



インストールの概要

Astra Control Center

NetApp
November 21, 2023

This PDF was generated from https://docs.netapp.com/ja-jp/astra-control-center-2208/get-started/acc_cluster_cr_options.html on November 21, 2023. Always check docs.netapp.com for the latest.

目次

インストールの概要	1
標準の手順で Astra Control Center をインストールします	1
OpenShift OperatorHub を使用して Astra Control Center をインストールします	27
Cloud Volumes ONTAP ストレージバックエンドに Astra Control Center をインストールします	34

インストールの概要

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

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

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

Astra Control Center をインストールするには、NetApp Support Siteからインストールバンドルをダウンロードし、次の手順を実行して、 Astra Control Center Operator と Astra Control Center を環境にインストールします。この手順を使用して、インターネット接続環境またはエアギャップ環境に Astra コントロールセンターをインストールできます。

Red Hat OpenShift環境では、を使用できます "代替手順" OpenShift OperatorHub を使用して Astra Control Center をインストールします。

必要なもの

- "インストールを開始する前に、 Astra Control Center の導入環境を準備します"。
- 使用環境でポッドセキュリティポリシーを設定または設定したい場合は、ポッドセキュリティポリシーと、それらがAstra Control Centerのインストールに与える影響について理解しておいてください。を参照してください "ポッドのセキュリティポリシーの制限事項を理解します"。
- すべてのクラスタオペレータが正常な状態であり、使用可能であることを確認します。

```
kubectl get clusteroperators
```

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

```
kubectl get apiservices
```

- 使用するネットアップFQDNがこのクラスタにルーティング可能であることを確認します。つまり、内部 DNS サーバに DNS エントリがあるか、すでに登録されているコア URL ルートを使用しています。
- クラスタにcert-managerがすでに存在する場合は、いくつかを実行する必要があります "事前に必要な手順" そのため、Astra Control Centerは独自の証明書管理ツールをインストールしません。

このタスクについて

Astra Control Center のインストールプロセスでは、次のことが実行されます。

- Astraコンポーネントをにインストールします netapp-acc （またはカスタム名）ネームスペース。
- デフォルトアカウントを作成します。
- デフォルトの管理ユーザのEメールアドレスとデフォルトのワンタイムパスワードを設定します。このユ

ーザには、UIへの初回ログインに必要なシステムのオーナーロールが割り当てられます。

- Astra Control Center のすべてのポッドが実行されていることを確認するのに役立ちます。
- Astra の UI をインストールします。



(環境 the Astra Data Store Early Access Program (EAP) リリースのみ) Astra Control Center を使用してAstraデータストアを管理し、VMwareワークフローを有効にする場合は、Astra Control Centerのみをに導入します pcloud ではなく、ネームスペースで指定します netapp-acc この手順 の手順で説明するネームスペースまたはカスタムネームスペース。



すべてのAstra Control Centerポッドが削除されないようにするため、インストールプロセス全体で次のコマンドを実行しないでください。 `kubectl delete -f astra_control_center_operator_deploy.yaml`



Docker Engine の代わりに Red Hat の Podman を使用している場合は、Docker コマンドの代わりに Podman コマンドを使用できます。

手順

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

- [Astra Control Centerバンドルをダウンロードして開梱します](#)
- [ネットアップAstra kubectlプラグインをインストール](#)
- [\[イメージをローカルレジストリに追加します\]](#)
- [\[認証要件を持つレジストリのネームスペースとシークレットを設定します\]](#)
- [Astra Control Center オペレータを設置します](#)
- [Astra Control Center を設定します](#)
- [Astra Control Center とオペレータのインストールを完了します](#)
- [\[システムステータスを確認します\]](#)
- [\[ロードバランシング用の入力を設定します\]](#)
- [Astra Control Center UI にログインします](#)

Astra Control Centerバンドルをダウンロードして開梱します

1. Astra Control Center バンドルをダウンロードします (astra-control-center-[version].tar.gz) をクリックします ["NetApp Support Site"](#)。
2. から Astra Control Center 証明書とキーの zip をダウンロードします ["NetApp Support Site"](#)。
3. (任意) 次のコマンドを使用して、バンドルのシグニチャを確認します。

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

4. 画像を抽出します。

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

ネットアップAstra kubectlプラグインをインストール

ネットアップアストラ kubectl コマンドラインプラグインにより、Astra Control Centerの導入とアップグレードに関連する一般的なタスクを実行する時間を節約できます。

必要なもの

ネットアップでは、プラグイン用のバイナリを提供しており、CPUアーキテクチャやオペレーティングシステムが異なる場合はそのプラグインをこのタスクを実行する前に、使用しているCPUとオペレーティングシステムを把握しておく必要があります。LinuxおよびMacオペレーティングシステムでは、`uname -a` この情報を収集するためのコマンドです。

手順

1. 使用可能なネットアップAstraを選択します kubectl プラグインバイナリ。オペレーティングシステムとCPUアーキテクチャに必要なファイルの名前をメモします。

```
ls kubectl-astra/
```

2. ファイルを標準と同じ場所にコピーします kubectl ユーティリティ。この例では、を使用しています kubectl ユーティリティはにあります /usr/local/bin ディレクトリ。交換してください <binary-name> 必要なファイル名を使用して、次の操作を実行します。

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

イメージをローカルレジストリに追加します

1. コンテナエンジンに応じた手順を実行します。

Docker です

1. Astraディレクトリに移動します。

```
cd acc
```

2. [[[[</Z1></Z1></Z1></Z1></Z1></Z1></Z1></Z1></Z1>_image_local_registry_push]]]]]]]]</Z2>
アストラControl Centerイメージディレクトリ内のパッケージイメージをローカルレジストリにプッシュします。</Z3>コマンドを実行する前に、次の置き換えを行ってください。

- bundle_fileをAstra Controlバンドルファイルの名前に置き換えます（例：acc.manifest.yaml）。
- my_registryをDockerリポジトリのURLに置き換えます。
- my_registry_userをユーザー名に置き換えます。
- my_registry_tokenをレジストリの認証済みトークンに置き換えます。

```
kubect1 astra packages push-images -m BUNDLE_FILE -r MY_REGISTRY  
-u MY_REGISTRY_USER -p MY_REGISTRY_TOKEN
```

ポドマン

1. レジストリにログインします。

```
podman login [your_registry_path]
```

2. 次のスクリプトを実行して、コメントに記載されているように<your _registry>を置き換えます。

```
# You need to be at the root of the tarball.
# You should see these files to confirm correct location:
#   acc.manifest.yaml
#   acc/

# Replace <YOUR_REGISTRY> with your own registry (e.g
registry.customer.com or registry.customer.com/testing, etc..)
export REGISTRY=<YOUR_REGISTRY>
export PACKAGENAME=acc
export PACKAGEVERSION=22.08.1-26
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
    # Load to local cache
    astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //'')

    # Remove path and keep imageName.
    astraImageNoPath=$(echo ${astraImage} | sed 's:.*/:::')

    # Tag with local image repo.
    podman tag ${astraImage} ${REGISTRY}/netapp/astra/${PACKAGENAME}
/${PACKAGEVERSION}/${astraImageNoPath}

    # Push to the local repo.
    podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

認証要件を持つレジストリのネームスペースとシークレットを設定します

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

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

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

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

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

対応：

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

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



プレースホルダ `your_registry_path` 以前にアップロードした画像の場所と一致する必要があります（例： `[Registry_URL]/netapp/astra/astracc/22.08.1-26`）。

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

回答例：

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



シークレットの生成後にネームスペースを削除する場合は、ネームスペースが再作成されたあとにネームスペースのシークレットを再生成する必要があります。

- c. を作成します netapp-acc（またはカスタムの名前付き）ネームスペース。

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

回答例：

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

- d. のシークレットを作成します netapp-acc（またはカスタムの名前付き）ネームスペース。Docker 情報を追加して次のコマンドを実行します。

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

応答

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


- a. `[[[sup_kubeconfig_secret]]]`（オプション）インストール後に Astra Control Center でクラスタを自動的に管理する場合は、このコマンドを使用して展開する Astra Control Center ネームスペース内のシークレットとして kubeconfig を指定する必要があります。

```
kubectl create secret generic [acc-kubeconfig-cred or custom secret name] --from-file=<path-to-your-kubeconfig> -n [netapp-acc or custom namespace]
```

Astra Control Center オペレータを設置します

1. ディレクトリを変更します。

```
cd manifests
```

2. Astra Control Center オペレータ配置YAMLを編集します
(`astra_control_center_operator_deploy.yaml`)を参照して、ローカルレジストリとシークレットを参照してください。

```
vim astra_control_center_operator_deploy.yaml
```



注釈付きサンプルYAMLは以下の手順に従います。

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

```
imagePullSecrets:  
- name: <astra-registry-cred>
```

- b. 変更 `[your_registry_path]` をクリックします `kube-rbac-proxy` でイメージをプッシュしたレジストリパスへのイメージ [前の手順](#)。
- c. 変更 `[your_registry_path]` をクリックします `acc-operator-controller-manager` でイメージをプッシュしたレジストリパスへのイメージ [前の手順](#)。
- d. （Astra データストアプレビューを使用するインストールの場合）この問題に関する既知の情報を参照してください ["ストレージクラスのプロビジョニングとYAML に対する追加の変更"](#)。

```

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
        - args:
            - --secure-listen-address=0.0.0.0:8443
            - --upstream=http://127.0.0.1:8080/
            - --logtostderr=true
            - --v=10
          image: [your_registry_path]/kube-rbac-proxy:v4.8.0
          name: kube-rbac-proxy
          ports:
            - containerPort: 8443
              name: https
        - args:
            - --health-probe-bind-address=:8081
            - --metrics-bind-address=127.0.0.1:8080
            - --leader-elect
          command:
            - /manager
          env:
            - name: ACCOP_LOG_LEVEL
              value: "2"
          image: [your_registry_path]/acc-operator:[version x.y.z]
          imagePullPolicy: IfNotPresent
      imagePullSecrets: []

```

3. Astra Control Center オペレータをインストールします。

