



安裝 **Astra Control Center**

Astra Control Center

NetApp
November 27, 2023

目錄

使用標準程序安裝Astra Control Center	1
下載並擷取Astra Control Center	2
安裝NetApp Astra kubectl外掛程式	2
將映像新增至本機登錄	3
設定具有驗證需求之登錄的命名空間和機密	5
安裝Astra Control Center操作員	6
設定Astra控制中心	10
完整的Astra控制中心和操作員安裝	19
驗證系統狀態	20
設定入口以進行負載平衡	25
登入Astra Control Center UI	29
疑難排解安裝	29
下一步	30
設定外部憑證管理程式	30

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 下列網址下載安裝套件、並執行下列步驟。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。

展開以進行其他安裝程序

- *使用RedHat OpenShift操作員中樞*安裝：請使用此功能 ["替代程序"](#) 使用作業系統集線器在OpenShift上安裝Astra Control Center。
- 以**Cloud Volumes ONTAP** 支援功能的方式在公有雲上安裝：使用 ["這些程序"](#) 若要在Amazon Web Services (AWS)、Google Cloud Platform (GCP) 或Microsoft Azure中安裝Astra Control Center、並提供Cloud Volumes ONTAP 一套支援整合式儲存後端的功能。

如需Astra Control Center安裝程序的示範、請參閱 ["這段影片"](#)。

開始之前

- ["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 ["Pod 安全限制"](#)。
- 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。
- 如果叢集中已存在憑證管理程式、您需要執行某些作業 ["必要步驟"](#) 因此Astra Control Center不會嘗試安裝自己的憑證管理程式。依預設、Astra Control Center會在安裝期間安裝自己的憑證管理程式。



在第三個故障網域或次要站台中部署 Astra Control Center。這是應用程式複寫和無縫災難恢復的建議。

步驟

若要安裝Astra Control Center、請執行下列步驟：

- [下載並擷取Astra Control Center](#)
- [安裝NetApp Astra kubectl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](#)



請勿刪除Astra Control Center運算子（例如、`kubectl delete -f astra_control_center_operator_deploy.yaml`）在Astra Control Center安裝或操作期間、隨時避免刪除Pod。

下載並擷取Astra Control Center

1. 下載包含Astra Control Center的套裝組合 (`astra-control-center-[version].tar.gz`) "[Astra Control Center 下載頁面](#)"。
2. （建議但可選）下載Astra Control Center的憑證與簽名套件 (`astra-control-center-certs-[version].tar.gz`) 驗證套件的簽名。

展開以取得詳細資料

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

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

隨即顯示輸出 `Verified OK` 驗證成功之後。

3. 從Astra Control Center套裝組合擷取映像：

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

安裝NetApp Astra kubectl外掛程式

您可以使用 `NetApp Astra kubectl` 命令列外掛程式、將影像推送至本機 Docker 儲存庫。

開始之前

NetApp為不同的CPU架構和作業系統提供外掛程式二進位檔。執行此工作之前、您必須先瞭解您的CPU和作業系統。

如果您已從先前的安裝中安裝外掛程式、"[請確定您擁有最新版本](#)" 完成這些步驟之前。

步驟

1. 列出可用的 `NetApp Astra Kubectl` 外掛程式二進位檔：



KECBECTI外掛程式庫是tar套件的一部分、會擷取到資料夾中 `kubectl-astra`。

```
ls kubectl-astra/
```

2. 將作業系統和 CPU 架構所需的檔案移至目前路徑、並將其重新命名為 `kubectl-astra`：

```
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. 切換到tar檔案的根目錄。您應該會看到 `acc.manifest.bundle.yaml` 檔案與這些目錄：

