Documentation Astra Control Automation 22.08

Astra Automation 22.08

NetApp
February 20, 2023

Table des matières

Documentation Astra Control Automation 22.08

Notes de mise à jour

À propos de cette version

La documentation de ce site décrit l'API REST Astra Control et les technologies d'automatisation connexes, disponibles avec la version d'Astra Control du 2022 août 22.08. En particulier, cette version de l'API REST est incluse avec les 22.08 versions correspondantes du centre de contrôle Astra et du service de contrôle Astra.

Consultez les pages et sites suivants pour plus d'informations sur cette version ainsi que sur les versions précédentes :

- "Nouveautés de l'API REST Astra Control"
- "Ressources REST et terminaux"
- "Documentation Astra Control Center 22.08"
- "Documentation relative au service après-vente Astra Control"
- "Versions antérieures de la documentation d'Astra Automation"

Suivez-nous sur Twitter @NetAppDoc. Envoyez vos commentaires sur la documentation en devenant un "Contributeur GitHub" ou en envoyant un e-mail à doccomments@netapp.com.

Nouveautés de l'API REST Astra Control

NetApp met régulièrement à jour l'API REST Astra Control pour vous apporter de nouvelles fonctionnalités, des améliorations et des correctifs.

10 août 2022 (22.08)

Cette version inclut une extension et une mise à jour de l'API REST, ainsi que des fonctions de sécurité et d'administration améliorées.

Nouvelles ressources Astra

Trois nouveaux types de ressources ont été ajoutés : **certificat**, **Groupe** et **AppMirror**. En outre, les versions de plusieurs ressources existantes ont été mises à jour.

Authentification LDAP

Vous pouvez configurer Astra Control Center pour qu'il s'intègre à un serveur LDAP afin d'authentifier les utilisateurs Astra sélectionnés. Voir "Configuration LDAP" pour en savoir plus.

Crochet d'exécution amélioré

Le support pour les crochets d'exécution a été ajouté avec la version 21.12 de l'Astra Control. En plus des crochets d'exécution pré-instantané et post-instantané existants, vous pouvez désormais configurer les types de crochets d'exécution suivants avec la version 22.08 :

- Avant sauvegarde

- Post-sauvegarde
- Post-restauration

Astra Control permet désormais d’utiliser le même script pour plusieurs crochets d’exécution.

Réplication des applications à l’aide de SnapMirror

Vous pouvez désormais répliquer les données et les changements d’applications entre les clusters à l’aide de la technologie NetApp SnapMirror. Cette amélioration peut être utilisée pour améliorer la continuité de l’activité et les capacités de restauration.

Informations associées

- "Astra Control Center : les nouveautés"
- "Astra Control Service : les nouveautés"

26 avril 2022 (22.04)

Cette version inclut une extension et une mise à jour de l’API REST, ainsi que des fonctions de sécurité et d’administration améliorées.

Nouvelles ressources Astra

Deux nouveaux types de ressources ont été ajoutés : **Package** et **Upgrade**. De plus, les versions de plusieurs ressources existantes ont été mises à niveau.

RBAC amélioré avec granularité de l’espace de noms

Lors de la liaison d’un rôle à un utilisateur associé, vous pouvez limiter les espaces de noms auxquels l’utilisateur a accès. Voir la référence **role Binding API** et "Sécurité RBAC" pour en savoir plus.

Dépose du godet

Vous pouvez retirer un godet lorsqu’il n’est plus nécessaire ou qu’il ne fonctionne pas correctement.

Prise en charge de Cloud Volumes ONTAP

Cloud Volumes ONTAP est désormais pris en charge en tant que système back-end de stockage.

Autres améliorations produit

Plusieurs améliorations supplémentaires ont été apportées aux deux versions d’Astra Control, notamment :

- Entrée générique pour Astra Control Center
- Cluster privé à AKS
- Prise en charge de Kubernetes 1.22
- Prise en charge de la gamme VMware Tanzu

Consultez la page **Nouveautés** des sites de documentation Astra Control Center et Astra Control Service.

Informations associées

- "Astra Control Center : les nouveautés"

* "Astra Control Service : les nouveautés"

14 décembre 2021 (21.12)

Cette version inclut une extension de l'API REST ainsi qu'un changement dans la structure de documentation pour mieux prendre en charge l'évolution d'Astra Control à travers les mises à jour futures.

Documentation distincte sur l'automatisation Astra pour chaque version d'Astra Control

Chaque nouvelle version d'Astra Control comprend une API REST distincte qui a été améliorée et adaptée aux caractéristiques de cette version. La documentation relative à chaque version de l'API REST Astra Control est désormais disponible sur son propre site Web dédié et dans le référentiel de contenu GitHub associé. Le site principal du document "Automatisation du contrôle d'Astra" contient toujours la documentation de la version la plus récente. Voir "Versions antérieures de la documentation Astra Control Automation" pour plus d'informations sur les versions précédentes.

Extension des types de ressources REST

Le nombre de types de ressources REST a continué de s'étendre, en mettant l'accent sur les crochets d'exécution et les systèmes back-end de stockage. Les nouvelles ressources incluent : compte, crochet d'exécution, source de hook, outrepassement de point d'exécution, nœud de cluster, gestion du système de stockage back-end, de l'espace de noms, du périphérique de stockage et du nœud de stockage. Voir "Ressources" pour en savoir plus.

Kit de développement logiciel NetApp Astra Control Python

Le kit de développement logiciel NetApp Astra Control Python est un pack open source qui facilite le développement du code d'automatisation pour votre environnement Astra Control. Au cœur du jeu de développement Astra, qui comprend un ensemble de classes pour extraire la complexité des appels de l'API REST. Il existe également un script de boîte à outils pour exécuter des tâches administratives spécifiques en enveloppant et en retirant les classes Python. Voir "Kit de développement logiciel NetApp Astra Control Python" pour en savoir plus.

5 août 2021 (21.08)

Avec cette version, il introduit un nouveau modèle de déploiement Astra et un important élargissement de l'API REST.

Modèle de déploiement d'Astra Control Center

Outre l'offre Astra Control Service proposée en tant que service de cloud public, cette version inclut également le modèle de déploiement sur site d'Astra Control Center. Vous pouvez installer Astra Control Center sur votre site pour gérer votre environnement Kubernetes local. Les deux modèles de déploiement Astra Control partagent la même API REST, avec de légères différences notées dans la documentation.

Extension des types de ressources REST

Avec l'API REST Astra Control, le nombre de ressources accessibles est considérablement étendu. Un grand nombre de ces nouvelles ressources constituent le socle de l'offre Astra Control Center sur site. Les nouvelles ressources disponibles sont : ASUP, droit, fonctionnalité, licence, définition abonnement, compartiment, cloud, cluster, cluster géré, système back-end et classe de stockage. Voir "Ressources" pour en savoir plus.

Terminaux supplémentaires prenant en charge un déploiement Astra

Outre les ressources REST étendues, plusieurs autres terminaux d'API sont disponibles pour prendre en charge le déploiement d'Astra Control.

Prise en charge d'OpenAPI
: Les noeuds finaux OpenAPI donnent accès au document JSON OpenAPI actuel et à d'autres ressources associées.

Prise en charge d'OpenMetrics
: Les noeuds finaux OpenMetrics fournissent un accès aux mesures du compte via la ressource OpenMetrics.

15 avril 2021 (21.04)

Cette version comprend de nouvelles fonctionnalités et améliorations suivantes.

Introduction de l'API REST

L'API REST Astra Control est disponible avec l'offre de service Astra Control. Sa création repose sur les technologies REST et les meilleures pratiques actuelles. Il constitue le socle de l'automatisation de vos déploiements Astra et inclut plusieurs fonctionnalités et avantages :

Ressources
: Quatorze types de ressources REST sont disponibles.

Accès au jeton d'API
: L'accès à l'API REST est assuré via un jeton d'accès à l'API que vous pouvez générer à partir de l'interface utilisateur Web Astra. Le jeton API fournit un accès sécurisé à l'API.

Prise en charge des collections
: Il existe un ensemble riche de paramètres de requête qui peuvent être utilisés pour accéder aux collections de ressources. Certaines opérations prises en charge incluent le filtrage, le tri et la pagination.

Problèmes connus

Il est recommandé de passer en revue tous les problèmes connus relatifs à la version actuelle de l'API REST Astra Control. Les problèmes connus identifient les problèmes susceptibles de vous empêcher d'utiliser le produit avec succès.

> Il n'y a pas de nouveaux problèmes connus avec la version 22.08 de l'API REST Astra Control. Les problèmes décrits ci-dessous ont été découverts dans les versions précédentes et sont toujours applicables avec la version actuelle.

Tous les périphériques de stockage d'un nœud de stockage interne ne sont pas détectés

Lors de l'émission d'un appel d'API REST pour récupérer les périphériques de stockage définis dans un nœud de stockage, seuls les périphériques de stockage de données Astra sont découverts. Tous les périphériques ne sont pas renvoyés.

Présentation des fonctionnalités et avantages

Astra Control Center et Astra Control Service fournissent une API REST commune que vous pouvez accéder directement via un langage de programmation ou un utilitaire tel que Curl. Les principaux points forts et avantages de l'API sont présentés ci-dessous.

> Pour accéder à l'API REST, vous devez d'abord vous connecter à l'interface utilisateur Web Astra et générer un jeton API. Vous devez inclure le token à chaque requête d'API.

Basée sur la technologie REST

L'API de contrôle d'Astra a été créée à l'aide de la technologie REST et des bonnes pratiques actuelles. La technologie centrale inclut HTTP, JSON et RBAC.

Prise en charge des deux modèles de déploiement Astra Control

Astra Control Service est utilisé dans l'environnement de cloud public, tandis qu'Astra Control Center est conçu pour vos déploiements sur site. Une API REST prend en charge ces deux modèles de déploiement.

Effacez le mappage entre les ressources du terminal REST et le modèle d'objet

Les terminaux REST externes utilisés pour accéder au mappage des ressources vers un modèle d'objet cohérent géré en interne par le service Astra. Le modèle d'objet est conçu à l'aide de la modélisation de relation d'entité (ER) qui permet de définir clairement les actions et les réponses de l'API.

Ensemble complet de paramètres de requête

L'API REST fournit un ensemble complet de paramètres de requête que vous pouvez utiliser pour accéder aux collections de ressources. Certaines opérations prises en charge incluent le filtrage, le tri et la pagination.

Alignement avec l'interface utilisateur Web Astra Control

La conception de l'interface utilisateur Web Astra est alignée avec l'API REST, ce qui garantit la cohérence entre les deux chemins d'accès et l'expérience utilisateur.

Données robustes de débogage et de détermination des problèmes

L'API REST Astra Control offre une fonctionnalité de débogage et de détermination des problèmes robuste, y compris les événements système et les notifications utilisateur.

Processus de flux de travail

Un ensemble de workflows est fourni pour vous aider au développement de votre code d'automatisation. Les workflows sont organisés en deux catégories principales : infrastructure et gestion.

Socle des technologies d'automatisation avancées

Vous pouvez non seulement accéder directement à l'API REST, mais aussi utiliser d'autres technologies d'automatisation basées sur l'API REST.

Fait partie de la documentation de la gamme Astra

La documentation relative à l'automatisation du contrôle Astra fait partie de la documentation de la gamme Astra la plus vaste. Voir "Documentation Astra" pour en savoir plus.

Commencez

Avant de commencer

Vous pouvez vous préparer rapidement à vous lancer avec l'API REST Astra Control en passant en revue les étapes ci-dessous.

Possèdent des identifiants de compte Astra

Vous aurez besoin d'informations d'identification Astra pour vous connecter à l'interface utilisateur Web Astra et générer un jeton API. Avec Astra Control Center, vous gérez ces informations d'identification localement. Avec le service de contrôle Astra, les informations d'identification du compte sont accessibles via le service **Auth0**.

Concepts de base de Kubernetes

Vous devez maîtriser plusieurs concepts Kubernetes de base. Voir "Concepts de base de Kubernetes" pour en savoir plus.

Examiner les concepts DE REPOS et la mise en œuvre

Assurez-vous de passer en revue "Implémentation DE l'APPLICATION REST de cœur" Pour plus d'informations sur les concepts DE REPOS et les détails concernant la conception de l'API REST Astra Control.

En savoir plus

Vous devez connaître les ressources d'information supplémentaires, comme le suggère le "Ressources supplémentaires".

Obtenir un jeton API

Vous devez obtenir un jeton API Astra pour utiliser l'API REST Astra Control.

Introduction

Un jeton API identifie l'appelant auprès d'Astra et doit être inclus avec chaque appel d'API REST.

- Vous pouvez générer un jeton API à l'aide de l'interface utilisateur Web Astra.
- L'identité de l'utilisateur portée avec le token est déterminée par l'utilisateur qui crée le token.
- Le jeton doit être inclus dans le `Authorization` En-tête de requête HTTP.
- Un jeton n'expire jamais après sa création.
- Vous pouvez révoquer un jeton dans l'interface utilisateur Web Astra.

Informations associées

- "Révoquer un jeton API"

Créez un jeton API Astra

Les étapes suivantes décrivent comment créer un jeton API Astra.

Avant de commencer

Vous avez besoin d’informations d’identification pour un compte Astra.

Description de la tâche

Cette tâche génère un jeton API sur l’interface Web Astra. Vous devez également récupérer l’ID de compte qui est également nécessaire lors de la réalisation d’appels API.

Étapes

1. Connectez-vous à Astra à l’aide de vos identifiants de compte.

   Accédez au site suivant pour le service Astra Control : "https://astra.netapp.io"

2. Cliquez sur l’icône figure en haut à droite de la page et sélectionnez **API Access**.
3. Cliquez sur **Generate API token** sur la page et, dans la fenêtre contextuelle, cliquez sur **Generate API token**.
4. Cliquez sur l’icône pour copier la chaîne de token dans le presse-papiers et l’enregistrer dans votre éditeur.
5. Copiez et enregistrez l’ID de compte disponible sur la même page.