```
kubectl apply -f astra_control_center_operator_deploy.yaml
```

回答例：

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

4. ポッドが実行中であることを確認します

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

Astra Control Center を設定します

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



astra_control_center_min.yaml はデフォルトのCRで、ほとんどのインストールに適しています。すべてのことをよく理解してください ["CRオプションとその可能性のある値"](#) お客様の環境にAstra Control Centerを正しく導入できるようにするため。環境で追加のカスタマイズが必要な場合は、を使用できます astra_control_center.yaml 代替CRとして。

```
vim astra_control_center_min.yaml
```



許可が不要なレジストリを使用している場合は、を削除する必要があります secret ラインの内側 imageRegistry または、インストールが失敗します。

- a. 変更 [your_registry_path] 前の手順でイメージをプッシュしたレジストリパスに移動します。
- b. を変更します accountName stringには、アカウントに関連付ける名前を指定します。

- c. を変更します `astraAddress` ブラウザからAstraにアクセスする際に使用するFQDNを文字列で指定します。使用しないでください `http://` または `https://` をクリックします。この FQDN をコピーしてで使います [後の手順](#)。
- d. を変更します `email` デフォルトの初期管理者アドレスを表す文字列。この E メールアドレスをコピーしてで使います [後の手順](#)。
- e. 変更 `enrolled` を選択します `AutoSupport false` インターネットに接続されていないか、または保持されているサイト `true` 接続されているサイト用。
- f. 外部証明書マネージャを使用する場合は、に次の行を追加します `spec` :

```
spec:
  crds:
    externalCertManager: true
```

- g. (オプション) 名を追加します `firstName` 姓を入力します `lastName` アカウントに関連付けられているユーザのこの手順は、UI ですぐに実行することもあとで実行することもできます。
- h. (オプション) を変更します `storageClass` インストールに必要な場合、別のTridentストレージクラスリソースへの値。
- i. (オプション) インストール後に Astra Control Center でクラスタを自動的に管理する場合は [このクラスタの kubeconfig を含むシークレットを作成しました](#) をクリックし、という名前のYAMLファイルに新しいフィールドを追加して、シークレットの名前を指定します `astraKubeConfigSecret`:
"acc-kubeconfig-cred or custom secret name"
- j. 次のいずれかの手順を実行します。

- * その他の入力コントローラ (`ingressType: Generic`) * : これはアストラコントロールセンターでのデフォルトのアクションです。Astra Control Center を展開したら、Astra Control Center を URL で公開するように入力コントローラを設定する必要があります。

デフォルトのAstra Control Centerインストールでは、ゲートウェイがセットアップされます (`service/traefik`) を入力します `ClusterIP`。このデフォルトのインストールでは、トラフィックをルーティングするために Kubernetes IngressController/Ingress を追加で設定する必要があります。入力を使用する場合は、を参照してください ["ロードバランシング用の入力を設定します"](#)。

- サービスロードバランサ (**`ingressType: AccTrafik`**) : IngressControllerをインストールしない場合、または入力リソースを作成しない場合は、を設定します `ingressType` 終了:
`AccTraefik`。

これにより、Astra Control Centerが導入されます `traefik Gateway as a Kubernetes LoadBalancer type service`の略。

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



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

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"
  ingressType: "Generic"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
```

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

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

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

回答例：

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

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

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

回答例：

```
astracontrolcenter.astra.netapp.io/astra created
```

システムステータスを確認します



OpenShift を使用する場合は、同等の OC コマンドを検証手順に使用できます。

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

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

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

回答例

NAME	READY	STATUS	RESTARTS
AGE			
acc-helm-repo-6b44d68d94-d8m55 13m	1/1	Running	0
activity-78f99ddf8-hltct 10m	1/1	Running	0
api-token-authentication-457nl 9m28s	1/1	Running	0
api-token-authentication-dgwsz 9m28s	1/1	Running	0
api-token-authentication-hmqqc 9m28s	1/1	Running	0
asup-75fd554dc6-m6qzh 9m38s	1/1	Running	0
authentication-6779b4c85d-92gds 8m11s	1/1	Running	0
bucket-service-7cc767f8f8-lqwr8 9m31s	1/1	Running	0
certificates-549fd5d6cb-5kmd6 9m56s	1/1	Running	0
certificates-549fd5d6cb-bkj9 9m56s	1/1	Running	0
cloud-extension-7bcb7948b-hn8h2 10m	1/1	Running	0
cloud-insights-service-56ccf86647-fgg69 9m46s	1/1	Running	0
composite-compute-677685b9bb-7vgsf 10m	1/1	Running	0
composite-volume-657d6c5585-dnq79 9m49s	1/1	Running	0
credentials-755fd867c8-vrlmt 11m	1/1	Running	0
entitlement-86495cdf5b-nwhh2 10m	1/1	Running	2
features-5684fb8b56-8d6s8 10m	1/1	Running	0
fluent-bit-ds-rhx7v 7m48s	1/1	Running	0
fluent-bit-ds-rjms4 7m48s	1/1	Running	0
fluent-bit-ds-zf5ph 7m48s	1/1	Running	0
graphql-server-66d895f544-w6hjd	1/1	Running	0

3m29s			
identity-744df448d5-rlcmm	1/1	Running	0
10m			
influxdb2-0	1/1	Running	0
13m			
keycloak-operator-75c965cc54-z7csw	1/1	Running	0
8m16s			
krakend-798d6df96f-9z2sk	1/1	Running	0
3m26s			
license-5fb7d75765-f8mjg	1/1	Running	0
9m50s			
login-ui-7d5b7df85d-l2s7s	1/1	Running	0
3m20s			
loki-0	1/1	Running	0
13m			
metrics-facade-599b9d7fcc-gtmgl	1/1	Running	0
9m40s			
monitoring-operator-67cc74f844-cdplp	2/2	Running	0
8m11s			
nats-0	1/1	Running	0
13m			
nats-1	1/1	Running	0
13m			
nats-2	1/1	Running	0
12m			
nautilus-769f5b74cd-k5jxm	1/1	Running	0
9m42s			
nautilus-769f5b74cd-kd9gd	1/1	Running	0
8m59s			
openapi-84f6ccd8ff-76kvp	1/1	Running	0
9m34s			
packages-6f59fc67dc-4g2f5	1/1	Running	0
9m52s			
polaris-consul-consul-server-0	1/1	Running	0
13m			
polaris-consul-consul-server-1	1/1	Running	0
13m			
polaris-consul-consul-server-2	1/1	Running	0
13m			
polaris-keycloak-0	1/1	Running	0
8m7s			
polaris-keycloak-1	1/1	Running	0
5m49s			
polaris-keycloak-2	1/1	Running	0
5m15s			
polaris-keycloak-db-0	1/1	Running	0

8m6s			
polaris-keycloak-db-1	1/1	Running	0
5m49s			
polaris-keycloak-db-2	1/1	Running	0
4m57s			
polaris-mongodb-0	2/2	Running	0
13m			
polaris-mongodb-1	2/2	Running	0
12m			
polaris-mongodb-2	2/2	Running	0
12m			
polaris-ui-565f56bf7b-zwr8b	1/1	Running	0
3m19s			
polaris-vault-0	1/1	Running	0
13m			
polaris-vault-1	1/1	Running	0
13m			
polaris-vault-2	1/1	Running	0
13m			
public-metrics-6d86d66444-2wbzl	1/1	Running	0
9m30s			
storage-backend-metrics-77c5d98dcd-dbhg5	1/1	Running	0
9m44s			
storage-provider-78c885f57c-6zcv4	1/1	Running	0
9m36s			
telegraf-ds-2l2m9	1/1	Running	0
7m48s			
telegraf-ds-qfzgh	1/1	Running	0
7m48s			
telegraf-ds-shrms	1/1	Running	0
7m48s			
telegraf-rs-bjpkt	1/1	Running	0
7m48s			
telemetry-service-6684696c64-qzfdf	1/1	Running	0
10m			
tenancy-6596b6c54d-vmppm	1/1	Running	0
10m			
traefik-7489dc59f9-6mnst	1/1	Running	0
3m19s			
traefik-7489dc59f9-xrkkg	1/1	Running	0
3m4s			
trident-svc-6c8dc458f5-jswcl	1/1	Running	0
10m			
vault-controller-6b954f9b76-gz9nm	1/1	Running	0
11m			

2. (オプション) インストールが完了したことを確認するには、を参照してください `acc-operator` 次のコマンドを使用してログを作成します。

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



`accHost` クラスタの登録は最後の処理の1つです。登録に失敗しても原因の導入は失敗しません。ログにクラスタ登録エラーが示された場合は、クラスタ追加ワークフローを通じて再度登録を試行できます ["UI で"](#) または API。

3. すべてのポッドが実行中の場合は、インストールが正常に完了したことを確認します (READY は `True`) を使用して Astra Control Center にログインする際に使用するワンタイムパスワードを取得します。

```
kubectl get AstraControlCenter -n netapp-acc
```

対応：

NAME	UUID	VERSION	ADDRESS
READY			
astra	ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f	22.08.1-26	
10.111.111.111	True		



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

ロードバランシング用の入力を設定します

Kubernetes 入力コントローラをセットアップして、クラスタのロードバランシングなどのサービスへの外部アクセスを管理できます。

この手順では、入力コントローラの設定方法について説明します (`ingressType:Generic`)。これは、Astra Control Center でのデフォルトのアクションです。Astra Control Center を展開したら、Astra Control Center を URL で公開するように入力コントローラを設定する必要があります。