```
acc/  
kubectl-astra/  
acc.manifest.bundle.yaml
```

2. 將Astra Control Center映像目錄中的套件映像推送到本機登錄。執行之前、請先進行下列替換 `push-images` 命令：

- 以 `<BUNDLE_FILE>` Astra Control套裝組合檔案的名稱取代 (`acc.manifest.bundle.yaml`)。
- 以 `<MY_FULL_REGISTRY_PATH>` Docker儲存庫的URL取代支援；例如 `<a href="https://<docker-registry>";" class="bare">https://<docker-registry>";`。
- 以 `<MY_REGISTRY_USER>` 使用者名稱取代。
- 以 `<MY_REGISTRY_TOKEN>` 登錄的授權權杖取代。

```
kubectl astra packages push-images -m <BUNDLE_FILE> -r  
<MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p  
<MY_REGISTRY_TOKEN>
```

Podman

1. 切換到tar檔案的根目錄。您應該會看到這個檔案和目錄：

```
acc.manifest.bundle.yaml  
acc/
```

2. 登入您的登錄：

```
podman login <YOUR_REGISTRY>
```

3. 針對您使用的Podman版本、準備並執行下列其中一個自訂指令碼。以包含任何子目錄的儲存庫URL取代 `<MY_FULL_REGISTRY_PATH>`。

```
<strong>Podman 4</strong>
```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=23.07.0-25
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //'')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/:::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

Podman 3

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=23.07.0-25
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //'')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/:::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```



指令碼所建立的映像路徑應如下所示、視登錄組態而定：

<https://netappdownloads.jfrog.io/docker-astra-control-prod/netapp/astra/acc/23.07.0-25/image:version>

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 Kribeconfig：

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



完成安裝之前、請確定您的 Kubeconfig 指向您要安裝 Astra Control Center 的叢集。

2. 如果您使用需要驗證的登錄、則需要執行下列動作：

展開步驟

a. 建立 netapp-acc-operator 命名空間：

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

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/23.07.0-25`）。

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



如果在產生機密之後刪除命名空間、請重新建立命名空間、然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]
```

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]
```

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 `ASTRA_IMAGE_REGISTRY` 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 `ASTRA_IMAGE_REGISTRY` 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](#)。

```
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
  strategy:
    type: Recreate
  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: ASTRA_IMAGE_REGISTRY/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
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_INSTALLTIMEOUT
          value: 5m
        image: ASTRA_IMAGE_REGISTRY/acc-operator:23.07.25
        imagePullPolicy: IfNotPresent
        livenessProbe:
          httpGet:
```

```
    path: /healthz
    port: 8081
    initialDelaySeconds: 15
    periodSeconds: 20
name: manager
readinessProbe:
  httpGet:
    path: /readyz
    port: 8081
    initialDelaySeconds: 5
    periodSeconds: 10
resources:
  limits:
    cpu: 300m
    memory: 750Mi
  requests:
    cpu: 100m
    memory: 75Mi
securityContext:
  allowPrivilegeEscalation: false
imagePullSecrets: []
securityContext:
  runAsUser: 65532
terminationGracePeriodSeconds: 10
```

3. 安裝Astra Control Center操作員：

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

展開範例回應：

```
namespace/netapp-acc-operator created
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.as
tra.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.yaml) 進行帳戶、支援、登錄及其他必要設定：

```
vim astra_control_center.yaml
```



附註的Y反洗錢範例遵循下列步驟。

2. 修改或確認下列設定：

<code>accountName</code>

設定	指導	類型	範例
accountName	變更 accountName 字串至您要與Astra Control Center帳戶建立關聯的名稱。只能有一個帳戶名稱。	字串	Example

<code>astraVersion</code>

設定	指導	類型	範例
astraVersion	要部署的Astra Control Center版本。此設定不需要任何動作、因為此值將預先填入。	字串	23.07.0-25

<code>astraAddress</code>

設定	指導	類型	範例
astraAddress	<p>變更 astraAddress 字串至您要在瀏覽器中使用的FQDN (建議) 或IP位址、以存取Astra Control Center。此位址定義Astra Control Center在資料中心的找到方式、以及當您完成配置時、從負載平衡器配置的相同FQDN或IP位址 "Astra Control Center需求"。</p> <p>附註：請勿使用 http:// 或 https:// 地址中。複製此FQDN以供在中使用 後續步驟。</p>	字串	astra.example.com