Une fois que vous avez terminé

Lorsque vous accédez à l’API REST Astra Control via Curl ou un langage de programmation, vous devez inclure le jeton de support d’API dans le protocole HTTP `Authorization` en-tête de demande.

Bonjour tout le monde

Vous pouvez lancer une commande curl simple sur la CLI de votre station de travail pour commencer à utiliser l’API REST Astra Control et confirmer sa disponibilité.

Avant de commencer

L’utilitaire Curl doit être disponible sur votre poste de travail local. Vous devez également disposer d’un jeton API et de l’identifiant de compte associé. Voir "Obtenir un jeton API" pour en savoir plus.

Exemple de boucle

La commande Curl suivante récupère la liste des utilisateurs Astra. Fournissez les <ACCOUNT_ID> et <API_TOKEN> appropriés comme indiqué.

```
curl --location --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/core/v1/users' --header
'Content-Type: application/json' --header 'Authorization: Bearer
<API_TOKEN>'
```

Exemple de sortie JSON


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


Préparez l'utilisation des workflows

Vous devez aussi connaître l'entreprise et le format des workflows Astra avant de les utiliser avec un déploiement en direct.

Introduction

Un *workflow* est une séquence d'une ou de plusieurs étapes nécessaires à la réalisation d'une tâche ou d'un objectif administratif spécifique. Chaque étape d'un workflow de contrôle Astra est l'une des suivantes :

- Appel d'API REST (avec des détails tels que des exemples Curl et JSON)
- Appel d'un autre flux de travail Astra
- Tâche associée divers (par exemple, prise d'une décision de conception requise)

Ces flux de travail incluent les étapes clés et les paramètres nécessaires à l'exécution de chaque tâche. Ils constituent un point de départ pour la personnalisation de votre environnement d'automatisation.

Paramètres d'entrée communs

Les paramètres d'entrée décrits ci-dessous sont communs à tous les échantillons curl utilisés pour illustrer un appel API REST.

> Comme ces paramètres d'entrée sont universellement requis, ils ne sont pas décrits plus en détail dans les flux de travail individuels. Si des paramètres d'entrée supplémentaires sont utilisés pour un exemple de boucle spécifique, ils sont décrits dans la section **Paramètres d'entrée supplémentaires**.

Paramètres de chemin

Le chemin du noeud final utilisé avec chaque appel d'API REST inclut les paramètres suivants. Voir aussi "Format d'URL" pour en savoir plus.

ID de compte

Il s'agit de la valeur UUIDv4 identifiant le compte Astra sur lequel l'opération API s'exécute. Voir "Obtenir un jeton API" Pour plus d'informations sur la localisation de votre identifiant de compte.

En-têtes de demande

En fonction de l'appel d'API REST, vous devrez peut-être inclure plusieurs en-têtes de requête.

Autorisation

Tous les appels d'API dans les workflows requièrent un jeton d'API pour identifier l'utilisateur. Vous devez inclure le token dans le `Authorization` en-tête de demande. Voir "Obtenir un jeton API" Pour plus d'informations sur la génération d'un jeton API.

Types de contenu

Avec LA PUBLICATION HTTP et LES requêtes PUT où JSON est inclus dans le corps de la demande, vous devez déclarer le type de support en fonction de la ressource Astra. Par exemple, vous pouvez inclure l'en-tête `Content-Type: application/astra-appSnap+json` lors de la création d'un snapshot pour une application gérée.

Accepter

Vous pouvez déclarer le type de support spécifique du contenu que vous attendez dans la réponse en fonction de la ressource Astra. Par exemple, vous pouvez inclure l'en-tête `Accept: application/astra-appBackup+json` lors de la liste des sauvegardes pour une application gérée. Cependant, pour plus de simplicité, les échantillons curl dans les flux de production acceptent tous les types de support.

Présentation des jetons et des identificateurs

Le jeton API et les autres valeurs d'ID utilisées avec les exemples de boucles sont opaques sans signification perceptible. Afin d'améliorer la lisibilité des échantillons, les valeurs réelles de jeton et d'ID ne sont pas utilisées. Des mots-clés réservés plus petits sont utilisés, ce qui présente plusieurs avantages :

- Les échantillons curl et JSON sont plus clairs et plus faciles à comprendre.
- Comme tous les mots-clés utilisent le même format avec des crochets et des lettres majuscules, vous pouvez rapidement identifier l'emplacement et le contenu à insérer ou extraire.
- Aucune valeur n'est perdue car les paramètres d'origine ne peuvent pas être copiés et utilisés avec un déploiement réel.

Voici quelques-uns des mots-clés réservés communs utilisés dans les exemples curl. Cette liste n'est pas exhaustive et des mots-clés supplémentaires sont utilisés au besoin. Leur signification devrait être évidente sur la base du contexte.

Mot-clé	Type	Description
<ID_COMPTE>	Chemin	Valeur UUIDv4 identifiant le compte sur lequel l'opération API s'exécute.
<API_TOKEN>	En-tête	Le jeton porteur identifiant et autorise l'appelant.

Mot-clé	Type	Description
<ID_APP>	Chemin	Valeur UUIDv4 identifiant l'application pour l'appel d'API.

Catégories de flux de travail

Selon votre modèle de déploiement, vous pouvez consulter deux catégories de workflows Astra, Si vous utilisez Astra Control Center, vous devez d'abord les workflows d'infrastructure, puis passer aux workflows de gestion. Avec Astra Control Service, vous pouvez généralement accéder directement aux workflows de gestion.

> ⓘ Les exemples de boucles des flux de travail utilisent l'URL du service de contrôle Astra. Vous devez modifier l'URL lorsque vous utilisez le centre de contrôle Astra sur site en fonction de votre environnement.

Workflows d'infrastructure

Ces workflows sont associés à l'infrastructure Astra, notamment les identifiants, les compartiments et les systèmes de stockage back-end. Elles sont nécessaires avec le centre de contrôle Astra, mais dans la plupart des cas peuvent également être utilisées avec le service de contrôle Astra. Les flux de travail se concentrent sur les tâches requises pour établir et gérer un cluster géré par Astra.

Flux de travail de gestion

Vous pouvez utiliser ces flux de travail une fois que vous avez un cluster géré. Les workflows sont axés sur la protection des applications, ainsi que sur les opérations de prise en charge, comme la sauvegarde, la restauration et le clonage d'une application.

Concepts de base de Kubernetes

Plusieurs concepts Kubernetes sont pertinents pour l'utilisation de l'API REST d'Astra.

Objets

Les objets gérés dans un environnement Kubernetes sont des entités permanentes qui représentent la configuration du cluster. Ces objets décrivent collectivement l'état du système, y compris la charge de travail du cluster.

Espaces de noms

Les espaces de noms fournissent une technique d'isolement des ressources au sein d'un même cluster. Cette structure organisationnelle est utile pour répartir les types de travail, d'utilisateurs et de ressources. Les objets avec un *namespace scope* doivent être uniques dans le namespace, tandis que ceux avec un *cluster scope* doivent être uniques dans tout le cluster.

Étiquettes

Des étiquettes peuvent être associées aux objets Kubernetes. Ils décrivent les attributs à l'aide de paires de valeurs clés et peuvent appliquer une organisation arbitraire sur le cluster, ce qui peut être utile à une entreprise, mais ne fonctionne pas dans le système Kubernetes de base.

Implémentation DE l'APPLICATION REST de cœur

Services web REST

Representational State Transfer (REST) est un style qui permet de créer des applications Web distribuées. Lorsqu'il est appliqué à la conception d'une API de services Web, il établit un ensemble de technologies classiques et de meilleures pratiques pour l'exposition des ressources basées sur serveur et la gestion de leurs États. REST fournit une base cohérente pour le développement d'applications. Les détails de chaque API peuvent varier en fonction des choix de conception spécifiques. Vous devez connaître les caractéristiques de l'API REST Astra Control avant de l'utiliser avec un déploiement en direct.

Ressources et représentation d'état

Les ressources sont les composants de base d'un système basé sur le Web. Lors de la création d'une application de services Web REST, les premières tâches de conception incluent :

- Identification des ressources système ou serveur

  Chaque système utilise et gère les ressources. Une ressource peut être un fichier, une transaction commerciale, un processus ou une entité administrative. L'une des premières tâches de conception d'une application basée sur des services Web REST consiste à identifier les ressources.

- Définition des États de ressource et des opérations d'état associées

  Les ressources se trouvent toujours dans un des États finis. Les États, ainsi que les opérations associées utilisées pour affecter les changements d'état, doivent être clairement définis.

Terminaux URI

Chaque ressource REST doit être définie et mise à disposition à l'aide d'un schéma d'adressage bien défini. Les noeuds finaux où les ressources sont situées et identifiées utilisent un URI (Uniform Resource identifier). L'URI fournit un cadre général pour créer un nom unique pour chaque ressource du réseau. L'URL (Uniform Resource Locator) est un type d'URI utilisé avec les services Web pour identifier et accéder aux ressources. Les ressources sont généralement exposées dans une structure hiérarchique similaire à un répertoire de fichiers.

Messages HTTP

Le protocole HTTP (Hypertext Transfer Protocol) est le protocole utilisé par le client et le serveur de services Web pour échanger des messages de requête et de réponse sur les ressources. Dans le cadre de la conception d'une application de services Web, les méthodes HTTP sont mappées aux ressources et aux actions de gestion d'état correspondantes. Le HTTP est sans état. Par conséquent, pour associer un ensemble de requêtes et de réponses associées dans le cadre d'une transaction, des informations supplémentaires doivent être incluses dans les en-têtes HTTP des flux de données de requête et de réponse.

Formatage JSON

Bien que l'information puisse être structurée et transférée de plusieurs façons entre un client de services Web et un serveur, l'option la plus populaire est JavaScript Object notation (JSON). JSON est une norme de l'industrie qui représente les structures de données simples en texte brut et permet de transférer les informations d'état décrivant les ressources. L'API REST Astra Control utilise JSON pour formater les données contenues dans le corps de chaque requête et réponse HTTP.

Ressources et collections

L'API REST Astra Control permet d'accéder aux instances de ressources et aux ensembles d'instances de ressources.

> Sur le plan conceptuel, une ressource REST * est similaire à un **objet** tel que défini avec les langages et systèmes de programmation orientés objet (OOP). Parfois, ces termes sont utilisés de manière interchangeable. Mais en général, la méthode « ressource » est préférée lorsqu'elle est utilisée dans le contexte de l'API REST externe tandis que l'option « objet » est utilisée pour les données d'instance avec état correspondantes stockées sur le serveur.

Caractéristiques des ressources Astra

L'API REST Astra Control est conforme aux principes de conception RESTful. Chaque instance de ressource Astra est créée en fonction d'un type de ressource bien défini. Un ensemble d'instances de ressource du même type est appelé **collection**. Les appels de l'API agissent sur des ressources individuelles ou des collections de ressources.

Types de ressource

Les types de ressource inclus avec l'API REST Astra Control présentent les caractéristiques suivantes :

- Chaque type de ressource est défini à l'aide d'un schéma (généralement au format JSON).
- Chaque schéma de ressource inclut le type et la version de ressource
- Les types de ressources sont globalement uniques

Instances de ressources

Les instances de ressources disponibles via l'API REST Astra Control présentent les caractéristiques suivantes :

- Les instances de ressources sont créées en fonction d'un type de ressource unique
- Le type de ressource est indiqué à l'aide de la valeur Type de support
- Les instances sont composées de données avec état qui sont conservées par le service Astra
- Chaque instance est accessible via une URL unique et longue durée
- Dans les cas où une instance de ressource peut avoir plusieurs représentations, différents types de support peuvent être utilisés pour demander la représentation souhaitée

Collections de ressources

Les collections de ressources disponibles via l'API REST Astra Control présentent les caractéristiques suivantes :

- L'ensemble des instances de ressource d'un type de ressource unique est appelé collection

- Les collections de ressources ont une URL unique et de longue durée

Identifiants d'instances

Un identifiant est attribué à chaque instance de ressource lors de sa création. Cet identifiant est une valeur UUIDv4 128 bits. Les valeurs UUIDv4 attribuées sont globalement uniques et immuables. Après l'émission d'un appel API qui crée une nouvelle instance, une URL avec l'ID associé est renvoyée à l'appelant dans un `Location` En-tête de la réponse HTTP. Vous pouvez extraire l'identificateur et l'utiliser sur les appels suivants lorsque vous faites référence à l'instance de ressource.

> L'identifiant de ressource est la clé principale utilisée pour les collections.

Structure commune pour les ressources Astra

Chaque ressource Astra Control est définie à l'aide d'une structure commune.

Les données communes

Chaque ressource Astra contient les valeurs-clés indiquées dans le tableau suivant.

Clé	Description
type	Type de ressource unique et global appelé **type de ressource**.
version	Un identificateur de version appelé **version de ressource**.
id	Un identificateur unique global appelé **identificateur de ressource**.
les métadonnées	Objet JSON contenant diverses informations, y compris les étiquettes de l'utilisateur et du système.

Objet de métadonnées

L'objet de métadonnées JSON inclus avec chaque ressource Astra contient les valeurs de clé indiquées dans le tableau suivant.

Clé	Description
étiquettes	Tableau JSON d'étiquettes spécifiées par le client associées à la ressource.
CréationTimestamp	Chaîne JSON contenant un horodatage indiquant quand la ressource a été créée.
ModificationTimestamp	Chaîne JSON contenant un horodatage au format ISO-8601 indiquant quand la ressource a été modifiée pour la dernière fois.
CreatedBy	Chaîne JSON contenant l'identifiant UUIDv4 de l'ID utilisateur qui a créé la ressource. Si la ressource a été créée par un composant système interne et qu'aucun UUID n'est associé à l'entité de création, l'UUID **null** est utilisé.

