인프라 워크플로우

Astra Automation

NetApp
August 11, 2025

This PDF was generated from https://docs.netapp.com/ko-kr/astra-automation/workflows_infra/workflows_infra_before.html on August 11, 2025. Always check docs.netapp.com for the latest.

목차

인프라 워크플로우

인프라 워크플로우 사용을 준비합니다

이러한 워크플로우를 사용하여 Astra Control Center 구축과 함께 사용되는 인프라를 생성하고 유지 관리할 수 있습니다. 대부분의 경우 Astra Control Service에서도 워크플로를 사용할 수 있습니다.

이러한 워크플로우는 NetApp이 언제든지 확장 및 개선할 수 있으므로 정기적으로 검토해야 합니다.

일반 준비

Astra 워크플로를 사용하기 전에 반드시 검토하십시오 "워크플로우 사용을 준비하십시오".

워크플로 범주

인프라 워크플로우는 다양한 범주로 구성되어 있어 원하는 워크플로를 쉽게 찾을 수 있습니다.

범주	설명
ID 및 액세스	이러한 워크플로를 통해 ID 및 Astra 액세스 방법을 관리할 수 있습니다. 리소스에는 사용자, 자격 증명 및 토큰이 포함됩니다.
LDAP 구성	선택적으로 LDAP를 사용하여 선택한 사용자를 인증하도록 Astra Control Center를 구성할 수 있습니다.
클러스터	관리 Kubernetes 클러스터를 추가하여 포함된 애플리케이션을 보호하고 지원할 수 있습니다.
클라우드	이러한 워크플로우를 통해 Astra Control REST API를 통해 사용 가능한 클라우드에 액세스할 수 있습니다.
버킷	이러한 워크플로우를 사용하여 백업을 저장하는 데 사용되는 S3 버킷을 생성하고 관리할 수 있습니다.
스토리지	이러한 워크플로우를 사용하여 스토리지 백엔드 및 볼륨을 추가하고 유지할 수 있습니다.

ID 및 액세스

사용자를 나열합니다

특정 Astra 계정에 대해 정의된 사용자를 나열할 수 있습니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/core/v1/users

추가 입력 매개변수

모든 REST API 호출에서 일반적으로 사용되는 매개 변수 외에도 이 단계의 curl 예제에도 다음 매개 변수가 사용됩니다.

매개 변수	유형	필수 요소입니다	설명
포함	쿼리	아니요	필요에 따라 응답에서 반환할 값을 선택합니다.

curl 예: 모든 사용자의 모든 데이터를 반환합니다

```
curl --request GET \
--location "https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/users" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

curl 예: 모든 사용자의 이름, 성 및 **ID**를 반환합니다

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/users?include=firstN
ame,lastName,id" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

JSON 출력 예

```
{
  "items": [
    [
    "David",
    "Anderson",
    "844ec6234-11e0-49ea-8434-a992a6270ec1"
    ],
    [
      "Jane",
      "Cohen",
      "2a3e227c-fda7-4145-a86c-ed9aa0183a6c"
    ]
  ],
  "metadata": {}
}
```

사용자를 생성합니다

특정 자격 증명과 사전 정의된 역할을 가진 사용자를 생성할 수 있습니다. 선택적으로 사용자의 특정 네임스페이스에 대한 액세스를 제한할 수도 있습니다.

단계 **1:** 사용자 이름을 선택합니다

워크플로우를 수행합니다 "사용자를 나열합니다" 를 클릭하고 현재 사용 중이 아닌 사용 가능한 이름을 선택합니다.

2단계**:** 사용자를 생성합니다

다음 REST API 호출을 수행하여 사용자를 생성합니다. 통화가 성공적으로 완료된 후 새 사용자는 아직 사용할 수 없습니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/users

컬의 예

```
curl --request POST \
--location "https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/users" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type" :  "application/astra-user",
  "version" : "1.1",
  "firstName" : "John",
  "lastName" : "West",
  "email" : "jwest@example.com"
}
```

JSON 출력 예


```
{
    "metadata": {
        "creationTimestamp": "2022-11-20T17:23:15Z",
        "modificationTimestamp": "2022-11-20T17:23:15Z",
        "createdBy": "a20e91f3-2c49-443b-b240-615d940ec5f3",
        "labels": []
    },
    "type": "application/astra-user",
    "version": "1.2",
    "id": "d07dac0a-a328-4840-a216-12de16bbd484",
    "authProvider": "local",
    "authID": "jwest@example.com",
    "firstName": "John",
    "lastName": "West",
    "companyName": "",
    "email": "jwest@example.com",
    "postalAddress": {
        "addressCountry": "",
        "addressLocality": "",
        "addressRegion": "",
        "streetAddress1": "",
        "streetAddress2": "",
        "postalCode": ""
    },
    "state": "active",
    "sendWelcomeEmail": "false",
    "isEnabled": "true",
    "isInviteAccepted": "true",
    "enableTimestamp": "2022-11-20T17:23:15Z",
    "lastActTimestamp": ""
}
```


3단계: 선택적으로 허용되는 네임스페이스를 선택합니다

워크플로우를 수행합니다 "네임스페이스를 나열합니다" 액세스를 제한할 네임스페이스를 선택합니다.

4단계: 사용자를 역할에 바인딩합니다

사용자를 역할에 바인딩하려면 다음 REST API 호출을 수행합니다. 아래 예제에서는 네임스페이스 액세스에 제한이 없습니다. 을 참조하십시오 "네임스페이스 세분화를 통해 RBAC 강화" 를 참조하십시오.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/roleBindings

컬의 예

```
curl --request POST \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/roleBindings" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type" :  "application/astra-roleBinding",
  "version" : "1.1",
  "userID" : "d07dac0a-a328-4840-a216-12de16bbd484",
  "accountID" : "29e1f39f-2bf4-44ba-a191-5b84ef414c95",
  "role" : "viewer",
  "roleConstraints": [ "*" ]
}
```

5단계: 자격 증명을 만듭니다

다음 REST API 호출을 수행하여 자격 증명을 생성하여 사용자와 연결합니다. 이 예제에서는 base64 값으로 제공된 암호를 사용합니다. 를 클릭합니다 `name` 속성에는 이전 단계에서 반환된 사용자의 ID가 포함되어야 합니다. 입력 속성 `change` 또한 base64로 인코딩되어야 하며 사용자가 처음 로그인할 때 암호를 변경해야 하는지 여부를 결정합니다 (`true` 또는 `false`)를 클릭합니다.

> ⓘ 이 단계는 로컬 인증을 사용하여 Astra Control Center를 구축하는 경우에만 필요합니다. LDAP 또는 Astra Control Service를 구축한 Astra Control Center 구축 환경에서는 필요하지 않습니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/credentials

컬의 예