<code>autoSupport</code>

您在本節中的選擇決定您是否會參與NetApp主動式支援應用程式NetApp Active IQ 功能、以及資料的傳送位置。需要網際網路連線（連接埠4442）、所有支援資料都會匿名。

設定	使用	指導	類型	範例
<code>autoSupport.enrolled</code>	也可以 <code>enrolled</code> 或 <code>url</code> 必須選取欄位	變更 <code>enrolled</code> 解決方 案AutoSupport <code>false</code> 適用於沒有網際網路連線或無法保留的網站 <code>true</code> 適用於連線站台。的設定 <code>true</code> 可將匿名資料傳送至 NetApp 以供支援之用。預設選項為 <code>false</code> 並表示不會將任何支援資料傳送給NetApp。	布林值	<code>false</code> （此值為預設值）
<code>autoSupport.url</code>	也可以 <code>enrolled</code> 或 <code>url</code> 必須選取欄位	此URL決定匿名資料的傳送位置。	字串	https://support.netapp.com/asupprod/post/1.0/postAsup

<code>email</code>

設定	指導	類型	範例
<code>email</code>	變更 <code>email</code> 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 。此電子郵件地址將作為初始帳戶登入UI的使用者名稱、並會收到Astra Control中事件的通知。	字串	<code>admin@example.com</code>

<code>firstName</code>

設定	指導	類型	範例
<code>firstName</code>	與Astra帳戶相關聯的預設初始系統管理員的名字。第一次登入後、此處使用的名稱會顯示在UI的標題中。	字串	SRE

<code>LastName</code>

設定	指導	類型	範例
lastName	與Astra帳戶相關聯的預設初始管理員姓氏。第一次登入後、此處使用的名稱會顯示在UI的標題中。	字串	Admin

<code>imageRegistry</code>

您在本節中的選擇定義了裝載Astra應用程式映像、Astra Control Center運算子和Astra Control Center Helm儲存庫的容器映像登錄。

設定	使用	指導	類型	範例
imageRegistry.name	必要	您在中推入映像的映像登錄名稱 上一步 。請勿使用 http:// 或 https:// 在登錄名稱中。	字串	example.registry.com/astra
imageRegistry.secret	如果您輸入的字串則為必要 imageRegistry.name' requires a secret. IMPORTANT: If you are using a registry that does not require authorization, you must delete this `secret` 行內 imageRegistry 否則安裝將會失敗。	用來驗證映像登錄的Kubernetes機密名稱。	字串	astra-registry-cred

<code>storageClass</code>

設定	指導	類型	範例
storageClass	<p>變更 storageClass 價值來源 <code>ontap-gold</code> 至安裝所需的另一個 Astra Trident storageClass 資源。執行命令 <code>kubectl get sc</code> 以判斷您現有的已設定儲存類別。必須在資訊清單檔案中輸入其中一個 Astra Trident 型儲存類別 (<code>astra-control-center-<version>.manifest</code>)、並將用於 Astra PV。如果未設定、則會使用預設的儲存類別。</p> <p>附註：如果已設定預設儲存類別、請確定它是唯一具有預設附註的儲存類別。</p>	字串	<code>ontap-gold</code>

<code>volumeReclaimPolicy</code>

設定	指導	類型	選項
volumeReclaimPolicy	<p>這為 Astra 的 PV 設定回收原則。將此原則設定為 <code>Retain</code> 刪除 Astra 後保留持續磁碟區。將此原則設定為 <code>Delete</code> 刪除 Astra 後刪除持續磁碟區。如果未設定此值、則會保留 PV。</p>	字串	<ul style="list-style-type: none">• <code>Retain</code> (這是預設值)• <code>Delete</code>