État de la ressource

Ressources sélectionnées a `state` valeur utilisée pour orchestrer les transitions de cycle de vie et contrôler l'accès.

Détails HTTP

L'API REST Astra Control utilise HTTP et les paramètres associés pour agir sur les instances et les collections de ressources. Les détails de l'implémentation HTTP sont présentés ci-dessous.

Transactions API et modèle CRUD

L'API REST d'Astra Control met en œuvre un modèle transactionnel avec des opérations et des transitions d'état clairement définies.

Transaction d'API de demande et de réponse

Chaque appel d'API REST est exécuté sous forme de requête HTTP auprès du service Astra. Chaque requête génère une réponse associée au client. Cette paire demande-réponse peut être considérée comme une transaction API.

Prise en charge du modèle opérationnel CRUD

Chaque instance et collection de ressources disponibles via l'API REST Astra Control est accessible en fonction du modèle **CRUD**. Il existe quatre opérations, chacune étant mappée à une seule méthode HTTP. Ses opérations sont les suivantes :

- Création
- Lecture
- Mise à jour
- Supprimer

Pour certaines ressources Astra, seul un sous-ensemble de ces opérations est pris en charge. Vous devez consulter le "Référence API" Pour plus d'informations sur un appel d'API spécifique.

Méthodes HTTP

Les méthodes HTTP ou verbes pris en charge par l'API sont présentées dans le tableau ci-dessous.

Méthode	CRUD	Description
OBTENEZ	Lecture	Récupère les propriétés d'un objet pour une instance ou une collection de ressources. Cette opération est considérée comme une opération **list** lorsqu'elle est utilisée avec une collection.
POST	Création	Crée une nouvelle instance de ressource basée sur les paramètres d'entrée. L'URL à long terme est renvoyée dans un `Location` en-tête de réponse.
EN	Mise à jour	Met à jour une instance de ressource entière avec le corps de demande JSON fourni. Les valeurs clés qui ne sont pas modifiables par l'utilisateur sont conservées.
SUPPRIMER	Supprimer	Supprime une instance de ressource existante.

En-têtes de demande et de réponse

Le tableau suivant résume les en-têtes HTTP utilisés avec l'API REST Astra Control.

Voir "RFC 7232" et "RFC 7233" pour en savoir plus.

En-tête	Type	Remarques sur l’utilisation
Accepter	Demande	Si la valeur est "/" ou n’est pas fournie, `application/json` Est renvoyé dans l’en-tête de réponse Content-Type. Si la valeur est définie sur le type de support de ressource Astra, le même type de support est renvoyé dans l’en-tête Type de contenu.
Autorisation	Demande	Jeton porteur avec la clé API pour l’utilisateur.
Type de contenu	Réponse	Renvoyé en fonction du `Accept` en-tête de demande.
ETAG	Réponse	Inclus avec un succès tel que défini dans RFC 7232. La valeur est une représentation hexadécimale de la valeur MD5 pour l’ensemble de la ressource JSON.
IF-match	Demande	En-tête de demande préalable mise en œuvre comme décrit à la section 3.1 RFC 7232 et prise en charge des requêtes **PUT**.
Si-modifié-depuis	Demande	En-tête de demande préalable mise en œuvre comme décrit à la section 3.4 RFC 7232 et prise en charge des requêtes **PUT**.
Si-non modifié-depuis	Demande	En-tête de demande préalable mise en œuvre comme décrit à la section 3.4 RFC 7232 et prise en charge des requêtes **PUT**.
Emplacement	Réponse	Contient l’URL complète de la nouvelle ressource créée.

Paramètres de requête

Les paramètres de requête suivants peuvent être utilisés avec les collections de ressources. Voir "Utilisation des collections" pour en savoir plus.

Paramètre de requête	Description
inclure	Contient les champs à retourner lors de la lecture d’une collection.
filtre	Indique les champs devant correspondre pour qu’une ressource soit renvoyée lors de la lecture d’une collection.
Orderby	Détermine l’ordre de tri des ressources renvoyées lors de la lecture d’une collection.
limite	Limite le nombre maximal de ressources renvoyées lors de la lecture d’une collection.
ignorer	Définit le nombre de ressources à passer et ignorer lors de la lecture d’une collection.
nombre	Indique si le nombre total de ressources doit être renvoyé dans l’objet métadonnées.

Codes d’état HTTP

Les codes d’état HTTP utilisés par l’API REST Astra Control sont décrits ci-dessous.

L’API REST Astra Control utilise également la norme **Détails du problème pour les API HTTP**. Voir "Diagnostics et support" pour en savoir plus.

Code	Signification	Description
200	OK	Indique la réussite des appels qui ne créent pas une nouvelle instance de ressource.
201	Créé	Un objet est créé avec succès et l'en-tête de réponse d'emplacement inclut l'identifiant unique de l'objet.
204	Aucun contenu	La demande a réussi bien qu'aucun contenu n'ait été renvoyé.
400	Demande incorrecte	L'entrée de la demande n'est pas reconnue ou est inappropriée.
401	Non autorisé	L'utilisateur n'est pas autorisé et doit être autorisé.
403	Interdit	L'accès est refusé en raison d'une erreur d'autorisation.
404	Introuvable	La ressource mentionnée dans la demande n'existe pas.
409	Conflit	La tentative de création d'un objet a échoué car celui-ci existe déjà.
500	Erreur interne	Une erreur interne générale s'est produite sur le serveur.
503	Service indisponible	Le service n'est pas prêt à traiter la demande pour une raison quelconque.

Format d'URL

La structure générale de l'URL utilisée pour accéder à une instance de ressource ou à une collection via l'API REST est composée de plusieurs valeurs. Cette structure reflète le modèle d'objet sous-jacent et la conception du système.

En tant que racine

Le compte Astra est la racine du chemin de ressource vers chaque point final REST. Ainsi, tous les chemins d'accès de l'URL commencent par `/account/{account_id}` où `account_id` Est la valeur UUIDv4 unique du compte. Structure interne cette conception reflète une conception où tout accès aux ressources est basé sur un compte spécifique.

Catégorie de ressource de point final

Les terminaux de ressources d'Astra se répartissent en trois catégories :

- Cœur (`/core`)
- Application gérée (`/k8s`)
- Topologie (`/topology`)

Voir "Ressources" pour en savoir plus.

Version de catégorie

Chacune des trois catégories de ressources possède une version globale qui contrôle la version des ressources consultées. Par convention et définition, passage à une nouvelle version majeure d'une catégorie de ressources (par exemple, de `/v1` à `/v2`) Introduira des changements de rupture dans l'API.

Instance ou collection de ressources

Une combinaison de types de ressources et d'identificateurs peut être utilisée dans le chemin, selon qu'une instance de ressource ou une collection est accédée.

Exemple

- Chemin de ressource

  En fonction de la structure présentée ci-dessus, un chemin type vers un noeud final est : `/accounts/{account_id}/core/v1/users`.

- URL complète

  L'URL complète du noeud final correspondant est : `https://astra.netapp.io/accounts/{account_id}/core/v1/users`.

Ressources et terminaux

Vous pouvez accéder aux ressources fournies par l'API REST Astra Control pour automatiser le déploiement d'Astra. Chaque ressource est disponible via un ou plusieurs terminaux. Vous trouverez ci-dessous une présentation des ressources REST que vous pouvez utiliser dans le cadre d'un déploiement d'automatisation.

> Le format du chemin et de l'URL complète utilisés pour accéder aux ressources de contrôle Astra est basé sur plusieurs valeurs. Voir "Format d'URL" pour en savoir plus. Voir aussi "Référence API" Pour en savoir plus sur l'utilisation des ressources et des terminaux Astra,

Résumé des ressources REST d'Astra Control

Les principaux terminaux de ressources de l'API REST Astra Control sont organisés en trois catégories. Chaque ressource est accessible avec l'ensemble complet des opérations CRUD (création, lecture, mise à jour, suppression) sauf mention contraire.

La colonne **version** indique la version d'Astra lorsque la ressource a été introduite pour la première fois. Ce champ est en gras pour les ressources récemment ajoutées avec la version actuelle.

Ressources centrales

Les terminaux de ressources de base fournissent les services de base nécessaires pour établir et maintenir l'environnement d'exécution Astra.

Ressource	Relâchez	Description
Compte	21.12	Les ressources de compte vous permettent de gérer les locataires isolés dans l'environnement de déploiement Astra Control mutualisé.
ASUP	21.08	Les ressources ASUP représentent les packs AutoSupport transférés au support NetApp.
Certificat	**22.08**	Les ressources de certificat représentent les certificats installés utilisés pour une authentification forte pour les connexions sortantes.
Informations d'identification	21.04	Les ressources de identifiants NetApp contiennent des informations relatives à la sécurité qui peuvent être utilisées avec les utilisateurs Astra, les clusters, les compartiments et les systèmes back-end de stockage.
Droits	21.08	Les ressources d'abonnement représentent les fonctions et capacités disponibles pour un compte en fonction des licences et des abonnements actifs.
Événement	21.04	Les ressources d'événements représentent tous les événements se produisant dans le système, y compris le sous-ensemble classé comme notifications.
Crochet d'exécution	21.12	Les ressources de hook d'exécution représentent des scripts personnalisés que vous pouvez exécuter avant ou après l'exécution d'un snapshot d'une application gérée.

Ressource	Relâchez	Description
Fonction	21.08	Les ressources de fonctionnalités représentent les fonctions Astra sélectionnées que vous pouvez interroger pour déterminer si elles sont activées ou désactivées dans le système. L'accès est limité en lecture seule.
Groupe	**22.08**	Les ressources du groupe représentent les groupes Astra et les ressources associées. Seuls les groupes LDAP sont pris en charge dans la version actuelle.
Source du crochet	21.12	Les ressources de la source de hook représentent le code source réel utilisé avec un crochet d'exécution. La séparation du code source et du contrôle d'exécution présente plusieurs avantages, tels que la possibilité de partager les scripts.
Licence	21.08	Les ressources de licence représentent les licences disponibles pour un compte Astra.
Notification	21.04	Les ressources de notification représentent les événements Astra qui ont une destination de notification. L'accès est fourni par utilisateur.
Création de package	22.04	Les ressources du package fournissent l'enregistrement et l'accès aux définitions de package. Les logiciels sont constitués de divers composants, notamment des fichiers, des images et d'autres artefacts.
Liaison de rôles	21.04	Les ressources liées au rôle représentent les relations entre des paires spécifiques d'utilisateurs et de comptes. En plus du lien entre les deux, un ensemble d'autorisations est spécifié pour chaque à l'aide d'un rôle spécifique.
Réglage	21.08	Les ressources de définition représentent un ensemble de paires de clé-valeur qui décrivent une fonction pour un compte Astra spécifique.
Abonnement	21.08	Les ressources d'abonnement représentent les abonnements actifs pour un compte Astra.
Jeton	21.04	Les ressources de token représentent les jetons disponibles pour accéder par programmation à l'API REST Astra Control.
Notification non lue	21.04	Les ressources de notification non lues représentent les notifications affectées à un utilisateur spécifique, mais pas encore lues.
Mise à niveau	22.04	Les ressources de mise à niveau permettent d'accéder aux composants logiciels et de lancer des mises à niveau.
Utilisateur	21.04	Les ressources utilisateur représentent les utilisateurs d'Astra qui ont accès au système en fonction de leur rôle défini.

Ressources applicatives gérées

Les terminaux de ressources d'application gérée permettent d'accéder aux applications Kubernetes gérées.

Ressource	Relâchez	Description
Ressources applicatives	21.04	Les ressources d'application représentent les ensembles internes d'informations d'état nécessaires à la gestion des applications Astra.
Sauvegarde des applications	21.04	Les ressources de sauvegarde de l'application représentent les sauvegardes des applications gérées.

Ressource	Relâchez	Description
Snapshot de l’application	21.04	Les ressources de snapshot de l’application représentent les snapshots des applications gérées.
Remplacement du crochet d’exécution	21.12	Les ressources de remplacement du crochet d’exécution vous permettent de désactiver les crochets d’exécution NetApp par défaut préchargés pour des applications spécifiques, selon vos besoins.
Planification	21.04	Les ressources de planification correspondent aux opérations de protection des données planifiées pour les applications gérées dans le cadre d’une stratégie de protection des données.

Ressources de topologie

Les points de terminaison de ressource de topologie fournissent un accès aux applications non gérées et aux ressources de stockage.

Ressource	Relâchez	Description
Appli	21.04	Les ressources d’application représentent toutes les applications Kubernetes, y compris celles qui ne sont pas gérées par Astra.
AppMirror	**22.08**	Les ressources AppMirror représentent les ressources AppMirror pour la gestion des relations de mise en miroir des applications.
Godet	21.08	Les ressources de compartiment représentent les compartiments cloud S3 utilisés pour stocker les sauvegardes des applications gérées par Astra.
Le cloud	21.08	Les ressources cloud sont des clouds avec lesquels les clients Astra peuvent se connecter pour gérer les clusters et les applications.
Cluster	21.08	Les ressources en cluster représentent les clusters Kubernetes qui ne sont pas gérés par Kubernetes.
Nœud de cluster	21.12	Les ressources des nœuds de cluster apportent une résolution supplémentaire en vous permettant d’accéder aux nœuds individuels dans un cluster Kubernetes.
Cluster géré	21.08	Les ressources du cluster géré représentent les clusters Kubernetes actuellement gérés par Kubernetes.
Stockage back-end géré	21.12	Les ressources du système de stockage back-end géré vous permettent d’accéder aux représentations extraites des fournisseurs de stockage back-end. Ces systèmes de stockage back-end peuvent être utilisés par les clusters et les applications gérés.
Espace de noms	21.12	Les ressources d’espace de noms permettent d’accéder aux espaces de noms utilisés dans un cluster Kubernetes.
Système back-end	21.08	Les ressources de stockage back-end représentent des fournisseurs de services de stockage utilisables par les clusters et les applications gérés Astra.
Classe de stockage	21.08	Les ressources de classe de stockage représentent différents types ou classes de stockage détectés et disponibles pour un cluster géré spécifique.
Volumétrie	21.04	Les ressources de volume représentent les volumes de stockage Kubernetes associés aux applications gérées.

