安裝Astra Control Center

Astra Control Center

NetApp
November 21, 2023

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

目錄

使用標準程序安裝Astra Control Center

若要安裝 Astra Control Center，請從 NetApp 支援網站下載安裝套件，然後執行下列步驟，在您的環境中安裝 Astra Control Center Operator 和 Astra Control Center。您可以使用此程序、在連線網際網路或無線環境中安裝Astra Control Center。

對於Red Hat OpenShift環境、您可以使用 "替代程序" 使用OpenShift作業系統集線器安裝Astra Control Center。

您需要的產品

- "開始安裝之前、請先準備好環境以進行Astra Control Center部署"。
- 如果您已設定或想要在環境中設定Pod安全性原則、請熟悉Pod安全性原則、以及這些原則如何影響Astra Control Center安裝。請參閱 "瞭解Pod安全性原則限制"。
- 確保所有叢集操作員都處於健全狀態且可用。

```
kubectl get clusteroperators
```

- 確保所有API服務均處於健全狀態且可供使用：

```
kubectl get apiservices
```

- 確保您打算使用的Astra FQDN可路由傳送至此叢集。這表示您在內部DNS伺服器中有DNS項目、或是使用已註冊的核心URL路由。
- 如果叢集中已存在認證管理程式、您需要執行某些作業 "必要步驟" 因此Astra Control Center不會安裝自己的認證管理程式。

關於這項工作

Astra Control Center安裝程序會執行下列作業：

- 將Astra元件安裝至 `netapp-acc` （或自訂命名）命名空間。
- 建立預設帳戶。
- 建立預設的管理使用者電子郵件地址和預設的一次性密碼。此使用者在系統中被指派第一次登入UI所需的擁有者角色。
- 協助您判斷所有Astra Control Center Pod都在執行中。
- 安裝Astra UI。

> （僅適用於Astra Data Store Early Access Program（EAP）版本）如果您打算使用Astra Control Center管理Astra Data Store並啟用VMware工作流程、請僅在上部署Astra Control Center `pcloud` 命名空間而非 `netapp-acc` 命名空間或自訂命名空間、請參閱本程序步驟。

> 請勿在整個安裝程序期間執行下列命令、以免刪除所有Astra Control Center Pod： `kubectl delete -f astra_control_center_operator_deploy.yaml`

如果您使用的是Red Hat的Podman而非Docker Engine、則可以使用Podman命令來取代Docker命令。

步驟

若要安裝Astra Control Center、請執行下列步驟：

- 下載並解壓縮Astra Control Center套裝組合
- 安裝NetApp Astra kubecl外掛程式
- [將映像新增至本機登錄]
- [設定具有驗證需求之登錄的命名空間和機密]
- 安裝Astra Control Center操作員
- 設定Astra控制中心
- 完整的Astra控制中心和操作員安裝
- [驗證系統狀態]
- [設定入口以進行負載平衡]
- 登入Astra Control Center UI

下載並解壓縮Astra Control Center套裝組合

1. 下載Astra Control Center套裝組合 (`astra-control-center-[version].tar.gz`）從 "NetApp 支援網站"。
2. 從下載Astra Control Center認證與金鑰的壓縮檔 "NetApp 支援網站"。
3. （可選）使用以下命令驗證套件的簽名：

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

4. 擷取影像：

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

安裝NetApp Astra kubecl外掛程式

NetApp Astra `kubectl` 命令列外掛程式可在執行與部署及升級Astra Control Center相關的一般工作時節省時間。

您需要的產品

NetApp為不同的CPU架構和作業系統提供外掛程式的二進位檔。執行此工作之前、您必須先瞭解您的CPU和作業系統。在Linux和Mac作業系統上、您可以使用 `uname -a` 命令來收集此資訊。

步驟

1. 列出可用的NetApp Astra `kubectl` 外掛程式二進位檔、並記下作業系統和CPU架構所需的檔案名稱：

```
ls kubectl-astra/
```

2. 將檔案複製到與標準相同的位置 `kubectl` 公用程式：在此範例中 `kubectl` 公用程式位於 `/usr/local/bin` 目錄。更換 `<binary-name>` 使用您需要的檔案名稱：