<code>ingressType</code>

設定	指導	類型	選項
ingressType	<p>使用下列其中一種入口類型：</p> <p>Generic (ingressType: "Generic") (預設) 如果您使用另一個入口控制器、或偏好使用自己的入口控制器、請使用此選項。部署Astra Control Center之後、您需要設定 "入口控制器" 使用URL公開Astra Control Center。</p> <p>AccTraefik (ingressType: "AccTraefik") 如果您不想設定入口控制器、請使用此選項。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。</p> <p>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。</p> <p>附註：如需「負載平衡器」和入口服務類型的詳細資訊、請參閱 "需求"。</p>	字串	<ul style="list-style-type: none">• Generic (這是預設值)• AccTraefik

<code>scaleSize</code>

設定	指導	類型	選項
scaleSize	<p>Astra 預設會使用高可用度 (HA) scaleSize 的 Medium，用於在 HA 中部署大多數服務並部署多個複本以實現冗餘。與 scaleSize 做為 Small、Astra 將減少所有服務的複本數量、但基本服務除外、以減少使用量。</p> <p>秘訣：Medium 部署包含約 100 個 Pod (不包括暫時性工作負載)。100 個 Pod 以三個主節點和三個工作節點組態為基礎)。請注意、在您的環境中、每個 Pod 的網路限制可能是個問題、特別是在考慮災難恢復案例時。</p>	字串	<ul style="list-style-type: none">• Small• Medium (這是預設值)

<code>astraResourcesScaler</code>

設定	指導	類型	選項
astraResourcesScaler	<p>適用的擴充選項適用於適用的適用範圍。依預設、Astra Control Center 會針對 Astra 內的大部分元件設定資源要求來進行部署。此組態可讓 Astra Control Center 軟體堆疊在應用程式負載和擴充性增加的環境中、發揮更佳效能。</p> <p>不過、在使用較小開發或測試叢集的案例中、則是使用「CR」欄位 astraResourcesScaler 可能設為 Off。這會停用資源要求、並允許在較小的叢集上部署。</p>	字串	<ul style="list-style-type: none">• Default (這是預設值)• Off

`<code>additionalValues</code>`



在 Astra Control Center CR 中新增下列額外值、以避免在 23.07 安裝中出現已知問題：

```
additionalValues:
  polaris-keycloak:
    livenessProbe:
      initialDelaySeconds: 180
    readinessProbe:
      initialDelaySeconds: 180
```

- 對於 Astral Control Center 和 Cloud Insights 通訊、依預設會停用 TLS 憑證驗證。您可以在中新增下一節、以啟用 Cloud Insights 與 Astra 控制中心主機叢集和託管叢集之間通訊的 TLS 憑證驗證 `additionalValues`。

```
additionalValues:
  netapp-monitoring-operator:
    config:
      ciSkipTlsVerify: false
  cloud-insights-service:
    config:
      ciSkipTlsVerify: false
  telemetry-service:
    config:
      ciSkipTlsVerify: false
```

`<code>crds</code>`

您在本節中的選擇決定Astra Control Center應如何處理客戶需求日。

設定	指導	類型	範例
<code>crds.externalCertManager</code>	<p>如果您使用外部憑證管理程式、請變更 <code>externalCertManager</code> 至 <code>true</code>。預設值 <code>false</code> 讓Astra Control Center在安裝期間安裝自己的憑證管理程式客戶檔案。</p> <p>CRD是整個叢集的物件、安裝這些物件可能會影響叢集的其他部分。您可以使用此旗標向Astra控制中心發出訊號、表示這些客戶需求日將由Astra控制中心外部的叢集管理員安裝及管理。</p>	布林值	False (此值為預設值)
<code>crds.externalTraefik</code>	<p>依預設、Astra Control Center會安裝必要的Traefik客戶需求日。CRD是整個叢集的物件、安裝這些物件可能會影響叢集的其他部分。您可以使用此旗標向Astra控制中心發出訊號、表示這些客戶需求日將由Astra控制中心外部的叢集管理員安裝及管理。</p>	布林值	False (此值為預設值)