```
curl --request POST \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/credentials" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type" :  "application/astra-credential",
  "version" : "1.1",
  "name" : "d07dac0a-a328-4840-a216-12de16bbd484",
  "keyType" : "passwordHash",
  "keyStore" : {
      "cleartext" : "TmV0QXBwMTIz",
      "change" : "ZmFsc2U="
  },
  "valid" : "true"
}
```

LDAP 구성

LDAP 구성을 준비합니다

선택적으로 Astra Control Center를 LDAP(Lightweight Directory Access Protocol) 서버와 통합하여 선택한 Astra 사용자에 대한 인증을 수행할 수 있습니다. LDAP는 분산된 디렉터리 정보에 액세스하기 위한 업계 표준 프로토콜이며 엔터프라이즈 인증에 널리 사용되는 프로토콜입니다.

관련 정보

- "LDAP 기술 사양 로드맵"
- "LDAP 버전 3"

구현 프로세스 개요

높은 수준에서는 Astra 사용자에게 인증을 제공하기 위해 LDAP 서버를 구성하기 위해 수행해야 하는 몇 가지 단계가 있습니다.

> 아래 제시된 단계는 순서대로 진행되는 반면, 경우에 따라 다른 순서로 수행할 수 있습니다. 예를 들어, LDAP 서버를 구성하기 전에 Astra 사용자 및 그룹을 정의할 수 있습니다.

1. 검토 "요구 사항 및 제한 사항" 옵션, 요구 사항 및 제한 사항을 이해합니다.
2. LDAP 서버와 원하는 구성 옵션(보안 포함)을 선택합니다.
3. 워크플로우를 수행합니다 "LDAP 서버를 사용하도록 Astra를 구성합니다" LDAP 서버와 Astra를 통합합니다.
4. LDAP 서버의 사용자 및 그룹을 검토하여 올바르게 정의되었는지 확인합니다.
5. 에서 적절한 워크플로를 수행합니다 "Astra에 LDAP 항목을 추가합니다" LDAP를 사용하여 인증할 사용자를 식별합니다.

요구 사항 및 제한 사항

인증을 위해 LDAP를 사용하도록 Astra를 구성하기 전에 제한 사항 및 구성 옵션을 포함하여 아래에 제시된 Astra 구성 필수 사항을 검토해야 합니다.

Astra Control Center에서만 지원됩니다

Astra Control 플랫폼은 두 가지 구축 모델을 제공합니다. LDAP 인증은 Astra Control Center 구축에서만 지원됩니다.

REST API 또는 웹 사용자 인터페이스를 사용한 구성

Astra Control Center의 현재 릴리즈는 Astra Control REST API와 Astra 웹 사용자 인터페이스를 모두 사용하는 LDAP 인증 구성을 지원합니다.

LDAP 서버가 필요합니다

Astra 인증 요청을 수락하고 처리하려면 LDAP 서버가 있어야 합니다. Microsoft의 Active Directory는 현재 Astra Control Center 릴리스에서 지원됩니다.

LDAP 서버에 대한 보안 연결

Astra에서 LDAP 서버를 구성할 때 선택적으로 보안 연결을 정의할 수 있습니다. 이 경우 LDAPS 프로토콜에 인증서가 필요합니다.

사용자 또는 그룹을 구성합니다

LDAP를 사용하여 인증할 사용자를 선택해야 합니다. 개별 사용자 또는 사용자 그룹을 식별하여 이 작업을 수행할 수 있습니다. LDAP 서버에서 계정을 정의해야 합니다. 또한 인증 요청을 LDAP로 전달할 수 있는 Astra(LDAP 입력)에서도 식별되어야 합니다.

사용자 또는 그룹을 바인딩할 때 역할 제약 조건

Astra Control Center의 현재 릴리스에서 "roleConstraint"에 대해 지원되는 유일한 값은 " * "입니다. 이는 사용자가 제한된 네임스페이스 집합으로 제한되지 않고 모든 네임스페이스에 액세스할 수 있음을 나타냅니다. 을 참조하십시오 "Astra에 LDAP 항목을 추가합니다" 를 참조하십시오.

LDAP 자격 증명

LDAP에서 사용하는 자격 증명에는 사용자 이름(이메일 주소) 및 관련 암호가 포함됩니다.

고유한 이메일 주소입니다

Astra Control Center 배포에서 사용자 이름으로 작동하는 모든 이메일 주소는 고유해야 합니다. Astra에 이미 정의된 이메일 주소를 가진 LDAP 사용자는 추가할 수 없습니다. 중복된 이메일이 있는 경우 먼저 Astra에서 삭제해야 합니다. 을 참조하십시오 "사용자를 제거합니다" 자세한 내용은 Astra Control Center 문서 사이트에서 확인하십시오.

선택적으로 **LDAP** 사용자 및 그룹을 먼저 정의합니다

LDAP 사용자 및 그룹이 LDAP에 없거나 LDAP 서버가 구성되어 있지 않은 경우에도 Astra Control Center에 LDAP 사용자 및 그룹을 추가할 수 있습니다. 이렇게 하면 LDAP 서버를 구성하기 전에 사용자 및 그룹을 미리 구성할 수

있습니다.

여러 **LDAP** 그룹에 정의된 사용자입니다

LDAP 사용자가 여러 LDAP 그룹에 속해 있고 Astra에서 그룹에 서로 다른 역할이 할당된 경우 인증 시 사용자의 유효 역할이 가장 큰 권한이 됩니다. 예를 들어, 사용자가 group1과 함께 '뷰어' 역할을 할당했지만 group2에서 '멤버' 역할을 갖고 있는 경우 사용자의 역할은 '멤버'가 됩니다. 이것은 Astra에서 사용하는 계층 구조를 기반으로 합니다(가장 높은 계층부터 가장 낮은 계층까지).

- 소유자
- 관리자
- 회원
- Viewer(뷰어)

주기적인 계정 동기화

Astra는 IT의 사용자 및 그룹을 약 60초마다 LDAP 서버와 동기화합니다. 따라서 LDAP에 사용자 또는 그룹을 추가하거나 LDAP에서 제거할 경우 Astra에서 사용할 수 있으려면 1분 정도 걸릴 수 있습니다.

LDAP 구성 비활성화 및 재설정

LDAP 구성을 재설정하기 전에 먼저 LDAP 인증을 비활성화해야 합니다. 또한 LDAP 서버("connectionHost")를 변경하려면 두 작업을 모두 수행해야 합니다. 을 참조하십시오 "LDAP를 비활성화하고 재설정합니다" 를 참조하십시오.

REST API 매개 변수

LDAP 구성 워크플로에서는 REST API 호출을 수행하여 특정 작업을 수행합니다. 각 API 호출은 제공된 샘플에 나와 있는 입력 매개 변수를 포함할 수 있습니다. 을 참조하십시오 "온라인 API 참조입니다" 참조 문서를 찾는 방법에 대한 자세한 내용은 를 참조하십시오.

LDAP 서버를 사용하도록 Astra를 구성합니다

LDAP 서버를 선택하고 인증 공급자로 서버를 사용하도록 Astra를 구성해야 합니다. 구성 작업은 아래에 설명된 단계로 구성됩니다. 각 단계에는 단일 REST API 호출이 포함됩니다.

1단계**: CA** 인증서 추가

다음 REST API 호출을 수행하여 CA 인증서를 Astra에 추가합니다.

> 이 단계는 선택 사항이며, Astra와 LDAP가 LDAPS를 사용하여 보안 채널을 통해 통신하도록 하려는 경우에만 필요합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/certificates

컬의 예

```
curl --request POST \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/certificates" \
--include \
--header "Content-Type: application/astra-certificate+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-certificate",
  "version": "1.0",
  "certUse": "rootCA",
  "cert": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUMyVEN",
  "isSelfSigned": "true"
}
```

입력 매개변수에 대한 다음 사항을 참고하십시오.

- cert는 base64로 인코딩된 PKCS-11 형식의 인증서(PEM 인코딩)를 포함하는 JSON 문자열입니다.
- 인증서가 자체 서명된 경우 isSelfSigned를 true로 설정해야 합니다. 기본값은 false 입니다.

JSON 출력 예