Ressources supplémentaires et terminaux

Vous pouvez utiliser plusieurs ressources et terminaux supplémentaires pour prendre en charge un déploiement Astra,

> Ces ressources et ces terminaux ne sont pas inclus dans la documentation de référence de l'API REST Astra Control.

OpenAPI

Les noeuds finaux OpenAPI donnent accès au document JSON OpenAPI actuel et à d'autres ressources associées.

OpenMetrics

Les noeuds finaux OpenMetrics fournissent un accès aux mesures du compte via la ressource OpenMetrics. Il est proposé avec le modèle de déploiement d'Astra Control Center.

Autres considérations d’utilisation

Sécurité RBAC

L’API REST Astra prend en charge le contrôle d’accès basé sur des rôles (RBAC) pour accorder et restreindre l’accès aux fonctions du système.

Rôles d’Astra

Chaque utilisateur Astra est affecté à un seul rôle qui détermine les actions qui peuvent être exécutées. Les rôles sont classés dans une hiérarchie comme décrit dans le tableau ci-dessous.

Rôle	Description
Propriétaire	Dispose de toutes les autorisations du rôle d’administrateur et peut également supprimer des comptes Astra.
Admin	Dispose de toutes les autorisations du rôle membre et peut également inviter des utilisateurs à rejoindre un compte.
Membre	Peut gérer entièrement l’application Astra et les ressources de calcul.
Visualiseur	Limité à l’affichage des ressources uniquement.

RBAC amélioré avec granularité de l’espace de noms

Cette fonctionnalité a été introduite avec la version 22.04 de l’API REST d’Astra.

Lorsqu’une liaison de rôle est établie pour un utilisateur spécifique, une contrainte peut être appliquée pour limiter les espaces de noms à lesquels l’utilisateur a accès. Il existe plusieurs façons de définir cette contrainte comme décrit dans le tableau ci-dessous. Voir le paramètre `roleContraints` Dans l’API de liaison de rôles pour plus d’informations.

Espaces de noms	Description
Tout	L’utilisateur peut accéder à tous les espaces de noms via le paramètre générique "*". Il s’agit de la valeur par défaut pour maintenir la compatibilité descendante.
Aucune	La liste des contraintes est spécifiée, bien qu’elle soit vide. Cela indique que l’utilisateur ne peut accéder à aucun espace de noms.
Liste d’espace de noms	L’UUID d’un namespace est inclus dans ce document qui limite l’utilisateur à un seul namespace. Une liste séparée par des virgules peut également être utilisée pour permettre l’accès à plusieurs espaces de noms.
Étiquette	Une étiquette est spécifiée et l’accès à tous les espaces de noms correspondants est autorisé.

Travailler avec les collections

L’API REST Astra Control offre plusieurs façons d’accéder aux collections de ressources via les paramètres de requête définis.

Sélection de valeurs

Vous pouvez spécifier les paires clé-valeur qui doivent être renvoyées pour chaque instance de ressource à l'aide de l' `include` paramètre. Toutes les instances sont renvoyées dans le corps de réponse.

Filtrage

Le filtrage des ressources de collection permet à un utilisateur API de spécifier des conditions qui déterminent si une ressource est renvoyée dans le corps de réponse. Le `filter` le paramètre est utilisé pour indiquer la condition de filtrage.

Tri

Le tri des ressources de collection permet à un utilisateur d'API de spécifier l'ordre dans lequel les ressources sont renvoyées dans le corps de réponse. Le `orderBy` le paramètre est utilisé pour indiquer la condition de filtrage.

Pagination

Vous pouvez appliquer la pagination en limitant le nombre d'instances de ressources renvoyées sur une demande à l'aide de l' `limit` paramètre.

Nombre

Si vous incluez le paramètre booléen `count` réglez sur `true`, le nombre de ressources du tableau renvoyé pour une réponse donnée est indiqué dans la section métadonnées.

Diagnostics et support

Il existe plusieurs fonctions de prise en charge disponibles avec l'API REST Astra Control qui peuvent être utilisées pour le diagnostic et le débogage.

Ressources API

Plusieurs fonctions Astra sont exposées via des ressources API qui fournissent des informations de diagnostic et une assistance.

Type	Description
Événement	Les activités du système enregistrées dans le cadre du traitement Astra.
Notification	Sous-ensemble des événements jugés suffisamment importants pour être présenté à l'utilisateur.
Notification non lue	Les notifications qui n'ont pas encore été lues ou récupérées par l'utilisateur.

Révoquer un jeton API

Vous pouvez révoquer un jeton API dans l'interface Web Astra lorsqu'il n'est plus nécessaire.

Avant de commencer

Vous avez besoin d'un compte Astra. Vous devez également identifier les jetons que vous souhaitez révoquer.

Description de la tâche

Une fois le token révoqué, il est immédiatement et définitivement inutilisable.

Étapes

1. Connectez-vous à Astra à l’aide de vos identifiants de compte.

   Accédez au site suivant pour le service Astra Control : "https://astra.netapp.io"

2. Cliquez sur l’icône figure en haut à droite de la page et sélectionnez **API Access**.
3. Sélectionnez le ou les jetons que vous souhaitez révoquer.
4. Dans la liste déroulante **actions**, cliquez sur **révoquer jetons**.

Workflows d’infrastructure

Avant de commencer

Vous pouvez utiliser ces workflows pour créer et entretenir l’infrastructure utilisée avec un déploiement d’Astra Control Center. Dans de nombreux cas, les flux de travail peuvent également être utilisés avec le service Astra Control.

Il est possible de développer et d’améliorer ces flux de travail par NetApp à tout moment, et nous vous recommandons de les consulter régulièrement.

Préparation générale

Avant d’utiliser l’un des flux de travail Astra, veillez à le lire "Préparez l’utilisation des workflows".

Catégories de flux de travail

Les workflows d’infrastructure sont organisés selon différentes catégories, pour faciliter la localisation.

Catégorie	Description
Identité et accès	Ces flux de travail vous permettent de gérer les identités et d’accéder à Astra. Les ressources comprennent les utilisateurs, les informations d’identification et les jetons.
Configuration LDAP	Vous pouvez configurer Astra Control Center pour qu’il utilise LDAP pour authentifier les utilisateurs sélectionnés.
Seaux	Vous pouvez utiliser ces workflows pour créer et gérer les compartiments S3 utilisés pour stocker les sauvegardes.
Stockage	Ces workflows permettent d’ajouter et de gérer des volumes et des systèmes back-end de stockage.
Clusters	Vous pouvez ajouter des clusters Kubernetes gérés, ce qui vous permet de protéger et de prendre en charge les applications qu’ils contiennent.

Identité et accès

Répertorier les utilisateurs

Vous pouvez lister les utilisateurs définis pour un compte Astra spécifique.

1. Dressez la liste des utilisateurs

Effectuez l’appel de l’API REST suivant.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/core/v1/users

Paramètres d’entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
inclure	Requête	Non	Sélectionner éventuellement les valeurs que vous souhaitez renvoyer dans la réponse.

Exemple Curl : renvoie toutes les données pour tous les utilisateurs

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/core/v1/users' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple Curl : renvoie le prénom, le nom et l’ID de tous les utilisateurs

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/core/v1/users?include=first
Name,lastName,id' --header 'Accept: */*' --header 'Authorization: Bearer
<API_TOKEN>'
```

Exemple de sortie JSON

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

Configuration LDAP

Préparation à la configuration LDAP

Vous pouvez intégrer Astra Control Center avec un serveur LDAP (Lightweight Directory Access Protocol) pour effectuer l'authentification pour certains utilisateurs Astra. LDAP est un protocole standard de l'industrie pour l'accès aux informations d'annuaires distribués et un choix populaire pour l'authentification d'entreprise.

Informations associées

- "Spécification technique LDAP feuille de route"
- "LDAP version 3"

Aperçu du processus de mise en œuvre

À un niveau élevé, il existe plusieurs étapes à suivre pour configurer un serveur LDAP afin de fournir une authentification aux utilisateurs d'Astra.

> Bien que les étapes présentées ci-dessous se trouvent dans une séquence, vous pouvez dans certains cas les exécuter dans un ordre différent. Par exemple, vous pouvez définir les utilisateurs et les groupes Astra avant de configurer le serveur LDAP.

1. Révision "Exigences et restrictions" connaître les options, les exigences et les limites.
2. Sélectionnez un serveur LDAP et les options de configuration souhaitées (y compris la sécurité).
3. Exécutez le flux de travail "Configurez Astra pour qu'il utilise un serveur LDAP" Pour intégrer Astra avec le serveur LDAP.
4. Vérifiez les utilisateurs et les groupes du serveur LDAP pour vous assurer qu'ils sont correctement définis.
5. Exécutez le flux de travail approprié dans "Ajoutez des entrées LDAP à Astra" Identifier les utilisateurs à authentifier à l'aide de LDAP.

Exigences et restrictions

Nous vous recommandons de consulter les éléments essentiels de configuration Astra présentés ci-dessous, y compris les limites et les options de configuration, avant de configurer Astra pour utiliser LDAP pour l'authentification.

Uniquement pris en charge avec Astra Control Center

La plateforme Astra Control propose deux modèles de déploiement. L'authentification LDAP est prise en charge uniquement avec les déploiements d'Astra Control Center.

Configuration de l'API REST uniquement

La version actuelle d'Astra Control Center prend uniquement en charge la configuration de l'authentification LDAP à l'aide de l'API REST Astra Control. Un aspect important de cette limitation est que les utilisateurs LDAP ne sont pas affichés dans l'onglet utilisateurs de l'interface Web Astra. Ils sont disponibles via l'API REST au niveau du terminal `../core/v1/users`.

Serveur LDAP requis

Vous devez disposer d'un serveur LDAP pour accepter et traiter les demandes d'authentification Astra. Microsoft Active Directory est pris en charge avec la version actuelle d'Astra Control Center.

Connexion sécurisée au serveur LDAP

Lors de la configuration du serveur LDAP dans Astra, vous pouvez éventuellement définir une connexion sécurisée. Dans ce cas, un certificat est nécessaire pour le protocole LDAPS.

Configurer des utilisateurs ou des groupes

Vous devez sélectionner les utilisateurs à authentifier à l'aide de LDAP. Pour ce faire, vous pouvez identifier les utilisateurs individuels ou un groupe d'utilisateurs. Les comptes doivent être définis au niveau du serveur LDAP. Elles doivent également être identifiées dans Astra (type LDAP) qui permet de transférer les demandes d'authentification vers LDAP.

Contrainte de rôle lors de la liaison d'un utilisateur ou d'un groupe

Avec la version actuelle d'Astra Control Center, la seule valeur prise en charge pour `roleConstraint` est "*". Cela indique que l'utilisateur n'est pas limité à un ensemble limité d'espaces de noms et qu'il peut y accéder tous. Voir "Ajoutez des entrées LDAP à Astra" pour en savoir plus.

Informations d'identification LDAP

Les informations d'identification utilisées par LDAP incluent le nom d'utilisateur (adresse e-mail) et le mot de passe associé.

Adresses e-mail uniques

Toutes les adresses e-mail agissant comme noms d'utilisateur dans un déploiement Astra Control Center doivent être uniques. Vous ne pouvez pas ajouter un utilisateur LDAP avec une adresse e-mail déjà définie pour Astra. Si un e-mail en double existe, vous devez d'abord le supprimer d'Astra. Voir "Supprimer des utilisateurs" Sur le site de documentation Astra Control Center pour plus d'informations.

Vous pouvez éventuellement définir d'abord les utilisateurs et les groupes LDAP

Vous pouvez ajouter les utilisateurs et groupes LDAP au Centre de contrôle Astra même s'ils n'existent pas encore dans LDAP ou si le serveur LDAP n'est pas configuré. Cela vous permet de préconfigurer les utilisateurs et les groupes avant de configurer le serveur LDAP.

Utilisateur défini dans plusieurs groupes LDAP

Si un utilisateur LDAP appartient à plusieurs groupes LDAP et que des rôles différents ont été attribués aux groupes dans Astra, le rôle effectif de l'utilisateur lors de l'authentification sera le plus privilégié. Par exemple, si un utilisateur est affecté à l' `viewer` rôle avec groupe1 mais a le `member` rôle dans le groupe 2, le rôle de l'utilisateur serait `member`. Ceci est basé sur la hiérarchie utilisée par Astra (la plus haute à la plus basse) :

- Propriétaire
- Admin
- Membre
- Visualiseur

Synchronisation périodique du compte

Astra synchronise ses utilisateurs et groupes avec le serveur LDAP environ toutes les 60 secondes. Par conséquent, si un utilisateur ou un groupe est ajouté à LDAP ou supprimé, il peut prendre jusqu'à une minute avant qu'il soit disponible dans Astra.

Désactivation et réinitialisation de la configuration LDAP

Avant de tenter de réinitialiser la configuration LDAP, vous devez d'abord désactiver l'authentification LDAP. De même, pour modifier le serveur LDAP (`connectionHost`), vous devez effectuer les deux opérations. Voir "Désactivez et réinitialisez LDAP" pour en savoir plus.

Paramètres de l’API REST

Les workflows de configuration LDAP appellent l’API REST pour accomplir des tâches spécifiques. Chaque appel API peut inclure des paramètres d’entrée comme indiqué dans les échantillons fournis. Voir "Référence API" pour plus d’informations sur la localisation de la documentation de référence.

Configurez Astra pour qu’il utilise un serveur LDAP

Vous devez sélectionner un serveur LDAP et configurer Astra pour qu’il utilise le serveur en tant que fournisseur d’authentification. La tâche de configuration comprend les étapes décrites ci-dessous. Chaque étape inclut un appel d’API REST unique.

1. Ajoutez un certificat CA

Effectuez l’appel d’API REST suivant pour ajouter un certificat d’autorité de certification à Astra.

> Cette étape est facultative et requise uniquement si vous voulez qu’Astra et le LDAP communiquent via un canal sécurisé à l’aide de LDAPS.

Méthode HTTP	Chemin
POST	/account/{account_id}/core/v1/certificats

Exemple d’entrée JSON

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

Notez les informations suivantes concernant les paramètres d’entrée :

- `cert` Est une chaîne JSON contenant un certificat au format PKCS-11 codé en base64 (codé PEM).
- `isSelfSigned` doit être réglé sur `true` si le certificat est auto-signé. La valeur par défaut est `false`.

Exemple de boucle

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

Exemple de réponse JSON