在完成安裝之前、請務必為您的組態選擇正確的儲存類別和入口類型。

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
  volumeReclaimPolicy: "Retain"
  ingressType: "Generic"
  scaleSize: "Medium"
  astraResourcesScaler: "Default"
  additionalValues:
    polaris-keycloak:
      livenessProbe:
        initialDelaySeconds: 180
      readinessProbe:
        initialDelaySeconds: 180
  crds:
    externalTraefik: false
    externalCertManager: false
```

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]
```

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

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



Astra Control Center 駕駛員將自動檢查環境需求。遺失 "需求" 可能導致安裝失敗、或 Astra Control Center 無法正常運作。請參閱 [下一節](#) 檢查與自動系統檢查相關的警告訊息。

驗證系統狀態

您可以使用 kubectl 命令來驗證系統狀態。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。

步驟

1. 確認安裝程序未產生與驗證檢查相關的警告訊息：

```
kubectl get acc [astra or custom Astra Control Center CR name] -n [netapp-acc or custom namespace] -o yaml
```



Astra Control Center 操作者記錄中也會報告其他警告訊息。

2. 修正自動化需求檢查所回報的環境問題。



您可以確保環境符合、以修正問題 "需求" 適用於 Astra Control Center。

3. 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。

展開以取得範例回應

NAME	READY	STATUS	
RESTARTS	AGE		
acc-helm-repo-6cc7696d8f-pmhm8 9h	1/1	Running	0
activity-597fb656dc-5rd4l 9h	1/1	Running	0
activity-597fb656dc-mqmcw 9h	1/1	Running	0
api-token-authentication-62f84 9h	1/1	Running	0
api-token-authentication-68nlf 9h	1/1	Running	0
api-token-authentication-ztgrm 9h	1/1	Running	0
asup-669d4ddbc4-fnmwp (9h ago) 9h	1/1	Running	1
authentication-78789d7549-1k686 9h	1/1	Running	0
bucket-service-65c7d95496-24x7l (9h ago) 9h	1/1	Running	3
cert-manager-c9f9fbf9f-k8zq2 9h	1/1	Running	0
cert-manager-c9f9fbf9f-qj1zm 9h	1/1	Running	0
cert-manager-cainjector-dbbbd8447-b5q1l 9h	1/1	Running	0
cert-manager-cainjector-dbbbd8447-p5whs 9h	1/1	Running	0
cert-manager-webhook-6f97bb7d84-4722b 9h	1/1	Running	0
cert-manager-webhook-6f97bb7d84-86kv5 9h	1/1	Running	0
certificates-59d9f6f4bd-2j899 9h	1/1	Running	0
certificates-59d9f6f4bd-9d9k6 9h	1/1	Running	0
certificates-expiry-check-28011180--1-8lkxz 9h	0/1	Completed	0
cloud-extension-5c9c9958f8-jdhrp 9h	1/1	Running	0
cloud-insights-service-5cdd5f7f-pp8r5 9h	1/1	Running	0
composite-compute-66585789f4-hxn5w	1/1	Running	0

9h			
composite-volume-68649f68fd-tb7p4	1/1	Running	0
9h			
credentials-dfc844c57-jsx92	1/1	Running	0
9h			
credentials-dfc844c57-xw26s	1/1	Running	0
9h			
entitlement-7b47769b87-4jb6c	1/1	Running	0
9h			
features-854d8444cc-c24b7	1/1	Running	0
9h			
features-854d8444cc-dv6sm	1/1	Running	0
9h			
fluent-bit-ds-9tlv4	1/1	Running	0
9h			
fluent-bit-ds-bpkcb	1/1	Running	0
9h			
fluent-bit-ds-cxmwx	1/1	Running	0
9h			
fluent-bit-ds-jgnhc	1/1	Running	0
9h			
fluent-bit-ds-vtr6k	1/1	Running	0
9h			
fluent-bit-ds-vxqd5	1/1	Running	0
9h			
graphql-server-7d4b9d44d5-zdbf5	1/1	Running	0
9h			
identity-6655c48769-4pwk8	1/1	Running	0
9h			
influxdb2-0	1/1	Running	0
9h			
keycloak-operator-55479d6fc6-slvmt	1/1	Running	0
9h			
krakend-f487cb465-78679	1/1	Running	0
9h			
krakend-f487cb465-rjsxx	1/1	Running	0
9h			
license-64cbc7cd9c-qxsr8	1/1	Running	0
9h			
login-ui-5db89b5589-ndb96	1/1	Running	0
9h			
loki-0	1/1	Running	0
9h			
metrics-facade-8446f64c94-x8h7b	1/1	Running	0
9h			
monitoring-operator-6b44586965-pvcl4	2/2	Running	0