入力コントローラを設定しない場合は、を設定できます (`ingressType:AccTraefik`)。Astra Control Center は、タイプ「LoadBalancer」のサービスを使用します。(svc/traefik Astra Control Center の名前空間) で、アクセス可能な外部 IP アドレスが割り当てられている必要があります。お使いの環境でロードバランサが許可されていて、設定されていない場合は、MetalLB または別の外部サービスロードバランサを使用して、外部 IP アドレスをサービスに割り当てることができます。内部 DNS サーバ構成では、Astra Control Center に選択した DNS 名を、負荷分散 IP アドレスに指定する必要があります。サービスタイプ「LoadBalancer」および入力の詳細については、を参照してください ["要件"](#)。

この手順は、使用する入力コントローラのタイプによって異なります。

- Istio入力
- nginx 入力コントローラ
- OpenShift 入力コントローラ

必要なもの

- が必要です ["入力コントローラ"](#) すでに導入されている必要があります。
- ["入力クラス"](#) 入力コントローラに対応するものがすでに作成されている必要があります。
- V1.19 と v1.22 の間で Kubernetes のバージョンを使用している。

Istio Ingressの手順

1. Istio Ingressを設定します。



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

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

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

コマンド例：

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

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

コマンド例：

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



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

4. 入力リソースをに配置します `netapp-acc`（またはカスタムネームド）ネームスペースで、`v1beta1`（Kubernetesバージョンで1.22未満で廃止）または`v1`リソースタイプを使用して、非推奨または新しいスキーマのいずれかに対応します。

出力：

```

apiVersion: networking.k8s.io/v1beta1
kind: IngressClass
metadata:
  name: istio
spec:
  controller: istio.io/ingress-controller
---
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
  namespace: istio-system
spec:
  ingressClassName: istio
  tls:
  - hosts:
    - <ACC address>
    secretName: [tls secret name]
  rules:
  - host: [ACC address]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          serviceName: traefik
          servicePort: 80

```

v1の新しいスキーマについては、次の例を参照してください。

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

出力：

```

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

```

5. 通常どおりAstra Control Centerを導入します。

6. 入力ステータスを確認します。

```
kubectl get ingress -n netapp-acc
```

対応：

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

Nginx Ingress Controller の手順

1. タイプのシークレットを作成します[kubernetes.io/tls]をクリックします netapp-acc （またはカス

タム名前付き) ネームスペース。を参照してください ["TLS シークレット"](#)。

2. 入力リソースをに配置します netapp-acc (またはカスタムの名前付き) ネームスペースのいずれかを
使用します v1beta1 (Kubernetesのバージョンが1.22より前の場合は廃止) または v1 非推奨または新
しいスキーマのリソースタイプ:
 - a. に設定します v1beta1 廃止されたスキーマの例を次に示します。

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

- b. をクリックします v1 新しいスキーマの例を次に示します。

```

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

```

OpenShift 入力コントローラの手順

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

```

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

```

Astra Control Center UI にログインします

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

手順

1. ブラウザで、で使用したFQDNを入力します astraAddress を参照してください
astra_control_center_min.yaml CR When (時間) [Astra Control Center をインストールした](#)。
2. プロンプトが表示されたら、自己署名証明書を受け入れます。



カスタム証明書はログイン後に作成できます。

3. Astra Control Centerのログインページで、に使用した値を入力します email インチ `astra_control_center_min.yaml` CR When (時間) [Astra Control Center をインストールしたを](#) クリックし、続けてワンタイムパスワードを入力します (ACC-[UUID])。



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

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



初めてログインする際にパスワードを忘れた場合、他の管理ユーザアカウントがまだ作成されていないときは、ネットアップのサポートに問い合わせ、パスワードのリカバリに関するサポートを依頼してください。

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

インストールのトラブルシューティングを行います

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

手順

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

```
kubectl logs --follow -n netapp-acc-operator $(kubectl get pods -n netapp-acc-operator -o name) -c manager
```

次のステップ

を実行して導入を完了します ["セットアップのタスク"](#)。

=
:allow-uri-read:

ポッドのセキュリティポリシーの制限事項を理解します

Astra Control Centerは、PoDセキュリティポリシー (PSP) による特権制限をサポートします。ポッドセキュリティポリシーを使用すると、ユーザまたはグループがコンテナを実行できる対象や、それらのコンテナに付与できる権限を制限できます。

RKE2などの一部のKubernetesディストリビューションには、デフォルトのポッドセキュリティポリシーが用意されていますが、制限が厳しく、Astra Control Centerのインストール時に問題が発生します。

ここに記載されている情報と例を使用して、Astra Control Centerによって作成されるポッドセキュリティポリシーを理解し、Astra Control Centerの機能を妨げずに必要な保護を提供するポッドセキュリティポリシー

を設定できます。

Astra Control CenterによってインストールされたPSP

Astra Control Centerは、インストール中にポッドのセキュリティポリシーをいくつか作成します。これらの中には永続的なものもあれば、一部のものは特定の処理中に作成され、処理が完了すると削除されます。

インストール中に作成されたPSP

Astra Control Centerのインストール中に、Astra Control Centerオペレータは、Astra Control CenterネームスペースへのAstra Control Centerサービスの展開をサポートするために、カスタムのPoDセキュリティポリシー、Roleオブジェクト、およびRoleBindingオブジェクトをインストールします。

新しいポリシーとオブジェクトには次の属性があります。

```
kubectl get psp
```

NAME	PRIV	CAPS	SELINUX	RUNASUSER
FSGROUP	SUPGROUP	READONLYROOTFS	VOLUMES	
avp-psp	false		RunAsAny	RunAsAny
RunAsAny	RunAsAny	false	*	
netapp-astra-deployment-psp	false		RunAsAny	RunAsAny
RunAsAny	RunAsAny	false	*	

```
kubectl get role
```

NAME	CREATED AT
netapp-astra-deployment-role	2022-06-27T19:34:58Z

```
kubectl get rolebinding
```

NAME	ROLE
AGE	
netapp-astra-deployment-rb	Role/netapp-astra-deployment-role
32m	

バックアップ処理中に作成されたPSP

バックアップ処理中に、Astra Control Centerは、動的なPODセキュリティポリシー、ClusterRoleオブジェクト、およびRoleBindingオブジェクトを作成します。これらの機能により、別のネームスペースで実行されるバックアッププロセスがサポートされます。

新しいポリシーとオブジェクトには次の属性があります。

```
kubectl get psp
```

NAME		PRIV	CAPS		
SELINUX	RUNASUSER	FSGROUP	SUPGROUP	READONLYROOTFS	
VOLUMES					
netapp-astra-backup		false	DAC_READ_SEARCH		
RunAsAny	RunAsAny	RunAsAny	RunAsAny	false	*

```
kubectl get role
```

NAME	CREATED AT
netapp-astra-backup	2022-07-21T00:00:00Z

```
kubectl get rolebinding
```

NAME	ROLE	AGE
netapp-astra-backup	Role/netapp-astra-backup	62s

クラスタ管理中に作成されたPSP

クラスターを管理する場合、Astra Control Centerは管理対象クラスタにNetApp Monitoringオペレータをインストールします。この演算子は、PODセキュリティポリシー、ClusterRoleオブジェクト、およびRoleBindingオブジェクトを作成して、Astra Control Center名前空間にテレメトリサービスを展開します。

新しいポリシーとオブジェクトには次の属性があります。

```
kubectl get psp
```

NAME		PRIV	CAPS		
SELINUX	RUNASUSER	FSGROUP	SUPGROUP	READONLYROOTFS	
VOLUMES					
netapp-monitoring-psp-nkmo		true	AUDIT_WRITE,NET_ADMIN,NET_RAW		
RunAsAny	RunAsAny	RunAsAny	RunAsAny	false	*

```
kubectl get role
```

NAME	CREATED AT
netapp-monitoring-role-privileged	2022-07-21T00:00:00Z

```
kubectl get rolebinding
```

NAME	ROLE	
AGE		
netapp-monitoring-role-binding-privileged	Role/netapp-monitoring-role-privileged	2m5s

ネームスペース間のネットワーク通信を有効にします

一部の環境では、NetworkPolicy構造体を使用してネームスペース間のトラフィックを制限します。Astra Control Centerオペレータ、Astra Control Center、およびAstra Plugin for VMware vSphereはすべて異なる名前空間に存在します。これらの異なるネームスペース内のサービスは、相互に通信できる必要があります。この通信をイネーブルにするには、次の手順を実行します。

手順

1. Astra Control Center名前空間に存在するNetworkPolicyリソースをすべて削除します。

```
kubectl get networkpolicy -n netapp-acc
```

2. 前のコマンドで返された各NetworkPolicyオブジェクトについて、次のコマンドを使用して削除します。object_name >を、返されたオブジェクトの名前に置き換えます。

```
kubectl delete networkpolicy <OBJECT_NAME> -n netapp-acc
```

3. 次のリソースファイルを適用して、Astra Plugin for VMware vSphereサービスがAstra Control Centerサービスに要求を送信できるように、accネットワーキングポリシーオブジェクトを設定します。角カッコ内の情報を環境内の情報に置き換えます。

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: acc-avp-network-policy
  namespace: <ACC_NAMESPACE_NAME> # REPLACE THIS WITH THE ASTRA CONTROL
CENTER NAMESPACE NAME
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from:
      - namespaceSelector:
          matchLabels:
            kubernetes.io/metadata.name: <PLUGIN_NAMESPACE_NAME> #
REPLACE THIS WITH THE ASTRA PLUGIN FOR VMWARE VSPHERE NAMESPACE NAME
```

4. 次のリソースファイルを適用して、Astra Control CenterオペレータがAstra Control Centerサービスと通信できるように、acc-operator-network-policyオブジェクトを設定します。角カッコ内の情報を環境内の情報に置き換えます。

