 이 레이블과 일치하는 리소스만 검색하는 데 사용됩니다. 레이블을 제공하지 않으면 Astra Control은 해당 클러스터에 대해 지정된 리소스 유형의 모든 인스턴스를 수집합니다.
   f. 항목에 따라 만들어진 규칙을 검토합니다.
   g. 추가 * 를 선택합니다.

   클러스터 범위의 리소스 규칙을 원하는 만큼 만들 수 있습니다. 규칙은 애플리케이션 요약 정의에 나타납니다.

9. 정의 * 를 선택합니다.
10. 정의 * 를 선택한 후 필요에 따라 다른 앱에 대해 프로세스를 반복합니다.

앱 정의를 마치면 앱이 에 나타납니다 `Healthy` 응용 프로그램 페이지의 응용 프로그램 목록에서 상태를 지정합니다. 이제 클론을 생성하고 백업과 스냅샷을 생성할 수 있습니다.

방금 추가한 앱에는 Protected(보호) 열 아래에 백업이 없고 아직 백업이 예약되지 않았음을 나타내는 경고 아이콘이 있을 수 있습니다.

특정 앱의 세부 정보를 보려면 앱 이름을 선택합니다.

이 앱에 추가된 리소스를 보려면 * 리소스 * 탭을 선택하십시오. 리소스 열에서 리소스 이름 뒤의 숫자를 선택하거나 검색에 리소스 이름을 입력하여 추가 클러스터 범위 리소스가 포함되도록 합니다.

앱으로 관리할 네임스페이스를 정의합니다

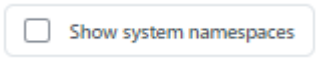
네임스페이스의 리소스를 애플리케이션으로 정의하여 Astra Control 관리에 네임스페이스의 모든 Kubernetes 리소스를 추가할 수 있습니다. 이 방법은 앱을 개별적으로 정의하는 것이 좋습니다 "특정 네임스페이스의 모든 리소스를 관리하고 보호하려고 합니다" 비슷한 방식으로, 일정한 간격으로.

단계

1. 클러스터 페이지에서 클러스터를 선택합니다.

2. Namespaces* 탭을 선택합니다.
3. 관리하려는 앱 리소스가 포함된 네임스페이스의 작업 메뉴를 선택하고 * 응용 프로그램으로 정의 * 를 선택합니다.

여러 응용 프로그램을 정의하려면 네임스페이스 목록에서 선택하고 왼쪽 위 모서리에 있는 * 작업 * 버튼을 선택한 다음 * 응용 프로그램으로 정의 * 를 선택합니다. 이렇게 하면 개별 네임스페이스에 여러 개의 개별 응용 프로그램이 정의됩니다. 다중 네임스페이스 응용 프로그램의 경우 을 참조하십시오 앱으로 관리할 리소스를 정의합니다.

기본적으로 앱 관리에 사용되지 않는 시스템 네임스페이스를 표시하려면 * Show system namespaces * 확인란을 선택합니다. 

"자세히 보기".

프로세스가 완료되면 해당 네임스페이스와 연결된 응용 프로그램이 '연결된 응용 프로그램' 열에 나타납니다.

[기술 미리보기] Kubernetes 맞춤형 리소스를 사용하여 애플리케이션을 정의합니다

Astra Control을 통해 관리할 Kubernetes 리소스를 사용자 지정 리소스(CR)를 사용하여 애플리케이션으로 정의하여 지정할 수 있습니다. 예를 들어, 특정 네임스페이스의 모든 리소스를 비슷한 방식으로 공통 간격으로 관리 및 보호하려는 경우, 해당 리소스를 개별적으로 또는 모든 Kubernetes 리소스를 네임스페이스에서 관리하려는 경우 클러스터 범위 리소스를 추가할 수 있습니다.

단계

1. 사용자 정의 리소스(CR) 파일을 만들고 이름을 지정합니다(예: `astra_mysql_app.yaml`)를 클릭합니다.
2. 애플리케이션 이름을 에 지정합니다 `metadata.name`.
3. 관리할 애플리케이션 리소스 정의:

spec.includedClusterScopedResources

Astra Control에 자동으로 포함되는 리소스 유형 외에 클러스터 범위 리소스 유형 포함

- **spec.includedClusterScopedResources**: _ (선택 사항) _ 포함할 클러스터 범위 리소스 유형의 목록입니다.
  - **groupVersionKind**:_(선택 사항) _ 종류를 명확하게 식별합니다.
    - * group *: _ (groupVersionKind가 사용되는 경우 필수) _ 포함할 리소스의 API 그룹입니다.
    - * VERSION *: _ (groupVersionKind를 사용하는 경우 필수) _ 포함할 리소스의 API 버전입니다.
    - * kind *: _ (groupVersionKind를 사용하는 경우 필수) _ 포함할 리소스의 종류.
  - * labelSelector *: _ (선택 사항) _ 리소스 집합에 대한 레이블 쿼리입니다. 레이블과 일치하는 리소스만 검색하는 데 사용됩니다. 레이블을 제공하지 않으면 Astra Control은 해당 클러스터에 대해 지정된 리소스 유형의 모든 인스턴스를 수집합니다. MatchLabels 및 MatchExpressions의 결과는 ANDed입니다.
    - * matchLabels *: _ (선택 사항) _ {key, value} 쌍의 맵입니다. matchLabels 맵의 단일 {key,value}은 키 필드가 "key"이고 연산자는 "in"이고 값 배열만 포함하는 matchExpressions의 요소와 같습니다. 요구 사항은 ANDed입니다.
    - * matchExpressions *:_(선택 사항) _ 라벨 선택기 요구 사항 목록입니다. 요구 사항은 ANDed입니다.
      - * key *: _ (matchExpressions를 사용하는 경우 필수) _ 라벨 선택기와 관련된 라벨 키입니다.
      - * operator *: _ (matchExpressions를 사용하는 경우 필수) _ 값 집합에 대한 키의 관계를 나타냅니다. 유효한 연산자는 다음과 같습니다 `In`, `NotIn`, `Exists` 및 `DoesNotExist`.
      - * values *: _ (matchExpressions를 사용하는 경우 필수) _ 문자열 값의 배열입니다. 오퍼레이터가 인 경우 `In` 또는 `NotIn`값 배열은 반드시 _not_이어야 합니다. 오퍼레이터가 인 경우 `Exists` 또는 `DoesNotExist`값 배열은 비어 있어야 합니다.

spec.includedNamespaces

응용 프로그램의 해당 리소스 내에 네임스페이스와 리소스를 포함합니다.

- **spec.includedNamespaces**: _ (필수) _ 리소스 선택을 위한 네임스페이스 및 선택적 필터를 정의합니다.
  - * namespace *: _ (필수) _ Astra Control을 사용하여 관리하려는 앱 리소스가 포함된 네임스페이스입니다.
  - * labelSelector *: _ (선택 사항) _ 리소스 집합에 대한 레이블 쿼리입니다. 레이블과 일치하는 리소스만 검색하는 데 사용됩니다. 레이블을 제공하지 않으면 Astra Control은 해당 클러스터에 대해 지정된 리소스 유형의 모든 인스턴스를 수집합니다. MatchLabels 및 MatchExpressions의 결과는 ANDed입니다.
    - * matchLabels *: _ (선택 사항) _ {key, value} 쌍의 맵입니다. matchLabels 맵의 단일 {key,value}은 키 필드가 "key"이고 연산자는 "in"이고 값 배열만 포함하는 matchExpressions의 요소와 같습니다. 요구 사항은 ANDed입니다.
    - * matchExpressions *:_(선택 사항) _ 라벨 선택기 요구 사항 목록입니다. `key` 및 `operator` 필수 항목입니다. 요구 사항은 ANDed입니다.
      - * key *: _ (matchExpressions를 사용하는 경우 필수) _ 라벨 선택기와 관련된 라벨 키입니다.

- * operator *: _ (matchExpressions를 사용하는 경우 필수) _ 값 집합에 대한 키의 관계를 나타냅니다. 유효한 연산자는 다음과 같습니다 `In`, `NotIn`, `Exists` 및 `DoesNotExist`.
- * values *: _ (matchExpressions를 사용하는 경우 필수) _ 문자열 값의 배열입니다. 오퍼레이터가 인 경우 `In` 또는 `NotIn`값 배열은 반드시 _not_이어야 합니다. 오퍼레이터가 인 경우 `Exists` 또는 `DoesNotExist`값 배열은 비어 있어야 합니다.

YAML 예:

```
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
```

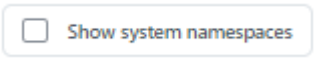
4. 를 채운 후 `astra_mysql_app.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

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

시스템 네임스페이스는 어떻습니까?

Astra Control은 Kubernetes 클러스터에서 시스템 네임스페이스를 검색합니다. 기본적으로 이러한 시스템 네임스페이스는 표시되지 않습니다. 시스템 앱 리소스를 백업해야 하는 경우는 드뭅니다.

선택한 클러스터의 Namespaces 탭에서 * Show system namespaces * 확인란을 선택하여 시스템 네임스페이스를 표시할 수 있습니다.



Astra Control 자체는 표준 앱이 아니며 "시스템 앱"입니다. Astra Control 자체를 관리하려고 해서는 안 됩니다. 관리 시 Astra Control 자체는 기본적으로 표시되지 않습니다.

스냅샷 및 백업으로 애플리케이션 보호

자동화된 보호 정책을 사용하거나 필요에 따라 스냅샷과 백업을 생성하여 앱을 보호합니다. Astra UI 또는 를 사용할 수 있습니다 "Astra Control API" 앱을 보호합니다.

에 대해 자세히 알아보십시오 "Astra Control의 데이터 보호".

앱 데이터 보호와 관련된 다음 작업을 수행할 수 있습니다.

- 보호 정책을 구성합니다
- 스냅샷을 생성합니다
- 백업을 생성합니다
- ONTAP - NAS - 경제성 작업을 위한 백업 및 복원 지원
- 변경 불가능한 백업을 생성합니다
- 스냅샷 및 백업을 봅니다
- 스냅샷을 삭제합니다
- 백업을 취소합니다
- 백업을 삭제합니다

보호 정책을 구성합니다

보호 정책은 정의된 일정에 따라 스냅샷, 백업 또는 둘 다를 생성하여 앱을 보호합니다. 시간별, 일별, 주별 및 월별 스냅샷과 백업을 생성하도록 선택할 수 있으며, 보존할 복제본 수를 지정할 수 있습니다. Astra Control 웹 UI 또는 맞춤형 리소스(CR) 파일을 사용하여 보호 정책을 정의할 수 있습니다.

시간당 한 번 이상 백업 또는 스냅샷을 자주 실행해야 하는 경우 를 수행할 수 있습니다 "Astra Control REST API를 사용하여 스냅샷과 백업을 생성합니다".

WORM(Write Once Read Many) 버킷에 대한 변경 불가능한 백업을 생성하는 보호 정책을 정의하는 경우 백업의 보존 시간이 버킷에 대해 구성된 보존 기간보다 짧지 않은지 확인합니다.

백업 및 복제 일정을 오프셋하여 일정이 겹치지 않도록 합니다. 예를 들어, 매시간 맨 위에서 백업을 수행하고 5분 오프셋 및 10분 간격으로 복제를 시작하도록 예약합니다.

웹 **UI**를 사용하여 보호 정책을 구성합니다

단계

1. 응용 프로그램 * 을 선택한 다음 앱 이름을 선택합니다.
2. 데이터 보호 * 를 선택합니다.
3. 보호 정책 구성 * 을 선택합니다.
4. 시간별, 일별, 주별 및 월별 스케줄에 대해 유지할 스냅샷 및 백업 수를 선택하여 보호 스케줄을 정의합니다.

   시간별, 일별, 주별 및 월별 스케줄을 동시에 정의할 수 있습니다. 보존 레벨을 설정하기 전에는 스케줄이 활성화되지 않습니다.

   백업의 보존 레벨을 설정할 때 백업을 저장할 버킷을 선택할 수 있습니다.

   다음 예에서는 스냅샷 및 백업의 경우 매시간, 일별, 주별 및 월별로 4개의 보호 스케줄을 설정합니다.

   [시간별, 일별, 주별 또는 월별 기준으로 스냅샷 및 백업을 수행하도록 선택할 수 있는 샘플 구성 정책의 스크린샷]

5. [* Tech preview *] 스토리지 버킷 목록에서 백업 또는 스냅샷의 대상 버킷을 선택합니다.
6. Review * 를 선택합니다.
7. 보호 정책 설정 * 을 선택합니다

[Tech Preview] CR을 사용하여 보호 정책을 구성합니다

단계

1. CR(사용자 정의 리소스) 파일을 만들고 이름을 지정합니다 `astra-control-schedule-cr.yaml`. Astra Control 환경, 클러스터 구성 및 데이터 보호 요구사항에 맞게 괄호 <> 의 값을 업데이트합니다.
   - <CR_NAME>: 이 사용자 지정 리소스의 이름입니다. 환경에 맞는 고유하고 합리적인 이름을 선택하십시오.
   - <APPLICATION_NAME>: 백업할 애플리케이션의 Kubernetes 이름입니다.
   - <APPVAULT_NAME>: 백업 컨텐츠를 저장해야 하는 AppVault의 이름입니다.
   - <BACKUPS_RETAINED>: 보존할 백업 수 0은 백업을 생성하지 않아야 함을 나타냅니다.
   - <SNAPSHOTS_RETAINED>: 보존할 스냅샷 수입니다. 0은 스냅샷을 생성하지 않아야 함을 나타냅니다.
   - <GRANULARITY>: 스케줄이 실행되는 빈도입니다. 가능한 값과 필수 관련 필드:
     - `hourly` (을(를) 지정해야 합니다 `spec.minute`)
     - `daily` (을(를) 지정해야 합니다 `spec.minute` 및 `spec.hour`)
     - `weekly` (을(를) 지정해야 합니다 `spec.minute`, `spec.hour`, 및 `spec.dayOfWeek`)
     - `monthly` (을(를) 지정해야 합니다 `spec.minute`, `spec.hour`, 및 `spec.dayOfMonth`)
   - <DAY_OF_MONTH>: _ (선택 사항) _ 일정을 실행할 요일(1-31)입니다. 세분화가 로 설정된 경우 이 필드는 필수입니다 `monthly`.
   - <DAY_OF_WEEK>: _ (선택 사항) _ 일정을 실행할 요일(0-7). 0 또는 7의 값은 일요일을 나타냅니다. 세분화가 로 설정된 경우 이 필드는 필수입니다 `weekly`.

- <HOUR_OF_DAY>: _ (선택 사항) _ 일정이 실행되는 시간(0 - 23)입니다. 세분화가 로 설정된 경우 이 필드는 필수입니다 `daily`, `weekly`, 또는 `monthly`.
- <MINUTE_OF_HOUR>: _ (선택 사항) _ 스케줄이 실행될 시간(0-59)입니다. 세분화가 로 설정된 경우 이 필드는 필수입니다 `hourly`, `daily`, `weekly`, 또는 `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. 를 채운 후 `astra-control-schedule-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

```
kubectl apply -f astra-control-schedule-cr.yaml
```

결과

Astra Control은 정의한 스케줄 및 보존 정책을 사용하여 스냅샷 및 백업을 생성하고 유지함으로써 데이터 보호 정책을 구현합니다.

스냅샷을 생성합니다

언제든지 주문형 스냅샷을 생성할 수 있습니다.

이 작업에 대해

Astra Control은 다음 드라이버를 통해 지원되는 스토리지 클래스를 사용하여 스냅샷 생성을 지원합니다.

- `ontap-nas`
- `ontap-san`
- `ontap-san-economy`

앱이 에서 지원하는 저장소 클래스를 사용하는 경우 `ontap-nas-economy` 드라이버, 스냅샷을 생성할 수 없습니다. 스냅샷에 대체 스토리지 클래스를 사용합니다.

웹 **UI**를 사용하여 스냅샷을 만듭니다

단계

1. 응용 프로그램 * 을 선택합니다.
2. 원하는 앱의 * Actions * 열에 있는 옵션 메뉴에서 * Snapshot * 을 선택합니다.
3. 스냅샷 이름을 사용자 지정하고 * 다음 * 을 선택합니다.
4. [* Tech preview *] 스토리지 버킷 목록에서 스냅샷의 대상 버킷을 선택합니다.
5. 스냅샷 요약을 검토하고 * Snapshot * 을 선택합니다.

[기술 미리보기**] CR**을 사용하여 스냅샷을 생성합니다

단계

1. CR(사용자 정의 리소스) 파일을 만들고 이름을 지정합니다 `astra-control-snapshot-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - <CR_NAME>: 이 사용자 지정 리소스의 이름입니다. 환경에 맞는 고유하고 합리적인 이름을 선택하십시오.
   - <APPLICATION_NAME>: 스냅샷을 생성할 애플리케이션의 Kubernetes 이름입니다.
   - <APPVAULT_NAME>: 스냅샷 컨텐츠를 저장해야 하는 AppVault의 이름입니다.
   - <RECLAIM_POLICY>: _ (선택 사항) _ 스냅샷 CR을 삭제할 때 스냅샷에 어떤 일이 발생하는지 정의합니다. 유효한 옵션:
     - `Retain`
     - `Delete` (기본값)

```
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. 를 채운 후 `astra-control-snapshot-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

```
kubectl apply -f astra-control-snapshot-cr.yaml
```

결과

스냅샷 프로세스가 시작됩니다. 데이터 보호 * > * 스냅샷 * 페이지의 * 상태 * 열에서 상태가 * 정상 * 인 경우 스냅샷이 성공합니다.

백업을 생성합니다

언제든지 앱을 백업할 수도 있습니다.

Azure NetApp Files 스토리지에서 호스트되는 애플리케이션을 백업할 때 스토리지 공간이 처리되는 방식에 유의하십시오. 을 참조하십시오 "애플리케이션 백업" 를 참조하십시오.

Astra Control은 다음 드라이버를 통해 지원되는 스토리지 클래스를 사용하여 백업 생성을 지원합니다.

- `ontap-nas`
- `ontap-nas-economy`
- `ontap-san`
- `ontap-san-economy`

이 작업에 대해

Astra Control의 버킷은 사용 가능한 용량을 보고하지 않습니다. Astra Control에서 관리되는 앱을 백업 또는 클론 복제하기 전에 적절한 스토리지 관리 시스템에서 버킷 정보를 확인하십시오.

앱이 에서 지원하는 저장소 클래스를 사용하는 경우 `ontap-nas-economy` 드라이버, 당신은 필요합니다 백업 및 복원을 활성화합니다 기능. 을(를) 정의했는지 확인합니다 `backendType` 매개 변수 을 선택합니다 "Kubernetes 스토리지 오브젝트입니다" 을 값으로 사용합니다 `ontap-nas-economy` 보호 작업을 수행하기 전에

웹 **UI**를 사용하여 백업을 만듭니다

단계

1. 응용 프로그램 * 을 선택합니다.
2. 원하는 앱의 * Actions * 열에 있는 옵션 메뉴에서 * Back Up * 을 선택합니다.
3. 백업 이름을 사용자 지정합니다.
4. 기존 스냅샷에서 앱을 백업할지 여부를 선택합니다. 이 옵션을 선택하면 기존 스냅샷 목록에서 선택할 수 있습니다.
5. [* Tech preview *] 스토리지 버킷 목록에서 백업할 대상 버킷을 선택합니다.
6. 다음 * 을 선택합니다.
7. 백업 요약을 검토하고 * 백업 * 을 선택합니다.

[기술 미리보기**] CR**을 사용하여 백업을 생성합니다

단계

1. CR(사용자 정의 리소스) 파일을 만들고 이름을 지정합니다 `astra-control-backup-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - <CR_NAME>: 이 사용자 지정 리소스의 이름입니다. 환경에 맞는 고유하고 합리적인 이름을 선택하십시오.
   - <APPLICATION_NAME>: 백업할 애플리케이션의 Kubernetes 이름입니다.
   - <APPVAULT_NAME>: 백업 컨텐츠를 저장해야 하는 AppVault의 이름입니다.

```
apiVersion: astra.netapp.io/v1
kind: Backup
metadata:
  namespace: astra-connector
  name: <CR_NAME>
spec:
  applicationRef: <APPLICATION_NAME>
  appVaultRef: <APPVAULT_NAME>
```

2. 를 채운 후 `astra-control-backup-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