9h			
nats-0	1/1	Running	0
9h			
nats-1	1/1	Running	0
9h			
nats-2	1/1	Running	0
9h			
nautilus-85754d87d7-756qb	1/1	Running	0
9h			
nautilus-85754d87d7-q8j7d	1/1	Running	0
9h			
openapi-5f9cc76544-7fnjm	1/1	Running	0
9h			
openapi-5f9cc76544-vzr7b	1/1	Running	0
9h			
packages-5db49f8b5-lrzhd	1/1	Running	0
9h			
polaris-consul-consul-server-0	1/1	Running	0
9h			
polaris-consul-consul-server-1	1/1	Running	0
9h			
polaris-consul-consul-server-2	1/1	Running	0
9h			
polaris-keycloak-0	1/1	Running	2
(9h ago) 9h			
polaris-keycloak-1	1/1	Running	0
9h			
polaris-keycloak-2	1/1	Running	0
9h			
polaris-keycloak-db-0	1/1	Running	0
9h			
polaris-keycloak-db-1	1/1	Running	0
9h			
polaris-keycloak-db-2	1/1	Running	0
9h			
polaris-mongodb-0	1/1	Running	0
9h			
polaris-mongodb-1	1/1	Running	0
9h			
polaris-mongodb-2	1/1	Running	0
9h			
polaris-ui-66fb99479-qp9gq	1/1	Running	0
9h			
polaris-vault-0	1/1	Running	0
9h			
polaris-vault-1	1/1	Running	0

9h	polaris-vault-2	1/1	Running	0
9h	public-metrics-76fbf9594d-zmxzw	1/1	Running	0
9h	storage-backend-metrics-7d7fbc9cb9-lmd25	1/1	Running	0
9h	storage-provider-5bdd456c4b-2fftc	1/1	Running	0
9h	task-service-87575df85-dnn2q	1/1	Running	3
(9h ago) 9h	task-service-task-purge-28011720--1-q6w4r	0/1	Completed	0
28m	task-service-task-purge-28011735--1-vk6pd	1/1	Running	0
13m	telegraf-ds-2r2kw	1/1	Running	0
9h	telegraf-ds-6s9d5	1/1	Running	0
9h	telegraf-ds-96jl7	1/1	Running	0
9h	telegraf-ds-hbp84	1/1	Running	0
9h	telegraf-ds-plwzv	1/1	Running	0
9h	telegraf-ds-sr22c	1/1	Running	0
9h	telegraf-rs-4sbg8	1/1	Running	0
9h	telemetry-service-fb9559f7b-mk917	1/1	Running	3
(9h ago) 9h	tenancy-559bbc6b48-5msgg	1/1	Running	0
9h	traefik-d997b8877-7xpf4	1/1	Running	0
9h	traefik-d997b8877-9xv96	1/1	Running	0
9h	trident-svc-585c97548c-d25z5	1/1	Running	0
9h	vault-controller-88484b454-2d6sr	1/1	Running	0
9h	vault-controller-88484b454-fc5cz	1/1	Running	0
9h	vault-controller-88484b454-jktld	1/1	Running	0
9h				