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

將映像新增至本機登錄

1. 為您的Container引擎完成適當的步驟順序：

Docker

1. 變更至Astra目錄：

```
cd acc
```

2. [Subforte_image_local_register_push]將Astra Control Center映像目錄中的套件映像推送到本機登錄。執行命令之前、請先進行下列替代：
   - 以Astra Control套件檔案名稱取代bunder_file（例如、 `acc.manifest.yaml`）。
   - 將my_registry取代為Docker儲存庫的URL。
   - 以使用者名稱取代my_register_user。
   - 以登錄的授權權杖取代my_register_token。

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

Podman

1. 登入您的登錄：

```
podman login [your_registry_path]
```

2. 執行下列指令碼、依照註解中的說明進行<your _inforation>替換：

```
# 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
```

設定具有驗證需求之登錄的命名空間和機密

1. 匯出Astra Control Center主機叢集的KUBECCONFIG：

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

2. 如果您使用需要驗證的登錄、則需要執行下列動作：

   a. 建立 `netapp-acc-operator` 命名空間：

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

   回應：

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

b. 為建立秘密 `netapp-acc-operator` 命名空間。新增Docker資訊並執行下列命令：

> 預留位置 `your_registry_path` 應與您先前上傳的影像位置相符（例如、`[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]
```

回應範例：

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

> 如果您在產生機密之後刪除命名空間、則需要在重新建立命名空間之後重新產生命名空間的機密。

c. 建立 `netapp-acc` （或自訂命名）命名空間。

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

回應範例：

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

d. 為建立秘密 `netapp-acc` （或自訂命名）命名空間。新增Docker資訊並執行下列命令：

```
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]
```

回應

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

a. [Substete_kubeconfig _secret]（選用）如果您希望叢集在安裝後由Astra Control Center自動管理、請確定您在Astra Control Center命名空間中提供了要使用此命令部署的kubeconfig作為機密：

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

安裝Astra Control Center操作員

1. 變更目錄：

```
cd manifests
```

2. 編輯Astra Control Center營運者部署Yaml (`astra_control_center_operator_deploy.yaml`）以參考您的本機登錄和機密。

```
vim astra_control_center_operator_deploy.yaml
```

> 附註的Y反 洗錢範例遵循下列步驟。

a. 如果您使用需要驗證的登錄、請取代的預設行 `imagePullSecrets: []` 提供下列功能：

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

b. 變更 `[your_registry_path]` 適用於 `kube-rbac-proxy` 映像到您在中推入映像的登錄路徑 上一步。

c. 變更 `[your_registry_path]` 適用於 `acc-operator-controller-manager` 映像到您在中推入映像的登錄路徑 上一步。

d. （若為使用Astra Data Store預覽的安裝）請參閱此已知問題 "儲存類別資源配置工具、以及您需要對Y反 洗錢進行的其他變更"。

```
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. 安裝Astra Control Center操作員：

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

回應範例：

```
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. 確認Pod正在執行：

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

設定Astra控制中心

1. 編輯Astra Control Center自訂資源（CR）檔案 (`astra_control_center_min.yaml`）進行帳戶、AutoSupport 功能、登錄及其他必要的設定：

> `astra_control_center_min.yaml` 為預設的CR、適用於大多數安裝。請熟悉所有資訊 "CR選項及其潛在價值" 確保您的環境正確部署Astra Control Center。如果您的環境需要額外的自訂功能、您可以使用 `astra_control_center.yaml` 作為替代的CR.