```
kubectl apply -f astra-control-backup-cr.yaml
```

결과

Astra Control은 앱 백업을 생성합니다.

- 네트워크에 정전이 발생했거나 비정상적으로 느린 경우 백업 작업이 시간 초과될 수 있습니다. 이로 인해 백업이 실패합니다.
- 실행 중인 백업을 취소해야 하는 경우 의 지침을 따릅니다 백업을 취소합니다. 백업을 삭제하려면 백업이 완료될 때까지 기다린 다음 의 지침을 따르십시오 백업을 삭제합니다.
- 데이터 보호 작업(클론, 백업, 복원)과 후속 영구 볼륨 크기 조정 후 UI에 새 볼륨 크기가 표시되기까지 최대 20분이 지연됩니다. 데이터 보호 작업이 몇 분 내에 성공적으로 완료되며 스토리지 백엔드에 관리 소프트웨어를 사용하여 볼륨 크기 변경을 확인할 수 있습니다.

ONTAP - NAS - 경제성 작업을 위한 백업 및 복원 지원

Astra Control Provisioner는 를 사용하는 스토리지 백엔드에 대해 설정할 수 있는 백업 및 복원 기능을 제공합니다 `ontap-nas-economy` 스토리지 클래스.

시작하기 전에

- Astra Control Provisioner 또는 Astra Trident를 활성화했습니다.
- Astra Control에서 애플리케이션을 정의했습니다. 이 응용 프로그램은 이 절차를 완료할 때까지 제한된 보호 기능을 제공합니다.
- 있습니다 `ontap-nas-economy` 스토리지 백엔드의 기본 스토리지 클래스로 선택됩니다.

구성 단계를 위해 확장합니다

1. ONTAP 스토리지 백엔드에서 다음을 수행합니다.

   a. 를 호스팅하는 SVM을 찾습니다 `ontap-nas-economy`응용 프로그램의 볼륨을 기반으로 합니다.

   b. 볼륨이 생성된 ONTAP에 연결된 터미널에 로그인합니다.

   c. SVM에 대한 스냅샷 디렉토리 숨기기:

   > 이러한 변경은 전체 SVM에 영향을 줍니다. 숨겨진 디렉토리에 계속 액세스할 수 있습니다.

   ```
   nfs modify -vserver <svm name> -v3-hide-snapshot enabled
   ```

   +

   > ONTAP 스토리지 백엔드의 스냅샷 디렉토리가 숨겨져 있는지 확인합니다. 이 디렉토리를 숨기지 않으면 특히 NFSv3을 사용하는 경우에는 애플리케이션에 대한 액세스가 손실될 수 있습니다.

2. Astra Control Provisioner 또는 Astra Trident에서 다음을 수행합니다.

   a. ONTAP-NAS 경제적이며 애플리케이션과 연결된 각 PV에 대해 스냅샷 디렉토리를 사용할 수 있습니다.

   ```
   tridentctl update volume <pv name> --snapshot-dir=true --pool
   -level=true -n trident
   ```

   b. 연결된 각 PV에 대해 스냅샷 디렉토리가 활성화되었는지 확인합니다.

   ```
   tridentctl get volume <pv name> -n trident -o yaml | grep
   snapshotDir
   ```

   응답:

   ```
   snapshotDirectory: "true"
   ```

3. Astra Control에서 연결된 모든 스냅샷 디렉토리를 활성화한 후 애플리케이션을 업데이트하여 Astra Control이 변경된 값을 인식하도록 합니다.

결과

Astra Control을 사용하여 애플리케이션을 백업 및 복원할 준비가 되었습니다. 각 PVC는 백업 및 복원을 위해 다른 응용 프로그램에서 사용할 수도 있습니다.

변경 불가능한 백업을 생성합니다

백업을 저장하는 버킷의 보존 정책에서 금지하는 한 변경 불가능한 백업은 수정, 삭제 또는 덮어쓸 수 없습니다. 보존 정책이 구성된 버킷에 애플리케이션을 백업하여 변경 불가능한 백업을 만들 수 있습니다. 을 참조하십시오 "데이터 보호" 변경 불가능한 백업 작업에 대한 중요한 정보를 참조하십시오.

시작하기 전에

보존 정책을 사용하여 대상 버킷을 구성해야 합니다. 사용하는 스토리지 공급자에 따라 이 방법이 달라집니다. 자세한 내용은 다음 스토리지 제공업체 설명서를 참조하십시오.

- * Amazon Web Services *: "버킷을 생성할 때 S3 오브젝트 잠금을 설정하고 기본 보존 기간으로 기본 보존 모드를 "거버넌스"로 설정합니다".
- * Google Cloud *: "보존 정책을 사용하여 버킷을 구성하고 보존 기간을 지정합니다".
- * Microsoft Azure *: "컨테이너 수준 범위에서 시간 기반 보존 정책을 사용하여 BLOB 스토리지 버킷을 구성합니다".
- * NetApp StorageGRID *: "버킷을 생성할 때 S3 오브젝트 잠금을 설정하고 기본 보존 기간을 사용하여 기본 보존 모드를 "규정 준수"로 설정합니다".

> Astra Control의 버킷은 사용 가능한 용량을 보고하지 않습니다. Astra Control에서 관리되는 앱을 백업 또는 클론 복제하기 전에 적절한 스토리지 관리 시스템에서 버킷 정보를 확인하십시오.

> 앱이 에서 지원하는 저장소 클래스를 사용하는 경우 `ontap-nas-economy` 드라이버, 을(를) 정의했는지 확인하십시오 `backendType` 매개 변수 을 선택합니다 "Kubernetes 스토리지 오브젝트입니다" 을 값으로 사용합니다 `ontap-nas-economy` 보호 작업을 수행하기 전에

단계

1. 응용 프로그램 * 을 선택합니다.
2. 원하는 앱의 * Actions * 열에 있는 옵션 메뉴에서 * Back Up * 을 선택합니다.
3. 백업 이름을 사용자 지정합니다.
4. 기존 스냅샷에서 앱을 백업할지 여부를 선택합니다. 이 옵션을 선택하면 기존 스냅샷 목록에서 선택할 수 있습니다.
5. 스토리지 버킷 목록에서 백업할 대상 버킷을 선택합니다. WORM(Write Once Read Many) 버킷은 버킷 이름 옆에 "잠김" 상태로 표시됩니다.

   > 버켓이 지원되지 않는 유형인 경우 버킷을 가리키거나 선택할 때 표시됩니다.

6. 다음 * 을 선택합니다.
7. 백업 요약을 검토하고 * 백업 * 을 선택합니다.

결과

Astra Control은 앱의 변경 불가능한 백업을 생성한다.

- 네트워크에 정전이 발생했거나 비정상적으로 느린 경우 백업 작업이 시간 초과될 수 있습니다. 이로 인해 백업이 실패합니다.
- 동일한 앱의 변경 불가능한 백업을 두 번 동일한 버킷에 동시에 생성하려는 경우 Astra Control이 두 번째 백업을 시작하지 못합니다. 첫 번째 백업이 완료될 때까지 기다린 후 다른 백업을 시작하십시오.
- 실행 중인 변경 불가능한 백업은 취소할 수 없습니다.
- 데이터 보호 작업(클론, 백업, 복원)과 후속 영구 볼륨 크기 조정 후 UI에 새 볼륨 크기가 표시되기까지 최대 20분이 지연됩니다. 데이터 보호 작업이 몇 분 내에 성공적으로 완료되며 스토리지 백엔드에 관리 소프트웨어를 사용하여 볼륨 크기 변경을 확인할 수 있습니다.

스냅샷 및 백업을 봅니다

Data Protection 탭에서 앱의 스냅샷 및 백업을 볼 수 있습니다.

변경 불가능한 백업은 사용 중인 버킷 옆에 "잠김" 상태로 표시됩니다.

단계

1. 응용 프로그램 * 을 선택한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
2. 데이터 보호 * 를 선택합니다.

   스냅샷은 기본적으로 표시됩니다.

3. 백업 목록을 참조하려면 * backups * 를 선택합니다.

스냅샷을 삭제합니다

더 이상 필요하지 않은 예약된 스냅샷 또는 주문형 스냅샷을 삭제합니다.

단계

1. 응용 프로그램 * 을 선택한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
2. 데이터 보호 * 를 선택합니다.
3. 원하는 스냅샷에 대한 * Actions * 열의 Options 메뉴에서 * Delete snapshot * 을 선택합니다.
4. 삭제를 확인하려면 "delete"라는 단어를 입력하고 * Yes, Delete snapshot * 을 선택합니다.

결과

Astra Control이 스냅샷을 삭제합니다.

백업을 취소합니다

진행 중인 백업을 취소할 수 있습니다.

백업을 취소하려면 백업이 에 있어야 합니다 `Running` 상태. 에 있는 백업은 취소할 수 없습니다 `Pending` 상태.

실행 중인 변경 불가능한 백업은 취소할 수 없습니다.

단계

1. 응용 프로그램 * 을 선택한 다음 앱 이름을 선택합니다.
2. 데이터 보호 * 를 선택합니다.
3. Backups * 를 선택합니다.
4. 원하는 백업에 대한 * Actions * 열의 Options 메뉴에서 * Cancel * 을 선택합니다.
5. 작업을 확인하려면 "취소"라는 단어를 입력하고 * 예, 백업 취소 * 를 선택합니다.

백업을 삭제합니다

더 이상 필요하지 않은 예약된 백업 또는 필요 시 백업을 삭제합니다.

실행 중인 백업을 취소해야 하는 경우 의 지침을 따릅니다 백업을 취소합니다. 백업을 삭제하려면 백업이 완료될 때까지 기다린 다음 이 지침을 따르십시오.

보존 기간이 만료되기 전에는 변경 불가능한 백업을 삭제할 수 없습니다.

단계

1. 응용 프로그램 * 을 선택한 다음 앱 이름을 선택합니다.
2. 데이터 보호 * 를 선택합니다.
3. Backups * 를 선택합니다.
4. 원하는 백업에 대한 * Actions * 열의 Options 메뉴에서 * Delete backup * 을 선택합니다.
5. 삭제를 확인하려면 "delete"라는 단어를 입력하고 * Yes, Delete backup * 을 선택합니다.

결과

Astra Control이 백업을 삭제합니다.

[기술 미리 보기] 전체 클러스터를 보호합니다

클러스터에서 관리되지 않는 네임스페이스의 예약 및 자동 백업을 생성할 수 있습니다. 이러한 워크플로우는 NetApp에서 Kubernetes 서비스 계정, 역할 바인딩 및 cron 작업으로 제공되며 Python 스크립트로 조정됩니다.

작동 방식

전체 클러스터 백업 워크플로우를 구성하고 설치하면 cron 작업이 주기적으로 실행되고 아직 관리되지 않은 모든 네임스페이스를 보호하므로 설치 중에 선택한 일정에 따라 보호 정책이 자동으로 생성됩니다.

전체 클러스터 백업 워크플로우로 클러스터의 모든 관리되지 않는 네임스페이스를 보호하지 않으려면 레이블 기반 백업 워크플로우를 활용할 수 있습니다. 레이블 기반 백업 워크플로는 또한 cron 작업을 사용하지만 관리되지 않는 모든 네임스페이스를 보호하는 대신 브론즈, 실버 또는 골드 백업 정책을 기반으로 선택적으로 네임스페이스를 보호하기 위해 제공하는 레이블을 통해 네임스페이스를 식별합니다.

선택한 워크플로의 범위에 속하는 새 네임스페이스가 만들어지면 관리자 작업 없이 자동으로 보호됩니다. 이러한 워크플로는 클러스터 단위로 구현되므로 클러스터의 중요도에 따라 각 클러스터에서 고유한 보호 수준을 가진 워크플로우를 사용할 수 있습니다.

예: 전체 클러스터 보호

예를 들어, 전체 클러스터 백업 워크플로우를 구성하고 설치하면 네임스페이스의 모든 앱이 관리자의 추가 작업 없이 주기적으로 관리 및 보호됩니다. 워크플로를 설치할 때 네임스페이스가 존재할 필요가 없습니다. 나중에 네임스페이스가 추가되면 네임스페이스가 보호됩니다.

예: 레이블 기반 보호

더 세분화하려면 레이블 기반 워크플로를 사용할 수 있습니다. 예를 들어 이 워크플로를 설치하고 필요한 보호 수준에 따라 보호할 네임스페이스에 여러 레이블 중 하나를 적용하도록 사용자에게 지시할 수 있습니다. 따라서 사용자는 이러한 레이블 중 하나로 네임스페이스를 만들 수 있으며 관리자에게 알릴 필요가 없습니다. 새로운 네임스페이스와 IT 내의 모든 앱이 자동으로 보호됩니다.

모든 네임스페이스의 예약된 백업을 생성합니다

전체 클러스터 백업 워크플로우를 사용하여 클러스터의 모든 네임스페이스에 대해 예약된 백업을 생성할 수 있습니다.

단계

1. 클러스터에 대한 네트워크 액세스 권한이 있는 시스템에 다음 파일을 다운로드합니다.
   - "components.yaml CRD 파일"
   - "protectCluster.py Python 스크립트"
2. 툴킷을 구성하고 설치하려면 "포함된 지침을 따릅니다".

특정 네임스페이스의 예약된 백업을 생성합니다

레이블 기반 백업 워크플로우를 사용하여 레이블을 기준으로 특정 네임스페이스의 예약된 백업을 만들 수 있습니다.

단계

1. 클러스터에 대한 네트워크 액세스 권한이 있는 시스템에 다음 파일을 다운로드합니다.
   - "components.yaml CRD 파일"
   - "protectCluster.py Python 스크립트"
2. 툴킷을 구성하고 설치하려면 "포함된 지침을 따릅니다".

앱 복원

Astra Control은 스냅샷 또는 백업에서 애플리케이션을 복원할 수 있습니다. 애플리케이션을 동일한 클러스터로 복구할 경우 기존 스냅샷에서 복구하는 속도가 빨라집니다. Astra Control UI 또는 를 사용할 수 있습니다 "Astra Control API" 앱을 복원합니다.

복원 또는 클론 작업 후에 실행되는 실행 후크에 네임스페이스 필터를 추가하고 복원 또는 클론 소스와 대상이 서로 다른 네임스페이스에 있는 경우 네임스페이스 필터는 대상 네임스페이스에만 적용됩니다.

시작하기 전에

- * 앱을 먼저 보호 *: 복원하기 전에 응용 프로그램의 스냅샷 또는 백업을 수행하는 것이 좋습니다. 이렇게 하면 복구가 실패할 경우 스냅샷 또는 백업에서 클론을 생성할 수 있습니다.
- * 대상 볼륨 확인 *: 다른 스토리지 클래스로 복원하는 경우 스토리지 클래스가 동일한 영구 볼륨 액세스 모드(예: ReadWriteMany)를 사용하는지 확인합니다. 대상 영구 볼륨 액세스 모드가 다르면 복원 작업이 실패합니다. 예를 들어, 소스 영구 볼륨에서 rwx 액세스 모드를 사용하는 경우 Azure Managed Disks, AWS EBS, Google

Persistent Disk 또는 와 같이 rwx를 제공할 수 없는 대상 스토리지 클래스를 선택합니다 `ontap-san`, 복구 작업이 실패합니다. 영구 볼륨 액세스 모드에 대한 자세한 내용은 를 참조하십시오 "쿠버네티스" 문서화:

- * 공간 요구사항 계획 *: NetApp ONTAP 스토리지를 사용하는 애플리케이션의 데이터 이동 없이 복원을 수행할 경우 복원된 앱에서 사용하는 공간이 두 배로 증가할 수 있습니다. 데이터 이동 없이 복구를 수행한 후 복구된 애플리케이션에서 원치 않는 스냅샷을 모두 제거하여 스토리지 공간을 확보합니다.
- * 지원되는 스토리지 클래스 드라이버 *: Astra Control은 다음 드라이버로 지원되는 스토리지 클래스를 사용하여 백업 복원을 지원합니다.
  - `ontap-nas`
  - `ontap-nas-economy`
  - `ontap-san`
  - `ontap-san-economy`
- * (ONTAP-NAS-이코노미 드라이버만 해당) 백업 및 복원 *: 에서 지원하는 스토리지 클래스를 사용하는 앱을 백업 또는 복원하기 전에 `ontap-nas-economy` 드라이버에서 을 확인합니다 "ONTAP 스토리지 백엔드의 스냅샷 디렉토리가 숨겨집니다". 이 디렉토리를 숨기지 않으면 특히 NFSv3을 사용하는 경우에는 애플리케이션에 대한 액세스가 손실될 수 있습니다.

> 다른 앱과 리소스를 공유하는 앱에서 데이터 이동 없이 복원 작업을 수행하면 의도하지 않은 결과가 발생할 수 있습니다. 앱 간에 공유되는 모든 리소스는 앱 중 하나에서 데이터 이동 없이 복원이 수행될 때 교체됩니다.

단계

1. 응용 프로그램 * 을 선택한 다음 앱 이름을 선택합니다.
2. 작업 열의 옵션 메뉴에서 * 복원 * 을 선택합니다.
3. 복원 유형 선택:
   - * 원래 네임스페이스로 복원 *: 이 절차를 사용하여 원래 클러스터로 응용 프로그램을 원래 상태로 복원할 수 있습니다.
     i. 앱을 원래 상태로 복원하는 데 사용할 스냅샷 또는 백업을 선택합니다. 그러면 앱이 이전 버전으로 되돌아갑니다.
     ii. 다음 * 을 선택합니다.

     > 이전에 삭제된 네임스페이스에 복원하는 경우 복원 프로세스의 일부로 동일한 이름의 새 네임스페이스가 만들어집니다. 이전에 삭제된 네임스페이스에서 앱을 관리할 권한이 있는 사용자는 새로 다시 생성된 네임스페이스에 대한 권한을 수동으로 복원해야 합니다.

   - * 새 네임스페이스로 복원 *: 이 절차를 사용하여 응용 프로그램을 다른 클러스터나 소스의 다른 네임스페이스로 복원할 수 있습니다. 이 절차를 사용하여 앱을 다른 스토리지 클래스로 마이그레이션할 수도 있습니다.
     i. 복원된 앱의 이름을 지정합니다.
     ii. 복원하려는 앱의 대상 클러스터를 선택합니다.
     iii. 앱과 연결된 각 소스 네임스페이스의 대상 네임스페이스를 입력합니다.

     > Astra Control은 이 복원 옵션의 일부로 새 대상 네임스페이스를 만듭니다. 지정한 대상 네임스페이스가 대상 클러스터에 이미 있으면 안 됩니다.

iv. 다음 * 을 선택합니다.

v. 앱을 복원하는 데 사용할 스냅샷 또는 백업을 선택합니다.

vi. 다음 * 을 선택합니다.

vii. 다음 중 하나를 선택합니다.

- * 원래 스토리지 클래스를 사용하여 복원 *: 대상 클러스터에 없는 경우 응용 프로그램은 원래 연결된 스토리지 클래스를 사용합니다. 이 경우 클러스터의 기본 스토리지 클래스가 사용됩니다.
- * 다른 스토리지 클래스를 사용하여 복구 *: 타겟 클러스터에 존재하는 스토리지 클래스를 선택합니다. 원래 연결된 스토리지 클래스에 관계없이 모든 애플리케이션 볼륨은 복구의 일부로 이 서로 다른 스토리지 클래스로 마이그레이션됩니다.

viii. 다음 * 을 선택합니다.

4. 필터링할 자원 선택:
   - * 모든 리소스 복원 *: 원래 앱과 연결된 모든 리소스를 복원합니다.
   - * 필터 리소스 *: 원래 응용 프로그램 리소스의 하위 집합을 복원하는 규칙을 지정합니다.
     i. 복원된 응용 프로그램에서 리소스를 포함하거나 제외하도록 선택합니다.
     ii. 포함 규칙 추가 * 또는 * 제외 규칙 추가 * 를 선택하고 응용 프로그램 복원 중에 올바른 리소스를 필터링하도록 규칙을 구성합니다. 규칙을 편집하거나 제거하고 구성이 올바른지 때까지 규칙을 다시 만들 수 있습니다.

> 포함 및 제외 규칙 구성에 대한 자세한 내용은 을 참조하십시오 응용 프로그램 복원 중에 리소스를 필터링합니다.

5. 다음 * 을 선택합니다.
6. 복원 작업에 대한 세부 정보를 주의 깊게 검토하고 "restore"를 입력하고(메시지가 나타나면) * Restore * 를 선택합니다.

[기술 미리보기] 사용자 지정 리소스(CR)를 사용하여 백업에서 복원

사용자 지정 리소스(CR) 파일을 사용하여 백업에서 데이터를 다른 네임스페이스 또는 원래 소스 네임스페이스로 복원할 수 있습니다.

CR을 사용하여 백업에서 복원합니다

단계

1. CR(사용자 정의 리소스) 파일을 만들고 이름을 지정합니다 `astra-control-backup-restore-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - <CR_NAME>: 이 CR 작업의 이름입니다. 환경에 적합한 이름을 선택하십시오.
   - <APPVAULT_NAME>: 백업 컨텐츠가 저장되는 AppVault의 이름입니다.
   - <BACKUP_PATH>: 백업 컨텐츠가 저장되는 AppVault 내부의 경로입니다. 예를 들면 다음과 같습니다.

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

   - <SOURCE_NAMESPACE>: 복구 작업의 소스 네임스페이스입니다.
   - <DESTINATION_NAMESPACE>: 복구 작업의 대상 네임스페이스입니다.

     ```
     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>"}]
     ```

<stdin>-include:../_include/selective-restore-cr.adoc [ ] 의 해결되지 않은 지시문

1. 를 채운 후 `astra-control-backup-restore-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

   ```
   kubectl apply -f astra-control-backup-restore-cr.yaml
   ```

CR을 사용하여 백업에서 원래 네임스페이스로 복원합니다

단계

1. CR(사용자 정의 리소스) 파일을 만들고 이름을 지정합니다 `astra-control-backup-ipr-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - <CR_NAME>: 이 CR 작업의 이름입니다. 환경에 적합한 이름을 선택하십시오.
   - <APPVAULT_NAME>: 백업 컨텐츠가 저장되는 AppVault의 이름입니다.
   - <BACKUP_PATH>: 백업 컨텐츠가 저장되는 AppVault 내부의 경로입니다. 예를 들면 다음과 같습니다.

```
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>
```

<stdin>-include:../_include/selective-restore-cr.adoc [ ] 의 해결되지 않은 지시문

1. 를 채운 후 `astra-control-backup-ipr-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

```
kubectl apply -f astra-control-backup-ipr-cr.yaml
```

[기술 미리보기] 사용자 정의 리소스**(CR)**를 사용하여 스냅샷에서 복원

사용자 지정 리소스(CR) 파일을 사용하여 스냅샷에서 데이터를 다른 네임스페이스 또는 원래 소스 네임스페이스로 복원할 수 있습니다.

CR을 사용하여 스냅샷에서 복원합니다

단계

1. CR(사용자 정의 리소스) 파일을 만들고 이름을 지정합니다 `astra-control-snapshot-restore-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - <CR_NAME>: 이 CR 작업의 이름입니다. 환경에 적합한 이름을 선택하십시오.
   - <APPVAULT_NAME>: 백업 컨텐츠가 저장되는 AppVault의 이름입니다.
   - <BACKUP_PATH>: 백업 컨텐츠가 저장되는 AppVault 내부의 경로입니다. 예를 들면 다음과 같습니다.

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

   - <SOURCE_NAMESPACE>: 복구 작업의 소스 네임스페이스입니다.
   - <DESTINATION_NAMESPACE>: 복구 작업의 대상 네임스페이스입니다.

     ```
     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>"}]
     ```

<stdin>-include:../_include/selective-restore-cr.adoc [ ] 의 해결되지 않은 지시문

1. 를 채운 후 `astra-control-snapshot-restore-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

   ```
   kubectl apply -f astra-control-snapshot-restore-cr.yaml
   ```

CR을 사용하여 스냅샷에서 원래 네임스페이스로 복원합니다

단계

1. CR(사용자 정의 리소스) 파일을 만들고 이름을 지정합니다 `astra-control-snapshot-ipr-cr.yaml`. 괄호 <> 의 값을 Astra Control 환경 및 클러스터 구성과 일치하도록 업데이트합니다.
   - <CR_NAME>: 이 CR 작업의 이름입니다. 환경에 적합한 이름을 선택하십시오.
   - <APPVAULT_NAME>: 백업 컨텐츠가 저장되는 AppVault의 이름입니다.
   - <BACKUP_PATH>: 백업 컨텐츠가 저장되는 AppVault 내부의 경로입니다. 예를 들면 다음과 같습니다.

```
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>
```

<stdin>-include:../_include/selective-restore-cr.adoc [ ] 의 해결되지 않은 지시문

1. 를 채운 후 `astra-control-snapshot-ipr-cr.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