```
{
  "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": []
    }
}
```

2. Ajoutez les informations d'identification de liaison

Effectuez l'appel d'API REST suivant pour ajouter les informations d'identification de liaison.

Méthode HTTP	Chemin
POST	/account/{account_id}/core/v1/credentials

Exemple d'entrée JSON

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

Notez les informations suivantes concernant les paramètres d'entrée :

- `bindDn` et `password` Sont les informations d'identification de liaison codées en base64 de l'utilisateur admin LDAP qui peut se connecter et rechercher dans le répertoire LDAP. `bindDn` Est l'adresse e-mail de l'utilisateur LDAP.

Exemple de boucle

```
curl --location -i --request POST --data @JSONinput
'https://astra.example.com/accounts/<ACCOUNT_ID>/core/v1/credentials'
--header 'Content-Type: application/astra-credential+json' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple de réponse JSON

```
{
    "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"
    }
}
```

Noter les paramètres de réponse suivants :

- Le `id` des informations d'identification sont utilisées dans les étapes suivantes du flux de travail.

3. Récupérez l'UUID du paramètre LDAP

Exécutez l'appel de l'API REST suivant pour récupérer l'UUID du `astra.account.ldap` Réglage inclus avec le centre de contrôle Astra.

> L'exemple curl ci-dessous utilise un paramètre de requête pour filtrer la collection de paramètres. Vous pouvez à la place supprimer le filtre pour obtenir tous les paramètres, puis rechercher `astra.account.ldap`.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/core/v1/settings

Exemple de boucle

```
curl --location -i --request GET
'https://astra.example.com/accounts/<ACCOUNT_ID>/core/v1/settings?filter=n
ame%20eq%20'astra.account.ldap'&include=name,id' --header 'Accept: */*'
--header 'Authorization: Bearer <API_TOKEN>'
```

Exemple de réponse JSON

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

4. Mettez à jour le paramètre LDAP

Effectuez l'appel d'API REST suivant pour mettre à jour le paramètre LDAP et terminer la configuration. Utilisez le `id` Valeur de l'appel API précédent pour le `<SETTING_ID>` Valeur dans le chemin d'accès à l'URL ci-dessous.

> Vous pouvez d'abord lancer une demande GET pour le paramètre spécifique afin de voir le schéma de configuration. Ceci fournira plus d'informations sur les champs requis dans la configuration.

Méthode HTTP	Chemin
EN	/account/{account_id}/core/v1/settings/{setting_id}

Exemple d'entrée JSON

```
{
  "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"
    }
}
```

Notez les informations suivantes concernant les paramètres d'entrée :

- `isEnabled` doit être réglé sur `true` ou une erreur peut se produire.

* `credentialId` est l'id des informations d'identification de liaison créées précédemment.
* `secureMode` doit être réglé sur `LDAP` ou `LDAPS` en fonction de votre configuration à l'étape précédente.
* Seul Active Directory est pris en charge en tant que fournisseur.

Exemple de boucle

```
curl --location -i --request PUT --data @JSONinput
'https://astra.example.com/accounts/<ACCOUNT_ID>/core/v1/settings/<SETTING
_ID>' --header 'Content-Type: application/astra-setting+json' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Si l'appel a réussi, la réponse HTTP 204 est renvoyée.

5. Récupérez le paramètre LDAP

Vous pouvez éventuellement effectuer l'appel d'API REST suivant pour récupérer les paramètres LDAP et confirmer la mise à jour.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/core/v1/settings/{setting_id}

Exemple de boucle

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

Exemple de réponse JSON

```
{
  "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": {}
}
```

Localisez le `state` champ de la réponse qui contient l'une des valeurs du tableau ci-dessous.

État	Description
en attente	Le processus de configuration est toujours actif et n'est pas encore terminé.
valide	La configuration a été effectuée avec succès et `currentConfig` la réponse correspond `desiredConfig`.
erreur	Le processus de configuration LDAP a échoué.

Ajoutez des entrées LDAP à Astra

Après avoir configuré LDAP en tant que fournisseur d'authentification pour Astra Control Center, vous pouvez sélectionner les utilisateurs LDAP qu'Astra authentifie à l'aide des informations d'identification LDAP. Chaque utilisateur doit avoir un rôle dans Astra avant d'avoir accès à Astra via l'API REST Astra Control.

Il existe deux façons de configurer Astra pour affecter des rôles. Choisissez celle qui convient le mieux à votre environnement.

- "Ajouter et lier un utilisateur individuel"
- "Ajouter et lier un groupe"

ⓘ Les informations d'identification LDAP se présentent sous la forme d'un nom d'utilisateur sous la forme d'une adresse e-mail et du mot de passe LDAP associé.

Ajouter et lier un utilisateur individuel

Vous pouvez attribuer un rôle à chaque utilisateur Astra qui est utilisé après l'authentification LDAP. Ceci est approprié lorsqu'il y a un petit nombre d'utilisateurs et que chacun peut avoir des caractéristiques administratives différentes.

1. Ajoutez un utilisateur

Effectuez l'appel d'API REST suivant pour ajouter un utilisateur à Astra et indiquer que LDAP est le fournisseur d'authentification.

Méthode HTTP	Chemin
POST	/account/{account_id}/core/v1/users

Exemple d'entrée JSON

```
{
  "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"
}
```

Notez les informations suivantes concernant les paramètres d'entrée :

- Les paramètres suivants sont requis :
  - `authProvider`
  - `authID`
  - `email`
- `authID` Est le nom distinctif (DN) de l'utilisateur dans LDAP
- `email` Doit être unique pour tous les utilisateurs définis dans Astra

Si le `email` La valeur n'est pas unique, une erreur se produit et un code d'état HTTP 409 est renvoyé dans la réponse.

Exemple de boucle

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

Exemple de réponse JSON

```
{
  "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": ""
}
```


2. Ajoutez une liaison de rôle pour l'utilisateur

Effectuez l'appel de l'API REST suivant pour lier l'utilisateur à un rôle spécifique. Vous devez créer l'UUID de l'utilisateur à l'étape précédente.

Méthode HTTP	Chemin
POST	/Account/{account_ID}/core/v1/roleliaisons

Exemple d'entrée JSON

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

Notez les informations suivantes concernant les paramètres d'entrée :

* Valeur utilisée ci-dessus pour `roleConstraint` Est la seule option disponible pour la version actuelle d'Astra. Il indique que l'utilisateur n'est pas limité à un ensemble limité d'espaces de noms et peut y accéder tous.

Exemple de boucle

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

Exemple de réponse JSON

```
{
  "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": ["*"]
}
```

Noter les éléments suivants concernant les paramètres de réponse :

- La valeur `user` pour le `principalType` champ indique que la liaison du rôle a été ajoutée pour un utilisateur (et non pour un groupe).

Ajouter et lier un groupe

Vous pouvez affecter un rôle à un groupe Astra qui est utilisé après l'authentification LDAP. Ceci est approprié lorsqu'il y a un grand nombre d'utilisateurs et que chacun peut avoir des caractéristiques administratives similaires.

1. Ajoutez un groupe

Effectuez l'appel d'API REST suivant pour ajouter un groupe à Astra et indiquer que LDAP est le fournisseur d'authentification.

Méthode HTTP	Chemin
POST	/account/{account_id}/core/v1/groupes

Exemple d'entrée JSON

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

Notez les informations suivantes concernant les paramètres d'entrée :

- Les paramètres suivants sont requis :
  - `authProvider`
  - `authID`

Exemple de boucle

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

Exemple de réponse JSON

```
{
  "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": []
  }
}
```

2. Ajoutez une liaison de rôle pour le groupe

Effectuez l'appel d'API REST suivant pour lier le groupe à un rôle spécifique. Vous devez créer l'UUID du groupe à l'étape précédente. Les utilisateurs qui sont membres du groupe pourront se connecter à Astra une fois que LDAP aura effectué l'authentification.

Méthode HTTP	Chemin
POST	/Account/{account_ID}/core/v1/roleliaisons

Exemple d'entrée JSON

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

Notez les informations suivantes concernant les paramètres d'entrée :

- Valeur utilisée ci-dessus pour `roleConstraint` Est la seule option disponible pour la version actuelle d'Astra. Il indique que l'utilisateur n'est pas limité à certains espaces de noms et peut y accéder tous.

Exemple de boucle

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

Exemple de réponse JSON


```
{
  "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": ["*"]
}
```


Noter les éléments suivants concernant les paramètres de réponse :

* La valeur `group` pour le `principalType` champ indique que la liaison de rôle a été ajoutée pour un groupe (et non pour un utilisateur).

Désactivez et réinitialisez LDAP

Il existe deux tâches administratives facultatives, bien que liées, que vous pouvez effectuer selon vos besoins pour un déploiement de centre de contrôle Astra. Vous pouvez désactiver globalement l'authentification LDAP et réinitialiser la configuration LDAP.

Les deux tâches de flux de travail requièrent l'ID pour le `astra.account.ldap` Réglage Astra. Les détails sur la récupération de l'ID de paramètre sont inclus dans **configurer le serveur LDAP**. Voir "Récupère l'UUID du paramètre LDAP" pour en savoir plus.

* "Désactivez l'authentification LDAP"
* "Réinitialisez la configuration de l'authentification LDAP"

Désactivez l'authentification LDAP

Vous pouvez effectuer l'appel d'API REST suivant pour désactiver globalement l'authentification LDAP pour un déploiement Astra spécifique. L'appel met à jour le `astra.account.ldap` réglage et `isEnabled` la valeur est définie sur `false`.

Méthode HTTP	Chemin
EN	/account/{account_id}/core/v1/settings/{setting_id}

Exemple d'entrée JSON

```
{
  "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"
    }
}
```

```
curl --location -i --request PUT --data @JSONinput
'https://astra.example.com/accounts/<ACCOUNT_ID>/core/v1/settings/<SETTING
_ID>' --header 'Content-Type: application/astra-setting+json' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Si l'appel a réussi, le `HTTP 204` la réponse est renvoyée. Vous pouvez éventuellement récupérer à nouveau les paramètres de configuration pour confirmer la modification.

Réinitialisez la configuration de l'authentification LDAP

Vous pouvez effectuer l'appel d'API REST suivant pour déconnecter Astra du serveur LDAP et réinitialiser la configuration LDAP dans Astra. L'appel met à jour le `astra.account.ldap` réglage et valeur de `connectionHost` est effacé.

La valeur de `isEnabled` doit également être défini sur `false`. Vous pouvez soit définir cette valeur avant d'effectuer l'appel de réinitialisation, soit dans le cadre de l'appel de réinitialisation. Dans le deuxième cas, `connectionHost` doit être effacé et `isEnabled` défini sur false lors du même appel de réinitialisation.

Il s’agit d’une opération perturbatrice et vous devez procéder avec précaution. Elle supprime tous les utilisateurs et groupes LDAP importés. Il supprime également tous les utilisateurs, groupes et liaisons Astra (type LDAP) associés que vous avez créés dans Astra Control Center.

Méthode HTTP	Chemin
EN	/account/{account_id}/core/v1/settings/{setting_id}

Exemple d’entrée JSON

```
{
  "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"
    }
}
```

Notez ce qui suit :

- Pour modifier le serveur LDAP, vous devez à la fois désactiver et réinitialiser la modification LDAP `connectHost` à une valeur nulle comme indiqué dans l’exemple ci-dessus.

```
curl --location -i --request PUT --data @JSONinput
'https://astra.example.com/accounts/<ACCOUNT_ID>/core/v1/settings/<SETTING
_ID>' --header 'Content-Type: application/astra-setting+json' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Si l’appel a réussi, le `HTTP 204` la réponse est renvoyée. Vous pouvez éventuellement récupérer à nouveau la configuration pour confirmer la modification.

Clusters

Lister les clusters

Vous pouvez afficher la liste des clusters disponibles dans un cloud spécifique.

1. Sélectionnez le nuage

Exécutez le flux de travail "Lister les clouds" et sélectionnez le cloud contenant les clusters.

2. Dressez la liste des clusters

Effectuez l’appel d’API REST suivant pour répertorier les clusters dans un cloud spécifique.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/topologique/v1/nuages/{cloud_id}/clusters

Exemple Curl : renvoie toutes les données de tous les clusters

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/clouds/<CLOUD_I
D>/clusters' --header 'Accept: */*' --header 'Authorization: Bearer
<API_TOKEN>'
```

Exemple de sortie JSON


```
{
    "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"
              }
          }
      ]
  }
```


Répertorier les clusters gérés

Vous pouvez lister les clusters Kubernetes actuellement gérés par Astra.