```
vim astra_control_center_min.yaml
```

> 如果您使用的登錄不需要授權、則必須刪除 `secret` 行內 `imageRegistry` 否則安裝將會失敗。

a. 變更 `[your_registry_path]` 移至您在上一個步驟中推送映像的登錄路徑。

b. 變更 `accountName` 字串至您要與帳戶建立關聯的名稱。

c. 變更 `astraAddress` 字串至您要在瀏覽器中用來存取Astra的FQDN。請勿使用 `http://` 或 `https://` 地址中。複製此FQDN以供在中使用 後續步驟。

d. 變更 `email` 字串至預設的初始系統管理員位址。複製此電子郵件地址以供在中使用 後續步驟。

e. 變更 `enrolled` for解決方案AutoSupport `false` 適用於沒有網際網路連線或無法保留的網站 `true` 適用於連線站台。

f. 如果您使用外部cert管理程式、請將下列行新增至 `spec`：

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

g. （選用）新增名字 `firstName` 和姓氏 `lastName` 與帳戶相關聯的使用者。您可以在UI中立即或稍後執行此步驟。

h. （選用）變更 `storageClass` 如果您的安裝需要、請將其值加到另一個Trident storageClass資源。

i. （選用）如果您希望叢集在安裝後由Astra Control Center自動管理、而且您已經擁有 已建立包含此叢集之Kbeconfig的秘密下、將新欄位新增至名為的Yaml檔案、以提供機密名稱 `astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"`

j. 請完成下列其中一個步驟：

- 其他入侵控制器（擷取類型：一般）：這是Astra控制中心的預設動作。部署Astra Control Center之後、您需要設定入口控制器、以URL顯示Astra Control Center。

  預設的Astra Control Center安裝會設定其閘道 (`service/traefik`）的類型 `ClusterIP`。此預設安裝需要您額外設定Kubernetes IngresController / Ingress、才能將流量路由傳送至該控制器。如果您想要使用入口、請參閱 "設定入口以進行負載平衡"。

- 服務負載平衡器（擷取類型：**AccTraefik**）：如果您不想安裝IngressController或建立Ingress資源、請設定 `ingressType` 至 `AccTraefik`。

  這會部署Astra控制中心 `traefik` 作為Kubernetes負載平衡器類型服務的閘道。

  Astra Control Center使用「負載平衡器」類型的服務 (`svc/traefik` （在Astra Control Center命名空間中）、並要求指派可存取的外部IP位址。如果您的環境允許負載平衡器、但您尚未設定負載平衡器、則可以使用MetalLB或其他外部服務負載平衡器、將外部IP位址指派給服務。在內部DNS伺服器組態中、您應該將Astra Control Center所選的DNS名稱指向負載平衡的IP位址。

  ⓘ 如需有關「負載平衡器」和入口服務類型的詳細資訊、請參閱 "需求"。

```
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"
```

完整的Astra控制中心和操作員安裝

1. 如果您尚未在上一步中執行此動作、請建立 `netapp-acc` （或自訂）命名空間：

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

回應範例：

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

2. 在中安裝Astra Control Center `netapp-acc` （或自訂）命名空間：

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

回應範例：

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

驗證系統狀態

如果您偏好使用OpenShift、您可以使用相似的相關命令來進行驗證步驟。

1. 驗證是否已成功安裝所有系統元件。

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

每個Pod的狀態應為 `Running`。部署系統Pod可能需要幾分鐘的時間。

回應範例