```
kubectl apply -f astra-control-snapshot-ipr-cr.yaml
```

결과

Astra Control은 사용자가 제공한 정보를 기반으로 앱을 복원합니다. 앱을 제자리에 복원한 경우 기존 영구 볼륨의 콘텐츠가 복원된 앱의 영구 볼륨 콘텐츠로 바뀝니다.

데이터 보호 작업(클론, 백업 또는 복원)과 후속 영구 볼륨 크기 조정 후 웹 UI에 새 볼륨 크기가 표시되기까지 최대 20분이 지연됩니다. 데이터 보호 작업이 몇 분 내에 성공적으로 완료되며 스토리지 백엔드에 관리 소프트웨어를 사용하여 볼륨 크기 변경을 확인할 수 있습니다.

네임스페이스 이름/ID 또는 네임스페이스 레이블에 의해 네임스페이스 제한이 있는 구성원 사용자는 동일한 클러스터 또는 조직 계정의 다른 클러스터에 있는 새 네임스페이스에 앱을 클론 복제하거나 복원할 수 있습니다. 그러나 동일한 사용자가 새 네임스페이스에서 복제되거나 복원된 앱에 액세스할 수 없습니다. 클론 또는 복원 작업에서 새 네임스페이스를 생성한 후 계정 관리자/소유자는 구성원 사용자 계정을 편집하고 영향을 받는 사용자의 역할 제약 조건을 업데이트하여 새 네임스페이스에 대한 액세스 권한을 부여할 수 있습니다.

응용 프로그램 복원 중에 리소스를 필터링합니다

에 필터 규칙을 추가할 수 있습니다 "복원" 복원된 응용 프로그램에서 포함하거나 제외할 기존 응용 프로그램 리소스를 지정하는 작업입니다. 지정된 네임스페이스, 레이블 또는 GVK(GroupVersionKind)를 기반으로 리소스를 포함하거나 제외할 수 있습니다.

포함 및 제외 시나리오에 대해 자세히 알아보세요

- * 원본 네임스페이스가 있는 포함 규칙(원본 위치 복원) * 을 선택합니다. 규칙에 정의된 기존 응용 프로그램 리소스는 삭제되며 복구에 사용하는 선택한 스냅숏 또는 백업의 리소스로 대체됩니다. 포함 규칙에 지정하지 않은 모든 리소스는 변경되지 않습니다.
- * 새 네임스페이스가 있는 포함 규칙 선택 *: 이 규칙을 사용하여 복원된 응용 프로그램에서 원하는 특정 리소스를 선택합니다. 포함 규칙에 지정하지 않은 리소스는 복원된 응용 프로그램에 포함되지 않습니다.
- * 원본 네임스페이스가 있는 제외 규칙(원본 위치 복원) * 선택: 제외하도록 지정한 리소스는 복원되지 않고 변경되지 않습니다. 제외하도록 지정하지 않은 리소스는 스냅샷 또는 백업에서 복구됩니다. 해당 StatefulSet 이 필터링된 리소스의 일부인 경우 영구 볼륨의 모든 데이터가 삭제되고 다시 생성됩니다.
- * 새 네임스페이스가 있는 제외 규칙을 선택합니다. * : 규칙을 사용하여 복원된 응용 프로그램에서 제거할 특정 리소스를 선택합니다. 제외하도록 지정하지 않은 리소스는 스냅샷 또는 백업에서 복구됩니다.

규칙은 포함 또는 제외 유형입니다. 자원 포함과 제외 를 결합하는 규칙은 사용할 수 없습니다.

단계

1. 리소스를 필터링하도록 선택하고 앱 복원 마법사에서 포함 또는 제외 옵션을 선택한 후 * 포함 규칙 추가 * 또는 * 제외 규칙 추가 * 를 선택합니다.

   Astra Control에 의해 자동으로 포함되는 클러스터 범위 리소스는 제외할 수 없습니다.

2. 필터 규칙 구성:

   적어도 하나의 네임스페이스, 레이블 또는 GVK를 지정해야 합니다. 필터 규칙을 적용한 후 유지하는 리소스가 복원된 응용 프로그램을 양호한 상태로 유지하는 데 충분한지 확인합니다.

   a. 규칙의 특정 네임스페이스를 선택합니다. 선택하지 않으면 모든 네임스페이스가 필터에 사용됩니다.

      응용 프로그램에 원래 여러 네임스페이스가 포함되어 있고 이를 새 네임스페이스로 복원하면 리소스에 포함되지 않은 네임스페이스도 모두 만들어집니다.

   b. (선택 사항) 리소스 이름을 입력합니다.

   c. (선택 사항) * 라벨 선택기 *: 포함 "라벨 선택기" 규칙에 추가합니다. 레이블 선택기는 선택한 레이블과 일치하는 자원만 필터링하는 데 사용됩니다.

   d. (선택 사항) 추가 필터링 옵션을 사용하려면 GVK(GroupVersionKind) SET * 를 선택하여 리소스 * 를 필터링합니다.

      GVK 필터를 사용하는 경우 버전 및 종류를 지정해야 합니다.

      i. (선택 사항) * Group *: 드롭다운 목록에서 Kubernetes API 그룹을 선택합니다.

      ii. * Kind *: 드롭다운 목록에서 필터에 사용할 Kubernetes 리소스 유형에 대한 오브젝트 스키마를 선택합니다.

      iii. * 버전 *: Kubernetes API 버전을 선택합니다.

3. 항목에 따라 만들어진 규칙을 검토합니다.

4. 추가 * 를 선택합니다.

원하는 만큼 리소스 포함 및 제외 규칙을 만들 수 있습니다. 작업을 시작하기 전에 복원 애플리케이션 요약에 규칙이 나타납니다.

애플리케이션 클론 복제 및 마이그레이션

기존 앱을 클론 복제하여 동일한 Kubernetes 클러스터 또는 다른 클러스터에 중복 앱을 생성할 수 있습니다. Astra Control은 앱을 클론할 때 애플리케이션 구성 및 영구 스토리지의 클론을 생성합니다.

Kubernetes 클러스터 간에 애플리케이션 및 스토리지를 이동해야 하는 경우 클로닝에 도움이 될 수 있습니다. 예를 들어, CI/CD 파이프라인과 Kubernetes 네임스페이스 전체에서 워크로드를 이동할 수 있습니다.

복원 또는 클론 작업 후에 실행되는 실행 후크에 네임스페이스 필터를 추가하고 복원 또는 클론 소스와 대상이 서로 다른 네임스페이스에 있는 경우 네임스페이스 필터는 대상 네임스페이스에만 적용됩니다.

시작하기 전에

- * 대상 볼륨 확인 *: 다른 스토리지 클래스에 클론을 생성하는 경우 스토리지 클래스가 동일한 영구 볼륨 액세스 모드(예: ReadWriteMany)를 사용하는지 확인합니다. 대상 영구 볼륨 액세스 모드가 다르면 클론 작업이 실패합니다. 예를 들어, 소스 영구 볼륨에서 rwx 액세스 모드를 사용하는 경우 Azure Managed Disks, AWS EBS, Google Persistent Disk 또는 와 같이 rwx를 제공할 수 없는 대상 스토리지 클래스를 선택합니다 `ontap-san`, 클론 작업이 실패합니다. 영구 볼륨 액세스 모드에 대한 자세한 내용은 를 참조하십시오 "쿠버네티스" 문서화:
- 앱을 다른 클러스터에 클론 복제하려면 소스 클러스터가 포함된 클라우드 인스턴스에 대한 기본 버킷을 할당해야 합니다. 소스 클라우드 인스턴스에 기본 버킷 세트가 없으면 클러스터 간 클론 작업이 실패합니다.
- 클론 작업 중에 IngressClass 리소스 또는 Webhook가 필요한 애플리케이션에는 대상 클러스터에 이미 정의된 리소스가 없어야 합니다.

클론 제한 사항

- * 명시적 스토리지 클래스 *: 스토리지 클래스가 명시적으로 설정된 앱을 배포하고 앱을 복제해야 하는 경우 타겟 클러스터에 원래 지정된 스토리지 클래스가 있어야 합니다. 명시적으로 설정된 스토리지 클래스를 가진 애플리케이션을 동일한 스토리지 클래스가 없는 클러스터로 클론 복제하면 실패합니다.
- * ONTAP - NAS - 경제적인 응용 프로그램 *: 응용 프로그램의 저장소 클래스가 에서 지원될 경우 복제 작업을 사용할 수 없습니다 `ontap-nas-economy` 드라이버. 그러나, "ONTAP - NAS - 경제성 작업을 위한 백업 및 복원 지원".
- * 클론 및 사용자 제약 조건 *: 네임스페이스 이름/ID 또는 네임스페이스 레이블에 의해 네임스페이스 제한이 있는 구성원 사용자는 동일한 클러스터의 새 네임스페이스 또는 조직 계정의 다른 클러스터에 앱을 클론 복제하거나 복원할 수 있습니다. 그러나 동일한 사용자가 새 네임스페이스에서 복제되거나 복원된 앱에 액세스할 수 없습니다. 클론 또는 복원 작업에서 새 네임스페이스를 생성한 후 계정 관리자/소유자는 구성원 사용자 계정을 편집하고 영향을 받는 사용자의 역할 제약 조건을 업데이트하여 새 네임스페이스에 대한 액세스 권한을 부여할 수 있습니다.
- * 클론은 기본 버킷 사용 *:
  - 애플리케이션 백업 또는 애플리케이션 복원 중에 사용할 버킷을 지정할 수 있습니다. 클러스터 간에 클론을 생성할 때 기본 버킷을 지정해야 하지만 같은 클러스터 내에서 클론을 생성할 때는 버킷을 지정할 필요가 없습니다.
  - 클러스터 간에 클론을 생성할 때 클론 작업의 소스 클러스터가 포함된 클라우드 인스턴스에 기본 버킷 세트가 있어야 합니다.

◦ 클론의 버킷을 변경할 수 있는 옵션은 없습니다. 어떤 버킷이 사용되는지 제어하려는 경우 이 두 가지 방법을 사용할 수 있습니다 "버킷 기본값을 변경합니다" 또는 을 수행합니다 "백업" 뒤에 가 있습니다 "복원" 별도.

- * Jenkins CI * 사용: Jenkins CI의 운영자 배포 인스턴스를 복제하는 경우 영구 데이터를 수동으로 복원해야 합니다. 이는 앱 배포 모델의 제한 사항입니다.

단계

1. 응용 프로그램 * 을 선택합니다.
2. 다음 중 하나를 수행합니다.
   ◦ 원하는 앱의 * Actions * 열에서 Options 메뉴를 선택합니다.
   ◦ 원하는 앱의 이름을 선택하고 페이지 오른쪽 상단의 상태 드롭다운 목록을 선택합니다.
3. 클론 * 을 선택합니다.
4. 클론의 세부 정보 지정:
   ◦ 이름을 입력합니다.
   ◦ 클론의 대상 클러스터를 선택합니다.
   ◦ 클론의 대상 네임스페이스를 입력합니다. 앱과 연결된 각 소스 네임스페이스는 대상 네임스페이스에 매핑됩니다.

   Astra Control은 클론 작업의 일부로 새 대상 네임스페이스를 생성합니다. 지정한 대상 네임스페이스가 대상 클러스터에 이미 있으면 안 됩니다.

   ◦ 다음 * 을 선택합니다.
   ◦ 앱과 연결된 원래 저장소 클래스를 유지하거나 다른 저장소 클래스를 선택합니다.

   앱의 스토리지 클래스를 기본 클라우드 공급자 스토리지 클래스나 기타 지원되는 스토리지 클래스로 마이그레이션하고 에서 지원하는 스토리지 클래스에서 앱을 마이그레이션할 수 있습니다 `ontap-nas-economy` 에서 지원하는 스토리지 클래스로 `ontap-nas` 또는 에서 지원하는 저장소 클래스가 있는 다른 클러스터로 앱을 복사합니다 `ontap-nas-economy` 드라이버.

   다른 스토리지 클래스를 선택했고 복원 시 이 스토리지 클래스가 존재하지 않는 경우 오류가 반환됩니다.

5. 다음 * 을 선택합니다.
6. 클론에 대한 정보를 검토하고 * Clone * 을 선택합니다.

결과

Astra Control은 사용자가 제공한 정보를 기반으로 앱을 복제합니다. 새 애플리케이션 클론이 에 있을 때 클론 작업이 성공적으로 수행됩니다 `Healthy` 상태를 표시합니다.

클론 또는 복원 작업에서 새 네임스페이스를 생성한 후 계정 관리자/소유자는 구성원 사용자 계정을 편집하고 영향을 받는 사용자의 역할 제약 조건을 업데이트하여 새 네임스페이스에 대한 액세스 권한을 부여할 수 있습니다.

앱 실행 후크 관리

실행 후크는 관리되는 앱의 데이터 보호 작업과 함께 실행되도록 구성할 수 있는 사용자 지정 작업입니다. 예를 들어 데이터베이스 앱이 있는 경우 실행 후크를 사용하여 스냅샷 전에 모든 데이터베이스 트랜잭션을 일시 중지하고 스냅샷이 완료된 후 트랜잭션을 다시 시작할 수 있습니다. 따라서 애플리케이션 정합성이 보장되는 스냅샷이 보장됩니다.

실행 후크 유형

Astra Control Service는 실행 가능한 시점을 기준으로 다음과 같은 유형의 실행 후크를 지원합니다.

- 사전 스냅샷
- 사후 스냅샷
- 사전 백업
- 백업 후
- 사후 복원

실행 후크 필터

실행 후크를 응용 프로그램에 추가하거나 편집할 때 실행 후크에 필터를 추가하여 후크가 일치시킬 컨테이너를 관리할 수 있습니다. 필터는 모든 컨테이너에서 동일한 컨테이너 이미지를 사용하는 응용 프로그램에 유용하지만 각 이미지를 다른 용도(예: Elasticsearch)로 사용할 수 있습니다. 필터를 사용하면 실행 후크가 실행되는 시나리오를 만들 수 있습니다. 단, 모든 동일한 컨테이너를 실행하는 것은 아닙니다. 단일 실행 후크에 대해 여러 개의 필터를 만들면 논리적 필터 및 연산자와 결합됩니다. 실행 후크당 최대 10개의 활성 필터를 사용할 수 있습니다.

실행 후크에 추가하는 각 필터는 클러스터의 컨테이너와 일치시키기 위해 정규식을 사용합니다. 후크가 컨테이너와 일치하면 후크는 해당 컨테이너에서 연결된 스크립트를 실행합니다. 필터에 대한 정규식은 정규식 2(RE2) 구문을 사용합니다. 이 구문은 일치 목록에서 컨테이너를 제외하는 필터를 만드는 것을 지원하지 않습니다. Astra Control이 실행 후크 필터의 정규식에 대해 지원하는 구문에 대한 자세한 내용은 을 참조하십시오 "정규식 2(RE2) 구문 지원".

> 복원 또는 클론 작업 후에 실행되는 실행 후크에 네임스페이스 필터를 추가하고 복원 또는 클론 소스와 대상이 서로 다른 네임스페이스에 있는 경우 네임스페이스 필터는 대상 네임스페이스에만 적용됩니다.

사용자 정의 실행 후크에 대한 중요 참고 사항

앱에 대한 실행 후크를 계획할 때 다음 사항을 고려하십시오.

> 실행 후크는 실행 중인 응용 프로그램의 기능을 줄이거나 완전히 비활성화하기 때문에 사용자 지정 실행 후크가 실행되는 시간을 최소화해야 합니다.
> 연결된 실행 후크와 함께 백업 또는 스냅샷 작업을 시작한 다음 취소하면 백업 또는 스냅샷 작업이 이미 시작된 경우에도 후크를 실행할 수 있습니다. 즉, 백업 후 실행 후크에 사용되는 논리는 백업이 완료된 것으로 가정할 수 없습니다.

- 새 Astra Control 구축 환경에서는 실행 후크 기능이 기본적으로 비활성화되어 있습니다.
  - 실행 후크를 사용하려면 먼저 실행 후크 기능을 활성화해야 합니다.
  - 소유자 또는 관리자 사용자는 현재 Astra Control 계정에 정의된 모든 사용자에 대해 실행 후크 기능을 활성화하거나 비활성화할 수 있습니다. 을 참조하십시오 실행 후크 기능을 활성화합니다 및 실행 후크 기능을

비활성화합니다 를 참조하십시오.

  - Astra Control 업그레이드 중에 기능 지원 상태가 유지됩니다.

- 실행 후크는 스크립트를 사용하여 작업을 수행해야 합니다. 많은 실행 후크가 동일한 스크립트를 참조할 수 있습니다.
- Astra Control에는 실행 후크가 실행 가능한 셸 스크립트 형식으로 기록하는 데 사용하는 스크립트가 필요합니다.
- 스크립트 크기는 96KB로 제한됩니다.
- Astra Control은 실행 후크 설정과 모든 일치 기준을 사용하여 스냅샷, 백업 또는 복구 작업에 적용할 수 있는 후크를 결정합니다.
- 모든 실행 후크 장애는 소프트 장애이며, 후크가 실패하더라도 다른 후크와 데이터 보호 작업은 계속 시도됩니다. 그러나 후크가 실패하면 * Activity * 페이지 이벤트 로그에 경고 이벤트가 기록됩니다.
- 실행 후크를 생성, 편집 또는 삭제하려면 소유자, 관리자 또는 구성원 권한이 있는 사용자여야 합니다.
- 실행 후크를 실행하는 데 25분 이상 걸리는 경우 후크에 장애가 발생하고 반환 코드가 "N/A"인 이벤트 로그 항목이 생성됩니다. 영향을 받는 모든 스냅샷은 시간 초과되어 실패로 표시되며, 그 결과 이벤트 로그 항목이 시간 초과를 나타냅니다.
- 임시 데이터 보호 작업의 경우 모든 후크 이벤트가 생성되고 * Activity * 페이지 이벤트 로그에 저장됩니다. 그러나 예약된 데이터 보호 작업의 경우 후크 장애 이벤트만 이벤트 로그에 기록됩니다(예약된 데이터 보호 작업 자체에서 생성되는 이벤트는 계속 기록됨).

실행 순서

데이터 보호 작업이 실행되면 실행 후크 이벤트가 다음 순서로 발생합니다.

1. 해당되는 모든 사용자 정의 사전 작업 실행 후크는 해당 컨테이너에서 실행됩니다. 필요한 만큼 사용자 지정 사전 작업 후크를 만들고 실행할 수 있지만, 이 후크의 실행 순서는 보장되거나 구성할 수 없습니다.
2. 데이터 보호 작업이 수행됩니다.
3. 해당되는 모든 사용자 지정 작업 후 실행 후크는 해당 컨테이너에서 실행됩니다. 필요한 만큼 사용자 지정 사후 작업 후크를 만들고 실행할 수 있지만 작업 후 후크의 실행 순서는 보장되거나 구성할 수 없습니다.

같은 유형의 실행 후크를 여러 개 생성하는 경우(예: 사전 스냅샷) 해당 후크의 실행 순서는 보장되지 않습니다. 그러나 다른 유형의 후크를 실행하는 순서는 보장됩니다. 예를 들어, 서로 다른 모든 유형의 후크가 있는 구성의 실행 순서는 다음과 같습니다.

1. 예비 후크가 실행되었습니다
2. 사전 스냅샷 후크가 실행되었습니다
3. 사후 스냅샷 후크가 실행되었습니다
4. 백업 후 후크가 실행되었습니다
5. 복원 후 후크가 실행되었습니다

시나리오 번호 2에서 이 구성의 예를 볼 수 있습니다 후크가 실행될지 여부를 결정합니다.

운영 환경에서 실행 후크 스크립트를 사용하려면 항상 해당 스크립트를 테스트해야 합니다. 'kubbeck exec' 명령을 사용하여 스크립트를 편리하게 테스트할 수 있습니다. 운영 환경에서 실행 후크를 사용하도록 설정한 후 결과 스냅샷과 백업을 테스트하여 정합성이 보장되는지 확인합니다. 앱을 임시 네임스페이스에 클론 복제하고, 스냅샷 또는 백업을 복원한 다음 앱을 테스트하여 이 작업을 수행할 수 있습니다.

다음 표를 사용하여 사용자 지정 실행 후크가 앱에 대해 실행되는지 여부를 확인할 수 있습니다.

모든 상위 수준 앱 작업은 스냅샷, 백업 또는 복원의 기본 작업 중 하나를 실행하는 것으로 구성됩니다. 시나리오에 따라 클론 작업은 이러한 작업의 다양한 조합으로 구성되므로 클론 작업이 실행되는 실행 후크는 달라집니다.

데이터 이동 없이 복원 작업을 수행하려면 기존 스냅샷 또는 백업이 필요하므로 이러한 작업은 스냅샷 또는 백업 후크를 실행하지 않습니다.

를 시작한 다음 스냅샷이 포함된 백업을 취소하고 연결된 실행 후크가 있는 경우 일부 후크가 실행될 수 있고 그렇지 않은 백업이 있을 수 있습니다. 즉, 백업 후 실행 후크는 백업이 완료된 것으로 가정할 수 없습니다. 연결된 실행 후크와 함께 취소된 백업의 경우 다음 사항에 유의하십시오.

- 예비 백업 및 예비 후크는 항상 실행됩니다.
- 백업에 새 스냅샷이 포함되어 있고 스냅샷이 시작된 경우 사전 스냅샷 및 사후 스냅샷 후크가 실행됩니다.
- 스냅샷을 시작하기 전에 백업을 취소하면 사전 스냅샷 및 사후 스냅샷 후크가 실행되지 않습니다.

시나리오	작동	기존 스냅샷	더 많은 워크로드 추가/제거	네임스페이스	클러스터	스냅샷 후크가 실행됩니다	백업 후크가 실행됩니다	후크 실행을 복원합니다
1	복제	해당 없음	해당 없음	신규	동일합니다	예	해당 없음	예
2	복제	해당 없음	해당 없음	신규	다릅니다	예	예	예
3	복제 또는 복원	예	해당 없음	신규	동일합니다	해당 없음	해당 없음	예
4	복제 또는 복원	해당 없음	예	신규	동일합니다	해당 없음	해당 없음	예
5	복제 또는 복원	예	해당 없음	신규	다릅니다	해당 없음	해당 없음	예
6	복제 또는 복원	해당 없음	예	신규	다릅니다	해당 없음	해당 없음	예
7	복원	예	해당 없음	기존	동일합니다	해당 없음	해당 없음	예
8	복원	해당 없음	예	기존	동일합니다	해당 없음	해당 없음	예
9	스냅샷	해당 없음	해당 없음	해당 없음	해당 없음	예	해당 없음	해당 없음
10	백업	해당 없음	해당 없음	해당 없음	해당 없음	예	예	해당 없음
11	백업	예	해당 없음	해당 없음	해당 없음	해당 없음	해당 없음	해당 없음

실행 후크 예

를 방문하십시오 "NetApp Verda GitHub 프로젝트" Apache Cassandra 및 Elasticsearch와 같은 인기 있는 앱의 실제 실행 후크를 다운로드하려면 다음을 수행합니다. 예제를 보고 사용자 지정 실행 후크를 구조화하는 아이디어를 얻을 수도 있습니다.

실행 후크 기능을 활성화합니다

소유자 또는 관리자 사용자인 경우 실행 후크 기능을 활성화할 수 있습니다. 이 기능을 활성화하면 이 Astra Control 계정에 정의된 모든 사용자가 실행 후크를 사용하고 기존 실행 후크와 후크 스크립트를 볼 수 있습니다.

단계

1. 응용 프로그램 * 으로 이동한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
2. Execution hook * 탭을 선택합니다.
3. 실행 후크 활성화 * 를 선택합니다.

   계정 * > * 기능 설정 * 탭이 나타납니다.
4. Execution Hooks * 창에서 설정 메뉴를 선택합니다.
5. 활성화 * 를 선택합니다.
6. 나타나는 보안 경고를 확인합니다.
7. Yes, enable execution hook * 를 선택합니다.

실행 후크 기능을 비활성화합니다

소유자 또는 관리자 사용자인 경우 이 Astra Control 계정에 정의된 모든 사용자에 대해 실행 후크 기능을 비활성화할 수 있습니다. 실행 후크 기능을 비활성화하려면 먼저 기존 실행 후크를 모두 삭제해야 합니다. 을 참조하십시오 실행 후크를 삭제합니다 기존 실행 후크를 삭제하는 방법에 대한 지침은 을 참조하십시오.

단계

1. 계정 * 으로 이동한 다음 * 기능 설정 * 탭을 선택합니다.
2. Execution hook * 탭을 선택합니다.
3. Execution Hooks * 창에서 설정 메뉴를 선택합니다.
4. 비활성화 * 를 선택합니다.
5. 나타나는 경고를 확인합니다.
6. 유형 `disable` 모든 사용자에 대해 이 기능을 사용하지 않도록 설정할 것인지 확인합니다.
7. 예, 사용 안 함 * 을 선택합니다.

기존 실행 후크를 봅니다

앱의 기존 사용자 지정 실행 후크를 볼 수 있습니다.

단계

1. 응용 프로그램 * 으로 이동한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
2. Execution hook * 탭을 선택합니다.

   결과 목록에서 사용 가능하거나 비활성화된 실행 후크를 모두 볼 수 있습니다. 후크의 상태, 일치하는 컨테이너 수, 생성 시간 및 실행 시간(사전 또는 사후 작업)을 확인할 수 있습니다. 를 선택할 수 있습니다 + 실행할 컨테이너 목록을 확장하려면 후크 이름 옆에 있는 아이콘을 클릭합니다. 이 응용 프로그램의 실행 후크를 둘러싼 이벤트 로그를 보려면 * Activity * 탭으로 이동하십시오.

기존 스크립트 보기

업로드된 기존 스크립트를 볼 수 있습니다. 또한 이 페이지에서 사용 중인 스크립트와 해당 스크립트를 사용하는 후크를 확인할 수 있습니다.

단계

1. 계정 * 으로 이동합니다.
2. 스크립트 * 탭을 선택합니다.

   이 페이지에서는 업로드된 기존 스크립트 목록을 볼 수 있습니다. Used By* 열에는 각 스크립트를 사용하는 실행 후크가 표시됩니다.

스크립트를 추가합니다

각 실행 후크는 스크립트를 사용하여 작업을 수행해야 합니다. 실행 후크가 참조할 수 있는 스크립트를 하나 이상 추가할 수 있습니다. 많은 실행 후크가 동일한 스크립트를 참조할 수 있으므로 하나의 스크립트만 변경하여 여러 실행 후크를 업데이트할 수 있습니다.

단계

1. 실행 후크 기능이 인지 확인합니다 활성화됨.
2. 계정 * 으로 이동합니다.
3. 스크립트 * 탭을 선택합니다.
4. 추가 * 를 선택합니다.
5. 다음 중 하나를 수행합니다.
   - 사용자 지정 스크립트를 업로드합니다.
     i. 파일 업로드 * 옵션을 선택합니다.
     ii. 파일을 찾아 업로드합니다.
     iii. 스크립트에 고유한 이름을 지정합니다.
     iv. (선택 사항) 다른 관리자가 스크립트에 대해 알아야 하는 참고 사항을 입력합니다.
     v. Save script * 를 선택합니다.
   - 클립보드에서 사용자 정의 스크립트를 붙여 넣습니다.
     i. 붙여넣기 또는 형식 * 옵션을 선택합니다.
     ii. 텍스트 필드를 선택하고 필드에 스크립트 텍스트를 붙여 넣습니다.
     iii. 스크립트에 고유한 이름을 지정합니다.
     iv. (선택 사항) 다른 관리자가 스크립트에 대해 알아야 하는 참고 사항을 입력합니다.
6. Save script * 를 선택합니다.

결과

새 스크립트가 * 스크립트 * 탭의 목록에 나타납니다.

스크립트를 삭제합니다

스크립트가 더 이상 필요하지 않고 실행 후크에서 사용되지 않는 경우 시스템에서 스크립트를 제거할 수 있습니다.

단계

1. 계정 * 으로 이동합니다.
2. 스크립트 * 탭을 선택합니다.
3. 제거할 스크립트를 선택하고 * Actions * 열에서 메뉴를 선택합니다.
4. 삭제 * 를 선택합니다.

> ⓘ 스크립트가 하나 이상의 실행 후크에 연결되어 있으면 * 삭제 * 작업을 사용할 수 없습니다. 스크립트를 삭제하려면 먼저 연결된 실행 후크를 편집하여 다른 스크립트에 연결합니다.

사용자 지정 실행 후크를 만듭니다

앱에 대한 사용자 정의 실행 후크를 생성하여 Astra Control에 추가할 수 있습니다. 을 참조하십시오 실행 후크 예 후크 예 실행 후크를 만들려면 소유자, 관리자 또는 구성원 권한이 있어야 합니다.

> ⓘ 실행 후크로 사용할 사용자 정의 쉘 스크립트를 작성할 때는 특정 명령을 실행하거나 실행 파일에 대한 전체 경로를 제공하지 않는 한 파일 시작 부분에 적절한 셸을 지정해야 합니다.

단계

1. 실행 후크 기능이 인지 확인합니다 활성화됨.
2. 응용 프로그램 * 을 선택한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
3. Execution hook * 탭을 선택합니다.
4. 추가 * 를 선택합니다.
5. 후크 세부 정보 * 영역에서:
   a. 작업 * 드롭다운 메뉴에서 작업 유형을 선택하여 후크를 언제 실행해야 하는지 결정합니다.
   b. 후크의 고유한 이름을 입력합니다.
   c. (선택 사항) 실행 중에 후크에 전달할 인수를 입력하고 각 인수 뒤에 Enter 키를 눌러 각 인수를 기록합니다.
6. (선택 사항) * Hook Filter Details * 영역에서 실행 후크가 실행되는 컨테이너를 제어하는 필터를 추가할 수 있습니다.
   a. 필터 추가 * 를 선택합니다.
   b. Hook filter type * 열의 드롭다운 메뉴에서 필터링할 특성을 선택합니다.
   c. Regex * 열에 필터로 사용할 정규식을 입력합니다. Astra Control은 를 사용합니다 "정규식 2(RE2) regex 구문".

      > ⓘ 정규식 필드에 다른 텍스트가 없는 특성(예: pod 이름)의 정확한 이름을 필터링하면 부분 문자열 일치가 수행됩니다. 정확한 이름과 해당 이름만 일치시키려면 정확한 문자열 일치 구문(예: `^exact_podname$`)를 클릭합니다.

   d. 필터를 더 추가하려면 * 필터 추가 * 를 선택합니다.

실행 후크에 대한 여러 필터가 논리 및 연산자와 결합됩니다. 실행 후크당 최대 10개의 활성 필터를 사용할 수 있습니다.

7. 완료되면 * Next * 를 선택합니다.
8. Script * 영역에서 다음 중 하나를 수행합니다.
   - 새 스크립트를 추가합니다.
     i. 추가 * 를 선택합니다.
     ii. 다음 중 하나를 수행합니다.
     - 사용자 지정 스크립트를 업로드합니다.
       I. 파일 업로드 * 옵션을 선택합니다.
       II. 파일을 찾아 업로드합니다.
       III. 스크립트에 고유한 이름을 지정합니다.
       IV. (선택 사항) 다른 관리자가 스크립트에 대해 알아야 하는 참고 사항을 입력합니다.
       V. Save script * 를 선택합니다.
     - 클립보드에서 사용자 정의 스크립트를 붙여 넣습니다.
       I. 붙여넣기 또는 형식 * 옵션을 선택합니다.
       II. 텍스트 필드를 선택하고 필드에 스크립트 텍스트를 붙여 넣습니다.
       III. 스크립트에 고유한 이름을 지정합니다.
       IV. (선택 사항) 다른 관리자가 스크립트에 대해 알아야 하는 참고 사항을 입력합니다.
   - 목록에서 기존 스크립트를 선택합니다.

     이렇게 하면 실행 후크에 이 스크립트를 사용하도록 지시합니다.
9. 다음 * 을 선택합니다.
10. 실행 후크 구성을 검토합니다.
11. 추가 * 를 선택합니다.

실행 후크의 상태를 확인합니다

스냅샷, 백업 또는 복원 작업이 실행된 후에 작업의 일부로 실행된 실행 후크의 상태를 확인할 수 있습니다. 이 상태 정보를 사용하여 실행 후크를 유지할지, 수정하거나 삭제할 것인지 결정할 수 있습니다.

단계

1. 응용 프로그램 * 을 선택한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
2. 데이터 보호 * 탭을 선택합니다.
3. 스냅샷 * 을 선택하여 실행 중인 스냅샷을 보거나 * 백업 * 을 선택하여 실행 중인 백업을 확인합니다.

   후크 상태 * 는 작업이 완료된 후 실행 후크의 상태를 표시합니다. 상태 위로 마우스를 가져가면 자세한 정보를 볼 수 있습니다. 예를 들어, 스냅샷 중에 실행 후크 오류가 발생한 경우 해당 스냅샷의 후크 상태 위로 마우스를 이동하면 실패한 실행 후크 목록이 표시됩니다. 각 오류의 원인을 확인하려면 왼쪽 탐색 영역의 * Activity * 페이지를 확인하십시오.

스크립트 사용을 봅니다

Astra Control 웹 UI에서 특정 스크립트를 사용하는 실행 후크를 확인할 수 있습니다.

단계

1. 계정 * 을 선택합니다.
2. 스크립트 * 탭을 선택합니다.

   스크립트 목록의 * Used By * 열에 목록의 각 스크립트를 사용하는 후크에 대한 세부 정보가 포함되어 있습니다.

3. 관심 있는 스크립트에 대해 * Used By *(사용 대상 *) 열에서 정보를 선택합니다.

   스크립트를 사용하는 후크의 이름 및 스크립트를 실행하도록 구성된 작업 유형과 함께 더 자세한 목록이 나타납니다.

실행 후크를 편집합니다

실행 후크를 편집하여 속성, 필터 또는 사용하는 스크립트를 변경할 수 있습니다. 실행 후크를 편집하려면 소유자, 관리자 또는 구성원 권한이 있어야 합니다.

단계

1. 응용 프로그램 * 을 선택한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
2. Execution hook * 탭을 선택합니다.
3. 편집할 후크의 경우 * Actions * 열에서 옵션 메뉴를 선택합니다.
4. 편집 * 을 선택합니다.
5. 필요한 사항을 변경하고 각 섹션을 완료한 후 * 다음 * 을 선택합니다.
6. 저장 * 을 선택합니다.

실행 후크를 비활성화합니다

앱 스냅샷 전후에 실행 후크가 실행되지 않도록 임시로 설정하려면 실행 후크를 사용하지 않도록 설정할 수 있습니다. 실행 후크를 비활성화하려면 소유자, 관리자 또는 구성원 권한이 있어야 합니다.

단계

1. 응용 프로그램 * 을 선택한 다음 관리되는 응용 프로그램의 이름을 선택합니다.
2. Execution hook * 탭을 선택합니다.
3. 비활성화할 후크의 경우 * Actions * 열에서 옵션 메뉴를 선택합니다.
4. 비활성화 * 를 선택합니다.

실행 후크를 삭제합니다

더 이상 필요 없는 경우 실행 후크를 완전히 제거할 수 있습니다. 실행 후크를 삭제하려면 소유자, 관리자 또는 구성원 권한이 있어야 합니다.

단계

1. 응용 프로그램 * 을 선택한 다음 관리되는 응용 프로그램의 이름을 선택합니다.

2. Execution hook * 탭을 선택합니다.
3. 삭제할 후크의 경우 * Actions * 열에서 옵션 메뉴를 선택합니다.
4. 삭제 * 를 선택합니다.
5. 결과 대화 상자에 "delete"를 입력하여 확인합니다.
6. 예, 실행 후크 삭제 * 를 선택합니다.

를 참조하십시오

- "NetApp Verda GitHub 프로젝트"

앱 및 컴퓨팅 상태 보기

앱 및 클러스터 상태 요약 보기

대시보드 * 를 클릭하여 앱, 클러스터 및 해당 상태를 한눈에 파악할 수 있습니다.

앱 타일은 다음을 식별하는 데 도움이 됩니다.

- 현재 관리하고 있는 앱 수
- 관리된 앱이 정상 상태인지 여부
- 애플리케이션이 완전히 보호되는지 여부(최근 백업을 사용할 수 있는 경우 보호됨)

이러한 항목은 단순한 숫자 또는 상태가 아니라 각 상태별로 드릴다운할 수 있습니다. 예를 들어 앱이 완전히 보호되지 않은 경우 아이콘 위로 마우스를 가져가면 완전히 보호되지 않은 앱을 확인할 수 있습니다. 여기에는 이유가 포함됩니다.

클러스터 타일은 클러스터 상태에 대한 유사한 세부 정보를 제공하며 앱을 사용하는 것처럼 드릴다운하여 자세한 정보를 얻을 수 있습니다.

클러스터의 상태 및 세부 정보를 봅니다

Astra Control에 Kubernetes 클러스터를 추가한 후에는 클러스터의 위치, 작업자 노드, 영구 볼륨 및 스토리지 클래스 등과 같은 클러스터에 대한 세부 정보를 볼 수 있습니다.

단계

1. Astra Control Service UI에서 * Clusters * 를 선택합니다.
2. 클러스터 * 페이지에서 세부 정보를 확인할 클러스터를 선택합니다.

> 클러스터가 "이동된" 상태에 있지만 클러스터 및 네트워크 연결이 양호한 것으로 나타나는 경우(Kubernetes API를 사용하여 클러스터에 액세스하려는 외부 시도가 성공한 경우), Astra Control에 제공한 kubeconfig는 더 이상 유효하지 않을 수 있습니다. 클러스터의 인증서 순환 또는 만료 때문일 수 있습니다. 이 문제를 해결하려면 을 사용하여 Astra Control의 클러스터와 연결된 자격 증명을 업데이트하십시오 "Astra Control API를 참조하십시오".

3. Overview *, * Storage * 및 * Activity * 탭에서 원하는 정보를 확인할 수 있습니다.
    - * 개요 *: 해당 상태를 포함한 작업자 노드에 대한 세부 정보.

- * 스토리지 *: 스토리지 클래스 및 상태를 비롯하여 컴퓨팅과 연관된 영구 볼륨입니다.
- * 활동 *: 클러스터와 관련된 활동

Astra Control Service * Dashboard * 부터 클러스터 정보를 볼 수도 있습니다. 리소스 요약 * 의 * 클러스터 * 탭에서 * 클러스터 * 페이지로 이동하는 관리 클러스터를 선택할 수 있습니다. 클러스터 * 페이지로 이동한 후 위에 설명된 단계를 따릅니다.

앱의 상태 및 세부 정보를 봅니다

앱 관리를 시작하면 Astra Control은 앱에 대한 세부 정보를 제공하여 통신 상태(Astra Control이 앱과 통신할 수 있는지 여부), 보호 상태(장애 발생 시 완전히 보호되는지 여부), Pod, 영구 스토리지 등을 식별할 수 있도록 합니다.

단계

1. 응용 프로그램 * 을 선택한 다음 앱 이름을 선택합니다.
2. 원하는 정보를 찾습니다.

앱 상태
: Astra Control이 애플리케이션과 통신할 수 있는지 여부를 나타내는 상태를 제공한다.

앱 보호 상태
: 앱이 얼마나 잘 보호되는지 상태를 제공합니다.

  - * 완전 보호 *: 이 앱에는 활성 백업 스케줄과 1주일 미만의 성공적인 백업이 있습니다
  - * 부분 보호됨 *: 응용 프로그램에 활성 백업 일정, 활성 스냅샷 일정 또는 백업 또는 스냅샷이 있습니다
  - * 보호되지 않음 *: 완전히 보호되거나 부분적으로 보호되지 않는 앱

    _최근 백업 _ 이(가) 있을 때까지 완전히 보호할 수 없습니다. 백업은 영구 볼륨으로부터 멀리 떨어진 개체 저장소에 저장되기 때문에 이 작업이 중요합니다. 장애 또는 사고로 인해 클러스터가 삭제되며 영구적 저장소인 경우 복구할 백업이 필요합니다. 스냅샷을 사용하면 복구할 수 없습니다.

개요
: 앱과 연결된 Pod의 상태에 대한 정보입니다.

데이터 보호
: 데이터 보호 정책을 구성하고 기존 스냅샷 및 백업을 볼 수 있습니다.

스토리지
: 에는 애플리케이션 레벨의 영구 볼륨이 나와 있습니다. 영구 볼륨의 상태는 Kubernetes 클러스터의 관점에서 나옵니다.

리소스
: 백업 및 관리되는 리소스를 확인할 수 있습니다.

활동입니다
: 앱과 관련된 Astra Control 활동입니다.

버킷을 관리합니다

Astra가 백업 및 클론에 사용하는 버킷을 관리할 수 있습니다. 클라우드 인스턴스에서 추가 버킷을 추가하고, 기존 버킷을 제거하고, Kubernetes 클러스터의 기본 버킷을 변경할 수 있습니다.

소유자 및 관리자만 버킷을 관리할 수 있습니다.

Astra Control이 버킷을 사용하는 방식

클라우드 인스턴스를 위한 첫 번째 Kubernetes 클러스터 관리를 시작할 때 Astra Control Service는 이를 위한 초기 버킷을 생성합니다 "클라우드 인스턴스".

수동으로 버킷을 클라우드 인스턴스의 기본 버킷으로 지정할 수 있습니다. 이 경우 Astra Control Service는 해당 클라우드 인스턴스의 관리되는 클러스터에서 생성한 백업 및 클론에 대해 기본적으로 이 버킷을 사용합니다(백업에 다른 버킷을 선택할 수 있음). 클라우드 인스턴스의 관리되는 클러스터에서 다른 클러스터로 애플리케이션의 라이브 클론을 수행하는 경우 Astra Control Service는 소스 클라우드 인스턴스에 대한 기본 버킷을 사용하여 클론 작업을 수행합니다.

여러 클라우드 인스턴스의 기본 버킷과 동일한 버킷을 설정할 수 있습니다.

보호 정책을 생성하거나 임시 백업을 시작할 때 모든 버킷에서 선택할 수 있습니다.

Astra Control Service는 백업 또는 클론을 시작하기 전에 대상 버킷에 액세스할 수 있는지 여부를 확인합니다.

기존 버킷을 봅니다

Astra Control Service에서 사용할 수 있는 버킷 목록을 보고 해당 상태를 확인하고 클라우드 인스턴스의 기본 버킷(정의된 경우)을 식별합니다.

버킷은 다음 상태 중 하나를 가질 수 있습니다.

보류 중
: 버킷을 추가하면 보류 중인 상태에서 시작되고 Astra Control이 이를 검색합니다.

사용 가능
: 이 버킷은 Astra Control에서 사용할 수 있습니다.

제거되었습니다
: 현재는 버킷이 작동하지 않습니다. 상태 아이콘 위에 마우스를 올려 놓으면 문제가 무엇인지 확인할 수 있습니다.

  버킷이 제거된 상태인 경우에도 기본 버킷으로 설정하고 보호 일정에 할당할 수 있습니다. 하지만 데이터 보호 작업이 시작될 때 버킷이 가용 상태가 아니면 해당 작업이 실패합니다.

단계

1. 버킷 * 으로 이동합니다.

   Astra Control Service에서 사용할 수 있는 버킷 목록이 표시됩니다.

추가 버킷을 추가합니다

언제든지 추가 버킷을 추가할 수 있습니다. 따라서 보호 정책을 생성하거나 임시 백업을 시작할 때 버킷 중에서 선택할 수 있으며, 클라우드 인스턴스에서 사용하는 기본 버킷을 변경할 수 있습니다.

다음 유형의 버킷을 추가할 수 있습니다.

- Amazon Web Services에서 직접 지원합니다
- 일반 S3
- Google 클라우드 플랫폼
- Microsoft Azure를 참조하십시오
- NetApp ONTAP S3
- NetApp StorageGRID S3

시작하기 전에

- 기존 버킷의 이름을 알고 있어야 합니다.
- Astra Control에 버킷 관리에 필요한 권한을 제공하는 버킷에 대한 자격 증명이 있어야 합니다.
- 버킷이 Microsoft Azure에 있는 경우:
  - 버킷은 _Astra-backup-rg_이라는 리소스 그룹에 속해야 합니다.
  - Azure 저장소 계정 인스턴스 성능 설정이 "프리미엄"으로 설정된 경우 "프리미엄 계정 유형" 설정을 "Blob 차단 "으로 설정해야 합니다.

단계

1. 버킷 * 으로 이동합니다.
2. 추가 * 를 선택하고 프롬프트에 따라 버킷을 추가합니다.
   - * 유형 *: 클라우드 공급자를 선택하십시오.
   - * 기존 버킷 이름 *: 버킷의 이름을 입력합니다.
   - * 설명 *: 버킷에 대한 설명을 선택적으로 입력합니다.
     - * 저장소 계정 * (Azure에만 해당): Azure 저장소 계정의 이름을 입력합니다. 이 버킷은 이름이 _Astra-backup-rg_인 리소스 그룹에 속해야 합니다.
     - * S3 서버 이름 또는 IP 주소 * (AWS 및 S3 버킷 유형만 해당): 해당 지역에 해당하는 S3 엔드포인트의 정규화된 도메인 이름을 에 입력합니다 `https://`. 을 참조하십시오 "아마존 문서" 를 참조하십시오.
     - * 자격 증명 선택 *: Astra Control Service에 버킷 관리에 필요한 권한을 제공하는 자격 증명을 입력합니다. 제공해야 하는 정보는 버킷 유형에 따라 다릅니다.
       a. 버킷을 추가하려면 * 추가 * 를 선택합니다.

결과

Astra Control Service가 버킷을 추가합니다. 이제 보호 정책을 생성하거나 임시 백업을 수행할 때 이 버킷을 선택할 수 있습니다. 이 버킷을 클라우드 인스턴스의 기본 버킷으로 설정할 수도 있습니다.

기본 버킷을 변경합니다

클라우드 인스턴스의 기본 버킷을 변경할 수 있습니다. Astra Control Service는 기본적으로 백업 및 클론에 이 버킷을 사용합니다. 각 클라우드 인스턴스에는 고유한 기본 버킷이 있습니다.

Astra Control은 클라우드 인스턴스에 대해 기본 버킷을 자동으로 할당하지 않습니다. 두 클러스터 간에 애플리케이션 클론 작업을 수행하기 전에 클라우드 인스턴스의 기본 버킷을 수동으로 설정해야 합니다.

단계

1. 클라우드 인스턴스 * 로 이동합니다.
2. 편집할 클라우드 인스턴스의 * 작업 * 열에서 구성 메뉴를 선택합니다.
3. 편집 * 을 선택합니다.
4. 버킷 목록에서 이 클라우드 인스턴스의 기본 버킷을 만들 버킷을 선택합니다.
5. Update * 를 선택합니다.

버킷을 탈거하십시오

더 이상 사용하지 않거나 상태가 불량한 버킷을 제거할 수 있습니다. 오브젝트 저장소 구성을 단순하고 최신 상태로 유지하기 위해 이 작업을 수행할 수 있습니다.

- 기본 버킷을 제거할 수 없습니다. 해당 버킷을 제거하려면 먼저 다른 버킷을 기본값으로 선택하십시오.
- 버킷의 클라우드 공급자 보존 기간이 만료되기 전에는 WORM(Write Once Read Many) 버킷을 제거할 수 없습니다. 웜 버킷은 버킷 이름 옆에 "잠김"으로 표시됩니다.

시작하기 전에

- 시작하기 전에 이 버킷에 대해 실행 중이거나 완료된 백업이 없는지 확인해야 합니다.
- 예약된 백업에 버킷이 사용되지 않는지 확인해야 합니다.

있는 경우 계속할 수 없습니다.

단계

1. 버킷 * 으로 이동합니다.
2. Actions * 메뉴에서 * Remove * 를 선택합니다.

Astra Control은 먼저 버킷에 백업을 사용하는 스케줄 정책이 없고 제거할 버킷에 활성 백업이 없음을 보장합니다.

3. 작업을 확인하려면 "remove"를 입력합니다.
4. 예, 버킷 제거 * 를 선택합니다.

[기술 미리보기] 사용자 지정 리소스를 사용하여 버킷을 관리합니다

애플리케이션 클러스터에서 Astra Control CR(사용자 지정 리소스)을 사용하여 버킷을 추가할 수 있습니다. 애플리케이션과 영구 스토리지를 백업하려는 경우나 클러스터 간에 애플리케이션을 클론 복제하려는 경우에는 오브젝트

저장소 버킷 공급자를 추가하는 것이 중요합니다. Astra Control은 이러한 백업 또는 클론을 정의한 오브젝트 저장소 버킷에 저장합니다. 사용자 지정 리소스 방법을 사용하는 경우 애플리케이션 스냅샷 기능을 사용하려면 버킷이 필요합니다.

애플리케이션 구성과 영구 스토리지를 동일한 클러스터에 클론 복제하려는 경우 Astra Control에 버킷이 필요하지 않습니다.

Astra Control의 버킷 맞춤형 리소스를 AppVault라고 합니다. 이 CR에는 보호 작업에 사용되는 버킷에 필요한 구성이 포함되어 있습니다.

시작하기 전에

- Astra Control Center에서 관리하는 클러스터에서 연결할 수 있는 버킷이 있어야 합니다.
- 버킷에 대한 자격 증명이 있는지 확인하십시오.
- 버킷이 다음 유형 중 하나인지 확인합니다.
  - NetApp ONTAP S3
  - NetApp StorageGRID S3
  - Microsoft Azure를 참조하십시오
  - 일반 S3

AWS(Amazon Web Services) 및 GCP(Google Cloud Platform)는 일반 S3 버킷 유형을 사용합니다.

Astra Control Center는 Amazon S3를 일반 S3 버킷 공급자로 지원하지만, Astra Control Center는 Amazon의 S3 지원을 주장하는 모든 오브젝트 저장소 공급업체를 지원하지 않을 수 있습니다.

단계

1. 사용자 정의 리소스(CR) 파일을 만들고 이름을 지정합니다(예: `astra-appvault.yaml`)를 클릭합니다.
2. 다음 특성을 구성합니다.
   - **metadata.name**: _ (필수) _ AppVault 사용자 정의 리소스의 이름입니다.
   - * spec.prefix *: _ (선택 사항) _ AppVault에 저장된 모든 요소의 이름 앞에 붙는 경로입니다.
   - **spec.providerConfig**: _ (필수) _ 지정된 공급자를 사용하여 AppVault에 액세스하는 데 필요한 구성을 저장합니다.
   - **spec.providerCredentials**: _ (필수) _ 지정된 공급자를 사용하여 AppVault에 액세스하는 데 필요한 자격 증명에 대한 참조를 저장합니다.
     - **spec.providerCredentials.valueFromSecret**: _ (선택 사항) _ 자격 증명 값이 비밀에서 와야 함을 나타냅니다.
       - * KEY *: _ (valueFromSecret을 사용하는 경우 필수) _ 선택할 암호의 유효한 키입니다.
       - * name *: _ (valueFromSecret을 사용하는 경우 필수) _ 이 필드의 값을 포함하는 암호의 이름입니다. 같은 네임스페이스에 있어야 합니다.
   - **spec.providerType**: _ (필수) _ 백업을 제공하는 항목을 결정합니다(예: NetApp ONTAP S3 또는 Microsoft Azure).

     YAML 예:

```
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. 를 채운 후 `astra-appvault.yaml` 올바른 값이 있는 파일에 CR을 적용합니다.

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

버킷을 추가하면 Astra Control이 기본 버킷 표시기로 하나의 버킷을 표시합니다. 사용자가 만든 첫 번째 버킷이 기본 버킷이 됩니다. 양동이 추가될 때 나중에 결정할 수 있습니다 "다른 기본 버킷을 설정합니다".

자세한 내용을 확인하십시오

- "Astra Control API를 사용합니다"

실행 중인 작업을 모니터링합니다

지난 24시간 동안 Astra Control에서 완료, 실패 또는 취소된 작업 및 실행 작업에 대한 세부 정보를 볼 수 있습니다. 예를 들어 실행 중인 백업, 복원 또는 클론 작업의 상태를 보고 완료율 및 남은 예상 시간과 같은 세부 정보를 볼 수 있습니다. 가 실행된 예약된 작업의 상태 또는 수동으로 시작한 작업을 볼 수 있습니다.

실행 중이거나 완료된 작업을 보는 동안 작업 세부 정보를 확장하여 각 하위 작업의 상태를 볼 수 있습니다. 진행 중이거나 완료된 작업의 경우 작업 진행률 표시줄이 녹색이고, 취소된 작업의 경우 파란색이고, 오류로 인해 실패한 작업의 경우 빨간색입니다.

클론 작업의 경우 작업 하위 작업은 스냅샷과 스냅샷 복구 작업으로 구성됩니다.

실패한 작업에 대한 자세한 내용은 을 참조하십시오 "계정 활동을 모니터링합니다".

단계

1. 작업을 실행하는 동안 * 응용 프로그램 * 으로 이동합니다.
2. 목록에서 응용 프로그램의 이름을 선택합니다.
3. 응용 프로그램의 세부 정보에서 * 작업 * 탭을 선택합니다.

현재 또는 과거 작업의 세부 정보를 보고 작업 상태별로 필터링할 수 있습니다.

태스크는 최대 24시간 동안 * 작업 * 목록에 유지됩니다. 을 사용하여 이 제한 및 기타 작업 모니터 설정을 구성할 수 있습니다 "Astra Control API를 참조하십시오".