```
{
  "type": "application/astra-certificate",
  "version": "1.0",
  "id": "a5212e7e-402b-4cff-bba0-63f3c6505199",
  "certUse": "rootCA",
  "cert": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUMyVEN",
  "cn": "adldap.example.com",
  "expiryTimestamp": "2023-07-08T20:22:07Z",
  "isSelfSigned": "true",
  "trustState": "trusted",
  "trustStateTransitions": [
  {
    "from": "untrusted",
    "to": [
      "trusted",
      "expired"
    ]
  },
  {
    "from": "trusted",
    "to": [
      "untrusted",
      "expired"
    ]
  },
  {
    "from": "expired",
    "to": [
     "untrusted",
     "trusted"
    ]
  }
  ],
  "trustStateDesired": "trusted",
  "trustStateDetails": [],
  "metadata": {
    "creationTimestamp": "2022-07-21T04:16:06Z",
    "modificationTimestamp": "2022-07-21T04:16:06Z",
    "createdBy": "8a02d2b8-a69d-4064-827f-36851b3e1e6e",
    "modifiedBy": "8a02d2b8-a69d-4064-827f-36851b3e1e6e",
    "labels": []
    }
}
```

2단계**:** 바인딩 자격 증명을 추가합니다

다음 REST API 호출을 수행하여 바인딩 자격 증명을 추가합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/credentials

컬의 예

```
curl --request POST \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/credentials" \
--include \
--header "Content-Type: application/astra-certificate+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "name": "ldapBindCredential",
  "type": "application/astra-credential",
  "version": "1.1",
  "keyStore": {
    "bindDn": "dWlkPWFkbWluLG91PXN5c3RlbQ==",
    "password": "cGFzc3dvcmQ="
  }
}
```

입력 매개변수에 대한 다음 사항을 참고하십시오.

- "bindDn"과 "password"는 LDAP 디렉토리를 연결하고 검색할 수 있는 LDAP 관리자 사용자의 base64로 인코딩된 바인딩 자격 증명입니다. bindDn은 LDAP 사용자의 이메일 주소입니다.

JSON 출력 예

```
{
    "type": "application/astra-credential",
    "version": "1.1",
    "id": "3bd9c8a7-f5a4-4c44-b778-90a85fc7d154",
    "name": "ldapBindCredential",
    "metadata": {
        "creationTimestamp": "2022-07-21T06:53:11Z",
        "modificationTimestamp": "2022-07-21T06:53:11Z",
        "createdBy": "527329f2-662c-41c0-ada9-2f428f14c137"
    }
}
```

다음과 같은 응답 매개 변수에 유의하십시오.

- 다음 워크플로우 단계에서 자격 증명의 ID가 사용됩니다.

3단계: LDAP 설정의 UUID를 검색합니다

Astra Control Center에 포함된 Astra.account.ldap 설정의 UUID를 조회하기 위한 REST API 호출은 다음과 같다.

> 아래 curl 예제에서는 query 매개 변수를 사용하여 설정 컬렉션을 필터링합니다. 대신 필터를 제거하여 모든 설정을 확인한 다음 Astra.account.ldap을 검색할 수 있습니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/core/v1/settings

컬의 예

```
curl --request GET \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/settings?filter=na
me%20eq%20'astra.account.ldap'&include=name,id" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
```

JSON 출력 예

```
{
  "items": [
    ["astra.account.ldap",
    "12072b56-e939-45ec-974d-2dd83b7815df"
    ]
  ],
  "metadata": {}
}
```

4단계**: LDAP** 설정을 업데이트합니다

다음 REST API 호출을 수행하여 LDAP 설정을 업데이트하고 구성을 완료합니다. 아래 URL path의 "<setting_ID>" 값에 대한 이전 API 호출의 id 값을 사용한다.

특정 설정에 대한 가져오기 요청을 먼저 발행하여 configSchema 를 볼 수 있습니다. 이렇게 하면 구성의 필수 필드에 대한 자세한 정보를 얻을 수 있습니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
를 누릅니다	/accounts/{account_id}/core/v1/settings/{setting_id}

컬의 예

```
curl --request PUT \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/settings/<SETTING_
ID>" \
--include \
--header "Content-Type: application/astra-setting+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-setting",
  "version": "1.0",
  "desiredConfig": {
    "connectionHost": "myldap.example.com",
    "credentialId": "3bd9c8a7-f5a4-4c44-b778-90a85fc7d154",
    "groupBaseDN": "OU=groups,OU=astra,DC=example,DC=com",
    "isEnabled": "true",
    "port": 686,
    "secureMode": "LDAPS",
    "userBaseDN": "OU=users,OU=astra,DC=example,dc=com",
    "userSearchFilter": "((objectClass=User))",
    "vendor": "Active Directory"
    }
}
```

입력 매개변수에 대한 다음 사항을 참고하십시오.

- IsEnabled가 true로 설정되어 있거나 오류가 발생할 수 있습니다.
- "credentialId"는 앞서 만든 바인딩 자격 증명의 ID입니다.
- 이전 단계의 구성을 기준으로 'ecureMode'를 'LDAP' 또는 'LDAPS'로 설정해야 합니다.
- 'Active Directory'만 공급업체로 지원됩니다.

호출이 성공하면 HTTP 204 응답이 반환됩니다.

5단계**: LDAP** 설정을 검색합니다

선택적으로 다음 REST API 호출을 수행하여 LDAP 설정을 검색하고 업데이트를 확인할 수 있습니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/core/v1/settings/{setting_id}

컬의 예

```
curl --request GET \
--location
"'https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/settings/<SETTING
_ID>" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

JSON 출력 예


```
{
  "items": [
  {
    "type": "application/astra-setting",
    "version": "1.0",
    "metadata": {
      "creationTimestamp": "2022-06-17T21:16:31Z",
      "modificationTimestamp": "2022-07-21T07:12:20Z",
      "labels": [],
      "createdBy": "system",
      "modifiedBy": "00000000-0000-0000-0000-000000000000"
    },
    "id": "12072b56-e939-45ec-974d-2dd83b7815df",
    "name": "astra.account.ldap",
    "desiredConfig": {
      "connectionHost": "10.193.61.88",
      "credentialId": "3bd9c8a7-f5a4-4c44-b778-90a85fc7d154",
      "groupBaseDN": "ou=groups,ou=astra,dc=example,dc=com",
      "isEnabled": "true",
      "port": 686,
      "secureMode": "LDAPS",
      "userBaseDN": "ou=users,ou=astra,dc=example,dc=com",
      "userSearchFilter": "((objectClass=User))",
      "vendor": "Active Directory"
    },
    "currentConfig": {
      "connectionHost": "10.193.160.209",
      "credentialId": "3bd9c8a7-f5a4-4c44-b778-90a85fc7d154",
      "groupBaseDN": "ou=groups,ou=astra,dc=example,dc=com",
      "isEnabled": "true",
      "port": 686,
      "secureMode": "LDAPS",
      "userBaseDN": "ou=users,ou=astra,dc=example,dc=com",
      "userSearchFilter": "((objectClass=User))",
```

```
      "vendor": "Active Directory"
    },
    "configSchema": {
      "$schema": "http://json-schema.org/draft-07/schema#",
      "title": "astra.account.ldap",
      "type": "object",
      "properties": {
        "connectionHost": {
          "type": "string",
          "description": "The hostname or IP address of your LDAP server."
        },
        "credentialId": {
          "type": "string",
          "description": "The credential ID for LDAP account."
        },
        "groupBaseDN": {
          "type": "string",
          "description": "The base DN of the tree used to start the group
search. The system searches the subtree from the specified location."
        },
        "groupSearchCustomFilter": {
          "type": "string",
          "description": "Type of search that controls the default group
search filter used."
        },
        "isEnabled": {
          "type": "string",
          "description": "This property determines if this setting is
enabled or not."
        },
        "port": {
          "type": "integer",
          "description": "The port on which the LDAP server is running."
        },
        "secureMode": {
          "type": "string",
          "description": "The secure mode LDAPS or LDAP."
        },
        "userBaseDN": {
          "type": "string",
          "description": "The base DN of the tree used to start the user
search. The system searches the subtree from the specified location."
        },
        "userSearchFilter": {
          "type": "string",
          "description": "The filter used to search for users according a
```

```
search criteria."
        },
        "vendor": {
          "type": "string",
          "description": "The LDAP provider you are using.",
          "enum": ["Active Directory"]
        }
      },
      "additionalProperties": false,
      "required": [
        "connectionHost",
        "secureMode",
        "credentialId",
        "userBaseDN",
        "userSearchFilter",
        "groupBaseDN",
        "vendor",
        "isEnabled"
      ]
      },
      "state": "valid",
    }
  ],
  "metadata": {}
}
```