```

apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: acc-operator-network-policy
  namespace: <ACC_NAMESPACE_NAME> # REPLACE THIS WITH THE ASTRA CONTROL
CENTER NAMESPACE NAME
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from:
      - namespaceSelector:
          matchLabels:
            kubernetes.io/metadata.name: <NETAPP-ACC-OPERATOR> #
REPLACE THIS WITH THE OPERATOR NAMESPACE NAME

```

リソースの制限を解除します

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

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

手順

1. NetApp-accネームスペース内のリソースクォータを取得します。

```
kubectl get quota -n netapp-acc
```

対応：

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

2. 名前別にすべてのリソースクォータを削除します。

```
kubectl delete resourcequota pods-high -n netapp-acc
```

```
kubectl delete resourcequota pods-low -n netapp-acc
```

```
kubectl delete resourcequota pods-medium -n netapp-acc
```

3. NetApp-accネームスペース内の制限範囲を取得します。

```
kubectl get limits -n netapp-acc
```

対応：

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

4. 制限範囲を名前で削除します。

```
kubectl delete limitrange cpu-limit-range -n netapp-acc
```

=
:allow-uri-read:

OpenShift OperatorHub を使用して Astra Control Center をインストールします

Red Hat OpenShift を使用する場合は、Red Hat 認定オペレータを使用して Astra Control Center をインストールできます。この手順を使用して、から Astra Control Center をインストールします ["Red Hat エコシステムカタログ"](#) または、Red Hat OpenShift Container Platform を使用します。

この手順を完了したら、インストール手順に戻ってを実行する必要があります ["残りのステップ"](#) インストールが成功したかどうかを確認し、ログオンします。

必要なもの

- ["インストールを開始する前に、Astra Control Center の導入環境を準備します"](#)。
- OpenShift クラスタから、すべてのクラスタオペレータが正常な状態にあることを確認します (available はです true) ；

```
oc get clusteroperators
```

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

```
oc get apiservices
```

- データセンターにAstra Control CenterのFQDNアドレスを作成します。
- 説明したインストール手順を実行するために必要な権限を取得し、Red Hat OpenShift Container Platform にアクセスします。
- クラスタにcert-managerがすでに存在する場合は、いくつかを実行する必要があります **"事前に必要な手順"**。そのため、Astra Control Centerは独自の証明書管理ツールをインストールしません。

手順

- [Astra Control Center](#)バンドルをダウンロードして開梱します
- ネットアップAstra kubectlプラグインをインストール
- [\[イメージをローカルレジストリに追加します\]](#)
- [\[オペレータインストールページを検索します\]](#)
- [\[オペレータをインストールします\]](#)
- [Astra Control Center](#) をインストールします

Astra Control Centerバンドルをダウンロードして開梱します

1. Astra Control Center バンドルをダウンロードします (astra-control-center-[version].tar.gz) をクリックします ["NetApp Support Site"](#)。
2. から Astra Control Center 証明書とキーの zip をダウンロードします ["NetApp Support Site"](#)。
3. (任意) 次のコマンドを使用して、バンドルのシグニチャを確認します。

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

4. 画像を抽出します。

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

ネットアップAstra kubectlプラグインをインストール

ネットアップアストラ kubectl コマンドラインプラグインにより、Astra Control Centerの導入とアップグレードに関連する一般的なタスクを実行する時間を節約できます。

必要なもの

ネットアップでは、プラグイン用のバイナリを提供しており、CPUアーキテクチャやオペレーティングシス

テムが異なる場合はそのプラグインをこのタスクを実行する前に、使用しているCPUとオペレーティングシステムを把握しておく必要があります。LinuxおよびMacオペレーティングシステムでは、`uname -a` この情報を収集するためのコマンドです。

手順

1. 使用可能なネットアップAstraを選択します `kubectl` プラグインバイナリ。オペレーティングシステムとCPUアーキテクチャに必要なファイルの名前をメモします。

```
ls kubectl-astra/
```

2. ファイルを標準と同じ場所にコピーします `kubectl` ユーティリティ。この例では、を使用しています `kubectl` ユーティリティはにありますが `/usr/local/bin` ディレクトリ。交換してください `<binary-name>` 必要なファイル名を使用して、次の操作を実行します。

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

イメージをローカルレジストリに追加します

1. コンテナエンジンに応じた手順を実行します。

Docker です

1. Astraディレクトリに移動します。

```
cd acc
```

2. [[[[</Z1></Z1></Z1></Z1></Z1></Z1></Z1></Z1></Z1>_image_local_registry_push]]]]]]]]</Z2>
アストラControl Centerイメージディレクトリ内のパッケージイメージをローカルレジストリにプッシュします。</Z3>コマンドを実行する前に、次の置き換えを行ってください。

- bundle_fileをAstra Controlバンドルファイルの名前に置き換えます（例：acc.manifest.yaml）。
- my_registryをDockerリポジトリのURLに置き換えます。
- my_registry_userをユーザー名に置き換えます。
- my_registry_tokenをレジストリの認証済みトークンに置き換えます。

```
kubect1 astra packages push-images -m BUNDLE_FILE -r MY_REGISTRY  
-u MY_REGISTRY_USER -p MY_REGISTRY_TOKEN
```

ポドマン

1. レジストリにログインします。

```
podman login [your_registry_path]
```

2. 次のスクリプトを実行して、コメントに記載されているように<your _registry>を置き換えます。


```

# You need to be at the root of the tarball.
# You should see these files to confirm correct location:
#   acc.manifest.yaml
#   acc/

# Replace <YOUR_REGISTRY> with your own registry (e.g
registry.customer.com or registry.customer.com/testing, etc..)
export REGISTRY=<YOUR_REGISTRY>
export PACKAGENAME=acc
export PACKAGEVERSION=22.08.1-26
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
    # Load to local cache
    astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //'')

    # Remove path and keep imageName.
    astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')

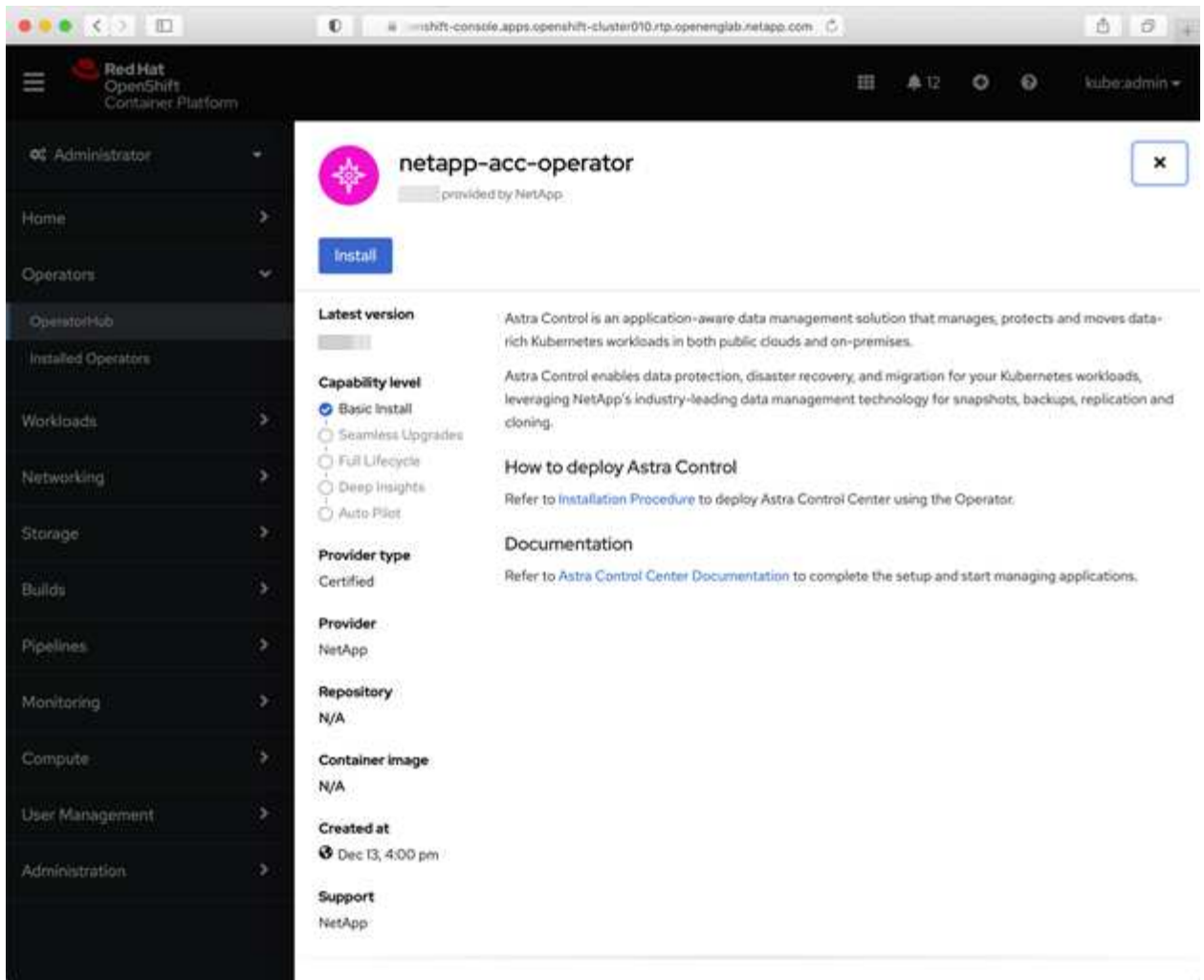
    # Tag with local image repo.
    podman tag ${astraImage} ${REGISTRY}/netapp/astra/${PACKAGENAME}
/${PACKAGEVERSION}/${astraImageNoPath}