계정을 관리합니다

대금 청구를 설정합니다

Astra Control Service 계정 청구를 관리하기 위해 두 가지 이상의 방법을 사용할 수 있습니다. Azure 또는 Amazon AWS를 사용하는 경우 Microsoft Azure Marketplace 또는 AWS Marketplace를 통해 Astra Control Service 계획을 신청할 수 있습니다. 이렇게 하면 마켓플레이스를 통해 청구 세부 정보를 관리할 수 있습니다. 또는 NetApp을 통해 직접 가입할 수 있습니다. NetApp에 직접 구독하면 Astra Control Service를 통해 청구 세부 정보를 관리할 수 있습니다. 가입 없이 Astra Control Service를 사용하는 경우 무료 요금제에 자동으로 가입됩니다.

Astra Control Service Free Plan을 사용하면 계정에서 최대 10개의 네임스페이스를 관리할 수 있습니다. 10개 이상의 네임스페이스를 관리하려면 무료 요금제로 업그레이드할 때 또는 Azure Marketplace 또는 AWS Marketplace를 통해 구독하여 청구를 설정해야 합니다.

대금 청구 개요

Astra Control Service와 관련된 비용에는 두 가지 유형이 있습니다. Astra Control Service에는 NetApp이 비용을 청구하며, 영구 볼륨 및 오브젝트 스토리지에 대해서는 클라우드 공급자가 비용을 청구합니다.

Astra Control 서비스 청구

Astra Control Service는 다음과 같은 세 가지 계획을 제공합니다.

무료 플랜
: 최대 10개의 네임스페이스를 무료로 관리할 수 있습니다.

프리미엄 페이고
: 네임스페이스당 특정 속도로 무제한의 네임스페이스 관리

프리미엄 구독

namespace pack _ 당 최대 20개의 네임스페이스를 관리할 수 있는 연간 구독으로 할인된 요금으로 선결제합니다. NetApp 세일즈 팀에 문의하여 조직에 필요한 만큼 팩을 구매합니다. 예를 들어 Astra Control Service에서 60개의 네임스페이스를 관리하려면 3개의 팩을 구입해야 합니다. 연간 구독에서 허용하는 것보다 많은 네임스페이스를 관리하는 경우 추가 네임스페이스당 구독 종속 초과 요금이 부과됩니다. 아직 Astra Control 계정이 없는 경우 Premium Subscription을 구매하면 자동으로 Astra Control 계정이 생성됩니다. 기존 무료 플랜이 있는 경우 프리미엄 구독으로 자동 변환됩니다.

Astra Control 계정을 만들면 무료 플랜이 자동으로 가입됩니다. Astra Control의 대시보드는 현재 허용 가능한 10개의 네임스페이스 중 관리하고 있는 네임스페이스 수를 보여 줍니다. 네임스페이스가 포함된 첫 번째 앱이 관리되는 경우 네임스페이스에 대한 청구가 시작되고 네임스페이스가 포함된 마지막 앱이 관리되지 않을 때 해당 네임스페이스에 대한 청구가 중지됩니다.

11번째 네임스페이스를 관리하려고 하면 Astra Control이 Free Plan의 한계에 도달했음을 알려줍니다. 그런 다음 무료 플랜에서 프리미엄 플랜으로 업그레이드하라는 메시지가 표시됩니다. 추가 네임스페이스당 구독 종속 초과 요금이 부과됩니다.

언제든지 프리미엄 요금제로 업그레이드할 수 있습니다. 업그레이드 후 Astra Control은 계정의 _ALL_Namespaces에 대해 귀하에게 요금을 부과하기 시작합니다. 처음 10개 네임스페이스는 무료 요금제에 머무르고 있지 않습니다.

Google Cloud 청구

영구 볼륨은 NetApp Cloud Volumes Service를 통해 지원되며, 앱의 백업은 Google 클라우드 스토리지 버킷에 저장됩니다.

- "Cloud Volumes Service에 대한 가격 세부 정보를 봅니다".

  Astra Control Service는 모든 서비스 유형과 서비스 수준을 지원합니다. 사용하는 서비스 유형은 에 따라 다릅니다 "Google Cloud 지역".

- "Google Cloud 스토리지 버킷의 가격 세부 정보를 확인하십시오".

Microsoft Azure 청구

영구 볼륨은 Azure NetApp Files에 의해 백업되고 앱 백업은 Azure Blob 컨테이너에 저장됩니다.

- "Azure NetApp Files에 대한 가격 세부 정보를 봅니다".
- "Microsoft Azure Blob 스토리지의 가격 세부 정보를 봅니다".
- "Azure 마켓플레이스에서 Astra Control Service 계획 및 가격을 확인하십시오"

Astra Control Service에 대한 Azure 청구 요금은 시간당 청구되며, 사용 시간의 29분이 경과한 후 새로운 청구 시간이 시작됩니다.

Amazon Web Services 청구

영구 볼륨은 EBS 또는 FSx for NetApp ONTAP에 의해 백업되고 앱 백업은 AWS 버킷에 저장됩니다.

- "Amazon Web Services에 대한 가격 세부 정보를 봅니다".

Azure Marketplace에서 **Astra Control Service**를 구독하십시오

Azure Marketplace를 사용하여 Astra Control Service에 가입할 수 있습니다. 계정 및 청구 세부 정보는 마켓플레이스를 통해 관리됩니다.

Azure Marketplace 구독 프로세스에 대한 비디오 안내를 보려면 를 방문하십시오 "NetApp TV를 참조하십시오".

단계

1. 로 이동합니다 "Azure 마켓플레이스 를 참조하십시오".
2. 지금 다운로드 * 를 선택합니다.
3. 지침에 따라 계획을 구독합니다.

AWS Marketplace에서 **Astra Control Service**를 구독하십시오

AWS Marketplace를 사용하여 Astra Control Service를 구독할 수 있습니다. 계정 및 청구 세부 정보는 마켓플레이스를 통해 관리됩니다.

단계

1. 로 이동합니다 "AWS 마켓플레이스 를 참조하십시오".
2. 구매 옵션 보기 * 를 선택합니다.
3. 메시지가 표시되면 AWS 계정에 로그인하거나 새 계정을 만듭니다.
4. 지침에 따라 계획을 구독합니다.

NetApp에 직접 **Astra Control Service**를 구독하십시오

Astra Control Service UI 내에서 또는 NetApp 세일즈 팀에 연락하여 Astra Control Service를 구독할 수 있습니다.

무료 요금제의 프리미엄 페이고 요금제로 업그레이드하세요

언제든지 결제 계획을 업그레이드하여 Astra Control에서 10개 이상의 네임스페이스를 관리할 수 있습니다. 유효한 신용 카드만 있으면 됩니다.

단계

1. 계정 * 을 선택한 다음 * 청구 * 를 선택합니다.
2. 요금제 * 에서 * Premium PayGo * 로 이동하여 * 지금 업그레이드 * 를 선택합니다.
3. 유효한 신용 카드에 대한 결제 세부 정보를 제공하고 * 프리미엄 요금제로 업그레이드 * 를 선택하십시오.

Astra Control은 신용 카드 만료일이 다가오면 이메일로 발송합니다.

결과

이제 10개 이상의 네임스페이스를 관리할 수 있습니다. Astra Control은 현재 관리하고 있는 _ all_namespaces에 대해 충전 작업을 시작합니다.

무료 요금제로부터 프리미엄 요금제로 업그레이드하세요

NetApp 세일즈 팀에 연락하여 연간 구독과 함께 할인된 요금으로 사전 결제할 수 있습니다.

단계

1. 계정 * 을 선택한 다음 * 청구 * 를 선택합니다.
2. 요금제 * 에서 * 프리미엄 이용 * 으로 이동하여 * 판매 연락처 * 를 선택합니다.
3. 프로세스를 시작할 수 있도록 세일즈 팀에 세부 정보를 제공합니다.

결과

NetApp 세일즈 담당자가 구매 주문서를 처리하기 위해 연락을 드릴 것입니다. 주문이 완료되면 Astra Control은 * 청구 * 탭에 현재 계획을 반영합니다.

현재 비용 및 청구 기록을 봅니다

Astra Control은 현재 월별 비용 및 네임스페이스별 상세 청구 내역을 보여 줍니다. 마켓플레이스를 통해 요금제를 구독한 경우 청구 내역이 표시되지 않지만 마켓플레이스에 로그인하여 확인할 수 있습니다.

단계

1. 계정 * 을 선택한 다음 * 청구 * 를 선택합니다.

   현재 비용이 청구 개요 아래에 표시됩니다.

2. 네임스페이스로 청구 내역을 보려면 * 청구 내역 * 을 선택합니다.

   Astra Control은 각 네임스페이스의 사용 시간 및 비용을 보여 줍니다. 사용 시간은 Astra Control이 청구 기간 동안 네임스페이스를 관리하는 시간(분)입니다.

3. 드롭다운 목록을 선택하여 이전 달을 선택합니다.

Premium PayGo의 신용 카드를 변경합니다

필요한 경우 Astra Control이 청구하기 위해 파일에 가지고 있는 신용 카드를 변경할 수 있습니다.

단계

1. 계정 > 청구 > 결제 방법 * 을 선택합니다.
2. 구성 아이콘을 선택합니다.
3. 신용 카드를 수정합니다.

중요 참고 사항

- 귀하의 청구 계획은 Astra Control 계정입니다.

  계정이 여러 개인 경우 각 계정마다 자체 청구 계획이 있습니다.

- Astra Control 청구서에는 네임스페이스 관리에 대한 비용이 포함되어 있습니다. 영구 볼륨의 스토리지 백엔드는 클라우드 공급자가 별도로 요금을 부과합니다.

  "Astra Control 가격에 대해 자세히 알아보십시오".

- 각 청구 기간은 해당 월의 마지막 날에 종료됩니다.
- 프리미엄 요금제의 경우 무료 요금제로 다운그레이드할 수 없습니다.

사용자를 초대하고 제거합니다

Astra Control 계정에 가입하도록 사용자를 초대하고 더 이상 계정에 액세스할 수 없는 사용자를 제거합니다.

사용자를 초대합니다

계정 소유자와 관리자는 다른 사용자를 Astra Control 계정에 가입하도록 초대할 수 있습니다.

단계

1. 사용자에게 가 있는지 확인합니다 "BlueXP 로그인".
2. 계정 * 을 선택합니다.
3. 사용자 * 탭에서 * 초대 * 를 선택합니다.
4. 사용자의 이름, 이메일 주소 및 역할을 입력합니다.

   다음 사항에 유의하십시오.

   - 이메일 주소는 사용자가 BlueXP에 등록하는 데 사용한 이메일 주소와 일치해야 합니다.
   - 각 역할은 다음과 같은 권한을 제공합니다.
     - 소유자 * 는 관리자 권한을 가지며 계정을 삭제할 수 있습니다.
     - Admin * 은 구성원 권한을 가지며 다른 사용자를 초대할 수 있습니다.
     - 회원 * 은 앱과 클러스터를 완벽하게 관리할 수 있습니다.
     - Viewer * 는 리소스를 볼 수 있습니다.
5. 멤버 또는 뷰어 역할이 있는 사용자에게 제약 조건을 추가하려면 * 제약 조건으로 역할 제한 * 확인란을 활성화합니다.

   제약 조건 추가에 대한 자세한 내용은 을 참조하십시오 "역할을 관리합니다".

6. 다른 사용자를 초대하려면 * 다른 사용자 추가 * 를 선택하고 새 사용자에 대한 정보를 입력합니다.

   한 번에 최대 10명의 사용자를 초대할 수 있습니다. 사용자 초대 * 대화 상자의 왼쪽에서 초대하는 사용자 사이를 탐색할 수 있습니다.

7. 사용자 초대 * 를 선택합니다.

결과

사용자 또는 사용자는 계정에 가입하도록 초대하는 이메일을 받게 됩니다.

사용자의 역할을 변경합니다

계정 소유자는 모든 사용자의 역할을 변경할 수 있고 계정 관리자는 관리자, 구성원 또는 뷰어 역할을 가진 사용자의 역할을 변경할 수 있습니다.

단계

1. 계정 * 을 선택합니다.
2. 사용자 * 탭의 * 작업 * 열에서 해당 사용자의 메뉴를 선택합니다.
3. 역할 편집 * 을 선택합니다.
4. 새 역할을 선택합니다.
5. 멤버 또는 뷰어 역할이 있는 사용자에게 제약 조건을 추가하려면 * 제약 조건으로 역할 제한 * 확인란을 활성화합니다.

   제약 조건 추가에 대한 자세한 내용은 을 참조하십시오 "역할을 관리합니다".
6. Confirm * 을 선택합니다.

결과

Astra Control은 선택한 새 역할에 따라 사용자의 권한을 업데이트합니다.

사용자를 제거합니다

소유자 역할을 가진 사용자는 언제든지 계정에서 다른 사용자를 제거할 수 있습니다.

단계

1. 계정 * 을 선택합니다.
2. 사용자 * 탭에서 제거할 사용자를 선택합니다.
3. Actions * 열에서 메뉴를 선택하고 * Remove user * 를 선택합니다.
4. 메시지가 표시되면 "remove"를 입력하여 삭제를 확인한 다음 * 예, 사용자 제거 * 를 선택합니다.

결과

Astra Control은 사용자를 계정에서 제거합니다.

역할을 관리합니다

네임스페이스 제약 조건을 추가하고 이러한 제약 조건에 대한 사용자 역할을 제한하여 역할을 관리할 수 있습니다. 이렇게 하면 조직 내의 리소스에 대한 액세스를 제어할 수 있습니다. Astra Control UI 또는 를 사용할 수 있습니다 "Astra Control API" 역할을 관리합니다.

역할에 네임스페이스 제약 조건을 추가합니다

관리자 또는 소유자 사용자는 네임스페이스 제약 조건을 추가할 수 있습니다.

단계

1. 계정 관리 * 탐색 영역에서 * 계정 * 을 선택합니다.
2. 사용자 * 탭을 선택합니다.
3. Actions * 열에서 Member 또는 Viewer 역할을 가진 사용자의 메뉴 버튼을 선택합니다.
4. 역할 편집 * 을 선택합니다.
5. 제약 조건으로 역할 제한 * 확인란을 활성화합니다.

이 확인란은 구성원 또는 뷰어 역할에만 사용할 수 있습니다. 역할 * 드롭다운 목록에서 다른 역할을 선택할 수 있습니다.

6. 구속 조건 추가 * 를 선택합니다.

   네임스페이스 또는 네임스페이스 레이블별로 사용 가능한 제약 조건 목록을 볼 수 있습니다.

7. 네임스페이스 구성 방법에 따라 * 제약 조건 유형 * 드롭다운 목록에서 * Kubernetes 네임스페이스 * 또는 * Kubernetes 네임스페이스 레이블 * 을 선택합니다.
8. 목록에서 하나 이상의 네임스페이스 또는 레이블을 선택하여 해당 네임스페이스로 역할을 제한하는 제약 조건을 구성합니다.
9. Confirm * 을 선택합니다.

   역할 편집 * 페이지에는 이 역할에 대해 선택한 제약 조건 목록이 표시됩니다.

10. Confirm * 을 선택합니다.

    계정 * 페이지의 * 역할 * 열에서 구성원 또는 뷰어 역할에 대한 제약 조건을 볼 수 있습니다.

> 역할에 대한 제약 조건을 설정하고 제약 조건을 추가하지 않고 * 확인 * 을 선택하면 역할이 전체 제한 사항으로 간주됩니다(역할에 네임스페이스가 할당된 리소스에 대한 액세스가 거부됨).

역할에서 네임스페이스 제약 조건을 제거합니다

관리자 또는 소유자 사용자는 역할에서 네임스페이스 제약 조건을 제거할 수 있습니다.

단계

1. 계정 관리 * 탐색 영역에서 * 계정 * 을 선택합니다.
2. 사용자 * 탭을 선택합니다.
3. Actions * 열에서 활성 제약 조건이 있는 Member 또는 Viewer 역할을 가진 사용자의 메뉴 버튼을 선택합니다.
4. 역할 편집 * 을 선택합니다.

   역할 편집 * 대화 상자에 해당 역할에 대한 활성 제약 조건이 표시됩니다.

5. 제거할 구속 조건의 오른쪽에 있는 * X * 를 선택합니다.
6. Confirm * 을 선택합니다.

를 참조하십시오

- "사용자 역할 및 네임스페이스"

자격 증명을 추가 및 제거합니다

언제든지 계정에서 클라우드 공급자 자격 증명을 추가 및 제거할 수 있습니다. Astra Control은 이러한 자격 증명을 사용하여 Kubernetes 클러스터, 클러스터에 있는 앱을 검색하고 대신 리소스를 프로비저닝합니다.

Astra Control의 모든 사용자는 동일한 자격 증명 세트를 공유합니다.

자격 증명을 추가합니다

Astra Control에 자격 증명을 추가하는 가장 일반적인 방법은 클러스터를 관리하는 것이지만 계정 페이지에서 자격 증명을 추가할 수도 있습니다. 그러면 추가 Kubernetes 클러스터를 관리할 때 자격 증명을 선택할 수 있습니다.

시작하기 전에

- Amazon Web Services의 경우 클러스터를 생성하는 데 사용되는 IAM 계정에 대한 자격 증명의 JSON 출력이 있어야 합니다. "IAM 사용자 설정 방법을 알아봅니다".
- GKE의 경우 필요한 권한이 있는 서비스 계정에 대한 서비스 계정 키 파일이 있어야 합니다. "서비스 계정 설정 방법에 대해 알아보십시오".
- AKS의 경우 서비스 보안 주체를 생성할 때 Azure CLI의 출력이 포함된 JSON 파일이 있어야 합니다. "서비스 보안 주체를 설정하는 방법에 대해 알아봅니다".

  JSON 파일에 추가하지 않은 경우 Azure 구독 ID도 필요합니다.

단계

1. 계정 > 자격 증명 * 을 선택합니다.
2. 자격 증명 추가 * 를 선택합니다.
3. Microsoft Azure * 를 선택합니다.
4. Google Cloud Platform * 을 선택합니다.
5. Amazon Web Services * 를 선택합니다.
6. Astra Control의 다른 자격 증명과 구별되는 자격 증명의 이름을 입력합니다.
7. 필요한 자격 증명을 입력합니다.
8. * Microsoft Azure *: JSON 파일을 업로드하거나 클립보드에서 해당 JSON 파일의 내용을 붙여넣어 Azure 서비스 교장에게 자세한 정보를 Astra Control에 제공합니다.

   JSON 파일에는 서비스 보안 주체를 생성할 때 Azure CLI의 출력이 포함되어야 합니다. 또한 구독 ID를 포함할 수 있으므로 Astra Control에 자동으로 추가됩니다. 그렇지 않으면 JSON을 제공한 후 ID를 수동으로 입력해야 합니다.
9. * Google Cloud Platform *: 파일을 업로드하거나 클립보드의 콘텐츠를 붙여넣어 Google Cloud 서비스 계정 키 파일을 제공합니다.
10. * Amazon Web Services *: 파일을 업로드하거나 클립보드의 콘텐츠를 붙여넣어 Amazon Web Services IAM 사용자 자격 증명을 제공합니다.
11. 자격 증명 추가 * 를 선택합니다.

결과

이제 Astra Control에 클러스터를 추가할 때 자격 증명을 선택할 수 있습니다.

자격 증명을 제거합니다

언제든지 계정에서 자격 증명을 제거합니다. 자격 증명은 이후에 제거해야 합니다 "모든 클러스터 관리를 취소합니다 "자격 증명을 순환하지 않는 경우(참조 자격 증명 회전)를 클릭합니다.

Astra Control에 추가하는 첫 번째 자격 증명 세트는 항상 사용 중입니다. Astra Control은 자격 증명을 사용하여 백업 버킷에 인증하기 때문입니다. 이러한 자격 증명을 제거하지 않는 것이 좋습니다.

단계

1. 계정 > 자격 증명 * 을 선택합니다.
2. 제거할 자격 증명에 대한 * 상태 * 열의 드롭다운 목록을 선택합니다.
3. 제거 * 를 선택합니다.
4. 삭제할 자격 증명의 이름을 입력한 다음 * 예, 자격 증명 제거 * 를 선택합니다.

결과

Astra Control은 계정에서 자격 증명을 제거합니다.

자격 증명 회전

계정에서 자격 증명을 회전할 수 있습니다. 자격 증명을 회전하는 경우 백업이 진행 중인 상태(예약 또는 필요 시)가 없을 때 유지 관리 창에서 자격 증명을 회전합니다.

단계

1. 의 단계에 따라 기존 자격 증명을 제거합니다 자격 증명을 제거합니다.
2. 의 단계에 따라 새 자격 증명을 추가합니다 자격 증명을 추가합니다.
3. 새 자격 증명을 사용하도록 모든 버킷을 업데이트합니다.
    a. 왼쪽 탐색 창에서 * Bucket * 을 선택합니다.
    b. 편집할 버킷의 * Actions * 열에서 드롭다운 목록을 선택합니다.
    c. 편집 * 을 선택합니다.
    d. 자격 증명 선택 * 섹션에서 Astra Control에 추가한 새 자격 증명을 선택합니다.
    e. Update * 를 선택합니다.
    f. 시스템에 남아 있는 버킷에 대해 * b * ~ * e * 단계를 반복합니다.

결과

Astra Control은 새로운 클라우드 공급자 자격 증명을 사용하기 시작합니다.

계정 활동을 모니터링합니다

Astra Control 계정의 활동에 대한 세부 정보를 볼 수 있습니다. 예를 들어, 새 사용자를 초대하거나, 클러스터를 추가하거나, 스냅샷을 생성할 때 사용할 수 있습니다. 계정 활동을 CSV 파일로 내보낼 수도 있습니다.

Astra Control에서 모든 계정 활동을 봅니다

1. Activity * 를 선택합니다.
2. 필터를 사용하여 활동 목록의 범위를 좁히거나 검색 상자를 사용하여 원하는 항목을 정확하게 찾을 수 있습니다.
3. CSV로 내보내기 * 를 선택하여 계정 활동을 CSV 파일로 다운로드합니다.

특정 앱의 계정 활동을 봅니다

1. 응용 프로그램 * 을 선택한 다음 앱 이름을 선택합니다.
2. Activity * 를 선택합니다.

클러스터의 계정 활동을 봅니다

1. 클러스터 * 를 선택한 다음 클러스터 이름을 선택합니다.
2. Activity * 를 선택합니다.

알림을 보고 관리합니다

Astra Control은 작업이 완료되거나 실패했을 때 알려줍니다. 예를 들어, 앱 백업이 성공적으로 완료되면 알림이 표시됩니다.

인터페이스의 오른쪽 상단에서 읽지 않은 알림 수를 사용할 수 있습니다.

이러한 알림을 보고 읽은 상태로 표시할 수 있습니다. 이렇게 하면 읽지 않은 알림을 지우는 것이 편리합니다.

단계

1. 오른쪽 상단에서 읽지 않은 알림 수를 선택합니다.
2. 알림을 검토한 후 * 읽은 상태로 표시 * 또는 * 모든 알림 표시 * 를 선택합니다.

   모든 알림 표시 * 를 선택한 경우 알림 페이지가 로드됩니다.

3. 알림 * 페이지에서 알림을 보고 읽음으로 표시할 알림을 선택하고 * 작업 * 을 선택한 다음 * 읽음으로 표시 * 를 선택합니다.

계정을 닫습니다

Astra Control 계정이 더 이상 필요하지 않으면 언제든지 계정을 종료할 수 있습니다.

계정을 닫으면 Astra Control이 자동으로 생성한 버킷이 자동으로 삭제됩니다.

단계

1. "모든 앱 및 클러스터 관리를 취소합니다".
2. "Astra Control에서 자격 증명을 제거합니다".
3. 계정 > 청구 > 결제 방법 * 을 선택합니다.
4. 계정 닫기 * 를 선택합니다.
5. 계정 이름을 입력하고 확인 을 클릭하여 계정을 닫습니다.

클라우드 인스턴스 관리

클라우드 인스턴스는 클라우드 공급자 내의 고유한 도메인입니다. 각 클라우드 공급자에 대해 여러 클라우드 인스턴스를 생성할 수 있으며 각 클라우드 인스턴스에는 고유한 이름, 자격 증명 및 관련 클러스터가 있습니다.

Astra Control에 새 클러스터를 추가할 때 클라우드 인스턴스를 생성합니다. Astra Control UI를 사용하여 클라우드 인스턴스를 편집하여 이름 또는 기본 버킷을 변경하고, Astra Control API를 사용하여 클라우드 인스턴스에서 다른 작업을 수행할 수 있습니다.