1. Dressez la liste des clusters gérés

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
OBTENEZ	/Account/{account_ID}/topologique/v1/managedclusters

Exemple Curl : renvoie toutes les données de tous les clusters

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/managedClusters
' --header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Cloud

Lister les clouds

Vous pouvez répertorier les nuages définis et disponibles un compte Astra spécifique.

1. Dresser la liste des clouds

Effectuez l'appel d'API REST suivant pour répertorier les clouds.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/topologique/v1/clouds

Exemple de curl : retournez toutes les données de tous les clouds

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/clouds'
--header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Seaux

Répertorier les rubriques

Vous pouvez afficher la liste des compartiments S3 définis pour un compte Astra spécifique.

1. Dressez la liste des compartiments

Effectuez l'appel suivant de l'API REST pour afficher la liste des compartiments.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/topologique/v1/seaux

Exemple de gondolage : renvoie toutes les données de tous les compartiments

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/buckets'
--header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Stockage

Répertorier les classes de stockage

Vous pouvez afficher la liste des classes de stockage disponibles.

1. Sélectionnez le nuage

Exécutez le flux de travail "Lister les clouds" et choisissez le cloud dans lequel vous travaillez.

2. Sélectionnez le cluster

Exécutez le flux de travail "Lister les clusters" et sélectionnez le cluster.

3. Énumérez les classes de stockage d'un cluster spécifique

Effectuez l'appel d'API REST suivant pour répertorier les classes de stockage d'un cluster et d'un cloud spécifiques.

Méthode HTTP	Chemin
OBTENEZ	/Account/{account_ID}/topologique/v1/nuages/<CLOUD_ID>/clusters/<CLUSTER_ID>/storageclasses

Exemple Curl : renvoie toutes les données de toutes les classes de stockage

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/clouds/<CLOUD_I
D>/clusters/<CLUSTER_ID>/storageClasses' --header 'Accept: */*' --header
'Authorization: Bearer <API_TOKEN>'
```

Exemple de sortie JSON


```
{
    "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": []
            }
        }
    ]
}
```


Liste des systèmes back-end

Vous pouvez lister les systèmes back-end disponibles.

1. Dressez la liste des systèmes back-end

Effectuez l’appel de l’API REST suivant.

Méthode HTTP	Chemin
OBTENEZ	/Account/{account_ID}/topologique/v1/storageBackends

Exemple de curl : renvoie toutes les données pour tous les systèmes back-end

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/storageBackends
' --header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple de sortie JSON

```
{
  "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"
    }
  ]
}
```

Flux de travail de gestion

Avant de commencer

Vous pouvez utiliser ces flux de travail dans le cadre de l'administration des applications dans un cluster géré Astra.

Il est possible de développer et d'améliorer ces flux de travail par NetApp à tout moment, et nous vous recommandons de les consulter régulièrement.

Préparation générale

Avant d'utiliser l'un des flux de travail Astra, veillez à le lire "Préparez l'utilisation des workflows".

Catégories de flux de travail

Les flux de travail de gestion sont organisés en différentes catégories, afin de localiser plus facilement celui que vous souhaitez.

Catégorie	Description
Contrôle des applications	Ces flux de production vous permettent de contrôler les applications gérées et non gérées. Vous pouvez afficher la liste des applications ainsi que créer et supprimer une application gérée.
Protection des applications	Ces flux de travail vous permettent de protéger vos applications gérées par le biais des snapshots et des sauvegardes.
Clonage et restauration des applications	Décrit le clonage et la restauration des applications gérées dans ce workflow.
Assistance	Plusieurs workflows sont disponibles pour débogage et prise en charge de vos applications, ainsi que pour l'environnement Kubernetes général.

Autres considérations

L'utilisation des flux de travail de gestion peut prendre en compte plusieurs considérations supplémentaires.

Clonage d'une application

Vous devez tenir compte de plusieurs facteurs lors du clonage d'une application. Les paramètres décrits ci-dessous font partie de l'entrée JSON.

Identificateur de cluster source

La valeur de `sourceClusterID` identifie toujours le cluster sur lequel l'application d'origine est installée.

Identificateur de cluster

La valeur de `clusterID` identifie le cluster sur lequel la nouvelle application sera installée.

- Lors du clonage au sein d'un même cluster, `clusterID` et `sourceClusterID` avoir la même valeur.
- Lors du clonage entre clusters, les deux valeurs sont différentes et `clusterID` Doit être l'ID du cluster

cible.

Espaces de noms

Le `namespace` la valeur doit être différente de celle de l'application source d'origine. De plus, le namespace du clone ne peut pas exister et Astra va le créer.

Sauvegardes et snapshots

Vous pouvez également cloner une application à partir d'une sauvegarde ou d'un snapshot existant à l'aide de `backupID` ou `snapshotID` paramètres. Si vous ne fournissez pas de sauvegarde ou de snapshot, Astra crée d'abord une sauvegarde de l'application, puis clone à partir de la sauvegarde.

Restauration d'une application

Voici quelques points à prendre en compte lors de la restauration d'une application.

- La restauration d'une application est très similaire à l'opération de clonage.
- Lors de la restauration d'une application, vous devez fournir une sauvegarde ou un instantané.

Contrôle des applications

Répertorier les applications

Vous pouvez lister les applications actuellement gérées par Astra. Pour rechercher des snapshots ou des sauvegardes d'une application spécifique, vous pouvez effectuer cette opération.

1. Dressez la liste des applications

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/k8s/v2/apps

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
inclure	Requête	Non	Sélectionner éventuellement les valeurs que vous souhaitez renvoyer dans la réponse.

Exemple de curl : renvoie toutes les données de toutes les applications

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple de curl : renvoie le nom, l'ID et l'état de toutes les applications

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps?include=name,id
,state' --header 'Accept: */*' --header 'Authorization: Bearer
<API_TOKEN>'
```

Exemple de sortie JSON


```
{
    "items": [
        [
            "mysql",
            "4ee2b8fa-3696-4f32-8879-399792f477c3",
            "ready"
        ],
        [
            "postgresql",
            "3b984474-e5c9-4b64-97ee-cdeb9bcd212e",
            "ready"
        ],
    ],
    "metadata": {}
}
```


Obtenir une application

Vous pouvez récupérer toutes les variables de ressource décrivant une seule application.

Avant de commencer

Vous devez avoir l'ID de l'application que vous souhaitez récupérer. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l'application.

1. Obtenez l'application

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/k8s/v2/apps/{app_id}

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l'application	Chemin	Oui.	Valeur ID de l'application à récupérer.

Exemple de curl : renvoie toutes les données de l'application

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps/<APP_ID>'
--header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Gérer une application

Vous pouvez créer une application gérée basée sur une application déjà connue d'Astra dans un espace de noms spécifique. Lorsqu'une application est gérée ou définie par Astra, vous pouvez la protéger en effectuant des sauvegardes et des copies Snapshot.

1. Sélectionnez l'espace de noms

Exécutez le flux de travail "Lister les espaces de noms" et sélectionnez l'espace de noms.

2. Sélectionnez le cluster

Exécutez le flux de travail "Lister les clusters" et sélectionnez le cluster.

3. Gérer l'application

Effectuez l'appel suivant de l'API REST pour gérer l'application.

Méthode HTTP	Chemin
POST	/account/{account_id}/k8s/v2/apps

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
JSON	Corps	Oui.	Fournit les paramètres nécessaires à l'identification de l'application à gérer. Voir l'exemple ci-dessous.

Exemple d'entrée JSON

```
{
  "clusterID": "7ce83fba-6aa1-4e0c-a194-26e714f5eb46",
  "name": "subtext",
  "namespaceScopedResources": [{"namespace": "kube-matrix"}],
  "type": "application/astra-app",
  "version": "2.0"
}
```

Exemple de curl : gérer une application

```
curl --location -i --request POST
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps' --header
'Content-Type: application/astra-app+json' --header 'Accept: */*' --header
'Authorization: Bearer <API_TOKEN>'  --data @JSONinput
```

Annuler la gestion d’une application

Vous pouvez supprimer une application gérée lorsqu’elle n’est plus nécessaire. La suppression d’une application gérée supprime également les planifications associées.

Avant de commencer

Vous devez avoir l’ID de l’application que vous souhaitez annuler la gestion. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l’application.

Les sauvegardes et snapshots de l’application ne sont pas automatiquement supprimés lorsqu’ils sont supprimés. Si vous n’avez plus besoin des sauvegardes et des snapshots, vous devez les supprimer avant de supprimer l’application.

1. Non géré de l’application

Effectuez l’appel suivant de l’API REST pour supprimer l’application.

Méthode HTTP	Chemin
SUPPRIMER	/account/{account_id}/k8s/v2/apps/{app_id}

Paramètres d’entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l’application	Chemin	Oui.	Identifie l’application à supprimer.

Exemple de curl : supprimez une application gérée

```
curl --location -i --request DELETE
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps/<APP_ID>'
--header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Protection des applications

Répertorier les snapshots

Vous pouvez afficher la liste des instantanés pris pour une application spécifique.

Avant de commencer

Vous devez disposer de l'ID de l'application pour laquelle vous souhaitez répertorier les instantanés. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l'application.

1. Dressez la liste des instantanés

Effectuez l'appel suivant de l'API REST pour afficher la liste des snapshots.

Méthode HTTP	Chemin
OBTENEZ	/Accounts/{account_ID}/k8s/v1/apps/{app_ID}/appSnaps

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l'application	Chemin	Oui.	Identifie l'application propriétaire des snapshots répertoriés.
nombre	Requête	Non	Si `count=true` le nombre de snapshots est inclus dans la section métadonnées de la réponse.

Exemple de curl : renvoie tous les snapshots de l'application

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appSna
ps' --header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple de boucle : renvoie tous les snapshots de l'application et du nombre

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appSna
ps?count=true' --header 'Accept: */*' --header 'Authorization: Bearer
<API_TOKEN>'
```

Exemple de sortie JSON


```
{
    "items": [
        {
            "type": "application/astra-appSnap",
            "version": "1.1",
            "id": "1ce34da4-bb0a-4926-b925-4a5d85dda8c2",
            "hookState": "success",
            "metadata": {
                "createdBy": "a530e865-23e8-4e2e-8020-e92c419a3867",
                "creationTimestamp": "2022-10-30T22:44:20Z",
                "modificationTimestamp": "2022-10-30T22:44:20Z",
                "labels": []
            },
            "snapshotAppAsset": "0ebfe3f8-40ed-4bdc-88c4-2144fbda85a0",
            "snapshotCreationTimestamp": "2022-10-30T22:44:33Z",
            "name": "snapshot-david-1",
            "state": "completed",
            "stateUnready": []
        }
    ],
    "metadata": {}
}
```


Répertoriez les sauvegardes

Vous pouvez afficher la liste des sauvegardes créées pour une application spécifique.

Avant de commencer

Vous devez disposer de l'ID de l'application pour laquelle vous souhaitez répertorier les sauvegardes. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l'application.

1. Dressez la liste des sauvegardes

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
OBTENEZ	/Accounts/{account_ID}/k8s/v1/apps/{app_ID}/appBackups

Paramètres d’entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l’application	Chemin	Oui.	Identifie l’application gérée propriétaire des sauvegardes répertoriées.

Exemple Curl : renvoie toutes les sauvegardes de l’application

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appBac
kups' --header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple de sortie JSON

```
{
    "items": [
        {
            "type": "application/astra-appBackup",
            "version": "1.1",
            "id": "8edeb4a4-fd8b-4222-a559-1013145b28fc",
            "name": "backup-david-oct28-1",
            "bucketID": "a443e58f-59bd-4d45-835a-1bc7813f659a",
            "snapshotID": "dfe237cb-57b7-4576-af4d-00ba3a8f2828",
            "state": "completed",
            "stateUnready": [],
            "hookState": "success",
            "totalBytes": 205219132,
            "bytesDone": 205219132,
            "percentDone": 100,
            "metadata": {
                "labels": [
                    {
                        "name": "astra.netapp.io/labels/read-
only/triggerType",
                        "value": "backup"
                    }
                ],
                "creationTimestamp": "2022-10-28T21:58:37Z",
                "modificationTimestamp": "2022-10-28T21:58:55Z",
                "createdBy": "a530e865-23e8-4e2e-8020-e92c419a3867"
            }
        }
    ],
    "metadata": {}
}
```


Créer un instantané pour une application

Vous pouvez créer un instantané pour une application spécifique.

Avant de commencer

Vous devez avoir l'ID de l'application pour laquelle vous souhaitez créer un snapshot. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l'application.

1. Créer un snapshot

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
POST	/Accounts/{account_ID}/k8s/v1/apps/{app_ID}/appSnaps

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l'application	Chemin	Oui.	Identifie l'application gérée où le snapshot sera créé.
JSON	Corps	Oui.	Fournit les paramètres de l'instantané. Voir l'exemple ci-dessous.

Exemple d'entrée JSON

```json
{
  "type": "application/astra-appSnap",
  "version": "1.1",
  "name": "snapshot-david-1"
}
```

Exemple de curl : créez un snapshot pour l'application

```
curl --location -i --request POST
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appSna
ps' --header 'Content-Type: application/astra-appSnap+json' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>' --data
@JSONinput
```

Créez une sauvegarde pour une application

Vous pouvez créer une sauvegarde pour une application spécifique, puis utiliser la sauvegarde pour restaurer ou cloner l'application.

Avant de commencer

Vous devez avoir l'ID de l'application que vous souhaitez sauvegarder. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l'application.