아래 표에 있는 값 중 하나를 가질 응답에 있는 '상태' 필드를 찾습니다.

상태	설명
보류 중	구성 프로세스가 아직 활성 상태이며 아직 완료되지 않았습니다.
유효합니다	구성이 성공적으로 완료되고 응답 결과가 'esiredConfig'와 일치합니다.
오류	LDAP 구성 프로세스가 실패했습니다.

Astra에 LDAP 항목을 추가합니다

LDAP가 Astra Control Center의 인증 공급자로 구성된 후에는 Astra가 LDAP 자격 증명을 사용하여 인증할 LDAP 사용자를 선택할 수 있습니다. 각 사용자는 Astra Control REST API를 통해 Astra에 액세스하려면 먼저 Astra에서 역할을 수행해야 합니다.

역할을 할당하도록 Astra를 구성하는 방법에는 두 가지가 있습니다. 해당 환경에 적합한 제품을 선택하십시오.

- "개별 사용자를 추가하고 바인딩합니다"
- "그룹을 추가하고 바인딩합니다"

LDAP 자격 증명은 이메일 주소 및 관련 LDAP 암호 형식의 사용자 이름입니다.

개별 사용자를 추가하고 바인딩합니다

LDAP 인증 후에 사용되는 각 Astra 사용자에게 역할을 할당할 수 있습니다. 이는 사용자 수가 적으며 각 사용자가 서로 다른 관리 특성을 가지고 있을 때 적합합니다.

1단계**:** 사용자 추가

다음 REST API 호출을 수행하여 Astra에 사용자를 추가하고 LDAP가 인증 제공자임을 나타냅니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/users

컬의 예

```
curl --request POST \
--location "https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/users"
\
--include \
--header "Content-Type: application/astra-user+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type" :  "application/astra-user",
  "version" : "1.1",
  "authID" : "cn=JohnDoe,ou=users,ou=astra,dc=example,dc=com",
  "authProvider" : "ldap",
  "firstName" : "John",
  "lastName" : "Doe",
  "email" : "john.doe@example.com"
}
```

입력 매개변수에 대한 다음 사항을 참고하십시오.

- 다음 매개 변수가 필요합니다.
  - 진정한 공급자
  - 진정한 아이디

◦ 이메일

- authID는 LDAP에 있는 사용자의 DN(고유 이름)입니다
- Astra에 정의된 모든 사용자는 '이메일'이 고유해야 합니다

e-메일 값이 고유하지 않으면 오류가 발생하고 응답에서 409 HTTP 상태 코드가 반환됩니다.

JSON 출력 예

```
{
  "metadata": {
    "creationTimestamp": "2022-07-21T17:44:18Z",
    "modificationTimestamp": "2022-07-21T17:44:18Z",
    "createdBy": "8a02d2b8-a69d-4064-827f-36851b3e1e6e",
    "labels": []
  },
  "type": "application/astra-user",
  "version": "1.2",
  "id": "a7b5e674-a1b1-48f6-9729-6a571426d49f",
  "authProvider": "ldap",
  "authID": "cn=JohnDoe,ou=users,ou=astra,dc=example,dc=com",
  "firstName": "John",
  "lastName": "Doe",
  "companyName": "",
  "email": "john.doe@example.com",
  "postalAddress": {
    "addressCountry": "",
    "addressLocality": "",
    "addressRegion": "",
    "streetAddress1": "",
    "streetAddress2": "",
    "postalCode": ""
  },
  "state": "active",
  "sendWelcomeEmail": "false",
  "isEnabled": "true",
  "isInviteAccepted": "true",
  "enableTimestamp": "2022-07-21T17:44:18Z",
  "lastActTimestamp": ""
}
```

2단계**:** 사용자에 대한 역할 바인딩을 추가합니다

다음 REST API 호출을 수행하여 사용자를 특정 역할에 바인딩합니다. 이전 단계에서 생성한 사용자의 UUID가 있어야 합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/roleBindings

컬의 예

```
curl --request POST \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/roleBindings" \
--include \
--header "Content-Type: application/astra-roleBinding+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-roleBinding",
  "version": "1.1",
  "accountID": "{account_id}",
  "userID": "a7b5e674-a1b1-48f6-9729-6a571426d49f",
  "role": "member",
  "roleConstraints": ["*"]
}
```

입력 매개변수에 대한 다음 사항을 참고하십시오.

- 이 값은 Astra의 현재 릴리스에서 사용할 수 있는 유일한 옵션입니다. 이 메시지는 사용자가 제한된 네임스페이스 집합으로 제한되지 않고 모든 네임스페이스에 액세스할 수 있음을 나타냅니다.

JSON 응답 예


```
{
  "metadata": {
    "creationTimestamp": "2022-07-21T18:08:24Z",
    "modificationTimestamp": "2022-07-21T18:08:24Z",
    "createdBy": "8a02d2b8-a69d-4064-827f-36851b3e1e6e",
    "labels": []
  },
  "type": "application/astra-roleBinding",
  "principalType": "user",
  "version": "1.1",
  "id": "b02c7e4d-d483-40d1-aaff-e1f900312114",
  "userID": "a7b5e674-a1b1-48f6-9729-6a571426d49f",
  "groupID": "00000000-0000-0000-0000-000000000000",
  "accountID": "d0fdbfa7-be32-4a71-b59d-13d95b42329a",
  "role": "member",
  "roleConstraints": ["*"]
}
```


응답 매개변수에 대한 다음 사항에 유의하십시오.

- 'princalType' 필드의 값 'user'는 사용자(그룹이 아님)에 대한 역할 바인딩이 추가되었음을 나타냅니다.

그룹을 추가하고 바인딩합니다

LDAP 인증 후에 사용되는 Astra 그룹에 역할을 할당할 수 있습니다. 이는 많은 수의 사용자가 있고 각 사용자가 유사한 관리 특성을 가지고 있을 때 적합합니다.