클라우드 인스턴스를 추가합니다

Astra Control에 새 클러스터를 추가할 때 새 클라우드 인스턴스를 추가할 수 있습니다. 을 참조하십시오 "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다" 를 참조하십시오.

클라우드 인스턴스를 편집합니다

클라우드 공급자의 기존 클라우드 인스턴스를 수정할 수 있습니다.

단계

1. 클라우드 인스턴스 * 로 이동합니다.
2. 클라우드 인스턴스 목록에서 편집하려는 클라우드 인스턴스의 * 작업 * 메뉴를 선택합니다.
3. 편집 * 을 선택합니다.

   이 페이지에서 클라우드 인스턴스의 이름과 기본 버킷을 업데이트할 수 있습니다.

   Astra Control의 각 클라우드 인스턴스는 고유한 이름을 가져야 합니다.

클라우드 인스턴스의 자격 증명을 회전합니다

Astra Control API를 사용하여 클라우드 인스턴스의 자격 증명을 회전할 수 있습니다. 자세한 내용은 "Astra 자동화 문서로 이동합니다".

클라우드 인스턴스를 제거합니다

Astra Control API를 사용하여 클라우드 공급자에서 클라우드 인스턴스를 제거할 수 있습니다. 자세한 내용은 "Astra 자동화 문서로 이동합니다".

Astra Control Provisioner를 활성화합니다

Astra Trident 버전 23.10 이상에는 라이센스를 보유한 Astra Control 사용자가 고급 스토리지 프로비저닝 기능에 액세스할 수 있도록 Astra Control Provisioner를 사용하는 옵션이 포함되어 있습니다. Astra Control Provisioner는 표준 Astra Trident CSI 기반 기능과 더불어 이 확장 기능을 제공합니다. 이 절차를 사용하여 Astra Control Provisioner를 활성화하고 설치할 수 있습니다.

Astra Control Service 구독에는 Astra Control Provisioner 사용에 대한 라이센스가 자동으로 포함됩니다.

향후 Astra Control 업데이트에서 Astra Control Provisioner는 Astra Trident를 스토리지 프로비저닝 및 오케스트레이터로 대체하며 Astra Control을 사용하려면 필수입니다. 따라서 Astra Control 사용자가 Astra Control Provisioner를 활성화하는 것이 좋습니다. Astra Trident는 오픈 소스를 계속 유지하며, NetApp의 새로운 CSI 및 기타 기능으로 릴리즈, 유지, 지원 및 업데이트될 것입니다.

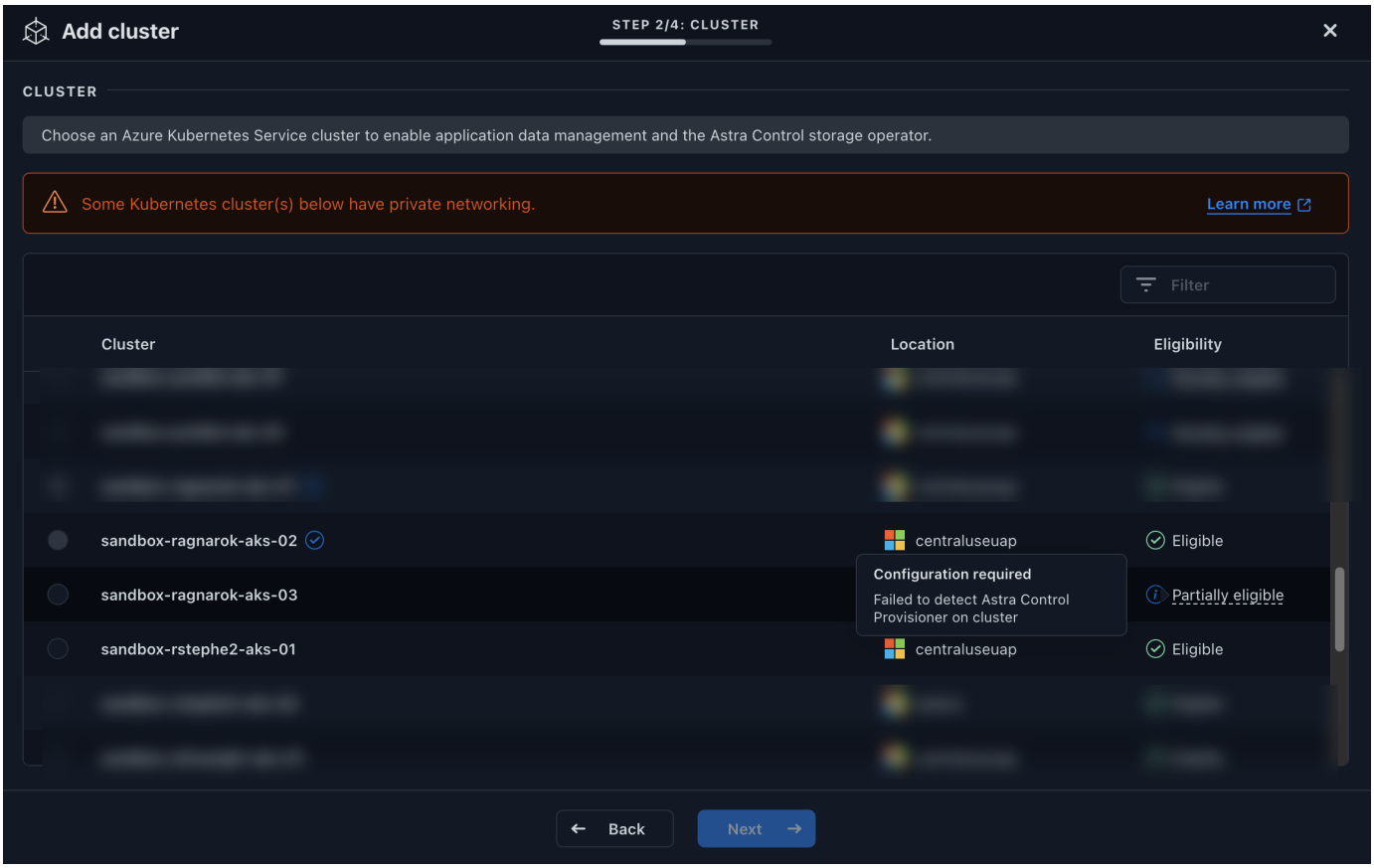
Astra Control Provisioner를 활성화해야 하는지 어떻게 알 수 있습니까**?**

이전에 Astra Trident를 설치하지 않은 클러스터를 Astra Control Service에 추가하면 클러스터가 로 표시됩니다 `Eligible`. 먼저 해 "클러스터를 Astra Control에 추가합니다", Astra Control Provisioner가 자동으로 활성화됩니다.

클러스터가 표시되어 있지 않은 경우 `Eligible`표시됩니다 `Partially eligible` 다음 중 하나로 인해 발생:

- 이전 버전의 Astra Trident를 사용하고 있습니다
- Provisioner 옵션이 아직 활성화되지 않은 Astra Trident 23.10을 사용하고 있습니다
- 자동 활성화를 허용하지 않는 클러스터 유형입니다

용 `Partially eligible` 경우 다음 지침을 사용하여 클러스터에 Astra Control Provisioner를 수동으로 활성화하십시오.



Astra Control Provisioner를 활성화하기 전에

Astra Control Provisioner가 없는 기존 Astra Trident가 있고 Astra Control Provisioner를 활성화하려면 먼저 다음을 수행합니다.

- * Astra Trident를 설치한 경우 해당 버전이 4개의 릴리즈 창 내에 있는지 확인 *: Astra Trident가 버전 24.02의 4개의 릴리즈 창 내에 있는 경우 Astra Control Provisioner를 사용하여 Astra Trident 24.02로 직접 업그레이드할 수 있습니다. 예를 들어, Astra Trident 23.04에서 24.02로 직접 업그레이드할 수 있습니다.
- * 클러스터에 AMD64 시스템 아키텍처가 있는지 확인 *: Astra Control Provisioner 이미지는 AMD64 및 ARM64 CPU 아키텍처 모두에서 제공되지만 Astra Control에서는 AMD64만 지원됩니다.

단계

1. NetApp Astra Control 이미지 레지스트리에 액세스:

a. Astra Control Service UI에 로그온하고 Astra Control 계정 ID를 기록합니다.
   i. 페이지 오른쪽 상단의 그림 아이콘을 선택합니다.
   ii. API 액세스 * 를 선택합니다.
   iii. 계정 ID를 기록합니다.
b. 같은 페이지에서 * API 토큰 생성 * 을 선택하고 API 토큰 문자열을 클립보드에 복사하여 편집기에 저장합니다.
c. 원하는 방법을 사용하여 Astra Control 레지스트리에 로그인합니다.

```
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. (사용자 지정 레지스트리에만 해당) 이미지를 사용자 지정 레지스트리로 이동하려면 다음 단계를 수행하십시오. 레지스트리를 사용하지 않는 경우 의 Trident 운영자 단계를 따르십시오 다음 섹션을 참조하십시오.

> 다음 명령에 Docker 대신 Podman을 사용할 수 있습니다. Windows 환경을 사용하는 경우 PowerShell을 사용하는 것이 좋습니다.

Docker 를 참조하십시오

a. 레지스트리에서 Astra Control Provisioner 이미지를 가져옵니다.

> 가져온 이미지는 여러 플랫폼을 지원하지 않으며 Linux AMD64와 같이 이미지를 가져온 호스트와 동일한 플랫폼만 지원합니다.

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

예:

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

b. 이미지에 태그 지정:

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

c. 이미지를 사용자 지정 레지스트리에 푸시합니다.

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

크레인

a. Astra Control Provisioner 매니페스트를 사용자 지정 레지스트리에 복사합니다.

```
crane copy cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

3. 원래의 Astra Trident 설치 방법에 가 있는지 확인합니다.
4. 원래 사용한 설치 방법을 사용하여 Astra Trident에서 Astra Control Provisioner를 활성화합니다.

Astra Trident 운영자

a. "Astra Trident 설치 프로그램을 다운로드하여 압축을 풉니다".

b. Astra Trident를 아직 설치하지 않았거나 원본 Astra Trident 구축에서 연산자를 제거한 경우 다음 단계를 완료하십시오.

i. CRD 생성:

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.1
6.yaml
```

ii. 트라이덴트 네임스페이스를 만듭니다 (`kubectl create namespace trident`) 또는 트리덴트 네임스페이스가 여전히 존재하는지 확인합니다 (`kubectl get all -n trident`)를 클릭합니다. 네임스페이스가 제거된 경우 다시 만듭니다.

c. Astra Trident를 24.02.0으로 업데이트:

Kubernetes 1.24 이하 버전을 실행하는 클러스터의 경우, 를 사용합니다 `bundle_pre_1_25.yaml`. Kubernetes 1.25 이상을 실행하는 클러스터의 경우 를 사용합니다 `bundle_post_1_25.yaml`.

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

d. Astra Trident가 실행 중인지 확인합니다.

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

응답:

```
NAME      AGE
trident   21m
```

e.] 비밀을 사용하는 레지스트리가 있는 경우 Astra Control Provisioner 이미지를 가져오는 데 사용할 비밀을 만듭니다.

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

f. TridentOrchestrator CR을 편집하고 다음과 같이 편집합니다.

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

i. Astra Trident 이미지에 대한 사용자 지정 레지스트리 위치를 설정하거나 Astra Control 레지스트리에서 가져옵니다 (`tridentImage: <my_custom_registry>/trident:24.02.0` 또는 `tridentImage: netapp/trident:24.02.0`)를 클릭합니다.

ii. Astra Control Provisioner를 활성화합니다 (`enableACP: true`)를 클릭합니다.

iii. Astra Control Provisioner 이미지의 사용자 지정 레지스트리 위치를 설정하거나 Astra Control 레지스트리에서 가져옵니다 (`acpImage: <my_custom_registry>/trident-acp:24.02.0` 또는 `acpImage: cr.astra.netapp.io/astra/trident-acp:24.02.0`)를 클릭합니다.

iv. 를 설정했는지 확인합니다 이미지 풀 암호 이 절차의 앞부분에서 여기에서 설정할 수 있습니다 (`imagePullSecrets: - <secret_name>`)를 클릭합니다. 이전 단계에서 설정한 것과 동일한 이름 암호 이름을 사용합니다.

```
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>
```

g. 파일을 저장하고 종료합니다. 배포 프로세스가 자동으로 시작됩니다.

h. 운영자, 배포 및 복제 세트가 생성되었는지 확인합니다.

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

Kubernetes 클러스터에는 운영자의 인스턴스 * 하나가 있어야 합니다. Astra Trident 연산자를 여러 번 구축해서는 안 됩니다.

i. 를 확인합니다 `trident-acp` 컨테이너가 실행 중이며 `acpVersion` 있습니다 `24.02.0` 의 상태입니다 `Installed`:

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

응답:

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

tridentctl 을 선택합니다

a. "Astra Trident 설치 프로그램을 다운로드하여 압축을 풉니다".

b. "기존 Astra Trident가 있는 경우 이를 호스팅하는 클러스터에서 제거합니다".

c. Astra Control Provisioner를 사용하도록 설정된 Astra Trident를 설치합니다 (`--enable-acp=true`):

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

d. Astra Control Provisioner가 활성화되었는지 확인합니다.

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

응답:

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

헬름

a. Astra Trident 23.07.1 이하를 설치한 경우 "설치 제거" 작업자 및 기타 구성품

b. Kubernetes 클러스터에서 1.24 이전 버전을 실행 중인 경우 psp:

```
kubectl delete psp tridentoperatorpod
```

c. Astra Trident Helm 리포지토리를 추가합니다.

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

d. 제어 차트 업데이트:

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

응답:

```
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!□
```

e. 영상을 나열합니다.

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

응답:

```
| 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)
```

f. 트라이덴트 - 운전자 24.02.0을 사용할 수 있는지 확인합니다.

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

응답:

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

g. 사용 `helm install` 을 클릭하고 다음 설정을 포함하는 옵션 중 하나를 실행합니다.

- 배포 위치의 이름입니다
- Astra Trident 버전
- Astra Control Provisioner 이미지의 이름
- Provisioner를 활성화하는 플래그입니다
- (선택 사항) 로컬 레지스트리 경로입니다. 로컬 레지스트리를 사용하는 경우, 을(를) 참조하십시오 "Trident 이미지" 하나의 레지스트리 또는 다른 레지스트리에 있을 수 있지만 모든 CSI 이미지는 동일한 레지스트리에 있어야 합니다.
- Trident 네임스페이스

옵션

◦ 레지스트리가 없는 이미지

```
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
```

◦ 하나 이상의 레지스트리에 있는 이미지

```
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
```

을 사용할 수 있습니다 `helm list` 이름, 네임스페이스, 차트, 상태, 앱 버전과 같은 설치 세부 정보를 검토하려면 수정본 번호.

Helm을 사용하여 Trident를 구축하는 데 문제가 있는 경우 다음 명령을 실행하여 Astra Trident를 완전히 제거합니다.

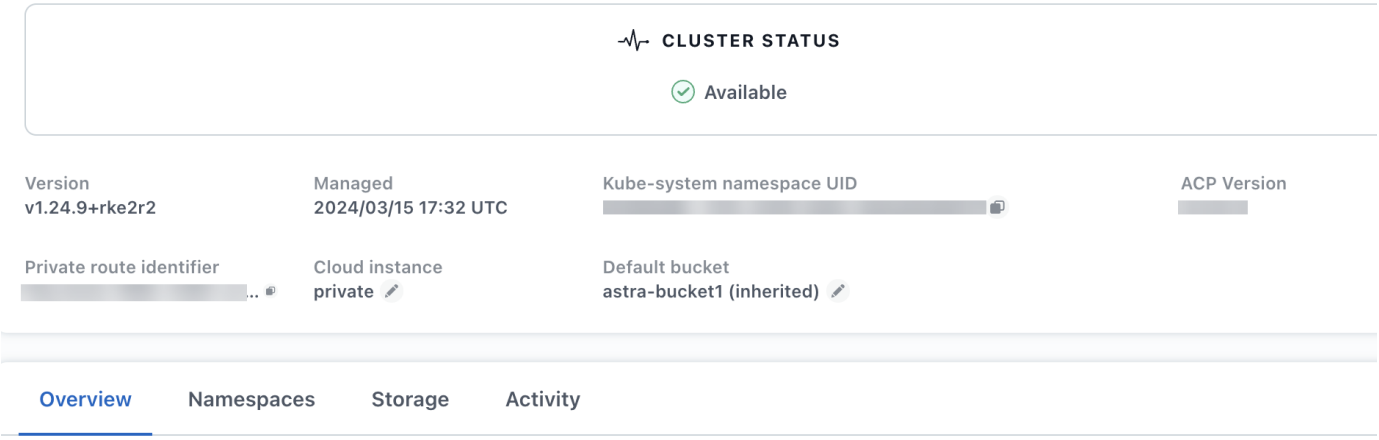
```
./tridentctl uninstall -n trident
```

- 하지 마십시오 * "Astra Trident CRD를 완전히 제거합니다" 설치 제거의 일부로 Astra Control Provisioner를 다시 활성화하려고 합니다.

결과

Astra Control Provisioner 기능이 활성화되어 있으며 실행 중인 버전에 제공되는 모든 기능을 사용할 수 있습니다.

Astra Control Provisioner를 설치하면 Astra Control UI에서 Provisioner를 호스팅하는 클러스터에 가 표시됩니다 `ACP version` 을 사용하지 마십시오 `Trident version` 필드 및 현재 설치된 버전 번호



를 참조하십시오

- "Astra Trident 업그레이드 설명서"

앱 및 클러스터 관리를 취소합니다

Astra Control에서 더 이상 관리하지 않으려는 응용 프로그램 또는 클러스터를 제거합니다.

앱 관리를 중지합니다

Astra Control에서 더 이상 백업, 스냅샷 또는 클론 복제하지 않을 애플리케이션 관리를 중지합니다.

앱 관리를 취소하는 경우:

- 기존 백업 및 스냅샷이 삭제됩니다.
- 애플리케이션과 데이터는 사용 가능한 상태로 유지됩니다.

단계

1. 왼쪽 탐색 모음에서 * 응용 프로그램 * 을 선택합니다.
2. 앱을 선택합니다.
3. 작업 열의 옵션 메뉴에서 * 관리 취소 * 를 선택합니다.

4. 정보를 검토합니다.
5. "unmanage"를 입력하여 확인합니다.
6. 예, 응용 프로그램 관리 취소 * 를 선택합니다.

결과

Astra Control은 앱 관리를 중지합니다.

클러스터 관리를 중지합니다

Astra Control에서 더 이상 관리하지 않으려는 클러스터 관리를 중지합니다.

클러스터를 관리하기 전에 클러스터와 연결된 앱의 관리를 해제해야 합니다.

모범 사례로서, GCP를 통해 클러스터를 삭제하기 전에 Astra Control에서 클러스터를 삭제하는 것이 좋습니다.

클러스터 관리를 취소하는 경우:

- 이 작업을 수행하면 Astra Control에서 클러스터를 관리할 수 없습니다. 클러스터 구성을 변경하지 않고 클러스터를 삭제하지 않습니다.
- Astra Control Provisioner 또는 Astra Trident는 클러스터에서 제거되지 않습니다. "Astra Trident를 제거하는 방법을 알아보십시오".

단계

1. 클러스터 * 를 선택합니다.
2. 더 이상 관리하지 않으려는 클러스터의 확인란을 선택합니다.
3. Actions * 열의 Options 메뉴에서 * Unmanage * 를 선택합니다.
4. 클러스터 관리를 해제할지 확인한 다음 * 예, 관리 취소 * 를 선택합니다.

결과

클러스터의 상태가 * Removing * 으로 변경됩니다. 그 이후에는 클러스터가 * Clusters * 페이지에서 제거되고 Astra Control에서 더 이상 관리되지 않습니다.

클라우드 공급자에서 클러스터 삭제

NetApp 스토리지 클래스에 상주하는 PV(영구적 볼륨)가 있는 Kubernetes 클러스터를 삭제하려면 먼저 아래 방법 중 하나를 따라 PVC(영구 볼륨 클레임)를 삭제해야 합니다. 클러스터를 삭제하기 전에 PVC 및 PV를 삭제하면 클라우드 공급자로부터 예상치 못한 청구서를 받지 않게 됩니다.

- * 방법 #1 *: 클러스터에서 애플리케이션 워크로드 네임스페이스를 삭제합니다. Trident 네임스페이스를_삭제하지 _마십시오.
- * 방법 #2 *: PVC와 POD를 삭제하거나 PVS가 마운트된 구축을 삭제합니다.

Astra Control에서 Kubernetes 클러스터를 관리할 경우, 해당 클러스터의 애플리케이션은 클라우드 공급자를 영구 볼륨의 스토리지 백엔드로 사용합니다. PVS를 먼저 제거하지 않고 클라우드 공급자에서 클러스터를 삭제하는 경우 백엔드 볼륨은 클러스터와 함께 _NOT_DELETED가 됩니다.

위의 방법 중 하나를 사용하면 클러스터에서 해당 PVS가 삭제됩니다. 삭제하기 전에 클러스터의 NetApp 스토리지

클래스에 PVS가 없는지 확인하십시오.

클러스터를 삭제하기 전에 영구 볼륨을 삭제하지 않은 경우 클라우드 공급자로부터 백엔드 볼륨을 수동으로 삭제해야 합니다.

Astra Control의 자체 관리형 인스턴스를 구축

네트워크 안에 있는 Astra Control의 자체 관리형 인스턴스를 원하는 경우 Astra Control Service에서 Astra Control Center를 직접 구축할 수 있습니다.

단계

1. 대시보드의 Getting Started 영역에서 * Astra Control의 자가 관리형 인스턴스 배포 * 를 선택합니다.
2. 다음 중 하나를 수행합니다.
   - Generate * 를 선택하여 새 API 토큰을 생성합니다.
   - 기존 Astra Control REST API 토큰에 붙여 넣습니다. 을 참조하십시오 "Astra 자동화 문서" API 토큰 생성에 대한 지침을 참조하십시오.
3. Astra Control Center * 배포 창의 지침을 따릅니다.

Astra Control Provisioner를 사용합니다

스토리지 백엔드 암호화를 구성합니다

Astra Control Provisioner를 사용하면 관리형 클러스터와 스토리지 백엔드 간의 트래픽 암호화를 활성화하여 데이터 액세스 보안을 개선할 수 있습니다.

Astra Control Provisioner는 두 가지 유형의 스토리지 백엔드에 대해 Kerberos 암호화를 지원합니다.

- * 온프레미스 ONTAP * - Astra Control Provisioner는 Red Hat OpenShift 및 업스트림 Kubernetes 클러스터에서 사내 ONTAP 볼륨까지 NFSv3 및 NFSv4 연결을 통해 Kerberos 암호화를 지원합니다.
- * Azure NetApp Files * - Astra Control Provisioner는 업스트림 Kubernetes 클러스터에서 Azure NetApp Files 볼륨으로의 NFSv4.1 연결을 통한 Kerberos 암호화를 지원합니다.

따라서 스냅샷을 생성하고, 삭제하고, 크기 조정하고, 스냅샷, 클론 읽기 전용 클론 생성 및 NFS 암호화를 사용하는 볼륨 가져오기

사내 **ONTAP** 볼륨과 전송 중인 **Kerberos** 암호화를 구성합니다

관리 클러스터와 온프레미스 ONTAP 스토리지 백엔드 사이의 스토리지 트래픽에 Kerberos 암호화를 활성화할 수 있습니다.

> 온프레미스 ONTAP 스토리지 백엔드를 통한 NFS 트래픽에 대한 Kerberos 암호화는 를 사용해야만 지원됩니다 `ontap-nas` 스토리지 드라이버.

시작하기 전에

- 가 있는지 확인합니다 "Astra Control Provisioner를 활성화했습니다" 관리 대상 클러스터에서.
- 에 액세스할 수 있는지 확인합니다 `tridentctl` 유틸리티.
- ONTAP 스토리지 백엔드에 대한 관리자 액세스 권한이 있어야 합니다.
- ONTAP 스토리지 백엔드에서 공유할 볼륨의 이름을 알고 있어야 합니다.
- NFS 볼륨에 대한 Kerberos 암호화를 지원하는 ONTAP 스토리지 VM을 준비했는지 확인합니다. 을 참조하십시오 "데이터 LIF에서 Kerberos를 사용하도록 설정합니다" 를 참조하십시오.
- Kerberos 암호화로 사용하는 NFSv4 볼륨이 올바르게 구성되어 있는지 확인합니다. 의 NetApp NFSv4 도메인 구성 섹션(13페이지)을 참조하십시오 "NetApp NFSv4의 향상된 기능 및 모범 사례 가이드 를 참조하십시오".

ONTAP 엑스포트 정책을 추가 또는 수정합니다

기존 ONTAP 엑스포트 정책에 규칙을 추가하거나, ONTAP 스토리지 VM 루트 볼륨뿐 아니라 업스트림 Kubernetes 클러스터와 공유된 ONTAP 볼륨에 대해 Kerberos 암호화를 지원하는 새 엑스포트 정책을 생성해야 합니다. 추가한 내보내기 정책 규칙 또는 새로 만든 내보내기 정책은 다음 액세스 프로토콜 및 액세스 권한을 지원해야 합니다.

액세스 프로토콜

NFS, NFSv3 및 NFSv4 액세스 프로토콜을 사용하여 엑스포트 정책을 구성합니다.

액세스 세부 정보

볼륨에 대한 필요에 따라 세 가지 Kerberos 암호화 버전 중 하나를 구성할 수 있습니다.

- * Kerberos 5 * - (인증 및 암호화)
- * Kerberos 5i * - (인증 및 ID 보호 암호화)
- * Kerberos 5p * - (인증 및 암호화, ID 및 개인 정보 보호)

적절한 액세스 권한을 사용하여 ONTAP 엑스포트 정책 규칙을 구성합니다. 예를 들어, 클러스터가 Kerberos 5i 및 Kerberos 5p 암호화가 혼합된 NFS 볼륨을 마운트하는 경우 다음 액세스 설정을 사용합니다.

유형	읽기 전용 액세스	읽기/쓰기 권한	고급 사용자 액세스
Unix	활성화됨	활성화됨	활성화됨
Kerberos 5i	활성화됨	활성화됨	활성화됨
Kerberos 5p	활성화됨	활성화됨	활성화됨

ONTAP 엑스포트 정책과 엑스포트 정책 규칙을 생성하는 방법은 다음 문서를 참조하십시오.

- "엑스포트 정책을 생성합니다"
- "엑스포트 정책에 규칙 추가"

스토리지 백엔드를 생성합니다

Kerberos 암호화 기능을 포함하는 Astra Control Provisioner 스토리지 백엔드 구성을 생성할 수 있습니다.

이 작업에 대해

Kerberos 암호화를 구성하는 스토리지 백엔드 구성 파일을 만들 때 를 사용하여 세 가지 Kerberos 암호화 버전 중 하나를 지정할 수 있습니다 `spec.nfsMountOptions` 매개 변수:

- `spec.nfsMountOptions: sec=krb5` (인증 및 암호화)
- `spec.nfsMountOptions: sec=krb5i` (인증 및 암호화, ID 보호)
- `spec.nfsMountOptions: sec=krb5p` (인증 및 암호화, ID 및 개인 정보 보호)

Kerberos 수준을 하나만 지정하십시오. 매개 변수 목록에서 Kerberos 암호화 수준을 두 개 이상 지정하면 첫 번째 옵션만 사용됩니다.

단계

1. 관리되는 클러스터에서 다음 예제를 사용하여 스토리지 백엔드 구성 파일을 생성합니다. 괄호 안의 값을 사용자 환경의 정보로 대체:

```
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. 이전 단계에서 생성한 구성 파일을 사용하여 백엔드를 생성합니다.

```
tridentctl create backend -f <backend-configuration-file>
```

백엔드 생성에 실패하면 백엔드 구성에 문제가 있는 것입니다. 다음 명령을 실행하여 로그를 보고 원인을 확인할 수 있습니다.

```
tridentctl logs
```

구성 파일의 문제를 확인하고 수정한 후 create 명령을 다시 실행할 수 있습니다.