1. Créez une sauvegarde

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
POST	/Accounts/{account_ID}/k8s/v1/apps/{app_ID}/appBackups

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l'application	Chemin	Oui.	Identifie l'application où la sauvegarde sera créée.
JSON	Corps	Oui.	Fournit les paramètres de la sauvegarde. Voir l'exemple ci-dessous.

Exemple d'entrée JSON

```
{
  "type": "application/astra-appBackup",
  "version": "1.1",
  "name": "backup-david-1"
}
```

Exemple Curl : créez une sauvegarde pour l'application

```
curl --location -i --request POST
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appBac
kups' --header 'Content-Type: application/astra-appBackup+json' --header
'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>' --data
@JSONinput
```

Supprime un snapshot

Vous pouvez supprimer un snapshot associé à une application.

Avant de commencer

Vous devez disposer des éléments suivants :

- ID de l'application propriétaire de l'instantané. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l'application.
- ID du snapshot à supprimer. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les snapshots" pour localiser l'instantané.

1. Supprimez le snapshot

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
SUPPRIMER	/Accounts/{account_ID}/k8s/v1/apps/{app_ID}/appSnaps/{appSnap_ID}

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l'application	Chemin	Oui.	Identifie l'application gérée propriétaire du snapshot.
id de snapshot	Chemin	Oui.	Identifie le snapshot à supprimer.

Exemple de curl : supprimez un seul snapshot pour l'application

```
curl --location -i --request DELETE
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appSna
ps/<SNAPSHOT_ID>' --header 'Accept: */*' --header 'Authorization: Bearer
<API_TOKEN>'
```

Supprimer une sauvegarde

Vous pouvez supprimer une sauvegarde associée à une application.

Avant de commencer

Vous devez disposer des éléments suivants :

- ID de l'application propriétaire de la sauvegarde. Si nécessaire, vous pouvez utiliser le workflow "Répertorier les applications" pour localiser l'application.
- ID de la sauvegarde à supprimer. Si nécessaire, vous pouvez utiliser le workflow "Répertoriez les sauvegardes" pour localiser l'instantané.

1. Supprimez la sauvegarde

Effectuez l'appel de l'API REST suivant.

> ⓘ Vous pouvez forcer la suppression d'une sauvegarde ayant échoué à l'aide de l'en-tête de demande facultatif comme décrit ci-dessous.

Méthode HTTP	Chemin
SUPPRIMER	/Accounts/{account_ID}/k8s/v1/apps/{app_ID}/appBackups/{appBackup_ID}

Paramètres d’entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
id de l’application	Chemin	Oui.	Identifie l’application gérée propriétaire de la sauvegarde.
id de sauvegarde	Chemin	Oui.	Identifie la sauvegarde à supprimer.
forcer la suppression	En-tête	Non	Utilisé pour forcer la suppression d’une sauvegarde ayant échoué.

Exemple de curl : supprimez une sauvegarde unique pour l’application

```
curl --location -i --request DELETE
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appBac
kups/<BACKUP_ID>' --header 'Accept: */*' --header 'Authorization: Bearer
<API_TOKEN>'
```

Exemple de curl : supprimez une sauvegarde unique pour l’application avec l’option forcer

```
curl --location -i --request DELETE
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v1/apps/<APP_ID>/appBac
kups/<BACKUP_ID>' --header 'Accept: */*' --header 'Authorization: Bearer
<API_TOKEN>' --header 'Force-Delete: true'
```

Clonage et restauration d’une application

Cloner une application

Vous pouvez créer une application en clonant une application existante.

Avant de commencer

Notez les éléments suivants concernant ce flux de travail :

- Aucune sauvegarde d’application ou snapshot n’est utilisée
- L’opération de clonage est effectuée au sein du même cluster
- La nouvelle application est placée dans un espace de noms différent

Pour cloner une application vers un autre cluster, vous devez mettre à jour le `clusterId` Paramètre JSON dans l’entrée correspondant à votre environnement.

1. Sélectionnez l’application à cloner

Exécutez le flux de travail "Répertorier les applications" et sélectionnez l’application à cloner. Plusieurs des valeurs de ressource sont nécessaires pour l’appel REST utilisé pour cloner l’application.

2. Clonez l’application

Effectuez l’appel suivant de l’API REST pour cloner l’application.

Méthode HTTP	Chemin
POST	/account/{account_id}/k8s/v2/apps

Paramètres d’entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
JSON	Corps	Oui.	Fournit les paramètres de l’application clonée. Voir l’exemple ci-dessous.

Exemple d’entrée JSON

```
{
  "type": "application/astra-app",
  "version": "2.0",
  "name": "mysql-clone",
  "clusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "sourceClusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "namespace": "mysql-ns",
  "sourceAppID": "e591ee59-ea90-4a9f-8e6c-d2b6e8647096"
}
```

Exemple de curl : clonez une application

```
curl --location -i --request POST
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps' --header
'Content-Type: application/astra-app+json' --header '*/*' --header
'Authorization: Bearer <API_TOKEN>' --data @JSONinput
```

Cloner une application à partir d’un snapshot

Vous pouvez créer une nouvelle application en la clonant à partir d’un snapshot.

Avant de commencer

Notez les éléments suivants concernant ce flux de travail :

- Un snapshot d'application est utilisé
- L'opération de clonage est effectuée au sein du même cluster

Pour cloner une application vers un autre cluster, vous devez mettre à jour le `clusterId` Paramètre JSON dans l'entrée correspondant à votre environnement.

1. Sélectionnez l'application à cloner

Exécutez le flux de travail "Répertorier les applications" et sélectionnez l'application à cloner. Plusieurs des valeurs de ressource sont nécessaires pour l'appel REST utilisé pour cloner l'application.

2. Sélectionnez le snapshot à utiliser

Exécutez le flux de travail "Répertorier les snapshots" et sélectionnez le snapshot que vous souhaitez utiliser.

3. Clonez l'application

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
POST	/account/{account_id}/k8s/v2/apps

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
JSON	Corps	Oui.	Fournit les paramètres de l'application clonée. Voir l'exemple ci-dessous.

Exemple d'entrée JSON

```
{
  "type": "application/astra-app",
  "version": "2.0",
  "name": "mysql-clone2",
  "clusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "sourceClusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "namespace": "mysql",
  "snapshotID": "e24515bd-a28e-4b28-b832-f3c74dbf32fb"
}
```

Exemple de curl : cloner une application à partir d'un snapshot

```
curl --location -i --request POST
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps' --header
'Content-Type: application/astra-app+json' --header '*/*' --header
'Authorization: Bearer <API_TOKEN>' --data @JSONinput
```

Cloner une application à partir d'une sauvegarde

Vous pouvez créer une nouvelle application en la clonant à partir d'une sauvegarde.

Avant de commencer

Notez les éléments suivants concernant ce flux de travail :

- Une sauvegarde d'application est utilisée
- L'opération de clonage est effectuée au sein du même cluster

Pour cloner une application vers un autre cluster, vous devez mettre à jour le `clusterId` Paramètre JSON dans l'entrée correspondant à votre environnement.

1. Sélectionnez l'application à cloner

Exécutez le flux de travail "Répertorier les applications" et sélectionnez l'application à cloner. Plusieurs des valeurs de ressource sont nécessaires pour l'appel REST utilisé pour cloner l'application.

2. Sélectionnez la sauvegarde à utiliser

Exécutez le flux de travail "Répertoriez les sauvegardes" et sélectionnez la sauvegarde que vous souhaitez utiliser.

3. Clonez l'application

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
POST	/account/{account_id}/k8s/v2/qpps

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
JSON	Corps	Oui.	Fournit les paramètres de l'application clonée. Voir l'exemple ci-dessous.

Exemple d’entrée JSON

```
{
  "type": "application/astra-app",
  "version": "2.0",
  "name": "mysql-clone3",
  "clusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "sourceClusterID": "30880586-d579-4d27-930f-a9633e59173b",
  "namespace": "mysql",
  "backupID": "e24515bd-a28e-4b28-b832-f3c74dbf32fb"
}
```

Exemple Curl : cloner une application à partir d’une sauvegarde

```
curl --location -i --request POST
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps' --header
'Content-Type: application/astra-app+json' --header '*/*' --header
'Authorization: Bearer <API_TOKEN>' --data @JSONinput
```

Restaurez une application à partir d’une sauvegarde

Vous pouvez restaurer une application en créant une nouvelle application à partir d’une sauvegarde.

1. Sélectionnez l’application à restaurer

Exécutez le flux de travail "Répertorier les applications" et sélectionnez l’application à cloner. Plusieurs des valeurs de ressources sont nécessaires pour l’appel DE REPOS utilisé pour restaurer l’application.

2. Sélectionnez la sauvegarde à utiliser

Exécutez le flux de travail "Répertoriez les sauvegardes" et sélectionnez la sauvegarde que vous souhaitez utiliser.

3. Restaurez l’application

Effectuez l’appel de l’API REST suivant. Vous devez fournir l’ID d’une sauvegarde (comme indiqué ci-dessous) ou d’un instantané.

Méthode HTTP	Chemin
EN	/account/{account_id}/k8s/v2/apps/{app_id}

Paramètres d’entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
JSON	Corps	Oui.	Fournit les paramètres de l’application clonée. Voir l’exemple ci-dessous.

Exemple d’entrée JSON

```
{
  "type": "application/astra-app",
  "version": "2.0",
  "backupID": "e24515bd-a28e-4b28-b832-f3c74dbf32fb"
}
```

Exemple Curl : restaurez une application à partir d’une sauvegarde

```
curl --location -i --request PUT
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/k8s/v2/apps/<APP_ID>'
--header 'Content-Type: application/astra-app+json' --header '*/*'
--header 'ForceUpdate: true' --header 'Authorization: Bearer <API_TOKEN>'
--data @JSONinput
```

Espaces de noms

Lister les espaces de noms

Vous pouvez lister les espaces de noms disponibles.

1. Dressez la liste des espaces de noms

Exécutez l’appel d’API REST suivant pour afficher la liste des namespaces.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/topologique/v1/namespaces

Exemple Curl : renvoie toutes les données de tous les espaces de noms

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/namespaces'
--header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple Curl : nom, état et ID de cluster pour tous les namespaces

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/topology/v1/namespaces?incl
ude=name,namespaceState,clusterID' --header 'Accept: */*' --header
'Authorization: Bearer <API_TOKEN>'
```

Exemple de sortie JSON


```
{
    "items": [
        [
            "default",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ],
        [
            "kube-node-lease",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ],
        [
            "kube-public",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ],
        [
            "kube-system",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ],
        [
            "mysql",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ],
        [
            "mysql-clone1",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ],
        [
            "netapp-acc-operator",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
```

```
        ],
        [
            "openshift",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ],
        [
            "trident",
            "discovered",
            "922f924a-a476-4a79-97f6-472571698154"
        ]
    ],
    "metadata": {}
}
```

Assistance

Dressez la liste des notifications

Vous pouvez lister les notifications d'un compte Astra spécifique. Vous pouvez le faire dans le cadre de la surveillance de l'activité du système ou du débogage d'un problème.

1. Dressez la liste des notifications

Effectuez l'appel de l'API REST suivant.

Méthode HTTP	Chemin
OBTENEZ	/account/{account_id}/core/v1/notifications

Paramètres d'entrée supplémentaires

Outre les paramètres communs à tous les appels API REST, les paramètres suivants sont également utilisés dans les exemples de boucles pour cette étape.

Paramètre	Type	Obligatoire	Description
filtre	Requête	Non	Vous pouvez éventuellement filtrer les notifications que vous souhaitez renvoyer dans la réponse.
inclure	Requête	Non	Sélectionner éventuellement les valeurs que vous souhaitez renvoyer dans la réponse.

Exemple de boucle : renvoie toutes les notifications

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/core/v1/notifications'
--header 'Accept: */*' --header 'Authorization: Bearer <API_TOKEN>'
```

Exemple Curl : renvoie la description des notifications avec gravité d'avertissement

```
curl --location -i --request GET
'https://astra.netapp.io/accounts/<ACCOUNT_ID>/core/v1/notifications?filte
r=severity%20eq%20'warning'&include=description' --header 'Accept: */*'
--header 'Authorization: Bearer <API_TOKEN>'
```

Exemple de sortie JSON


```
{
  "items": [
    [
      "Trident on cluster david-ie-00 has failed or timed out;
installation of the Trident operator failed or is not yet complete;
operator failed to reach an installed state within 300.00 seconds;
container trident-operator not found in operator deployment"
    ],
    [
      "Trident on cluster david-ie-00 has failed or timed out;
installation of the Trident operator failed or is not yet complete;
operator failed to reach an installed state within 300.00 seconds;
container trident-operator not found in operator deployment"
    ]
  ],
  "metadata": {}
}
```


Supprimer une application ayant échoué

Il se peut que vous ne puissiez pas supprimer une application gérée si elle a une sauvegarde ou un snapshot en état d'échec. Dans ce cas, vous pouvez supprimer manuellement l'application à l'aide du workflow décrit ci-dessous.

1. Sélectionnez l'application à supprimer

Exécutez le flux de travail "Répertorier les applications" et sélectionnez l'application à supprimer.

2. Dressez la liste des sauvegardes existantes de l'application

Exécutez le flux de travail "Répertoriez les sauvegardes".

3. Supprimez toutes les sauvegardes

Supprimez toutes les sauvegardes de l'application en exécutant le flux de travail "Supprimer une sauvegarde" pour chaque sauvegarde de la liste.

4. Dressez la liste des instantanés existants de l'application

Exécutez le flux de travail "Répertorier les snapshots".

5. Supprimez tous les instantanés

Exécutez le flux de travail "Supprime un snapshot" à partir de chaque instantané de la liste.