1단계**:** 그룹을 추가합니다

다음 REST API 호출을 수행하여 Astra에 그룹을 추가하고 LDAP가 인증 제공자임을 나타냅니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/groups

컬의 예

```
curl --request POST \
--location "https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/groups"
\
--include \
--header "Content-Type: application/astra-group+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-group",
  "version": "1.0",
  "name": "Engineering",
  "authProvider": "ldap",
  "authID": "CN=Engineering,OU=groups,OU=astra,DC=example,DC=com"
}
```

입력 매개변수에 대한 다음 사항을 참고하십시오.

- 다음 매개 변수가 필요합니다.
  - 진정한 공급자
  - 진정한 아이디

JSON 응답 예

```
{
  "type": "application/astra-group",
  "version": "1.0",
  "id": "8b5b54da-ae53-497a-963d-1fc89990525b",
  "name": "Engineering",
  "authProvider": "ldap",
  "authID": "CN=Engineering,OU=groups,OU=astra,DC=example,DC=com",
  "metadata": {
    "creationTimestamp": "2022-07-21T18:42:52Z",
    "modificationTimestamp": "2022-07-21T18:42:52Z",
    "createdBy": "8a02d2b8-a69d-4064-827f-36851b3e1e6e",
    "labels": []
  }
}
```

2단계**:** 그룹에 대한 역할 바인딩을 추가합니다

다음 REST API 호출을 수행하여 그룹을 특정 역할에 바인딩합니다. 이전 단계에서 생성한 그룹의 UUID가 있어야 합니다. 그룹 구성원인 사용자는 LDAP가 인증을 수행한 후 Astra에 로그인할 수 있습니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/roleBindings

컬의 예

```
curl --request POST \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/roleBindings" \
--include \
--header "Content-Type: application/astra-roleBinding+json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-roleBinding",
  "version": "1.1",
  "accountID": "{account_id}",
  "groupID": "8b5b54da-ae53-497a-963d-1fc89990525b",
  "role": "viewer",
  "roleConstraints": ["*"]
}
```

입력 매개변수에 대한 다음 사항을 참고하십시오.

- 이 값은 Astra의 현재 릴리스에서 사용할 수 있는 유일한 옵션입니다. 이 메시지는 사용자가 특정 네임스페이스에만 제한되지 않고 모든 네임스페이스에 액세스할 수 있음을 나타냅니다.

JSON 응답 예

```
{
  "metadata": {
    "creationTimestamp": "2022-07-21T18:59:43Z",
    "modificationTimestamp": "2022-07-21T18:59:43Z",
    "createdBy": "527329f2-662c-41c0-ada9-2f428f14c137",
    "labels": []
  },
  "type": "application/astra-roleBinding",
  "principalType": "group",
  "version": "1.1",
  "id": "2f91b06d-315e-41d8-ae18-7df7c08fbb77",
  "userID": "00000000-0000-0000-0000-000000000000",
  "groupID": "8b5b54da-ae53-497a-963d-1fc89990525b",
  "accountID": "d0fdbfa7-be32-4a71-b59d-13d95b42329a",
  "role": "viewer",
  "roleConstraints": ["*"]
}
```

응답 매개변수에 대한 다음 사항에 유의하십시오.

- 'princalType' 필드의 값 'group'은 사용자가 아닌 그룹에 대해 역할 바인딩이 추가되었음을 나타냅니다.

LDAP를 비활성화하고 재설정합니다

Astra Control Center 배포용으로 필요에 따라 수행할 수 있는 두 가지 관련 관리 작업은 선택 사항입니다. LDAP 인증을 전역적으로 비활성화하고 LDAP 구성을 재설정할 수 있습니다.

두 워크플로우 모두 Astra.account.ldap.astra 설정에 대한 ID가 필요합니다. 설정 ID를 검색하는 방법에 대한 자세한 내용은 * LDAP 서버 구성 * 에 포함되어 있습니다. 을 참조하십시오 "LDAP 설정의 UUID를 조회한다" 를 참조하십시오.

- "LDAP 인증을 비활성화합니다"
- "LDAP 인증 구성을 재설정합니다"

LDAP 인증을 비활성화합니다

다음 REST API 호출을 수행하여 특정 Astra 구축에 대한 LDAP 인증을 전역적으로 비활성화할 수 있습니다. 이 호출은 Astra.account.ldap 설정을 업데이트하고 IsEnabled 값은 false로 설정된다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
를 누릅니다	/accounts/{account_id}/core/v1/settings/{setting_id}

```
curl --request PUT \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/settings/<SETTING_
ID>" \
--include \
--header "Content-Type: application/astra-setting+json"
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-setting",
  "version": "1.0",
  "desiredConfig": {
    "connectionHost": "myldap.example.com",
    "credentialId": "3bd9c8a7-f5a4-4c44-b778-90a85fc7d154",
    "groupBaseDN": "OU=groups,OU=astra,DC=example,DC=com",
    "isEnabled": "false",
    "port": 686,
    "secureMode": "LDAPS",
    "userBaseDN": "OU=users,OU=astra,DC=example,dc=com",
    "userSearchFilter": "((objectClass=User))",
    "vendor": "Active Directory"
    }
}
```

호출이 성공하면 HTTP 204 응답이 반환됩니다. 필요에 따라 구성 설정을 다시 검색하여 변경을 확인할 수 있습니다.

LDAP 인증 구성을 재설정합니다

다음 REST API 호출을 수행하여 LDAP 서버에서 Astra 연결을 끊고 Astra에서 LDAP 구성을 재설정할 수 있습니다. 호에는 Astra.account.ldap 설정이 업데이트되고 connectionHost 값이 지워집니다.

IsEnabled의 값도 false로 설정되어야 합니다. 재설정 통화를 하기 전에 또는 재설정 통화를 하기 위해 이 값을 설정할 수 있습니다. 두 번째 경우는 connectionHost를 지워지고 동일한 재설정 호출에서 IsEnabled를 false로 설정해야 합니다.

이는 중단을 야기하는 작업이므로 주의하여 진행해야 합니다. 가져온 LDAP 사용자 및 그룹이 모두 삭제됩니다. 또한 Astra Control Center에서 만든 관련 Astra 사용자, 그룹 및 roleBindings(LDAP 유형)도 모두 삭제합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
를 누릅니다	/accounts/{account_id}/core/v1/settings/{setting_id}

```
curl --request PUT \
--location
"https://astra.example.com/accounts/$ACCOUNT_ID/core/v1/settings/<SETTING_
ID>" \
--include \
--header "Content-Type: application/astra-setting+json"
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
--data  @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-setting",
  "version": "1.0",
  "desiredConfig": {
    "connectionHost": "",
    "credentialId": "3bd9c8a7-f5a4-4c44-b778-90a85fc7d154",
    "groupBaseDN": "OU=groups,OU=astra,DC=example,DC=com",
    "isEnabled": "false",
    "port": 686,
    "secureMode": "LDAPS",
    "userBaseDN": "OU=users,OU=astra,DC=example,dc=com",
    "userSearchFilter": "((objectClass=User))",
    "vendor": "Active Directory"
    }
}
```

다음 사항에 유의하십시오.

- LDAP 서버를 변경하려면 위의 예와 같이 "connectHost"를 null 값으로 변경하는 LDAP를 해제하고 재설정해야 합니다.
- 호출이 성공하면 HTTP 204 응답이 반환됩니다. 선택적으로 구성을 다시 검색하여 변경을 확인할 수 있습니다.