스토리지 클래스를 생성합니다

스토리지 클래스를 만들어 Kerberos 암호화를 사용하여 볼륨을 프로비저닝할 수 있습니다.

이 작업에 대해

저장소 클래스 개체를 만들 때 를 사용하여 세 가지 Kerberos 암호화 버전 중 하나를 지정할 수 있습니다 `mountOptions` 매개 변수:

- `mountOptions: sec=krb5` (인증 및 암호화)
- `mountOptions: sec=krb5i` (인증 및 암호화, ID 보호)
- `mountOptions: sec=krb5p` (인증 및 암호화, ID 및 개인 정보 보호)

Kerberos 수준을 하나만 지정하십시오. 매개 변수 목록에서 Kerberos 암호화 수준을 두 개 이상 지정하면 첫 번째 옵션만 사용됩니다. 스토리지 백엔드 구성에서 지정한 암호화 수준이 스토리지 클래스 객체에 지정한 레벨과 다른 경우 스토리지 클래스 객체가 우선합니다.

단계

1. 다음 예제를 사용하여 StorageClass Kubernetes 개체를 생성합니다.

```
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. 스토리지 클래스를 생성합니다.

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. 스토리지 클래스가 생성되었는지 확인합니다.

```
kubectl get sc ontap-nas-sc
```

다음과 유사한 출력이 표시됩니다.

```
NAME            PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

볼륨 프로비저닝

스토리지 백엔드와 스토리지 클래스를 생성한 후 이제 볼륨을 프로비저닝할 수 있습니다. 에 대해서는 이 지침을

참조하십시오 "볼륨 프로비저닝".

Azure NetApp Files 볼륨과 함께 전송 중인 Kerberos 암호화를 구성합니다

관리 클러스터와 단일 Azure NetApp Files 스토리지 백엔드 또는 Azure NetApp Files 스토리지 백엔드의 가상 풀 사이의 스토리지 트래픽에 Kerberos 암호화를 활성화할 수 있습니다.

시작하기 전에

- 관리형 Red Hat OpenShift 클러스터에서 Astra Control Provisioner를 활성화했는지 확인합니다. 을 참조하십시오 "Astra Control Provisioner를 활성화합니다" 를 참조하십시오.
- 에 액세스할 수 있는지 확인합니다 `tridentctl` 유틸리티.
- 요구 사항을 확인하고 의 지침에 따라 Kerberos 암호화용 Azure NetApp Files 스토리지 백엔드를 준비했는지 확인합니다 "Azure NetApp Files 설명서".
- Kerberos 암호화로 사용하는 NFSv4 볼륨이 올바르게 구성되어 있는지 확인합니다. 의 NetApp NFSv4 도메인 구성 섹션(13페이지)을 참조하십시오 "NetApp NFSv4의 향상된 기능 및 모범 사례 가이드 를 참조하십시오".

스토리지 백엔드를 생성합니다

Kerberos 암호화 기능을 포함하는 Azure NetApp Files 스토리지 백엔드 구성을 만들 수 있습니다.

이 작업에 대해

Kerberos 암호화를 구성하는 스토리지 백엔드 구성 파일을 만들 때 다음 두 가지 가능한 수준 중 하나에 적용되도록 정의할 수 있습니다.

- 를 사용하는 * 스토리지 백엔드 레벨 * `spec.kerberos` 필드에 입력합니다
- 를 사용하는 * 가상 풀 레벨 * `spec.storage.kerberos` 필드에 입력합니다

가상 풀 레벨에서 구성을 정의하면 스토리지 클래스의 레이블을 사용하여 풀이 선택됩니다.

두 레벨에서 Kerberos 암호화의 세 가지 버전 중 하나를 지정할 수 있습니다.

- `kerberos: sec=krb5` (인증 및 암호화)
- `kerberos: sec=krb5i` (인증 및 암호화, ID 보호)
- `kerberos: sec=krb5p` (인증 및 암호화, ID 및 개인 정보 보호)

단계

1. 관리되는 클러스터에서 스토리지 백엔드(스토리지 백엔드 레벨 또는 가상 풀 레벨)를 정의해야 하는 위치에 따라 다음 예제 중 하나를 사용하여 스토리지 백엔드 구성 파일을 생성합니다. 괄호 안의 값을 사용자 환경의 정보로 대체:

스토리지 백엔드 레벨의 예

```
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
```

가상 풀 레벨 예

```
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. 이전 단계에서 생성한 구성 파일을 사용하여 백엔드를 생성합니다.

```
tridentctl create backend -f <backend-configuration-file>
```

백엔드 생성에 실패하면 백엔드 구성에 문제가 있는 것입니다. 다음 명령을 실행하여 로그를 보고 원인을 확인할 수 있습니다.

```
tridentctl logs
```

구성 파일의 문제를 확인하고 수정한 후 create 명령을 다시 실행할 수 있습니다.

스토리지 클래스를 생성합니다

스토리지 클래스를 만들어 Kerberos 암호화를 사용하여 볼륨을 프로비저닝할 수 있습니다.

단계

1. 다음 예제를 사용하여 StorageClass Kubernetes 개체를 생성합니다.

```
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. 스토리지 클래스를 생성합니다.

```
kubectl create -f sample-input/storage-class-anf-sc-nfs.yaml
```

3. 스토리지 클래스가 생성되었는지 확인합니다.

```
kubectl get sc anf-sc-nfs
```

다음과 유사한 출력이 표시됩니다.

```
NAME          PROVISIONER             AGE
anf-sc-nfs    csi.trident.netapp.io   15h
```

볼륨 프로비저닝

스토리지 백엔드와 스토리지 클래스를 생성한 후 이제 볼륨을 프로비저닝할 수 있습니다. 에 대해서는 이 지침을 참조하십시오 "볼륨 프로비저닝".

스냅샷을 사용하여 볼륨 데이터를 복구합니다

Astra Control Provisioner는 를 사용하여 스냅샷에서 데이터를 데이터 이동 없이 빠르게 복원할 수 있도록 합니다 `TridentActionSnapshotRestore` (TASR) CR 이 CR은 필수 Kubernetes 조치로 작동하며 작업이 완료된 후에도 유지되지 않습니다.

Astra Control Provisioner는 에서 스냅샷 복원을 지원합니다 `ontap-san`, `ontap-san-economy`, `ontap-nas`, `ontap-nas-flexgroup`, `azure-netapp-files`, `gcp-cvs`, 및 `solidfire-san` 드라이버.

시작하기 전에

바인딩된 PVC 및 사용 가능한 볼륨 스냅샷이 있어야 합니다.

- PVC 상태가 Bound인지 확인한다.

```
kubectl get pvc
```

- 볼륨 스냅샷을 사용할 준비가 되었는지 확인합니다.

```
kubectl get vs
```

단계

1. TASR CR을 생성합니다. 이 예에서는 PVC용 CR을 생성합니다 `pvc1` 및 볼륨 스냅샷 `pvc1-snapshot`.

```
cat tasr-pvc1-snapshot.yaml

apiVersion: v1
kind: TridentActionSnapshotRestore
metadata:
  name: this-doesnt-matter
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. CR을 적용하여 스냅샷에서 복원합니다. 이 예는 스냅샷에서 복구합니다 `pvc1`.

```
kubectl create -f tasr-pvc1-snapshot.yaml

tridentactionsnapshotrestore.trident.netapp.io/this-doesnt-matter
created
```

결과

Astra Control Provisioner는 스냅샷에서 데이터를 복원합니다. 스냅샷 복구 상태를 확인할 수 있습니다.

```
kubectl get tasr -o yaml

apiVersion: 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: ""
```

- 대부분의 경우 Astra Control Provisioner는 장애 발생 시 작업을 자동으로 재시도하지 않습니다. 작업을 다시 수행해야 합니다.
- 관리자 권한이 없는 Kubernetes 사용자는 애플리케이션 네임스페이스에서 TASR CR을 생성할 수 있는 관리자의 권한을 받아야 할 수 있습니다.

SnapMirror를 사용하여 볼륨을 복제합니다

Astra Control Provisioner를 사용하면 재해 복구를 위한 데이터 복제를 위해 한 클러스터의 소스 볼륨과 피어링된 클러스터의 타겟 볼륨 간에 미러 관계를 생성할 수 있습니다. 이름이 지정된 CRD(사용자 지정 리소스 정의)를 사용하여 다음 작업을 수행할 수 있습니다.

- 볼륨 간 미러 관계 생성(PVC)
- 볼륨 간 미러 관계를 제거합니다
- 미러 관계를 해제합니다
- 재해 상태(페일오버) 중에 2차 볼륨 승격
- 계획된 페일오버 또는 마이그레이션 중에 클러스터에서 클러스터로 애플리케이션의 무손실 전환 수행

복제 사전 요구 사항

시작하기 전에 다음과 같은 사전 요구 사항이 충족되는지 확인하십시오.

ONTAP 클러스터

- * Astra Control Provisioner *: Astra Control Provisioner 버전 23.10 이상이 ONTAP를 백엔드로 사용하는 소스 및 대상 Kubernetes 클러스터 모두에 있어야 합니다.
- * 라이센스 *: 소스 및 대상 ONTAP 클러스터 모두에서 데이터 보호 번들을 사용하는 ONTAP SnapMirror 비동기 라이센스를 활성화해야 합니다. 을 참조하십시오 "ONTAP의 SnapMirror 라이센스 개요" 를 참조하십시오.

피어링

- * 클러스터 및 SVM *: ONTAP 스토리지 백엔드를 피어링해야 합니다. 을 참조하십시오 "클러스터 및 SVM 피어링 개요" 를 참조하십시오.

  두 ONTAP 클러스터 간의 복제 관계에 사용되는 SVM 이름이 고유한지 확인합니다.

- * Astra Control Provisioner 및 SVM *: 피어링된 원격 SVM을 대상 클러스터의 Astra Control Provisioner에서 사용할 수 있어야 합니다.

지원되는 드라이버

- 볼륨 복제는 ONTAP-NAS 및 ONTAP-SAN 드라이버에서 지원됩니다.

대칭 복사된 **PVC**를 작성합니다

다음 단계를 수행하고 CRD 예제를 사용하여 운영 볼륨과 보조 볼륨 간의 미러 관계를 생성합니다.

단계

1. 운영 Kubernetes 클러스터에서 다음 단계를 수행합니다.
   a. 를 사용하여 StorageClass 객체를 생성합니다 `trident.netapp.io/replication: true` 매개 변수.

      예

      ```
      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. 이전에 생성된 StorageClass를 사용하여 PVC를 생성합니다.

예

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

c. 로컬 정보로 MirrorRelationship CR을 생성합니다.

예

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

Astra Control Provisioner는 볼륨과 볼륨의 현재 DP(Data Protection) 상태에 대한 내부 정보를 가져온 다음 MirrorRelationship의 상태 필드를 채웁니다.

d. TridentMirrorRelationship CR을 가져와 PVC의 내부 이름과 SVM을 얻습니다.

```
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. 보조 Kubernetes 클러스터에서 다음 단계를 수행합니다.
   a. trident.netapp.io/replication: true 매개 변수를 사용하여 StorageClass 를 만듭니다.

      예

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        trident.netapp.io/replication: true
      ```

   b. 대상 및 소스 정보를 사용하여 MirrorRelationship CR을 생성합니다.

      예

      ```
      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 Provisioner는 구성된 관계 정책 이름(또는 ONTAP의 기본값)을 사용하여 SnapMirror 관계를 생성하고 초기화합니다.

c. 이전에 생성한 StorageClass를 사용하여 PVC를 생성하여 보조(SnapMirror 대상) 역할을 합니다.