6. Retirez l'application

Exécutez le flux de travail "Annuler la gestion d'une application" pour supprimer l'application.

À l’aide de Python

Kit de développement logiciel NetApp Astra Control Python

Le kit de développement logiciel NetApp Astra Control Python est un logiciel open source qui permet d’automatiser le déploiement d’Astra Control. Il constitue également une ressource précieuse pour en savoir plus sur l’API REST Astra Control, peut-être dans le cadre de la création de votre propre plateforme d’automatisation.

> Dans un souci de simplicité, le kit de développement NetApp Astra Control Python sera appelé **SDK** au cours du reste de cette page.

Deux outils logiciels connexes

Le SDK inclut deux outils différents bien que associés qui fonctionnent à différents niveaux d’abstraction lors de l’accès à l’API REST Astra Control.

Kit de développement Astra

Le kit de développement logiciel Astra fournit les principales fonctionnalités de la plateforme. Il inclut un ensemble de classes Python qui abstraite les appels de l’API REST sous-jacente. Ces classes prennent en charge les actions administratives sur diverses ressources Astra Control, notamment les applications, les sauvegardes, les instantanés et les clusters.

Le kit de développement Astra fait partie de l’offre et est fourni en une seule pièce `astraSDK.py` fichier. Vous pouvez importer ce fichier dans votre environnement et utiliser les classes directement.

> Le **SDK Python de contrôle Astra** (ou uniquement SDK) de NetApp est le nom de l’ensemble du package. Le **Astra SDK** fait référence aux classes Python de base dans le fichier unique `astraSDK.py`.

Script Toolkit

En plus du fichier Astra SDK, le `toolkit.py` script également disponible. Ce script fonctionne à un niveau d’abstraction plus élevé en donnant accès à des actions administratives discrètes définies en interne comme des fonctions Python. Le script importe le SDK Astra et fait des appels aux classes selon les besoins.

Comment y accéder

Vous pouvez accéder au SDK de la manière suivante.

Pack Python

Le SDK est disponible à l’adresse "Index des paquets Python" sous le nom **actoolkit**. Un numéro de version est attribué au package et continuera d’être mis à jour au besoin. Vous devez utiliser l’utilitaire de gestion de paquets **PIP** pour installer le package dans votre environnement.

Une fois installé, les astraSDK.py classes peuvent être utilisées en plaçant `import astraSDK` dans vos scripts. En outre, `actoolkit` peut être appelé directement sur votre invite de commande et équivaut à `toolkit.py` (`actoolkit list clusters` est identique à `./toolkit.py list clusters`).

Voir "PyPI : kit de développement Python de contrôle NetApp Astra" pour en savoir plus.

Code source GitHub

Le code source du SDK est également disponible sur GitHub. Le référentiel inclut les éléments suivants :

- `astraSDK.py` (SDK Astra avec classes Python)
- `toolkit.py` (script basé sur les fonctions de niveau supérieur)
- Conditions requises et instructions détaillées pour l'installation
- Scripts d'installation
- Documentation complémentaire

Vous pouvez cloner le "GitHub : kits NetApp/netapp-astra-toolkits" référentiel pour votre environnement local.

Conditions requises pour l'installation et de base

Il y a plusieurs options et conditions requises à considérer dans le cadre de l'installation de l'emballage et de la préparation à l'utilisation.

Résumé des options d'installation

Vous pouvez installer le SDK de l'une des manières suivantes :

- Utilisez le préparé "Docker : kits de ressources NetApp/astra" image, avec toutes les dépendances nécessaires installées, y compris `actoolkit`
- Utiliser le PIP pour installer le `actoolkit` Package de PyPI dans votre environnement Python
- Clonez le référentiel GitHub et copiez/modifiez les deux fichiers Python de base afin qu'ils soient accessibles à votre code client Python

Reportez-vous aux pages PyPI et GitHub pour plus d'informations.

Exigences relatives à l'environnement Astra Control

Qu'il s'agisse d'utiliser directement les classes Python du SDK Astra ou des fonctions du `toolkit.py` À terme, vous accéderez à l'API REST lors d'un déploiement d'Astra Control. Vous aurez donc besoin d'un compte Astra et d'un jeton API. Voir "Avant de commencer" Et les autres pages de la section **Get Started** de cette documentation pour plus d'informations.

Exigences relatives au kit de développement logiciel NetApp Astra Control Python

Le SDK a plusieurs conditions préalables liées à l'environnement Python local. Par exemple, vous devez utiliser Python 3.8 ou ultérieur. En outre, plusieurs paquets Python sont requis. Pour plus d'informations, consultez la page de référentiel GitHub ou la page de package PyPI.

Résumé des ressources utiles

Voici quelques ressources utiles pour commencer.

- "PyPI : kit de développement Python de contrôle NetApp Astra"
- "GitHub : kits NetApp/netapp-astra-toolkits"
- "Docker : kits de ressources NetApp/astra"

Python natif

Avant de commencer

Python est un langage de développement populaire pour l'automatisation des data centers. Avant d'utiliser les fonctions natives de Python avec plusieurs packages courants, vous devez préparer l'environnement et les fichiers d'entrée requis.

> En plus d'accéder directement à l'API REST Astra Control avec Python, NetApp propose un kit d'outils qui résume l'API et élimine une partie de la complexité. Voir "Kit de développement logiciel NetApp Astra Control Python" pour en savoir plus.

Préparation de l'environnement

Les exigences de configuration de base pour exécuter les scripts Python sont décrites ci-dessous.

Python 3

La dernière version de Python 3 doit être installée.

Bibliothèques supplémentaires

Les bibliothèques **requêtes** et **urllib3** doivent être installées. Vous pouvez utiliser pip ou un autre outil de gestion Python, selon les besoins de votre environnement.

Accès réseau

Le poste de travail sur lequel les scripts s'exécutent doit disposer d'un accès réseau et pouvoir accéder à Astra Control. Lorsque vous utilisez le service Astra Control, vous devez être connecté à Internet et être en mesure de vous connecter au service à `https://astra.netapp.io`.

Informations d'identité

Vous avez besoin d'un compte Astra valide avec l'identifiant de compte et le jeton API. Voir "Obtenir un jeton API" pour en savoir plus.

Créez les fichiers d'entrée JSON

Les scripts Python s'appuient sur les informations de configuration contenues dans les fichiers d'entrée JSON. Des exemples de fichiers sont fournis ci-dessous.

> Vous devez mettre à jour les échantillons en fonction de votre environnement.

Informations d'identité

Le fichier suivant contient le jeton API et le compte Astra. Vous devez transmettre ce fichier aux scripts Python à l'aide de `-i` (ou `--identity`) Paramètre CLI.

```
{
  "api_token": "kH4CA_uVIa8q9UuPzhJaAHaGlaR7-no901DkkrVjIXk=",
  "account_id": "5131dfdf-03a4-5218-ad4b-fe84442b9786"
}
```

Répertorier les applications

Vous pouvez utiliser le script suivant pour répertorier les applications de votre compte Astra.

Voir "Avant de commencer" Par exemple le fichier d'entrée JSON requis.

```
#!/usr/bin/env python3
##-----------------------------------------------------------------------
------
#
# Usage: python3 list_man_apps.py -i identity_file.json
#
# (C) Copyright 2022 NetApp, Inc.
#
# This sample code is provided AS IS, with no support or warranties of
# any kind, including but not limited for warranties of merchantability
# or fitness of any kind, expressed or implied. Permission to use,
# reproduce, modify and create derivatives of the sample code is granted
# solely for the purpose of researching, designing, developing and
# testing a software application product for use with NetApp products,
# provided that the above copyright notice appears in all copies and
# that the software application product is distributed pursuant to terms
# no less restrictive than those set forth herein.
#
##-----------------------------------------------------------------------
------

import argparse
import json
import requests
import urllib3
import sys

# Global variables
api_token = ""
account_id = ""

def get_managed_apps():
    ''' Get and print the list of apps '''

    # Global variables
    global api_token
    global account_id

    # Create an HTTP session
```

```
    sess1 = requests.Session()

    # Suppress SSL unsigned certificate warning
    urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

    # Create URL
    url1 = "https://astra.netapp.io/accounts/" + account_id +
"/k8s/v2/apps"

    # Headers and response output
    req_headers  = {}
    resp_headers = {}
    resp_data    = {}

    # Prepare the request headers
    req_headers.clear
    req_headers['Authorization'] = "Bearer " + api_token
    req_headers['Content-Type'] = "application/astra-app+json"
    req_headers['Accept'] = "application/astra-app+json"

    # Make the REST call
    try:
        resp1 = sess1.request('get', url1, headers=req_headers,
allow_redirects=True, verify=False)

    except requests.exceptions.ConnectionError:
        print("Connection failed")
        sys.exit(1)

    # Retrieve the output
    http_code = resp1.status_code
    resp_headers = resp1.headers

    # Print the list of apps
    if resp1.ok:
        resp_data = json.loads(resp1.text)
        items = resp_data['items']
        for i in items:
            print(" ")
            print("Name: " + i['name'])
            print("ID: " + i['id'])
            print("State: " + i['state'])
    else:
        print("Failed with HTTP status code: " + str(http_code))

    print(" ")
```

```
    # Close the session
    sess1.close()

    return

def read_id_file(idf):
    ''' Read the identity file and save values '''

    # Global variables
    global api_token
    global account_id

    with open(idf) as f:
        data = json.load(f)

    api_token = data['api_token']
    account_id = data['account_id']

    return

def main(args):
    ''' Main top level function '''

    # Global variables
    global api_token
    global account_id

    # Retrieve name of JSON input file
    identity_file = args.id_file

    # Get token and account
    read_id_file(identity_file)

    # Issue REST call
    get_managed_apps()

    return

def parseArgs():
        ''' Parse the CLI input parameters '''

        parser = argparse.ArgumentParser(description='Astra REST API -
List the apps',
                                 add_help = True)
        parser.add_argument("-i", "--identity", action="store", dest
="id_file", default=None,
                                 help='(Req) Name of the identity input file',
```

```
required=True)

        return parser.parse_args()

if __name__ == '__main__':
    ''' Begin here '''

    # Parse input parameters
    args = parseArgs()

    # Call main function
    main(args)
```

Référence API

Vous pouvez accéder aux détails des appels de l’API REST Astra Control, y compris les méthodes HTTP, les paramètres d’entrée et les réponses. Cette référence complète est utile lors du développement d’applications d’automatisation à l’aide de l’API REST.

> La documentation de référence de l’API REST est actuellement fournie avec Astra Control et est disponible en ligne.

Avant de commencer

Vous avez besoin d’un compte pour Astra Control Center ou Astra Control Service.

Étapes

1. Connectez-vous à Astra à l’aide de vos identifiants de compte.

   Accédez au site suivant pour le service Astra Control : "https://astra.netapp.io"

2. Cliquez sur l’icône figure en haut à droite de la page et sélectionnez **API Access**.
3. En haut de la page, cliquez sur l’URL affichée sous **Documentation API**.
4. Indiquez à nouveau les informations d’identification de votre compte si vous y êtes invité.

Ressources supplémentaires

Vous trouverez des ressources d'aide supplémentaires sur les services cloud et le support NetApp, ainsi que des informations générales sur REST et cloud.

Astra

- "Documentation Astra Control Center 22.08"

  Documentation relative à la version actuelle du logiciel Astra Control Center déployé dans les locaux du client.

- "Documentation relative au service après-vente Astra Control"

  Documentation relative à la version actuelle du logiciel Astra Control Service disponible dans le Cloud public.

- "Documentation Astra Trident"

  Documentation relative à la version actuelle du logiciel Astra Trident, un orchestrateur de stockage open source géré par NetApp.

- "Documentation de la gamme Astra"

  Un emplacement central permettant d'accéder à toute la documentation d'Astra tant pour les déploiements sur site que dans le cloud public.

Ressources cloud NetApp

- "NetApp BlueXP"

  Site central des solutions clouds NetApp.

- "Console NetApp Cloud Central"

  Console de services NetApp Cloud Central avec connexion.

- "Support NetApp"

  Accédez aux outils de dépannage, à la documentation et à l'assistance technique.

Concepts DE REPOS et cloud

- Doctorat "thèse" Par Roy Fielding

  Cette publication a introduit et établi le modèle de développement des applications REST.

- "Auth0"

  Il s'agit du service de plateforme d'authentification et d'autorisation utilisé par le service Astra pour l'accès au Web.

- "Éditeur RFC"

  Source faisant autorité pour les normes Web et Internet, maintenue comme un ensemble de documents RFC numérotés de façon unique.

Versions antérieures de la documentation Astra Control Automation

Vous pouvez accéder à la documentation relative à l'automatisation des versions antérieures d'Astra Control en cliquant sur les liens ci-dessous.

- "Documentation Astra Control Automation 22.04"
- "Documentation Astra Control Automation 21.12"
- "Documentation Astra Control Automation 21.08"

Mentions légales

Les mentions légales donnent accès aux déclarations de copyright, aux marques, aux brevets, etc.

Droits d'auteur

http://www.netapp.com/us/legal/copyright.aspx

Marques déposées

NetApp, le logo NETAPP et les marques mentionnées sur la page des marques commerciales NetApp sont des marques commerciales de NetApp, Inc. Les autres noms de sociétés et de produits peuvent être des marques commerciales de leurs propriétaires respectifs.

http://www.netapp.com/us/legal/netapptmlist.aspx

Brevets

Vous trouverez une liste actuelle des brevets appartenant à NetApp à l'adresse suivante :

https://www.netapp.com/us/media/patents-page.pdf

Politique de confidentialité

https://www.netapp.com/us/legal/privacypolicy/index.aspx

Licence API Astra Control

https://docs.netapp.com/us-en/astra-automation/media/astra-api-license.pdf