```
NAME                                        READY   STATUS    RESTARTS
AGE
acc-helm-repo-6b44d68d94-d8m55              1/1     Running   0
13m
activity-78f99ddf8-hltct                    1/1     Running   0
10m
api-token-authentication-457nl              1/1     Running   0
9m28s
api-token-authentication-dgwsz              1/1     Running   0
9m28s
api-token-authentication-hmqqc              1/1     Running   0
9m28s
asup-75fd554dc6-m6qzh                       1/1     Running   0
9m38s
authentication-6779b4c85d-92gds             1/1     Running   0
8m11s
bucketservice-7cc767f8f8-lqwr8              1/1     Running   0
9m31s
certificates-549fd5d6cb-5kmd6               1/1     Running   0
9m56s
certificates-549fd5d6cb-bkjh9               1/1     Running   0
9m56s
cloud-extension-7bcb7948b-hn8h2             1/1     Running   0
10m
cloud-insights-service-56ccf86647-fgg69     1/1     Running   0
9m46s
composite-compute-677685b9bb-7vgsf          1/1     Running   0
10m
composite-volume-657d6c5585-dnq79           1/1     Running   0
9m49s
credentials-755fd867c8-vrlmt                1/1     Running   0
11m
entitlement-86495cdf5b-nwhh2                1/1     Running   2
10m
features-5684fb8b56-8d6s8                   1/1     Running   0
10m
fluent-bit-ds-rhx7v                         1/1     Running   0
7m48s
fluent-bit-ds-rjms4                         1/1     Running   0
7m48s
fluent-bit-ds-zf5ph                         1/1     Running   0
7m48s
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-vmpsm                  1/1     Running   0
10m
traefik-7489dc59f9-6mnst                  1/1     Running   0
3m19s
traefik-7489dc59f9-xrkgg                  1/1     Running   0
3m4s
trident-svc-6c8dc458f5-jswcl              1/1     Running   0
10m
vault-controller-6b954f9b76-gz9nm         1/1     Running   0
11m
```

2. （選用）若要確保安裝完成、您可以觀看 `acc-operator` 使用下列命令記錄。

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

> `accHost` 叢集登錄是最後一項作業、如果失敗、也不會導致部署失敗。如果記錄中指出叢集登錄失敗、您可以透過新增叢集工作流程再次嘗試登錄 "在UI中" 或API。

3. 當所有Pod都在執行時、請確認安裝成功 (`READY` 是 `True`）並取得登入Astra Control Center時使用的一次性密碼：

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

回應：

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

> 複製UUID值。密碼是 `ACC-` 接著是UUID值 (`ACC-[UUID]` 或者、在此範例中、 `ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f`）。

設定入口以進行負載平衡

您可以設定Kubernetes入口控制器來管理外部服務存取、例如叢集中的負載平衡。

本程序說明如何設定入口控制器 (`ingressType:Generic`）。這是Astra Control Center的預設動作。部署Astra Control Center之後、您需要設定入口控制器、以URL顯示Astra Control Center。

> 如果您不想設定入口控制器、可以設定 `ingressType:AccTraefik`)。Astra Control Center使用「負載平衡器」類型的服務 (`svc/traefik` （在Astra Control Center命名空間中）、並要求指派可存取的外部IP位址。如果您的環境允許負載平衡器、但您尚未設定負載平衡器、則可以使用MetalLB或其他外部服務負載平衡器、將外部IP位址指派給服務。在內部DNS伺服器組態中、您應該將Astra Control Center所選的DNS名稱指向負載平衡的IP位址。如需有關「負載平衡器」和入口服務類型的詳細資訊、請參閱 "需求"。

這些步驟會因您使用的入口控制器類型而有所不同：

- Istio入口
- Nginx入口控制器
- OpenShift入口控制器

您需要的產品

- 必要的 "入口控制器" 應已部署。
- 。 "入口等級" 應已建立對應於入口控制器的。
- 您使用的Kubernetes版本介於v1.19和v1.22之間、甚至包括在內。

Istio入侵步驟

1. 設定Istio入口。

   此程序假設使用「預設」組態設定檔來部署Istio。

2. 收集或建立Ingress閘道所需的憑證和私密金鑰檔案。

   您可以使用CA簽署或自我簽署的憑證。一般名稱必須是Astra位址（FQDN）。

   命令範例：

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

3. 建立秘密 `tls secret name` 類型 `kubernetes.io/tls` 中的TLS私密金鑰和憑證 `istio-system namespace` 如TLS機密所述。

   命令範例：

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

   機密名稱應與相符 `spec.tls.secretName` 提供於 `istio-ingress.yaml` 檔案：

4. 在中部署入口資源 `netapp-acc` （或自訂命名）命名空間、使用v1beta1（Kubernetes版本低於或1.22的版本已過時）或v1資源類型來取代過時的或新的架構：

   輸出：

```yaml
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 addess>
    secretName: [tls secret name]
  rules:
  - host: [ACC addess]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          serviceName: traefik
          servicePort: 80
```

如需v1新架構、請遵循下列範例：

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

輸出：

```
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 addess>
    secretName: [tls secret name]
  rules:
  - host: [ACC addess]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: traefik
            port:
              number: 80
```

5. 如常部署Astra Control Center。
6. 檢查入侵狀態：

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

回應：

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

適用於**Nginvin**像 控制器的步驟

1. 建立類型的秘密[`kubernetes.io/tls`]用於中的TLS私密金鑰和憑證 `netapp-acc` （或自訂命名）命名空

間、如所述 "TLS機密"。

2. 在中部署入口資源 `netapp-acc` （或自訂命名）命名空間、使用任一項 `v1beta1` （在Kubernetes版本低於或1.22的版本中已過時）或 `v1` 過時或新架構的資源類型：

   a. 適用於 `v1beta1` 過時的架構、請遵循以下範例：