예

```
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 Provisioner가 TridentMirrorRelationship CRD를 확인하고 관계가 없는 경우 볼륨을 생성하지 못합니다. 이 관계가 있으면 Astra Control Provisioner는 새로운 FlexVol 볼륨을 MirrorRelationship에 정의된 원격 SVM과 함께 피어링된 SVM에 배치합니다.

볼륨 복제 상태입니다

Trident Mirror Relationship(TMR)은 PVC 간 복제 관계의 한쪽 끝을 나타내는 CRD입니다. 대상 TMR에는 Astra Control Provisioner에게 원하는 상태를 알려주는 상태가 있습니다. 대상 TMR의 상태는 다음과 같습니다.

- * 설립 * : 로컬 PVC는 미러 관계의 대상 볼륨이며, 이것은 새로운 관계입니다.
- * 승진된 * : 로컬 PVC는 현재 유효한 미러 관계가 없는 ReadWrite 및 마운트 가능합니다.
- * 재설립 * : 로컬 PVC는 미러 관계의 대상 볼륨이며 이전에 해당 미러 관계에 있었습니다.
  - 대상 볼륨이 대상 볼륨 내용을 덮어쓰므로 대상 볼륨이 소스 볼륨과 관계가 있는 경우 다시 설정된 상태를 사용해야 합니다.
  - 볼륨이 소스와 이전에 관계가 없는 경우 재설정된 상태가 실패합니다.

비계획 페일오버 중에 보조 **PVC**를 승격합니다

보조 Kubernetes 클러스터에서 다음 단계를 수행합니다.

- TridentMirrorRelationship의 _spec.state_field를 로 업데이트합니다 `promoted`.

계획된 페일오버 중에 보조 **PVC**를 승격합니다

계획된 장애 조치(마이그레이션) 중에 다음 단계를 수행하여 보조 PVC를 승격합니다.

단계

1. 운영 Kubernetes 클러스터에서 PVC의 스냅샷을 생성하고 스냅샷이 생성될 때까지 기다립니다.
2. 운영 Kubernetes 클러스터에서 SnapshotInfo CR을 생성하여 내부 세부 정보를 가져옵니다.

   예

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. 보조 Kubernetes 클러스터에서 *TridentMirrorRelationship_CR* 의 *_spec.state_field*를 *_promitted* 및 *spec.promotedSnapshotHandle* 으로 업데이트하여 스냅샷의 내부 이름으로 업데이트합니다.
4. 보조 Kubernetes 클러스터에서 승격될 TridentMirrorRelationship의 상태(status.state 필드)를 확인합니다.

페일오버 후 미러 관계를 복구합니다

미러 관계를 복구하기 전에 새 1차 사이트로 만들 측면을 선택합니다.

단계

1. 보조 Kubernetes 클러스터에서 TridentMirrorRelationship의 _spec.remoteVolumeHandle_field 값이 업데이트되었는지 확인합니다.
2. 보조 Kubernetes 클러스터에서 TridentMirrorRelationship의 _spec.mirror_field를 로 업데이트합니다 `reestablished`.

추가 작업

Astra Control Provisioner는 운영 볼륨과 2차 볼륨에서 다음 작업을 지원합니다.

1차 **PVC**를 새로운 **2**차 **PVC**로 복제합니다

이미 1차 PVC와 2차 PVC가 있는지 확인하십시오.

단계

1. 설정된 보조(대상) 클러스터에서 PersistentVolumeClaim 및 TridentMirrorRelationship CRD를 삭제합니다.
2. 운영(소스) 클러스터에서 TridentMirrorRelationship CRD를 삭제합니다.
3. 설정하려는 새 2차(대상) PVC에 대해 1차(소스) 클러스터에 새 TridentMirrorRelationship CRD를 생성합니다.

대칭 복사**, 1**차 또는 **2**차 **PVC**의 크기를 조정합니다

PVC는 평소대로 크기를 조정할 수 있으며, 데이터 양이 현재 크기를 초과할 경우 ONTAP는 자동으로 대상 flevxols를 확장합니다.

PVC에서 복제를 제거합니다

복제를 제거하려면 현재 보조 볼륨에 대해 다음 작업 중 하나를 수행합니다.

- 2차 PVC에서 MirrorRelationship을 삭제합니다. 이렇게 하면 복제 관계가 끊어집니다.
- 또는 spec.state 필드를 _promessed_로 업데이트합니다.

PVC 삭제**(**이전에 미러링됨**)**

Astra Control Provisioner는 복제된 PVC를 확인하고 볼륨을 삭제하기 전에 복제 관계를 해제합니다.

TMR을 삭제합니다

미러링된 관계의 한 쪽에서 TMR을 삭제하면 Astra Control Provisioner가 삭제를 완료하기 전에 나머지 TMR이 _promessed_state로 전환됩니다. 삭제하도록 선택한 TMR이 이미 _PROJED_STATE에 있는 경우 기존 미러 관계가 없으며 TMR이 제거되고 Astra Control Provisioner가 로컬 PVC를 _ReadWrite_로 승격합니다. 이렇게 삭제하면 ONTAP의 로컬 볼륨에 대한 SnapMirror 메타데이터가 해제됩니다. 이 볼륨이 향후 미러 관계에 사용될 경우 새 미러 관계를 생성할 때 _establed_volume 복제 상태의 새 TMR을 사용해야 합니다.

ONTAP가 온라인 상태일 때 미러 관계를 업데이트합니다

미러 관계는 설정된 후 언제든지 업데이트할 수 있습니다. 를 사용할 수 있습니다 `state: promoted` 또는 `state: reestablished` 관계를 업데이트하는 필드
대상 볼륨을 일반 ReadWrite 볼륨으로 승격할 때 _promotedSnapshotHandle_을 사용하여 현재 볼륨을 복구할 특정 스냅샷을 지정할 수 있습니다.

ONTAP이 오프라인일 때 미러 관계를 업데이트합니다

Astra Control이 ONTAP 클러스터에 직접 연결되지 않은 상태에서 CRD를 사용하여 SnapMirror 업데이트를 수행할 수 있습니다. 다음 TridentActionMirrorUpdate 예제 형식을 참조하십시오.

예

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` TridentActionMirrorUpdate CRD의 상태를 반영합니다. 이 값은 *SUCCEEDED*,*In Progress* 또는_Failed_에서 가져올 수 있습니다.

Astra Control REST API를 사용한 자동화

Astra Control에는 Curl과 같은 프로그래밍 언어나 유틸리티를 사용하여 Astra Control 기능에 직접 액세스할 수 있는 REST API가 있습니다. Ansible 및 기타 자동화 기술을 사용하여 Astra Control 배포를 관리할 수도 있습니다.

자세한 내용은 "Astra 자동화 문서로 이동합니다".

지식 및 지원

지원을 위해 등록하십시오

Astra Control은 계정을 설정할 때 자동으로 계정을 등록하여 지원을 받려고 시도합니다. 그럴 수 없는 경우 직접 지원을 등록할 수 있습니다. NetApp 기술 지원의 도움을 받으려면 지원 등록을 해야 합니다.

지원 등록을 확인합니다

Astra Control에는 지원 등록을 확인할 수 있는 지원 상태 필드가 포함되어 있습니다.

단계

1. 지원 * 을 선택합니다.
2. 지원 상태 필드를 살펴봅니다.

   지원 상태는 "등록되지 않음"으로 시작되지만 "진행 중"으로 이동한 후 "등록됨"으로 이동합니다.

   이 지원 등록 상태는 15분마다 폴링됩니다. 신규 NetApp 고객은 다음 영업일까지 온보딩 및 지원 등록을 완료할 수 있었습니다. 일련 번호가 48시간 이내에 "Registered"로 표시되지 않는 경우, astra.feedback@netapp.com 을 사용하여 NetApp에 문의하거나 에서 수동으로 등록할 수 있습니다 https://register.netapp.com.

일련 번호를 확인합니다

계정을 등록할 때 Astra Control은 귀사에 제공한 정보를 사용하여 "941"으로 시작하는 20자리의 NetApp 일련 번호를 생성합니다.

NetApp 일련 번호는 Astra Control 계정을 나타냅니다. 웹 티켓을 열 때 이 일련 번호를 사용해야 합니다.

일련 번호는 Astra Control 인터페이스의 * 지원 * 페이지에서 확인할 수 있습니다.

지원 권한을 활성화합니다

Astra Control에서 지원 계정을 자동으로 등록할 수 없는 경우 지원 자격 부여를 활성화하려면 Astra Control과 연결된 NetApp 일련 번호를 등록해야 합니다. NetApp은 2가지 지원 등록 옵션을 제공합니다.

1. 기존 NetApp NSS(Support Site) SSO 계정을 가진 현재 NetApp 고객
2. 기존 NetApp Support Site(NSS) SSO 계정이 없는 새로운 NetApp 고객

옵션 **1:** 기존 **NetApp NSS(Support Site)** 계정이 있는 현재 **NetApp** 고객

단계

1. 로 이동합니다 "클라우드 데이터 서비스 지원 등록" 페이지.
2. SELECT * 이미 NetApp 고객으로 등록되었습니다 *.
3. 로그인하려면 NetApp Support 사이트 자격 증명을 입력하십시오.

기존 고객 등록 페이지가 나타납니다.

4. 양식에 필요한 정보를 입력합니다.
   a. 이름, 회사 및 이메일 주소를 입력합니다.
   b. 제품 라인으로 * Astra Control Service * 를 선택합니다.
   c. 청구 공급자를 선택합니다.
   d. 일련 번호를 입력합니다.
   e. 제출 * 을 선택합니다.

결과

"Registration Submitted successfully(등록 제출이 완료되었습니다)" 페이지로 리디렉션됩니다. 등록과 관련된 이메일 주소는 몇 분 이내에 "귀하의 제품은 이제 지원을 받을 수 있습니다."라는 이메일을 받게 됩니다.

해당 일련 번호에 대한 1회 지원 등록입니다.

옵션 **2:** 기존 **NetApp Support Site(NSS)** 계정이 없는 신규 **NetApp** 고객

단계

1. 로 이동합니다 "클라우드 데이터 서비스 지원 등록" 페이지.
2. SELECT * 저는 등록된 NetApp 고객이 아닙니다. *

   New Customer Registration 페이지가 나타납니다.

3. 양식에 필요한 정보를 입력합니다.
   a. 이름, 회사 정보 및 연락처 정보를 입력합니다.
   b. 제품 라인으로 * Astra Control Service * 를 선택합니다.
   c. 청구 공급자를 선택합니다.
   d. 일련 번호를 입력합니다.
   e. captcha 값을 입력합니다.
   f. 확인란을 선택하여 NetApp 개인 정보 보호 정책을 읽었는지 확인합니다.
   g. 제출 * 을 선택합니다.

      제출된 등록으로부터 확인 이메일을 받게 됩니다. 오류가 발생하지 않으면 "Registration Submitted successfully(등록 제출이 완료되었습니다)" 페이지로 리디렉션됩니다. 또한 1시간 이내에 "Your product is now eligible for support"라는 이메일을 받게 됩니다.

   해당 일련 번호에 대한 1회 지원 등록입니다.

4. 또한 새로운 NetApp 고객은 이후의 지원 활성화를 위해 NetApp NSS(Support Site) 사용자 계정을 만들고 기술 지원 채팅 및 웹 티켓 판매에 대한 지원 포털에 액세스해야 합니다.

   로 이동합니다 "NetApp Support 등록 사이트" 를 눌러 이 작업을 수행합니다. 새로 등록된 Astra Control 일련 번호를 제공하여 프로세스를 신속하게 처리할 수 있습니다.

문제 해결

발생할 수 있는 몇 가지 일반적인 문제를 해결하는 방법에 대해 알아봅니다.

https://kb.netapp.com/Cloud/Astra/Control

를 참조하십시오

- "문제 해결"

도움을 받으십시오

NetApp은 다양한 방법으로 Astra Control을 지원합니다. 기술 자료(KB) 기사 및 불화 채널 같은 광범위한 무료 셀프 지원 옵션이 24x7 제공됩니다. Astra Control 계정에는 웹 발권 서비스를 통한 원격 기술 지원이 포함되어 있습니다.

먼저 해야 합니다 "NetApp 일련 번호에 대한 지원을 활성화합니다" 이러한 비 셀프 서비스 지원 옵션을 사용하려면 NetApp NSS(Support Site) SSO 계정은 케이스 관리와 함께 채팅 및 웹 티켓팅에 필요합니다.

기본 메뉴에서 * 지원 * 탭을 선택하면 Astra Control UI에서 지원 옵션에 액세스할 수 있습니다.

자체 지원

이러한 옵션은 24x7 무료로 제공됩니다.

- "기술 자료"

  Astra Control과 관련된 문서, FAQ 또는 고장 수리 정보를 검색합니다.

- 문서화

  현재 보고 있는 문서 사이트입니다.

- "불화를 통해 도움을 받습니다"

  Pub 카테고리의 Astra로 이동하여 동료 및 전문가와 교류하십시오.

- 피드백 이메일

  귀하의 생각, 아이디어 또는 우려 사항을 당사에 알릴 수 있도록 astra.feedback@netapp.com 으로 이메일을 보내주십시오.

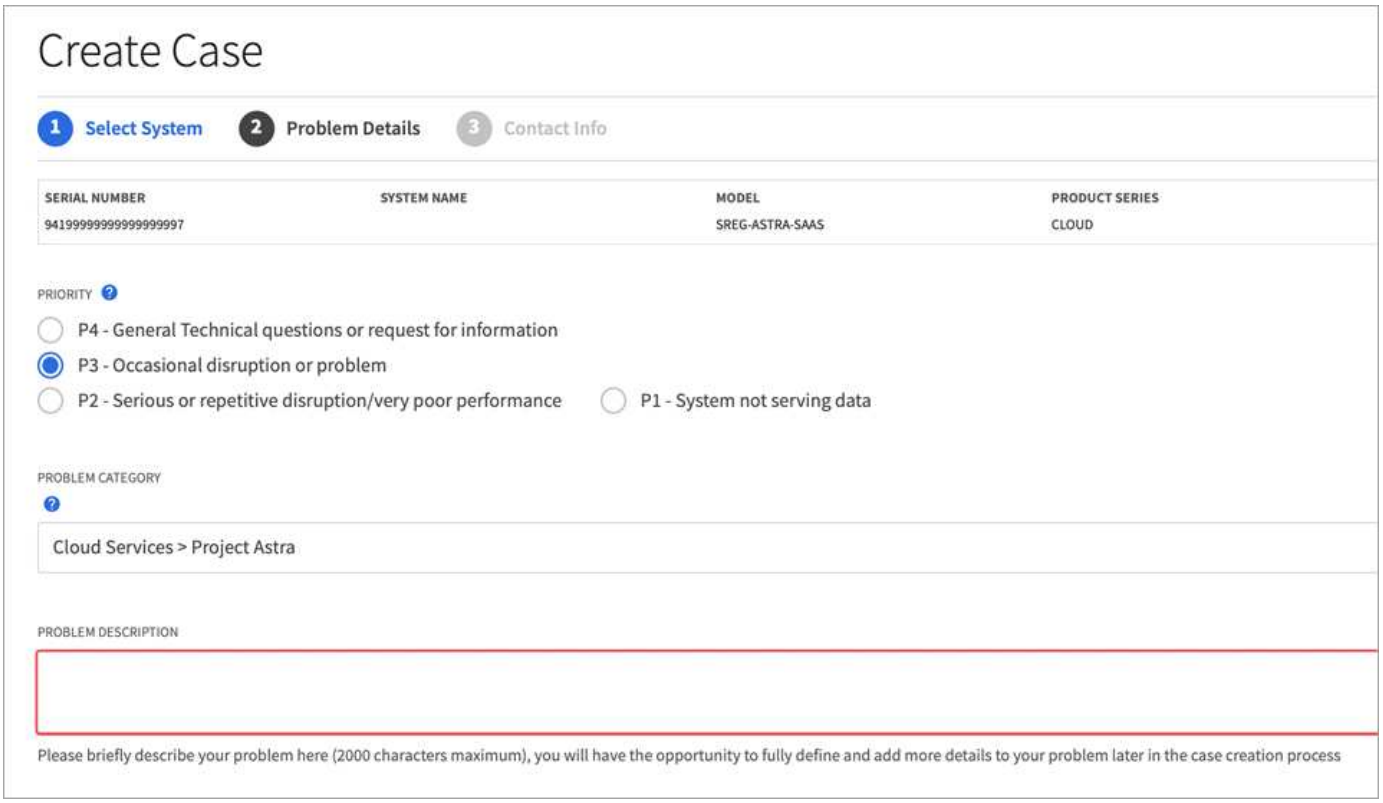
구독 지원

위의 자체 지원 옵션 외에도 NetApp 지원 엔지니어와 협력하여 이후의 문제를 해결할 수 있습니다 "NetApp 일련 번호에 대한 지원을 활성화합니다".

Astra Control 일련 번호가 활성화되면 을 생성하여 NetApp 기술 지원 리소스에 액세스할 수 있습니다 "지원 티켓".

클라우드 데이터 서비스 > Astra * 를 선택합니다.

"941" 일련 번호를 사용하여 웹 티켓을 여십시오. "일련 번호에 대해 자세히 알아보십시오".



자주 묻는 질문

이 FAQ는 질문에 대한 간단한 답변을 찾는 경우에 도움이 될 수 있습니다.

개요

Astra Control은 Kubernetes 네이티브 애플리케이션의 애플리케이션 데이터 라이프사이클 관리 작업을 단순화하는 것을 목표로 합니다. Astra Control Service는 여러 클라우드 공급자 환경에서 실행되는 Kubernetes 클러스터를 지원합니다.

다음 섹션에서는 Astra Control을 사용할 때 나타날 수 있는 몇 가지 추가 질문에 대한 답변을 제공합니다. 자세한 내용은 astra.feedback@netapp.com 으로 문의하십시오

Astra Control에 액세스합니다

Astra Control에 등록할 때 이렇게 많은 세부 정보를 제공해야 하는 이유는 무엇입니까**?**

등록 시 Astra Control에 정확한 고객 정보가 필요합니다. 이 정보는 글로벌 무역 규정 준수(GTC) 확인을 거쳐야 합니다.

Astra Control에 등록할 때 **"**등록 실패**"** 오류가 발생하는 이유는 무엇입니까**?**

Astra Control은 온보딩 섹션에서 정확한 고객 정보를 제공해야 합니다. 잘못된 정보를 입력하면 "등록 실패" 오류가 표시됩니다. 자신이 속한 다른 계정도 잠깁니다.

Astra Control Service URL이란**?**

에서 Astra Control Service에 액세스할 수 있습니다 https://astra.netapp.io.

동료에게 이메일 초대장을 보냈지만 받지 못했습니다**.** 어떻게 해야 합니까**?**

스팸 폴더에서 do-not-reply@netapp.com 이메일을 확인하거나 받은 편지함에서 "초대장"을 검색하도록 요청하십시오. 사용자를 제거하고 다시 추가할 수도 있습니다.

무료 요금제에서 프리미엄 **PayGO** 요금제로 업그레이드했습니다**.** 처음 **10**개의 네임스페이스에 대한 비용이 청구됩니까**?**

예. Premium Plan으로 업그레이드한 후 Astra Control에서 계정의 관리되는 모든 네임스페이스에 대해 요금을 부과하기 시작합니다.

한 달 중순에 프리미엄 **PayGO** 플랜으로 업그레이드했습니다**.** 한 달 동안 요금이 부과됩니까**?**

아니요 청구는 프리미엄 요금제로 업그레이드한 시점부터 시작됩니다.

무료 요금제를 사용하고 있습니다**.** 영구 볼륨 청구에 대한 요금이 부과됩니까**?**

예, 클라우드 공급자로부터 클러스터에서 사용하는 영구 볼륨에 대한 비용이 청구됩니다.

Kubernetes 클러스터를 등록하는 중입니다

Astra Control Service에 **CSI** 드라이버를 추가하기 전에 클러스터에 설치해야 합니까**?**

아니요 클러스터를 Astra Control에 추가하면 서비스에서 Kubernetes 클러스터에 Astra Trident Container Storage Interface(CSI) 드라이버를 자동으로 설치합니다. 이 CSI 드라이버는 클라우드 공급자가 지원하는 클러스터에 영구 볼륨을 프로비저닝하는 데 사용됩니다.

Astra Control Service에 추가한 후 클러스터에 작업자 노드를 추가해야 합니다**.** 어떻게 해야 합니까**?**

새 작업자 노드를 기존 풀에 추가하거나 가 있는 한 새 풀을 생성할 수 있습니다 `COS_CONTAINERD` 이미지 유형. 이러한 정보는 Astra Control에서 자동으로 발견됩니다. Astra Control에서 새 노드가 보이지 않으면 새 작업자 노드가 지원되는 이미지 유형을 실행하고 있는지 확인합니다. 을 사용하여 새 작업자 노드의 상태를 확인할 수도 있습니다 `kubectl get nodes` 명령.

EKS(Elastic Kubernetes Service) 클러스터를 등록하는 중입니다

프라이빗 **EKS** 클러스터를 **Astra Control Service**에 추가할 수 있습니까**?**

예, Astra Control Service에 전용 EKS 클러스터를 추가할 수 있습니다. 전용 EKS 클러스터를 추가하려면 을 참조하십시오 "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다".

Azure Kubernetes Service(AKS) 클러스터를 등록 중입니다

프라이빗 **AKS** 클러스터를 **Astra Control Service**에 추가할 수 있습니까**?**

예, Astra Control Service에 전용 AKS 클러스터를 추가할 수 있습니다. 전용 AKS 클러스터를 추가하려면 을 참조하십시오 "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다".

Active Directory를 사용하여 내 **AKS** 클러스터의 인증을 관리할 수 있습니까**?**

예. 인증 및 ID 관리에 Azure Active Directory(Azure AD)를 사용하도록 AKS 클러스터를 구성할 수 있습니다. 클러스터를 생성할 때 의 지침을 따릅니다 "공식 문서" Azure AD를 사용하도록 클러스터를 구성합니다. 클러스터가 AKS로 관리되는 Azure AD 통합에 대한 요구 사항을 충족하는지 확인해야 합니다.

GKE(Google Kubernetes Engine) 클러스터를 등록하는 중입니다

프라이빗 **GKE** 클러스터를 **Astra Control Service**에 추가할 수 있습니까**?**

예, Astra Control Service에 개인 GKE 클러스터를 추가할 수 있습니다. 전용 GKE 클러스터를 추가하려면 을 참조하십시오 "Astra Control Service에서 Kubernetes 클러스터 관리를 시작합니다".

전용 GKE 클러스터에는 가 있어야 합니다 "인증된 네트워크" Astra Control IP 주소를 허용하도록 설정합니다.

52.188.218.166/32

GKE 클러스터가 공유 **VPC**에 상주할 수 있습니까**?**

예. Astra Control은 공유 VPC에 상주하는 클러스터를 관리할 수 있다. "공유 VPC 구성을 위해 Astra 서비스 계정을 설정하는 방법에 대해 알아보십시오".

GCP에서 서비스 계정 자격 증명을 어디서 찾을 수 있습니까**?**

에 로그인한 후 "Google Cloud Console을 선택합니다"서비스 계정 세부 정보는 * IAM 및 Admin * 섹션에서 확인할 수 있습니다. 자세한 내용은 을 참조하십시오 "Google Cloud for Astra Control을 설정하는 방법".

서로 다른 **GCP** 프로젝트의 서로 다른 **GKE** 클러스터를 추가하려고 합니다**. Astra Control**에서 지원됩니까**?**

아니요. 이 구성은 지원되지 않습니다. 하나의 GCP 프로젝트만 지원됩니다.

클러스터를 제거하는 중입니다

올바르게 등록을 취소하고 클러스터를 종료하며 연결된 볼륨을 삭제하려면 어떻게 해야 합니까**?**

1. "Astra Control에서 애플리케이션을 관리합니다".
2. "Astra Control에서 클러스터 등록을 취소합니다".
3. "영구 볼륨 클레임을 삭제합니다".
4. 클러스터를 삭제합니다.

Astra Control에서 클러스터를 제거한 후 애플리케이션 및 데이터는 어떻게 됩니까**?**

Astra Control에서 클러스터를 제거해도 클러스터의 구성(애플리케이션 및 영구 스토리지)은 변경되지 않습니다. Astra Control 스냅샷 또는 해당 클러스터의 애플리케이션 백업을 복구할 수 없습니다. 스토리지 백엔드 내에 저장된 볼륨 스냅샷 데이터는 제거되지 않습니다. Astra Control에서 생성한 영구 스토리지 백업은 클라우드 공급자의 오브젝트 저장소 내에 남아 있지만 복원에는 사용할 수 없습니다.

> GCP를 통해 삭제하기 전에 항상 Astra Control에서 클러스터를 제거하십시오. Astra Control에서 관리하는 동안 GCP에서 클러스터를 삭제하면 Astra Control 계정에 문제가 발생할 수 있습니다.

Astra Control Provisioner는 관리를 취소할 때 클러스터에서 자동으로 제거됩니까**?**

Astra Control Center에서 클러스터를 관리 취소하면 Astra Control Provisioner 또는 Astra Trident가 클러스터에서 자동으로 제거되지 않습니다. Astra Control Provisioner 및 해당 구성 요소 또는 Astra Trident를 제거하려면 다음을 수행해야 합니다 "다음 단계에 따라 Astra Control Provisioner 서비스가 포함된 Astra Trident 인스턴스를 제거하십시오".

응용 프로그램 관리

Astra Control에서 애플리케이션을 구축할 수 있습니까**?**

Astra Control은 애플리케이션을 배포하지 않습니다. 응용 프로그램은 Astra Control 외부에서 배포해야 합니다.

GCP CVS에 바인딩된 애플리케이션의 **PVC**가 표시되지 않습니다**.** 무엇이 문제입니까**?**

Astra Trident 운영자는 Astra Control에 성공적으로 추가된 후 기본 스토리지 클래스를 'NetApp-cvs-perf-premium'으로 설정합니다. 애플리케이션의 PVC가 Cloud Volumes Service for Google Cloud에 바인딩되지 않은 경우 다음과 같은 몇 가지 단계를 수행할 수 있습니다.

- kubeck get SC를 실행하고 기본 스토리지 클래스를 확인합니다.
- 애플리케이션 배포에 사용된 YAML 파일 또는 H제어 차트를 확인하고 다른 스토리지 클래스가 정의되어 있는지 확인하십시오.
- GKE 버전 1.24 이상은 Docker 기반 노드 이미지를 지원하지 않습니다. GKE의 작업자 노드 이미지 유형이 인지 확인합니다 `COS_CONTAINERD` 그리고 NFS 마운트가 성공했습니다.

Astra Control에서 애플리케이션 관리를 중지하면 애플리케이션은 어떻게 됩니까**?**

기존 백업 또는 스냅샷이 삭제됩니다. 애플리케이션과 데이터는 사용 가능한 상태로 유지됩니다. 관리되지 않는 응용 프로그램 또는 해당 응용 프로그램에 속한 백업 또는 스냅숏에는 데이터 관리 작업을 사용할 수 없습니다.

데이터 관리 작업

Astra Control에서 오브젝트 저장소 버킷을 생성하는 위치는 어디입니까**?**

첫 번째 관리되는 클러스터의 지리적 위치에 따라 오브젝트 저장소의 위치가 결정됩니다. 예를 들어, 추가한 첫 번째 클러스터가 유럽 구역에 있는 경우 해당 지역에서 버킷이 생성됩니다. 필요한 경우, 할 수 있습니다 "추가 버킷을 추가합니다".

내 계정에 생성하지 않은 스냅샷이 있습니다. 그들은 어디에서 왔습니까?

경우에 따라 Astra Control은 다른 프로세스를 수행할 때 스냅샷을 자동으로 생성합니다. 스냅샷이 몇 분 이상 오래된 경우 안전하게 삭제할 수 있습니다.

애플리케이션에서 여러 **PVS**를 사용합니다. **Astra Control**은 이러한 모든 **PVC**의 스냅샷과 백업을 수행합니까?

예. Astra Control의 애플리케이션에 대한 스냅샷 작업에는 애플리케이션의 PVC에 바인딩된 모든 PVS의 스냅샷이 포함됩니다.

Astra Control에서 생성한 스냅샷을 클라우드 공급자를 통해 직접 관리할 수 있습니까?

아니요 Astra Control에서 생성된 스냅샷 및 백업은 Astra Control을 통해서만 관리할 수 있다.

Astra Control Provisioner

Astra Control Provisioner의 스토리지 프로비저닝 기능은 **Astra Trident**의 스토리지 프로비저닝 기능과 어떻게 다릅니까?

Astra Control Provisioner는 Astra Control의 일부로 오픈 소스 Astra Trident에서 사용할 수 없는 상위 스토리지 프로비저닝 기능을 지원합니다. 이러한 기능은 오픈 소스 Trident에서 사용할 수 있는 모든 기능에 추가됩니다.

Astra Control Provisioner가 **Astra Trident**를 대체합니까?

Astra Control Provisioner는 Astra Control 아키텍처에서 스토리지 프로비저닝 및 오케스트레이터로 대체되었습니다. Astra Control 사용자가 수행해야 합니다 "Astra Control Provisioner를 활성화합니다" Astra Control을 사용하려면 Astra Trident는 이 릴리즈에서 계속 지원되지만 향후 릴리즈에서 지원되지는 않습니다. Astra Trident는 오픈 소스를 그대로 유지하며 NetApp의 새로운 CSI 및 기타 기능으로 릴리즈, 유지, 지원 및 업데이트됩니다. 하지만 Astra Control Provisioner에는 Astra Trident CSI 기능과 함께 확장 스토리지 관리 기능이 포함되어 있는 Astra Control Provisioner만 사용할 수 있습니다.

Astra Trident에 대한 비용을 지불해야 합니까?

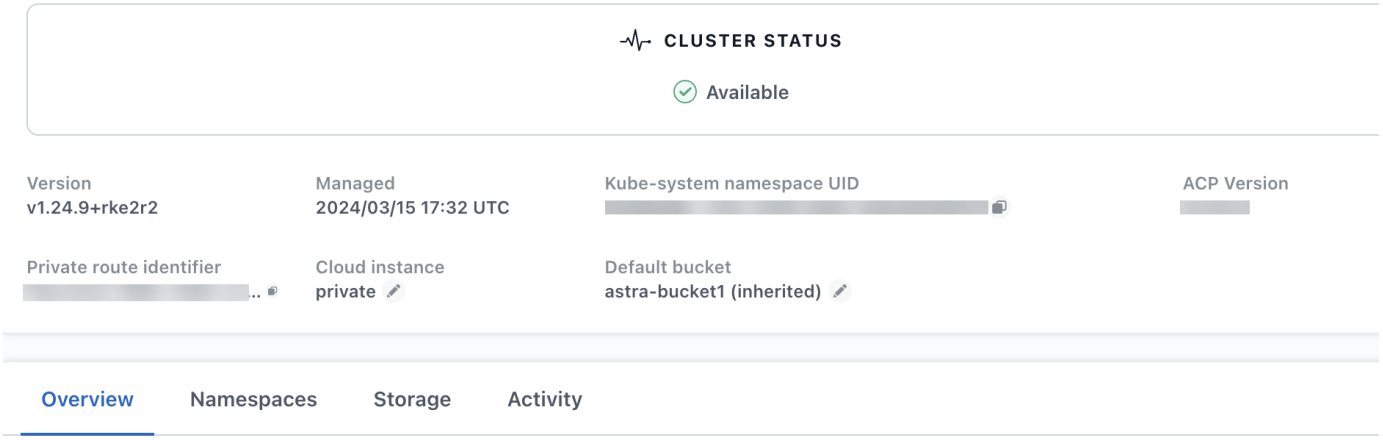
아니요 Astra Trident는 계속해서 오픈 소스이며 무료로 다운로드할 수 있습니다. Astra Control Provisioner 기능을 사용하려면 이제 Astra Control 라이센스가 필요합니다.

모든 **Astra Control**을 설치 및 사용하지 않고 **Astra Control**의 스토리지 관리 및 프로비저닝 기능을 사용할 수 있습니까?

예, Astra Control 데이터 관리 기능의 전체 기능을 사용하지 않으려는 경우에도 Astra Control Provisioner로 업그레이드하고 기능을 사용할 수 있습니다.

Astra Control Provisioner가 클러스터에서 **Astra Trident**를 대체했는지 어떻게 알 수 있습니까?

Astra Control Provisioner를 설치하면 Astra Control UI의 호스트 클러스터에 가 표시됩니다 `ACP version` 을 사용하지 마십시오 `Trident version` 필드 및 현재 설치된 버전 번호



UI에 액세스할 수 없는 경우 다음 방법을 사용하여 설치를 확인할 수 있습니다.

Astra Trident 운영자

를 확인합니다 `trident-acp` 컨테이너가 실행 중이며 `acpVersion` 있습니다 `23.10.0` 또는 이후 의 상태로 표시됩니다 `Installed`:

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

응답:

```
status:
  acpVersion: 23.10.0
  currentInstallationParams:
    ...
    acpImage: <my_custom_registry>/trident-acp:v23.10.0
    enableACP: "true"
    ...
  ...
  status: Installed
```

tridentctl 을 선택합니다

Astra Control Provisioner가 활성화되었는지 확인합니다.

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

응답:

```
+----------------+----------------+-------------+ | SERVER VERSION |
CLIENT VERSION | ACP VERSION | +----------------+----------------
+-------------+ | 23.10.0 | 23.10.0 | 23.10.0. | +----------------
+----------------+-------------+
```

법적 고지

법적 고지 사항은 저작권 선언, 상표, 특허 등에 대한 액세스를 제공합니다.

저작권

"https://www.netapp.com/company/legal/copyright/"

상표

NetApp, NetApp 로고, NetApp 상표 페이지에 나열된 마크는 NetApp Inc.의 상표입니다. 기타 회사 및 제품 이름은 해당 소유자의 상표일 수 있습니다.

"https://www.netapp.com/company/legal/trademarks/"

특허

NetApp 소유 특허 목록은 다음 사이트에서 확인할 수 있습니다.

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

개인 정보 보호 정책

"https://www.netapp.com/company/legal/privacy-policy/"

오픈 소스

통지 파일은 NetApp 소프트웨어에 사용된 타사의 저작권 및 라이센스에 대한 정보를 제공합니다.

"Astra에 대한 고지"

Astra Control API 라이센스

https://docs.netapp.com/us-en/astra-automation/media/astra-api-license.pdf