    # Push to the local repo.
    podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done

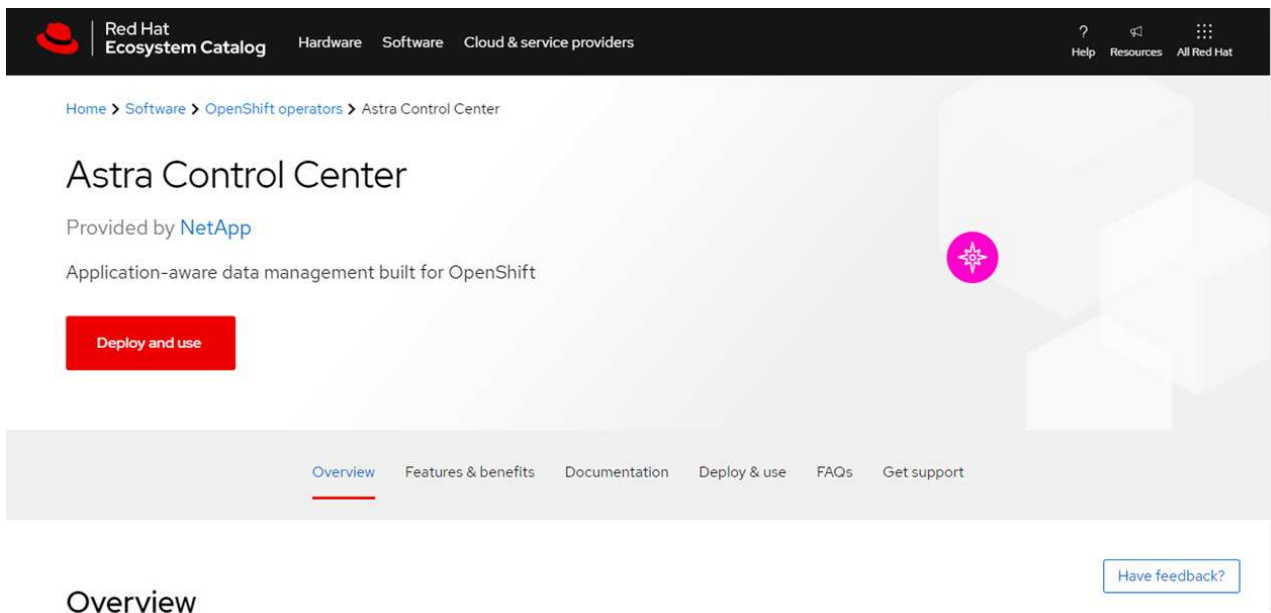
```

オペレータインストールページを検索します

1. 次のいずれかの手順を実行して、オペレータインストールページにアクセスします。
 - Red Hat OpenShift の Web コンソールから



- i. OpenShift Container Platform UI にログインします。
 - ii. サイドメニューから、* 演算子 > OperatorHub * を選択します。
 - iii. NetApp Astra Control Center オペレータを選択します。
 - iv. 「* Install *」を選択します。
- Red Hat エコシステムカタログから
：



- i. NetApp Astra Control Center を選択します "演算子".
- ii. [Deploy and Use] を選択します。

オペレータをインストールします

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



オペレータはすべてのクラスターネームスペースで使用できます。

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



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

- c. 「* Install *」 を選択します。



手動承認方式を選択した場合は、このオペレータの手動インストール計画を承認するように求められます。

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

Astra Control Center をインストールします

1. コンソールのAstra Control Centerオペレータの詳細ビューで、を選択します Create instance を参照してください。
2. を実行します Create AstraControlCenter フォームフィールド：
 - a. Astra Control Center の名前を保持または調整します。

- b. (オプション) AutoSupport を有効または無効にします。Auto Support 機能の保持を推奨します。
 - c. Astra Control Center のアドレスを入力します。入らないでください <http://> または <https://> をクリックします。
 - d. Astra Control Center のバージョンを入力します。たとえば、21.12.60 と入力します。
 - e. アカウント名、E メールアドレス、および管理者の姓を入力します。
 - f. デフォルトのボリューム再利用ポリシーをそのまま使用します。
 - g. * Image Registry * に、ローカルコンテナイメージのレジストリパスを入力します。入らないでください <http://> または <https://> をクリックします。
 - h. 認証が必要なレジストリを使用する場合は、シークレットを入力します。
 - i. 管理者の名を入力します。
 - j. リソースの拡張を構成する。
 - k. デフォルトのストレージクラスは保持します。
 - l. CRD 処理の環境設定を定義します。
3. 選択するオプション Create。

次のステップ

Astra Control Center が正しくインストールされたことを確認し、を完了します ["残りのステップ"](#) ログインしてください。さらに、の導入も完了します ["セットアップのタスク"](#)。

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

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

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

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

- [Amazon Web Services に Astra Control Center を導入](#)
- [Astra Control Center を Google Cloud Platform に導入](#)
- [Microsoft Azure に Astra Control Center を導入](#)

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

Amazon Web Services に Astra Control Center を導入

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

AWSに必要なもの

AWS に Astra Control Center を導入する前に、次のものがが必要です。

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

AWS の運用環境の要件

Astra Control Center を使用するには、AWS 向けに次の運用環境が必要です。

- Red Hat OpenShift Container Platform 4.8 の場合



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

Astra Control Center では、環境のリソース要件に加え、次のリソースが必要です。

コンポーネント	要件
バックエンドの NetApp Cloud Volumes ONTAP ストレージ容量	300GB 以上のデータがあります
ワーカーノード（ AWS EC2 の要件）	少なくとも 3 つのワーカーノードが必要です。vCPU コア 4 基、RAM はそれぞれ 12GB です
ロードバランサ	動作環境クラスタ内のサービスに送信される入力トラフィックに使用できるサービスタイプ「LoadBalancer」
FQDN	Astra Control Center の FQDN をロードバランシング IP アドレスに指定する方法
Astra Trident （ NetApp Cloud Manager の Kubernetes クラスタ検出の一部としてインストール）	Trident 21.04 以降がインストールおよび設定され、NetApp ONTAP バージョン 9.5 以降がストレージバックエンドとしてインストールされている必要があります

コンポーネント	要件
イメージレジストリ	<p>Astra Control Center のビルドイメージをプッシュできる、AWS Elastic Container Registry などの既存のプライベートレジストリが必要です。イメージをアップロードするイメージレジストリの URL を指定する必要があります。</p> <div>  <p>Restic ベースのイメージを使用してアプリケーションをバックアップおよび復元するには、Astra Control Center ホストクラスと管理対象クラスが同じイメージレジストリにアクセスする必要があります。</p> </div>
Astra Trident / ONTAP 構成	<p>Astra Control Center を使用するには、ストレージクラスを作成してデフォルトのストレージクラスとして設定する必要があります。Astra Control Center では、Kubernetes クラスタを NetApp Cloud Manager にインポートする際に作成される次の ONTAP Kubernetes ストレージクラスがサポートされます。Astra Trident によって提供される機能は次のとおりです。</p> <ul style="list-style-type: none"> • vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io • vsaworkingenvironment-<>-ha-san csi.trident.netapp.io • vsaworkingenvironment-<>-single-nas csi.trident.netapp.io • vsaworkingenvironment-<>-single-san csi.trident.netapp.io



これらの要件は、運用環境で実行されている唯一のアプリケーションが Astra Control Center であることを前提としています。環境で追加のアプリケーションを実行している場合は、それに応じてこれらの最小要件を調整します。



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

AWS の導入の概要を参照してください

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

これらの各手順については、以下で詳しく説明します。

1. [十分な IAM 権限があることを確認します。](#)
2. [AWS に Red Hat OpenShift クラスタをインストールします。](#)
3. [AWS を設定します。](#)
4. [NetApp Cloud Manager を設定します。](#)

5. [Astra Control Center](#) をインストールします。

十分な **IAM** 権限があることを確認します

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

を参照してください "[AWS の初期クレデンシャル](#)".

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

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

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

AWS を設定します

次に、仮想ネットワークの作成、EC2 コンピューティングインスタンスのセットアップ、AWS S3 バケットの作成、Astra Control Center イメージをホストする Elastic Container Register (ECR) の作成、このレジストリへのイメージのプッシュを行うように AWS を設定します。

AWS のドキュメントに従って次の手順を実行します。を参照してください "[AWS インストールドキュメント](#)".