```
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. 適用於 `v1` 新架構、請遵循以下範例：

```yaml
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 addess>
    http:
      paths:
        - path:
          backend:
            service:
              name: traefik
              port:
                number: 80
          pathType: ImplementationSpecific
```

OpenShift入口控制器的步驟

1. 取得您的憑證、取得可供OpenShift路由使用的金鑰、憑證和CA檔案。
2. 建立OpenShift路由：

```
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
```

登入Astra Control Center UI

安裝Astra Control Center之後、您將變更預設管理員的密碼、並登入Astra Control Center UI儀表板。

步驟

1. 在瀏覽器中、輸入您在中使用的FQDN `astraAddress` 在中 `astra_control_center_min.yaml` 請於何時進行 您安裝了Astra Control Center。
2. 出現提示時、請接受自我簽署的憑證。

> 您可以在登入後建立自訂憑證。

3. 在Astra Control Center登入頁面、輸入您使用的值 `email` 在中 `astra_control_center_min.yaml` 請於何時進行 您安裝了Astra Control Center，然後是一次性密碼 (`ACC-[UUID]`）。

> 如果您輸入錯誤密碼三次、系統將鎖定管理員帳戶15分鐘。

4. 選擇*登入*。
5. 出現提示時變更密碼。

> 如果這是您第一次登入、但您忘記密碼、而且尚未建立其他管理使用者帳戶、請聯絡NetApp支援部門以取得密碼恢復協助。

6. （選用）移除現有的自我簽署TLS憑證、並以取代 "由憑證授權單位（CA）簽署的自訂TLS憑證"。

疑難排解安裝

如果有任何服務存在 `Error` 狀態、您可以檢查記錄。尋找400到500範圍內的API回應代碼。這些都表示發生故障的地點。

步驟

1. 若要檢查Astra控制中心的操作員記錄、請輸入下列內容：

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

下一步

執行以完成部署 "設定工作"。

=
:allow-uri-read:

瞭解Pod安全性原則限制

Astra Control Center透過Pod安全性原則（ASP）來支援權限限制。Pod安全性原則可讓您限制使用者或群組能夠執行容器的項目、以及容器可以擁有的權限。

某些Kubernetes發佈版本（例如RKE2）的預設Pod安全性原則限制太多、因此在安裝Astra Control Center時會造成問題。

您可以使用此處提供的資訊和範例來瞭解Astra Control Center所建立的Pod安全性原則、並設定Pod安全性原則、以提供所需的保護、而不會干擾Astra Control Center功能。

由Astra Control Center安裝的PSPS

Astra Control Center會在安裝期間建立數個Pod安全性原則。其中有些是永久性的、有些是在特定作業期間建立、一旦作業完成、就會移除。

在安裝期間建立PSPS

在Astra Control Center安裝期間、Astra Control Center操作員會安裝自訂的Pod安全性原則、角色物件和角色繫結物件、以支援Astra Control Center命名空間中的Astra Control Center服務部署。

新原則和物件具有下列屬性：

```
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
```

在備份作業期間建立PSPS

在備份作業期間、Astra Control Center會建立動態pod安全性原則、ClusterRole物件和角色繫結物件。這些支援在個別命名空間中執行的備份程序。

新原則和物件具有下列屬性：

```
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
```

叢集管理期間建立的**PSPS**

當您管理叢集時、Astra Control Center會在託管叢集中安裝NetApp監控操作員。此運算子會建立一個Pod安全性原則、叢集角色物件和角色繫結物件、以便在Astra Control Center命名空間中部署遙測服務。

新原則和物件具有下列屬性：

```
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
```

啟用命名空間之間的網路通訊

有些環境使用網路原則架構來限制命名空間之間的流量。Astra Control Center營運者、Astra Control Center和Astra Plugin for VMware vSphere都位於不同的命名空間中。這些不同命名空間中的服務必須能夠彼此通訊。若要啟用此通訊、請遵循下列步驟。