클러스터

클러스터 나열

특정 클라우드에 사용 가능한 클러스터를 나열할 수 있습니다.

1단계: 클라우드를 선택합니다

워크플로우를 수행합니다 "구름 목록을 표시합니다" 클러스터가 포함된 클라우드를 선택합니다.

2단계: 클러스터를 나열합니다

다음 REST API 호출을 수행하여 특정 클라우드에 있는 클러스터를 나열합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/topology/v1/cloud/{cloud_id}/cluster

curl 예: 모든 클러스터의 모든 데이터를 반환합니다

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/clouds/<CLOUD_ID
>/clusters" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

JSON 출력 예

```
{
    "items": [
        {
            "type": "application/astra-cluster",
            "version": "1.1",
            "id": "7ce83fba-6aa1-4e0c-a194-26e714f5eb46",
            "name": "openshift-clstr-ol-07",
            "state": "running",
            "stateUnready": [],
            "managedState": "managed",
            "protectionState": "full",
            "protectionStateDetails": [],
            "restoreTargetSupported": "true",
            "snapshotSupported": "true",
            "managedStateUnready": [],
            "managedTimestamp": "2022-11-03T15:50:59Z",
            "inUse": "true",
            "clusterType": "openshift",
            "accHost": "true",
            "clusterVersion": "1.23",
```

```
            "clusterVersionString": "v1.23.12+6b34f32",
            "namespaces": [
                "default",
                "kube-node-lease",
                "kube-public",
                "kube-system",
                "metallb-system",
                "mysql",
                "mysql-clone1",
                "mysql-clone2",
                "mysql-clone3",
                "mysql-clone4",
                "netapp-acc-operator",
                "netapp-monitoring",
                "openshift",
                "openshift-apiserver",
                "openshift-apiserver-operator",
                "openshift-authentication",
                "openshift-authentication-operator",
                "openshift-cloud-controller-manager",
                "openshift-cloud-controller-manager-operator",
                "openshift-cloud-credential-operator",
                "openshift-cloud-network-config-controller",
                "openshift-cluster-csi-drivers",
                "openshift-cluster-machine-approver",
                "openshift-cluster-node-tuning-operator",
                "openshift-cluster-samples-operator",
                "openshift-cluster-storage-operator",
                "openshift-cluster-version",
                "openshift-config",
                "openshift-config-managed",
                "openshift-config-operator",
                "openshift-console",
                "openshift-console-operator",
                "openshift-console-user-settings",
                "openshift-controller-manager",
                "openshift-controller-manager-operator",
                "openshift-dns",
                "openshift-dns-operator",
                "openshift-etcd",
                "openshift-etcd-operator",
                "openshift-host-network",
                "openshift-image-registry",
                "openshift-infra",
                "openshift-ingress",
                "openshift-ingress-canary",
```

```
            "openshift-ingress-operator",
            "openshift-insights",
            "openshift-kni-infra",
            "openshift-kube-apiserver",
            "openshift-kube-apiserver-operator",
            "openshift-kube-controller-manager",
            "openshift-kube-controller-manager-operator",
            "openshift-kube-scheduler",
            "openshift-kube-scheduler-operator",
            "openshift-kube-storage-version-migrator",
            "openshift-kube-storage-version-migrator-operator",
            "openshift-machine-api",
            "openshift-machine-config-operator",
            "openshift-marketplace",
            "openshift-monitoring",
            "openshift-multus",
            "openshift-network-diagnostics",
            "openshift-network-operator",
            "openshift-node",
            "openshift-oauth-apiserver",
            "openshift-openstack-infra",
            "openshift-operator-lifecycle-manager",
            "openshift-operators",
            "openshift-ovirt-infra",
            "openshift-sdn",
            "openshift-service-ca",
            "openshift-service-ca-operator",
            "openshift-user-workload-monitoring",
            "openshift-vsphere-infra",
            "pcloud",
            "postgreql",
            "trident"
        ],
        "defaultStorageClass": "4bacbb3c-0727-4f58-b13c-3a2a069baf89",
        "cloudID": "4f1e1086-f415-4451-a051-c7299cd672ff",
        "credentialID": "7ffd7354-b6c2-4efa-8e7b-cf64d5598463",
        "isMultizonal": "false",
        "tridentManagedStateAllowed": [
            "unmanaged"
        ],
        "tridentVersion": "22.10.0",
        "apiServiceID": "98df44dc-2baf-40d5-8826-e198b1b40909",
        "metadata": {
            "labels": [
                {
                    "name": "astra.netapp.io/labels/read-
```

```
only/cloudName",
                        "value": "private"
                    }
                ],
                "creationTimestamp": "2022-11-03T15:50:59Z",
                "modificationTimestamp": "2022-11-04T14:42:32Z",
                "createdBy": "00000000-0000-0000-0000-000000000000"
            }
        }
    ]
}
```

자격 증명을 사용하여 클러스터를 추가합니다

클러스터를 추가하면 Astra에서 관리할 수 있습니다. Astra 22.11 릴리즈부터 Astra Control Center와 Astra Control Service가 모두 포함된 클러스터를 추가할 수 있습니다.

주요 클라우드 공급자(AKS, EKS, GKE) 중 하나에서 Kubernetes 서비스를 사용할 때는 클러스터를 추가할 필요가 없습니다.

단계 **1: kubeconfig** 파일을 얻습니다

Kubernetes 관리자 또는 서비스로부터 * kubconfig * 파일의 복사본을 얻어야 합니다.

단계 **2: kubeconfig** 파일을 준비합니다

kubecononfig * 파일을 사용하기 전에 다음 작업을 수행해야 합니다.

1. 파일을 YAML 형식에서 JSON으로 변환:

   YAML 형식의 kubecononfig 파일을 받으면 JSON으로 변환해야 합니다.

2. Base64에서 JSON 인코딩:

   base64에서 JSON 파일을 인코딩해야 합니다.

예

다음은 kubecononfig 파일을 YAML에서 JSON으로 변환한 후 base64로 인코딩하는 예입니다.

```
yq -o=json ~/.kube/config | base64
```

3단계**:** 클라우드를 선택합니다

워크플로우를 수행합니다 "구름 목록을 표시합니다" 클러스터를 추가할 클라우드를 선택합니다.

선택할 수 있는 유일한 클라우드는 * 프라이빗 * 클라우드입니다.

4단계: 자격 증명을 만듭니다

kubecononfig 파일을 사용하여 자격 증명을 생성하려면 다음 REST API 호출을 수행합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/credentials

컬의 예

```
curl --request POST \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/credentials" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type" :  "application/astra-credential",
  "version" : "1.1",
  "name" : "Cloud One",
  "keyType" : "kubeconfig",
  "keyStore" : {
    "base64": encoded_kubeconfig
  },
  "valid" : "true"
}
```

5단계: 클러스터를 추가합니다

다음 REST API 호출을 수행하여 클러스터를 클라우드에 추가합니다. 의 값 `credentialID` 입력 필드는 이전 단계의 REST API 호출에서 얻어집니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/topology/v1/cloud/{cloud_id}/cluster

컬의 예

```
curl --request POST \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/clouds/<CLOUD_ID
>/clusters" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type" :  "application/astra-cluster",
  "version" : "1.1",
  "credentialID": credential_id
}
```

관리되는 클러스터를 나열합니다

현재 Astra에서 관리하는 Kubernetes 클러스터를 나열할 수 있습니다.