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



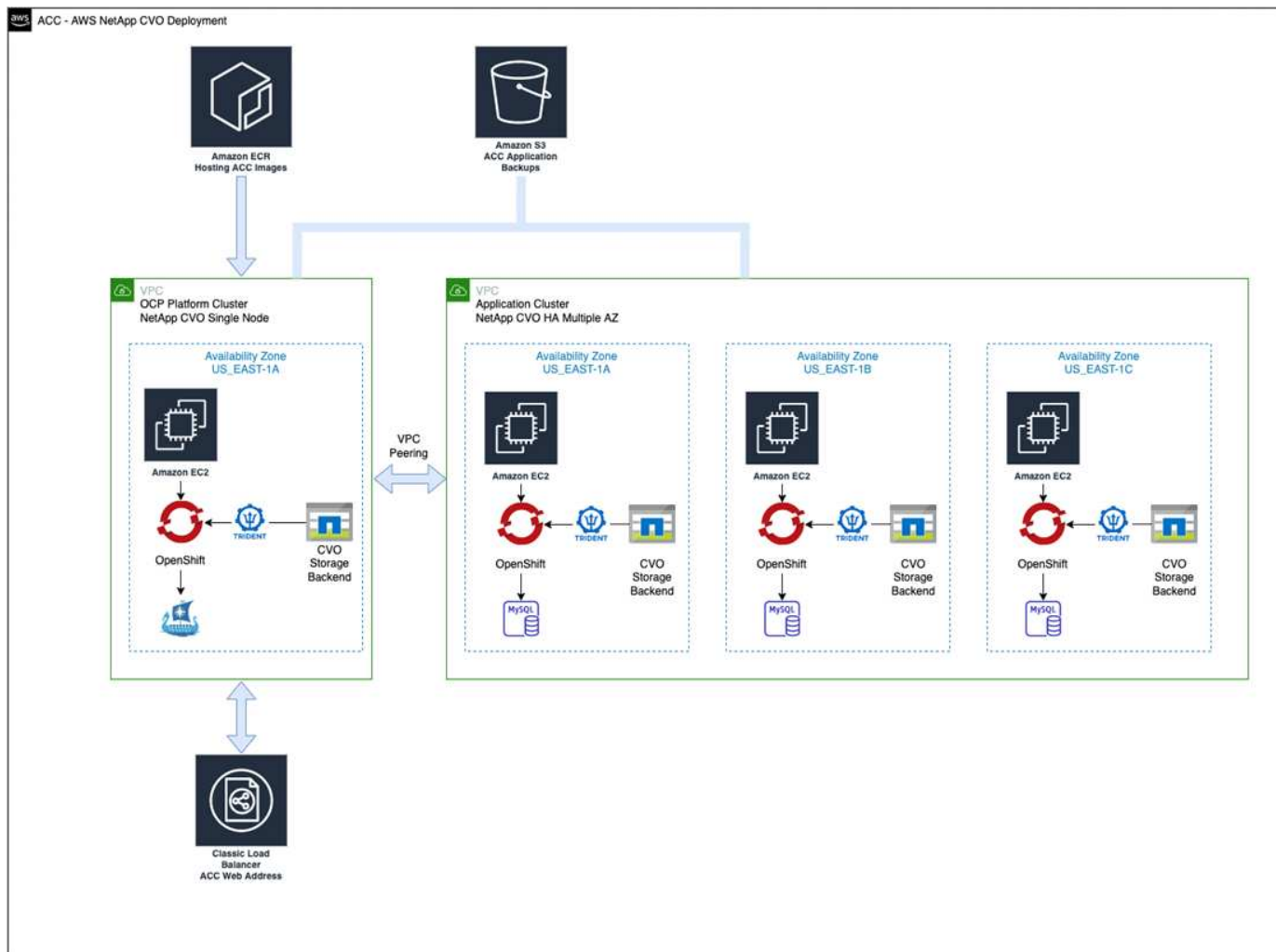
ECR を作成しないと、Astra Control Center は、AWS バックエンドを持つ Cloud Volumes ONTAP を含むクラスタからモニタリングデータにアクセスできません。問題は、Astra Control Center を使用して検出および管理しようとしたクラスタに AWS ECR アクセスがない場合に発生します。

6. ACC イメージを定義済みのレジストリにプッシュします。



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

AWS 環境の例を次に示します。



NetApp Cloud Manager を設定します

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

Cloud Manager のドキュメントに従って、次の手順を実行します。以下を参照してください。

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

手順

1. Cloud Manager にクレデンシャルを追加します。
2. ワークスペースを作成します。
3. AWS 用のコネクタを追加します。プロバイダとして AWS を選択します。
4. クラウド環境の作業環境を構築
 - a. 場所：「Amazon Web Services (AWS)」
 - b. 「Cloud Volumes ONTAP HA」と入力します。
5. OpenShift クラスタをインポートします。作成した作業環境にクラスタが接続されます。

- a. ネットアップクラスタの詳細を表示するには、* K8s * > * Cluster list * > * Cluster Details * を選択します。
- b. 右上隅に Trident のバージョンが表示されていることを確認します。
- c. Cloud Volumes ONTAP クラスタのストレージクラスは、プロビジョニングツールとしてネットアップを使用していることに注目してください。

これにより、Red Hat OpenShift クラスタがインポートされ、デフォルトのストレージクラスに割り当てられます。ストレージクラスを選択します。Trident は、インポートと検出のプロセスの一環として自動的にインストールされます。

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



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

Astra Control Center をインストールします

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



AWSでは汎用のS3バケットタイプが使用されます。

Astra Control CenterをGoogle Cloud Platformに導入

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

GCPに必要なもの

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

- Astra Control Center ライセンス。を参照してください ["Astra Control Center のライセンス要件"](#)。
- ["Astra Control Center の要件を満たす"](#)。
- NetApp Cloud Central アカウント
- OCPを使用している場合は、Red Hat OpenShift Container Platform（OCP）4.10
- OCPを使用する場合は、Red Hat OpenShift Container Platform（OCP）権限（ポッドを作成するためのネームスペースレベル）
- バケットとコネクタの作成を可能にする権限を持つGCPサービスアカウント

GCPの運用環境の要件



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

Astra Control Center では、環境のリソース要件に加え、次のリソースが必要です。

コンポーネント	要件
バックエンドの NetApp Cloud Volumes ONTAP ストレージ容量	300GB 以上のデータがあります
ワーカーノード (GCP コンピューティング要件)	少なくとも 3 つのワーカーノードが必要です。vCPU コア 4 基、RAM はそれぞれ 12GB です
ロードバランサ	動作環境クラスタ内のサービスに送信される入力トラフィックに使用できるサービスタイプ「LoadBalancer」
FQDN (GCP DNS ゾーン)	Astra Control Center の FQDN をロードバランシング IP アドレスに指定する方法
Astra Trident (NetApp Cloud Manager の Kubernetes クラスタ検出の一部としてインストール)	Trident 21.04 以降がインストールおよび設定され、NetApp ONTAP バージョン 9.5 以降がストレージバックエンドとしてインストールされている必要があります
イメージレジストリ	<p>Astra Control Centerビルドイメージをプッシュできる、Google Container Registryなどの既存のプライベートレジストリが必要です。イメージをアップロードするイメージレジストリの URL を指定する必要があります。</p> <div>  <p>バックアップ用にリストイメージを取得するには、匿名アクセスを有効にする必要があります。</p> </div>
Astra Trident / ONTAP 構成	<p>Astra Control Center を使用するには、ストレージクラスを作成してデフォルトのストレージクラスとして設定する必要があります。Astra Control Center では、Kubernetes クラスタを NetApp Cloud Manager にインポートする際に作成される次の ONTAP Kubernetes ストレージクラスがサポートされます。Astra Trident によって提供される機能は次のとおりです。</p> <ul style="list-style-type: none"> • vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io • vsaworkingenvironment-<>-ha-san csi.trident.netapp.io • vsaworkingenvironment-<>-single-nas csi.trident.netapp.io • vsaworkingenvironment-<>-single-san csi.trident.netapp.io



これらの要件は、運用環境で実行されている唯一のアプリケーションが Astra Control Center であることを前提としています。環境で追加のアプリケーションを実行している場合は、それに応じてこれらの最小要件を調整します。

GCPの導入の概要

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

これらの各手順については、以下で詳しく説明します。

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

GCPにRed Hat OpenShiftクラスタをインストールします

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

インストール手順については、次を参照してください。

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

GCPプロジェクトとVirtual Private Cloudを作成します

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



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

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

十分な **IAM** 権限があることを確認します

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

を参照してください ["GCPの初期資格情報と権限"](#)。

GCPを設定します

次に、VPCの作成、コンピューティングインスタンスのセットアップ、Google Cloud Object Storageの作成、Astra Control CenterイメージのホストにGoogle Container Registerの作成、このレジストリへのイメージのプッシュを行うようにGCPを設定します。

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

1. GCPでGCPプロジェクトとVPCを作成します。GCPでは、CVOバックエンドでOCPクラスタ用に使用する予定です。
2. コンピューティングインスタンスを確認します。GCP内のベアメタルサーバまたはVMです。
3. インスタンスタイプが、マスターノードとワーカーノードのAstra最小リソース要件と一致していない場合は、GCPでインスタンスタイプを変更してAstraの要件を満たします。を参照してください "[Astra Control Center の要件](#)"。
4. バックアップを保存するGCP Cloud Storageバケットを少なくとも1つ作成します。
5. バケットへのアクセスに必要なシークレットを作成します。
6. すべてのAstra Control CenterイメージをホストするGoogle Container Registryを作成します。
7. すべてのAstra Control Centerイメージに対して、Dockerプッシュ/プル用のGoogle Container Registryアクセスを設定します。

例：次のスクリプトを入力すると、ACCイメージをこのレジストリにプッシュできます。

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

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

例