4. (選用) 觀看 acc-operator 監控進度的記錄：

```
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、您可以透過再次嘗試登錄 ["在UI中新增叢集工作流程"](#) 或API。

5. 當所有Pod都在執行時、請確認安裝成功 (READY 是 True) 並取得您登入Astra Control Center時所使用的初始設定密碼：

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

回應：

NAME	UUID	VERSION	ADDRESS
READY			
astra	9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f	23.07.0-25	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。如果您使用的預設值、這些程序會提供入口控制器的設定範例 `ingressType: "Generic"` Astra Control Center自訂資源 (`astra_control_center.yaml`)。如果您指定、則不需要使用此程序 `ingressType: "AccTraefik"` Astra Control Center自訂資源 (`astra_control_center.yaml`)。

部署Astra Control Center之後、您需要設定入口控制器、以URL顯示Astra Control Center。

設定步驟視您使用的入口控制器類型而有所不同。Astra Control Center支援多種入站控制器類型。這些設定程序提供一些常見入口控制器類型的範例步驟。

開始之前

- 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。

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v1資源類型作為架構 (`istio-ingress.yaml` 在本例中使用)：

```
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: [netapp-acc or custom namespace]
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. 套用變更：

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

6. 檢查入侵狀態：

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

回應：

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

7. 完成Astra Control Center安裝。

適用於Nginx像 控制器的步驟

1. 建立類型的秘密 `kubernetes.io/tls` 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v1資源類型作為架構 (`nginx-Ingress.yaml` 在本例中使用)：

```
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
```

3. 套用變更：

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



NetApp建議將Nginx像 控制器安裝為部署、而非 `daemonSet`。

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 FQDN（包括 `https://` 字首） `astraAddress` 在中 `astra_control_center.yaml` 請於何時進行 [您安裝了Astra Control Center](#)。
2. 收到提示時、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.yaml` 請於何時進行 [您安裝了Astra Control Center](#)，然後輸入初始設定密碼 (`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"](#) 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。

選項

- 若要檢查Astra控制中心的操作員記錄、請輸入下列內容：

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

- 若要檢查 Astra Control Center CR 的輸出：

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

下一步

- (選用) 視您的環境而定、請在安裝後完成 "組態步驟"。
- 執行以完成部署 "設定工作"。

設定外部憑證管理程式

如果Kubernetes叢集中已存在憑證管理程式、您需要執行一些必要步驟、使Astra Control Center不會安裝自己的憑證管理程式。

步驟

1. 確認您已安裝憑證管理程式：

```
kubectl get pods -A | grep 'cert-manager'
```

回應範例：

```
cert-manager   essential-cert-manager-84446f49d5-sf2zd   1/1
Running        0     6d5h
cert-manager   essential-cert-manager-cainjector-66dc99cc56-91dmt   1/1
Running        0     6d5h
cert-manager   essential-cert-manager-webhook-56b76db9cc-fjqrq     1/1
Running        0     6d5h
```

2. 為建立憑證/金鑰配對 astraAddress FQDN：

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

回應範例：


```
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'tls.key'
```

3. 使用先前產生的檔案建立秘密：

```
kubectl create secret tls selfsigned-tls --key tls.key --cert tls.crt -n
<cert-manager-namespace>
```

回應範例：

```
secret/selfsigned-tls created
```

4. 建立 ClusterIssuer 以下*確切*的檔案包含您的命名空間位置 cert-manager 已安裝Pod：

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: astra-ca-clusterissuer
  namespace: <cert-manager-namespace>
spec:
  ca:
    secretName: selfsigned-tls
```

```
kubectl apply -f ClusterIssuer.yaml
```

回應範例：

```
clusterissuer.cert-manager.io/astra-ca-clusterissuer created
```

5. 確認 ClusterIssuer 已正確啟動。Ready 必須是 True 在繼續之前：

```
kubectl get ClusterIssuer
```

回應範例：

NAME	READY	AGE
astra-ca-clusterissuer	True	9s

6. 完成 "Astra Control Center安裝程序"。有 "Astra Control Center叢集Yaml所需的組態步驟" 您可在其中變更CRD值、以表示外部安裝了憑證管理程式。您必須在安裝期間完成此步驟、Astra Control Center才能辨識外部憑證管理程式。

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。