다음과 같은 REST API 호출을 수행한다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/topology/v1/managedClusters

curl 예: 모든 클러스터의 모든 데이터를 반환합니다

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/managedClusters"
\
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

클러스터 관리

데이터 보호를 수행할 수 있도록 Kubernetes 클러스터를 관리할 수 있습니다.

1단계: 관리할 클러스터를 선택합니다

워크플로우를 수행합니다 "클러스터 나열" 원하는 클러스터를 선택합니다. 속성 `managedState` 의 클러스터가 이어야 합니다 `unmanaged`.

2단계: 필요에 따라 스토리지 클래스를 선택합니다

필요한 경우 워크플로우를 수행합니다 "스토리지 클래스를 나열합니다" 원하는 스토리지 클래스를 선택합니다.

> 호출 중에 클러스터 관리를 위한 스토리지 클래스를 제공하지 않으면 기본 스토리지 클래스가 사용됩니다.

3단계: 클러스터 관리

다음 REST API 호출을 수행하여 클러스터를 관리합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/topology/v1/managedClusters

컬의 예

```
curl --request POST \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/managedClusters"
\
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예

```
{
  "type": "application/astra-managedCluster",
  "version": "1.0",
  "id": "d0fdf455-4330-476d-bb5d-4d109714e07d"
}
```

클라우드

구름 목록을 표시합니다

특정 Astra 계정을 정의하고 사용할 수 있는 클라우드를 나열할 수 있습니다.

다음 REST API 호출을 수행하여 클라우드를 나열합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/topology/v1/클라우드

curl 예: 모든 클라우드에 대한 모든 데이터를 반환합니다

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/clouds" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

버킷

버킷을 나열하십시오

특정 Astra 계정에 대해 정의된 S3 버킷을 나열할 수 있습니다.

다음 REST API 호출을 수행하여 Bucket을 나열합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/topology/v1/버킷

CURL 예: 모든 버킷에 대한 모든 데이터를 반환합니다

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/buckets" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

스토리지

스토리지 클래스를 나열합니다

사용 가능한 스토리지 클래스를 나열할 수 있습니다.

1단계: 클라우드를 선택합니다

워크플로우를 수행합니다 "구름 목록을 표시합니다" 작업할 클라우드를 선택하십시오.

2단계: 클러스터를 선택합니다

워크플로우를 수행합니다 "클러스터 나열" 클러스터를 선택합니다.

3단계: 특정 클러스터의 스토리지 클래스를 나열합니다

다음 REST API 호출을 수행하여 특정 클러스터 및 클라우드에 대한 스토리지 클래스를 표시합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/topology/v1/클라우드/<CLOUD_ID>/클러스터/<CLUSTER_ID>/storage Claes

curl 예: 모든 저장소 클래스에 대한 모든 데이터를 반환합니다

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/clouds/<CLOUD_ID
>/clusters/<CLUSTER_ID>/storageClasses" \
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

JSON 출력 예

```
{
    "items": [
        {
            "type": "application/astra-storageClass",
            "version": "1.1",
            "id": "4bacbb3c-0727-4f58-b13c-3a2a069baf89",
            "name": "ontap-basic",
            "provisioner": "csi.trident.netapp.io",
            "available": "eligible",
            "allowVolumeExpansion": "true",
            "reclaimPolicy": "Delete",
            "volumeBindingMode": "Immediate",
```

```
            "isDefault": "true",
            "metadata": {
                "createdBy": "system",
                "creationTimestamp": "2022-10-26T05:16:19Z",
                "modificationTimestamp": "2022-10-26T05:16:19Z",
                "labels": []
            }
        },
        {
            "type": "application/astra-storageClass",
            "version": "1.1",
            "id": "150fe657-4a42-47a3-abc6-5dafba3de8bf",
            "name": "thin",
            "provisioner": "kubernetes.io/vsphere-volume",
            "available": "ineligible",
            "reclaimPolicy": "Delete",
            "volumeBindingMode": "Immediate",
            "metadata": {
                "createdBy": "system",
                "creationTimestamp": "2022-10-26T04:46:08Z",
                "modificationTimestamp": "2022-11-04T14:58:19Z",
                "labels": []
            }
        },
        {
            "type": "application/astra-storageClass",
            "version": "1.1",
            "id": "7c6a5c58-6a0d-4cb6-98a0-8202ad2de74a",
            "name": "thin-csi",
            "provisioner": "csi.vsphere.vmware.com",
            "available": "ineligible",
            "allowVolumeExpansion": "true",
            "reclaimPolicy": "Delete",
            "volumeBindingMode": "WaitForFirstConsumer",
            "metadata": {
                "createdBy": "system",
                "creationTimestamp": "2022-10-26T04:46:17Z",
                "modificationTimestamp": "2022-10-26T04:46:17Z",
                "labels": []
            }
        },
        {
            "type": "application/astra-storageClass",
            "version": "1.1",
            "id": "7010ef09-92a5-4c90-a5e5-3118e02dc9a7",
            "name": "vsim-san",
```

```
            "provisioner": "csi.trident.netapp.io",
            "available": "eligible",
            "allowVolumeExpansion": "true",
            "reclaimPolicy": "Delete",
            "volumeBindingMode": "Immediate",
            "metadata": {
                "createdBy": "system",
                "creationTimestamp": "2022-11-03T18:40:03Z",
                "modificationTimestamp": "2022-11-03T18:40:03Z",
                "labels": []
            }
        }
    ]
}
```

저장소 백엔드를 나열합니다

사용 가능한 저장소 백엔드를 나열할 수 있습니다.

다음과 같은 REST API 호출을 수행한다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
가져오기	/accounts/{account_id}/topology/v1/storageBackends

curl 예: 모든 저장소 백엔드에 대한 모든 데이터를 반환합니다

```
curl --request GET \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/topology/v1/storageBackends"
\
--include \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

JSON 출력 예


```
{
  "items": [
    {
      "backendCredentialsName": "10.191.77.177",
      "backendName": "myinchunhcluster-1",
      "backendType": "ONTAP",
      "backendVersion": "9.8.0",
      "configVersion": "Not applicable",
      "health": "Not applicable",
      "id": "46467c16-1585-4b71-8e7f-f0bc5ff9da15",
      "location": "nalab2",
      "metadata": {
        "createdBy": "4c483a7e-207b-4f9a-87b7-799a4629d7c8",
        "creationTimestamp": "2021-07-30T14:26:19Z",
        "modificationTimestamp": "2021-07-30T14:26:19Z"
      },
      "ontap": {
        "backendManagementIP": "10.191.77.177",
        "managementIPs": [
          "10.191.77.177",
          "10.191.77.179"
        ]
      },
      "protectionPolicy": "Not applicable",
      "region": "Not applicable",
      "state": "Running",
      "stateUnready": [],
      "type": "application/astra-storageBackend",
      "version": "1.0",
      "zone": "Not applicable"
    }
  ]
}
```


자가 관리 클러스터에 동적 **ANF** 풀 사용

ANF 스토리지 백엔드가 있는 전용 사내 클러스터에서 관리되는 앱을 백업할 때 동적 ANF 풀 기능을 활성화해야 합니다. 이는 용량 풀을 확장 및 계약할 때 사용할 구독 ID를 제공하여 수행됩니다.