```
manifestfile=astraccontrol-center-<version>.manifest
GCP_CR_REGISTRY=<target image repository>
ASTRA_REGISTRY=<source ACC image repository>

while IFS= read -r image; do
    echo "image: $ASTRA_REGISTRY/$image $GCP_CR_REGISTRY/$image"
    root_image=${image%:*}
    echo $root_image
    docker pull $ASTRA_REGISTRY/$image
    docker tag $ASTRA_REGISTRY/$image $GCP_CR_REGISTRY/$image
    docker push $GCP_CR_REGISTRY/$image
done < astraccontrol-center-22.04.41.manifest
```

8. DNS ゾーンを設定します。

NetApp Cloud Manager を設定します

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

Cloud Manager のドキュメントに従って、次の手順を実行します。を参照してください "[GCPでCloud Volumes ONTAP の使用を開始する](#)"。

必要なもの

- 必要なIAM権限と役割を持つGCPサービスアカウントにアクセスします

手順

1. Cloud Manager にクレデンシャルを追加します。を参照してください ["GCP アカウントの追加"](#)。
2. GCPのコネクタを追加します。
 - a. プロバイダーとして[GCP]を選択します。
 - b. GCP資格情報を入力します。を参照してください ["Cloud ManagerからGCPでコネクタを作成する"](#)。
 - c. コネクタが動作していることを確認し、コネクタに切り替えます。
3. クラウド環境の作業環境を構築
 - a. 場所: "GCP"
 - b. 「Cloud Volumes ONTAP HA」と入力します。
4. OpenShift クラスタをインポートします。作成した作業環境にクラスタが接続されます。
 - a. ネットアップクラスタの詳細を表示するには、* K8s * > * Cluster list * > * Cluster Details * を選択します。
 - b. 右上隅に Trident のバージョンが表示されていることを確認します。
 - c. Cloud Volumes ONTAP クラスタのストレージクラスは、プロビジョニングツールとして「ネットアップ」を使用していることに注目してください。

これにより、Red Hat OpenShift クラスタがインポートされ、デフォルトのストレージクラスに割り当てられます。ストレージクラスを選択します。Trident は、インポートと検出のプロセスの一環として自動的にインストールされます。
5. このCloud Volumes ONTAP 環境内のすべての永続ボリュームとボリュームをメモします。



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

Astra Control Center をインストールします

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



GCPでは汎用S3バケットタイプが使用されます。

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

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

Microsoft Azure に Astra Control Center を導入

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

Azureに必要なもの

Azure に Astra Control Center を導入する前に、次のものがが必要です。

- Astra Control Center ライセンス。を参照してください ["Astra Control Center のライセンス要件"](#)。
- ["Astra Control Center の要件を満たす"](#)。
- NetApp Cloud Central アカウント
- OCPを使用する場合、Red Hat OpenShift Container Platform（OCP）4.8
- OCPを使用する場合は、Red Hat OpenShift Container Platform（OCP）権限（ポッドを作成するためのネームスペースレベル）
- バケットとコネクタの作成を可能にする権限を持つ Azure クレデンシャル

Azure の運用環境の要件

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

Astra Control Center では、環境のリソース要件に加え、次のリソースが必要です。

を参照してください ["Astra Control Center の運用環境要件"](#)。

コンポーネント	要件
バックエンドの NetApp Cloud Volumes ONTAP ストレージ容量	300GB 以上のデータがあります
ワーカーノード（ Azure コンピューティング要件）	少なくとも 3 つのワーカーノードが必要です。vCPU コア 4 基、RAM はそれぞれ 12GB です
ロードバランサ	動作環境クラスタ内のサービスに送信される入力トラフィックに使用できるサービスタイプ「LoadBalancer」
FQDN （ Azure DNS ゾーン）	Astra Control Center の FQDN をロードバランシング IP アドレスに指定する方法
Astra Trident （ NetApp Cloud Manager の Kubernetes クラスタ 検出の一部としてインストール）	Trident 21.04 以降がインストールおよび設定され、NetApp ONTAP バージョン 9.5 以降がストレージバックエンドとして使用されます

コンポーネント	要件
イメージレジストリ	<p>Astra Control Center ビルドイメージをプッシュできる、Azure Container Registry（ACR）などの既存のプライベートレジストリが必要です。イメージをアップロードするイメージレジストリの URL を指定する必要があります。</p> <div>  <p>バックアップ用にリストイメージを取得するには、匿名アクセスを有効にする必要があります。</p> </div>
Astra Trident / ONTAP 構成	<p>Astra Control Center を使用するには、ストレージクラスを作成してデフォルトのストレージクラスとして設定する必要があります。Astra Control Center では、Kubernetes クラスタを NetApp Cloud Manager にインポートする際に作成される次の ONTAP Kubernetes ストレージクラスがサポートされます。Astra Trident によって提供される機能は次のとおりです。</p> <ul style="list-style-type: none"> • vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io • vsaworkingenvironment-<>-ha-san csi.trident.netapp.io • vsaworkingenvironment-<>-single-nas csi.trident.netapp.io • vsaworkingenvironment-<>-single-san csi.trident.netapp.io



これらの要件は、運用環境で実行されている唯一のアプリケーションが Astra Control Center であることを前提としています。環境で追加のアプリケーションを実行している場合は、それに応じてこれらの最小要件を調整します。

Azure の導入の概要

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

これらの各手順については、以下で詳しく説明します。

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

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

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

インストール手順については、のRedHatのマニュアルを参照してください ["AzureにOpenShiftクラスタをインストールしています"](#) および ["Azureアカウントをインストールしています"](#)。

Azure リソースグループを作成する

Azure リソースグループを少なくとも 1 つ作成します。



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

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

十分な **IAM** 権限があることを確認します

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

を参照してください ["Azure のクレデンシャルと権限"](#)。

Azure を設定

次に、仮想ネットワークの作成、コンピューティングインスタンスのセットアップ、Azure Blobコンテナの作成、Astra Control CenterイメージをホストするAzure Container Register (ACR) の作成、このレジストリへのイメージのプッシュを行うようにAzureを設定します。

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

1. Azure Virtual Networkの作成
2. コンピューティングインスタンスを確認します。Azure の場合、ベアメタルサーバまたは VM を使用できます。
3. インスタンスタイプがまだマスターノードとワーカーノードの Astra 最小リソース要件に一致していない場合は、Azure でインスタンスタイプを変更して Astra の要件を満たします。を参照してください ["Astra Control Center の要件"](#)。
4. バックアップを格納するAzure BLOBコンテナを少なくとも1つ作成します。
5. ストレージアカウントを作成します。Astra Control Center でバケットとして使用するコンテナを作成するには、ストレージアカウントが必要です。
6. バケットへのアクセスに必要なシークレットを作成します。
7. Azure Container Registry (ACR) を作成して、すべての Astra Control Center イメージをホストします。
8. ACR アクセスを設定して Docker プッシュ / プルをすべての Astra Control Center イメージに適用します。
9. 次のスクリプトを入力して、ACC イメージをこのレジストリにプッシュします。


```
az acr login -n <AZ ACR URL/Location>
```

This script requires ACC manifest file and your Azure ACR location.

◦ 例 * :

```
manifestfile=astra-control-center-<version>.manifest
AZ_ACR_REGISTRY=<target image repository>
ASTRA_REGISTRY=<source ACC image repository>