步驟

1. 刪除Astra Control Center命名空間中的任何網路原則資源：

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

2. 對於上述命令傳回的每個網路原則物件、請使用下列命令加以刪除。以傳回物件的名稱取代<object_name>：

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

3. 套用下列資源檔案、以設定acc型VP-network-policy物件、讓Astra Plugin for VMware vSphere服務能夠向Astra Control Center服務提出要求。將方括弧<>中的資訊取代為您環境中的資訊：

```
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. 套用下列資源檔案、以設定acc操作者網路原則物件、讓Astra Control Center操作者能夠與Astra Control Center服務通訊。將方括弧<>中的資訊取代為您環境中的資訊：

```
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
```

移除資源限制

某些環境使用資源配額和限制範圍物件、以防止命名空間中的資源消耗叢集上的所有可用CPU和記憶體。Astra Control Center並未設定上限、因此不符合這些資源。您需要將它們從您計畫安裝Astra Control Center的命名空間中移除。

您可以使用下列步驟擷取及移除這些配額和限制。在這些範例中、命令輸出會在命令之後立即顯示。

步驟

1. 在NetApp-acc命名空間中取得資源配額：

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

回應：

```
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. 依名稱刪除所有資源配額：

```
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. 取得NetApp-acc命名空間中的限制範圍：

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

回應：

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

4. 依名稱刪除限制範圍：

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

=
:allow-uri-read:

版權資訊

Copyright © 2023 NetApp, Inc. 版權所有。台灣印製。非經版權所有人事先書面同意，不得將本受版權保護文件的任何部分以任何形式或任何方法（圖形、電子或機械）重製，包括影印、錄影、錄音或儲存至電子檢索系統中。

由 NetApp 版權資料衍伸之軟體必須遵守下列授權和免責聲明：

此軟體以 NETAPP「原樣」提供，不含任何明示或暗示的擔保，包括但不限於有關適售性或特定目的適用性之擔保，特此聲明。於任何情況下，就任何已造成或基於任何理論上責任之直接性、間接性、附隨性、特殊性、懲罰性或衍生性損害（包括但不限於替代商品或服務之採購；使用、資料或利潤上的損失；或企業營運中斷），無論是在使用此軟體時以任何方式所產生的契約、嚴格責任或侵權行為（包括疏忽或其他）等方面，NetApp 概不負責，即使已被告知有前述損害存在之可能性亦然。

NetApp 保留隨時變更本文所述之任何產品的權利，恕不另行通知。NetApp 不承擔因使用本文所述之產品而產生的責任或義務，除非明確經過 NetApp 書面同意。使用或購買此產品並不會在依據任何專利權、商標權或任何其他 NetApp 智慧財產權的情況下轉讓授權。

本手冊所述之產品受到一項（含）以上的美國專利、國外專利或申請中專利所保障。

有限權利說明：政府機關的使用、複製或公開揭露須受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中的「技術資料權利 - 非商業項目」條款 (b)(3) 小段所述之限制。

此處所含屬於商業產品和 / 或商業服務（如 FAR 2.101 所定義）的資料均為 NetApp, Inc. 所有。根據本協議提供的所有 NetApp 技術資料和電腦軟體皆屬於商業性質，並且完全由私人出資開發。 美國政府對於該資料具有非專屬、非轉讓、非轉授權、全球性、有限且不可撤銷的使用權限，僅限於美國政府為傳輸此資料所訂合約所允許之範圍，並基於履行該合約之目的方可使用。除非本文另有規定，否則未經 NetApp Inc. 事前書面許可，不得逕行使用、揭露、重製、修改、履行或展示該資料。美國政府授予國防部之許可權利，僅適用於 DFARS 條款 252.227-7015(b)（2014 年 2 月）所述權利。

商標資訊

NETAPP、NETAPP 標誌及 http://www.netapp.com/TM 所列之標章均為 NetApp, Inc. 的商標。文中所涉及的所有其他公司或產品名稱，均為其各自所有者的商標，不得侵犯。