동적 ANF 풀은 ANF(Azure NetApp Files) 스토리지 백엔드를 사용하는 Astra 관리 애플리케이션의 기능입니다. 이러한 애플리케이션을 백업할 때 Astra는 영구 볼륨이 속한 용량 풀을 자동으로 확장 및 축소하며, 이는 1.5배 정도입니다. 이를 통해 추가 영구 비용을 부과하지 않고도 백업에 충분한 공간을 확보할 수 있습니다. 을 참조하십시오 "Azure 애플리케이션 백업" 를 참조하십시오.

1단계**: Azure** 구독 식별자를 추가합니다

다음과 같은 REST API 호출을 수행한다.

서비스 보안 주체에 대한 구독 ID 및 base64 값을 포함하여 사용자 환경에 맞게 JSON 입력 예제를 업데이트해야 합니다.

HTTP 메서드 및 끝점입니다

이 REST API 호출은 다음과 같은 메소드와 엔드포인트를 사용합니다.

HTTP 메소드	경로
게시	/accounts/{account_id}/core/v1/credentials

컬의 예

```
curl --request POST \
--location
"https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/credentials" \
--include \
--header "Content-Type: application/astra-credential+json"
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN" \
--data  @JSONinput
```

JSON 입력 예


```
{
  "keyStore": {
    "privKey": "SGkh",
    "pubKey": "UGhpcyCpcyBhbiBleGFtcGxlLg==",
    "base64":
"fwogICAgJmFwcElkIjogIjY4ZmSiODFiLTY0YWYtNDdjNC04ZjUzLWE2NDdlZTUzMGZkZCIsC
iAgICAiZGlzcGxheU5hbWUiOiAic3AtYXN0cmEtZGV2LXFhIiwKICAgICJuYW1lIjogImh0dHA
6Ly9zcC1hc3RyYS1kZXYtcWEiLAogICAgInBhc3N3b3JkIjogIllLQThRfk9IVVJkZWZYM0pST
WJlLnpUeFBleVE0UnNwTG9DcUJjazAiLAogICAgInRlbmFudCI6ICIwMTFjZGY2Yy03NTEyLTQ
3MDUtYjI0ZS03NzIxYWZkOGNhMzciLAogICAgInN1YnNjcmlwdGlvbklkIjogImIyMDAxNTVmL
TAwMWEtNDNiZS04N2JlLTNlZGRlODNhY2VmNCIKfQ=="
  },
  "name": "myCert",
  "type": "application/astra-credential",
  "version": "1.1",
  "metadata": {
    "labels": [
      {
        "name": "astra.netapp.io/labels/read-only/credType",
        "value": "service-account"
      },
      {
        "name": "astra.netapp.io/labels/read-only/cloudName",
        "value": "OCP"
      },
      {
        "name": "astra.netapp.io/labels/read-only/azure/subscriptionID",
        "value": "b212156f-001a-43be-87be-3edde83acef5"
      }
    ]
  }
}
```


2단계**:** 필요한 경우 버킷을 추가합니다

필요한 경우 관리되는 애플리케이션에 버킷을 추가해야 합니다.

3단계**:** 관리되는 앱의 백업을 수행합니다

워크플로우를 수행합니다 "앱에 대한 백업을 생성합니다". 원래 영구 볼륨이 있는 용량 풀은 자동으로 확장 및 축소됩니다.

4단계**:** 이벤트 로그를 검토합니다

작업 이벤트는 백업 중에 기록됩니다. 워크플로우를 수행합니다 "알림을 나열합니다" 를 눌러 메시지를 봅니다.

저작권 정보

Copyright © 2025 NetApp, Inc. All Rights Reserved. 미국에서 인쇄됨 본 문서의 어떠한 부분도 저작권 소유자의 사전 서면 승인 없이는 어떠한 형식이나 수단(복사, 녹음, 녹화 또는 전자 검색 시스템에 저장하는 것을 비롯한 그래픽, 전자적 또는 기계적 방법)으로도 복제될 수 없습니다.

NetApp이 저작권을 가진 자료에 있는 소프트웨어에는 아래의 라이센스와 고지사항이 적용됩니다.

본 소프트웨어는 NetApp에 의해 '있는 그대로' 제공되며 상품성 및 특정 목적에의 적합성에 대한 명시적 또는 묵시적 보증을 포함하여(이에 제한되지 않음) 어떠한 보증도 하지 않습니다. NetApp은 대체품 또는 대체 서비스의 조달, 사용 불능, 데이터 손실, 이익 손실, 영업 중단을 포함하여(이에 국한되지 않음), 이 소프트웨어의 사용으로 인해 발생하는 모든 직접 및 간접 손해, 우발적 손해, 특별 손해, 징벌적 손해, 결과적 손해의 발생에 대하여 그 발생 이유, 책임론, 계약 여부, 엄격한 책임, 불법 행위(과실 또는 그렇지 않은 경우)와 관계없이 어떠한 책임도 지지 않으며, 이와 같은 손실의 발생 가능성이 통지되었다 하더라도 마찬가지입니다.

NetApp은 본 문서에 설명된 제품을 언제든지 예고 없이 변경할 권리를 보유합니다. NetApp은 NetApp의 명시적인 서면 동의를 받은 경우를 제외하고 본 문서에 설명된 제품을 사용하여 발생하는 어떠한 문제에도 책임을 지지 않습니다. 본 제품의 사용 또는 구매의 경우 NetApp에서는 어떠한 특허권, 상표권 또는 기타 지적 재산권이 적용되는 라이센스도 제공하지 않습니다.

본 설명서에 설명된 제품은 하나 이상의 미국 특허, 해외 특허 또는 출원 중인 특허로 보호됩니다.

제한적 권리 표시: 정부에 의한 사용, 복제 또는 공개에는 DFARS 252.227-7013(2014년 2월) 및 FAR 52.227-19(2007년 12월)의 기술 데이터-비상업적 품목에 대한 권리(Rights in Technical Data -Noncommercial Items) 조항의 하위 조항 (b)(3)에 설명된 제한사항이 적용됩니다.

여기에 포함된 데이터는 상업용 제품 및/또는 상업용 서비스(FAR 2.101에 정의)에 해당하며 NetApp, Inc.의 독점 자산입니다. 본 계약에 따라 제공되는 모든 NetApp 기술 데이터 및 컴퓨터 소프트웨어는 본질적으로 상업용이며 개인 비용만으로 개발되었습니다. 미국 정부는 데이터가 제공된 미국 계약과 관련하여 해당 계약을 지원하는 데에만 데이터에 대한 전 세계적으로 비독점적이고 양도할 수 없으며 재사용이 불가능하며 취소 불가능한 라이센스를 제한적으로 가집니다. 여기에 제공된 경우를 제외하고 NetApp, Inc.의 사전 서면 승인 없이는 이 데이터를 사용, 공개, 재생산, 수정, 수행 또는 표시할 수 없습니다. 미국 국방부에 대한 정부 라이센스는 DFARS 조항 252.227-7015(b)(2014년 2월)에 명시된 권한으로 제한됩니다.

상표 정보

NETAPP, NETAPP 로고 및 http://www.netapp.com/TM에 나열된 마크는 NetApp, Inc.의 상표입니다. 기타 회사 및 제품 이름은 해당 소유자의 상표일 수 있습니다.