while IFS= read -r image; do
    echo "image: $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRY/$image"
    root_image=${image%:*}
    echo $root_image
    docker pull $ASTRA_REGISTRY/$image
    docker tag $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRY/$image
    docker push $AZ_ACR_REGISTRY/$image
done < astra-control-center-22.04.41.manifest
```

10. DNS ゾーンを設定します。

NetApp Cloud Manager を設定します

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

Cloud Manager のドキュメントに従って、次の手順を実行します。を参照してください ["Azure で Cloud Manager を使用する準備をしています"](#)。

必要なもの

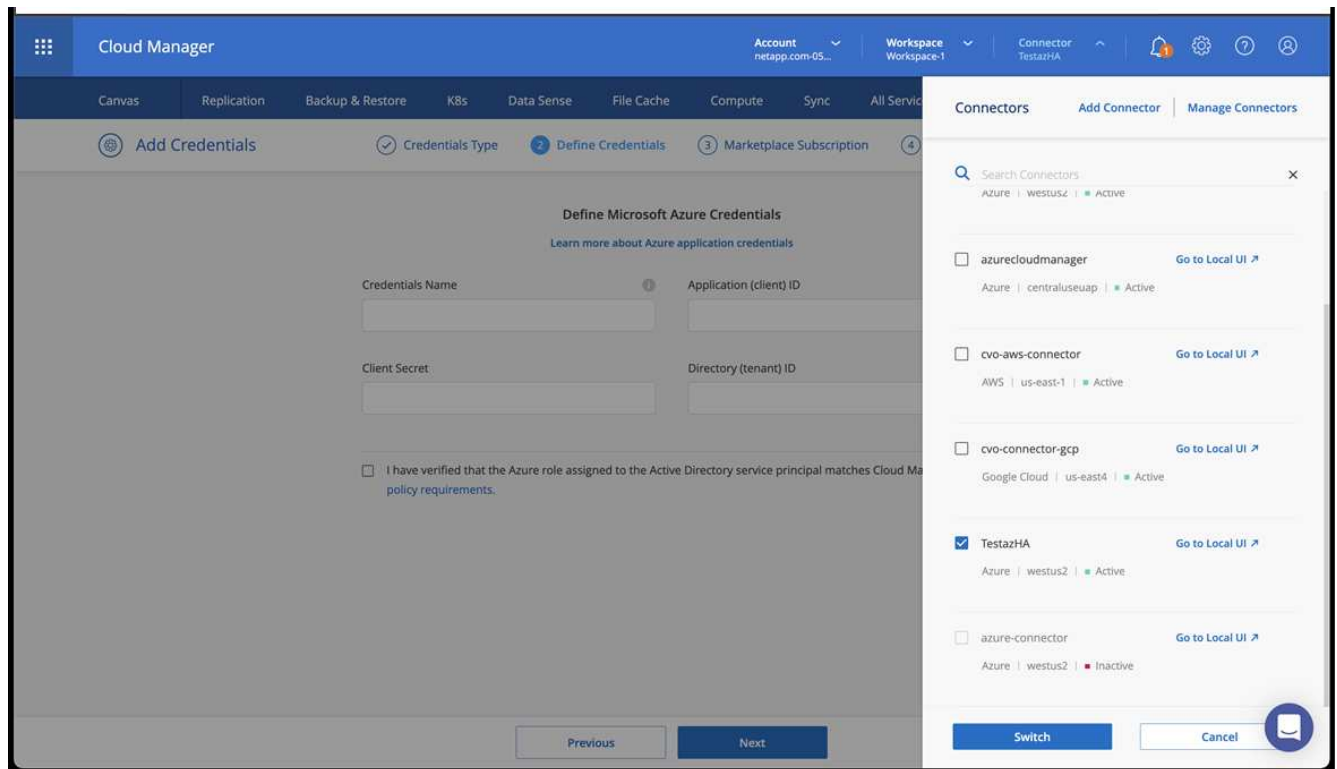
必要な IAM 権限とロールを持つ Azure アカウントにアクセスします

手順

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

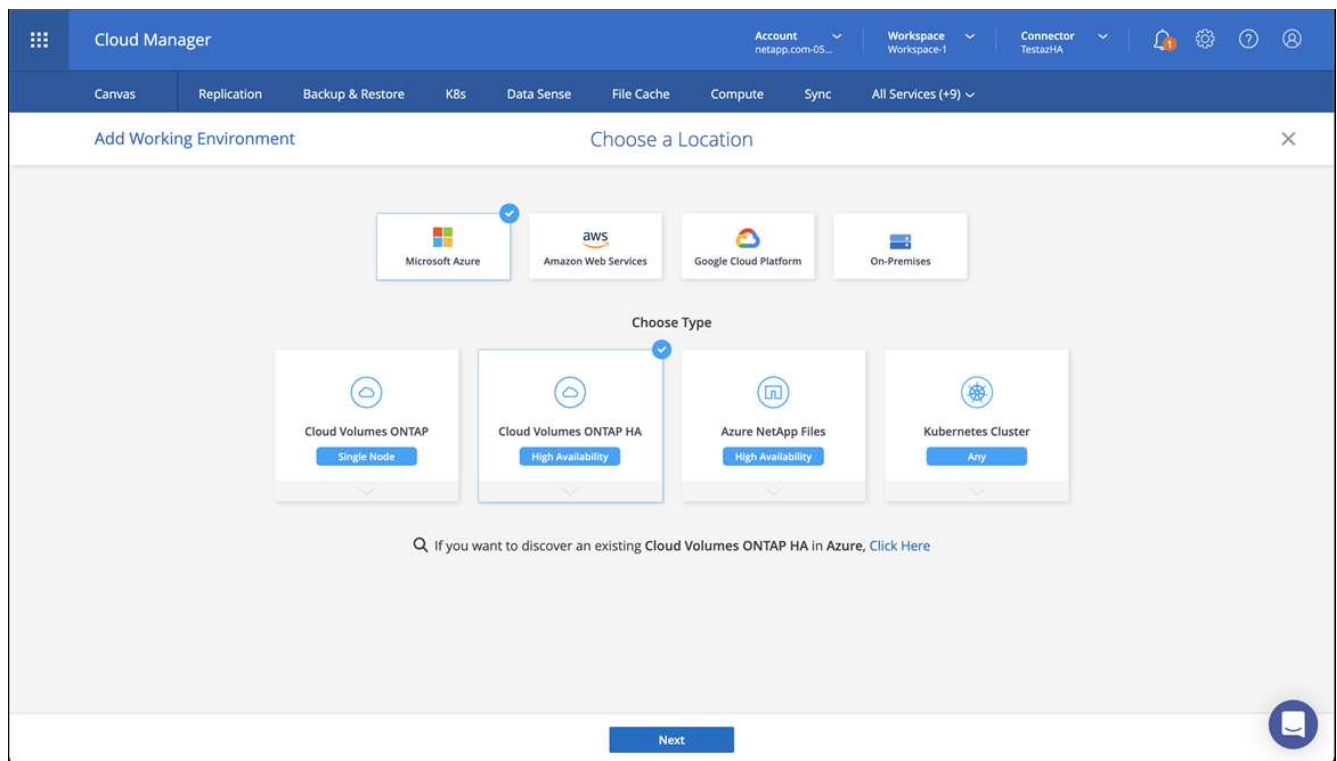
を参照してください ["Cloud Manager から Azure にコネクタを作成する"](#)。

3. コネクタが動作していることを確認し、コネクタに切り替えます。



4. クラウド環境の作業環境を構築

- a. 場所：「Microsoft Azure」。
- b. 「Cloud Volumes ONTAP HA」と入力します。



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

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

The screenshot shows the 'Cluster Details' page in the Cloud Manager. The top navigation bar includes 'Account', 'Workspace', and 'Connector'. The main navigation bar has tabs for 'Canvas', 'Replication', 'Backup & Restore', 'K8s', 'Data Sense', 'File Cache', 'Compute', 'Sync', and 'All Services (+9)'. The 'K8s' tab is selected, and the 'Cluster List' > 'Cluster Details' breadcrumb is visible. The cluster name 'targetazacc' is at the top, with buttons for 'Update Kubeconfig' and 'Connect to Working Environment'. Below this, a summary card shows: Status: Running (green checkmark), Cluster Version: v1.21.6+bb8d50a, Added by: Import, Volumes: 3, VPC: -, Date Added: April 14, 2022, Trident Version: v21.04.1, and Provider: Microsoft Azure. A section titled '1 Working Environments' contains a table with columns: Name, Provider, Region, Zone, Subnet, and Capacity. It lists one environment: 'testHAenvaz HA' with Provider 'Microsoft Azure', Region 'westus2', Subnet '10.0.0.0/16', and Capacity '0.00 used of 500 GB available'. Below this, a section titled '3 Storage Classes' contains a table with columns: Storage Class ID, Provisioner, Volumes, and Labels. It lists two storage classes: 'managed-premium' with Provisioner 'Microsoft Azure' and 0 volumes, and 'vsaworkingenvironment-xr1hs5pd-ha-nas' with Provisioner 'NetApp' and 3 volumes. The 'vsaworkingenvironment-xr1hs5pd-ha-nas' class is marked as 'Default'. A tooltip for this class shows labels: 'trident.netapp.io/backend=Vsaworkingenvironment-xr1hs5pd-ha', 'trident.netapp.io/ha=true', and 'trident.netapp.io/protocol=NAS'. The footer shows 'Cloud Manager 3.9.17 Build: 2 Apr 12, 2022 03:04:23 pm UTC'.

- b. 右上隅に Trident のバージョンが表示されていることを確認します。
- c. Cloud Volumes ONTAP クラスタのストレージクラスは、プロビジョニングツールとしてネットアップを使用していることに注目してください。

これにより、Red Hat OpenShift クラスタがインポートされ、デフォルトのストレージクラスが割り当てられます。ストレージクラスを選択します。Trident は、インポートと検出のプロセスの一環として自動的にインストールされます。

6. このCloud Volumes ONTAP 環境内のすべての永続ボリュームとボリュームをメモします。
7. Cloud Volumes ONTAP は、シングルノードまたはハイアベイラビリティとして動作できます。HA が有効になっている場合は、Azure で実行されている HA ステータスとノード導入ステータスを確認します。

Astra Control Center をインストールして設定します

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

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

著作権に関する情報

Copyright © 2023 NetApp, Inc. All Rights Reserved. Printed in the U.S. このドキュメントは著作権によって保護されています。著作権所有者の書面による事前承諾がある場合を除き、画像媒体、電子媒体、および写真複写、記録媒体、テープ媒体、電子検索システムへの組み込みを含む機械媒体など、いかなる形式および方法による複製も禁止します。

ネットアップの著作物から派生したソフトウェアは、次に示す使用許諾条項および免責条項の対象となります。

このソフトウェアは、ネットアップによって「現状のまま」提供されています。ネットアップは明示的な保証、または商品性および特定目的に対する適合性の暗示的保証を含み、かつこれに限定されないいかなる暗示的な保証も行いません。ネットアップは、代替品または代替サービスの調達、使用不能、データ損失、利益損失、業務中断を含み、かつこれに限定されない、このソフトウェアの使用により生じたすべての直接的損害、間接的損害、偶発的損害、特別損害、懲罰的損害、必然的損害の発生に対して、損失の発生の可能性が通知されていたとしても、その発生理由、根拠とする責任論、契約の有無、厳格責任、不法行為（過失またはそうでない場合を含む）にかかわらず、一切の責任を負いません。

ネットアップは、ここに記載されているすべての製品に対する変更を随時、予告なく行う権利を保有します。ネットアップによる明示的な書面による合意がある場合を除き、ここに記載されている製品の使用により生じる責任および義務に対して、ネットアップは責任を負いません。この製品の使用または購入は、ネットアップの特許権、商標権、または他の知的所有権に基づくライセンスの供与とはみなされません。

このマニュアルに記載されている製品は、1つ以上の米国特許、その他の国の特許、および出願中の特許によって保護されている場合があります。

権利の制限について：政府による使用、複製、開示は、DFARS 252.227-7013（2014年2月）およびFAR 5252.227-19（2007年12月）のRights in Technical Data -Noncommercial Items（技術データ - 非商用品目に関する諸権利）条項の(b)(3)項、に規定された制限が適用されます。

本書に含まれるデータは商用製品および / または商用サービス（FAR 2.101の定義に基づく）に関係し、データの所有権はNetApp, Inc.にあります。本契約に基づき提供されるすべてのネットアップの技術データおよびコンピュータ ソフトウェアは、商用目的であり、私費のみで開発されたものです。米国政府は本データに対し、非独占的かつ移転およびサブライセンス不可で、全世界を対象とする取り消し不能の制限付き使用权を有し、本データの提供の根拠となった米国政府契約に関連し、当該契約の裏付けとする場合にのみ本データを使用できます。前述の場合を除き、NetApp, Inc.の書面による許可を事前に得ることなく、本データを使用、開示、転載、改変するほか、上演または展示することはできません。国防総省にかかる米国政府のデータ使用权については、DFARS 252.227-7015(b)項（2014年2月）で定められた権利のみが認められます。

商標に関する情報

NetApp、NetAppのロゴ、<http://www.netapp.com/TM>に記載されているマークは、NetApp, Inc.の商標です。その他の会社名と製品名は、それを所有する各社の商標である場合があります。