



開始使用

Astra Control Center

NetApp
August 11, 2025

目錄

開始使用	1
瞭解Astra Control	1
功能	1
部署模式	1
Astra Control Service的運作方式	3
Astra控制中心的運作方式	4
以取得更多資訊	5
Astra Control Center需求	5
支援的主機叢集 Kubernetes 環境	5
主機叢集資源需求	6
服務網狀網路需求	6
Astra Trident	6
Astra Control 資源配置程式	6
儲存設備後端	7
Astra Control Center 授權	8
網路需求	8
內部部署Kubernetes叢集的入口	9
支援的網頁瀏覽器	9
應用程式叢集的其他需求	9
下一步	10
Astra Control Center快速入門	10
以取得更多資訊	11
安裝總覽	11
使用標準程序安裝Astra Control Center	11
使用OpenShift作業系統集線器安裝Astra Control Center	46
安裝Astra Control Center搭配Cloud Volumes ONTAP 一套功能性儲存後端	57
安裝後設定Astra Control Center	69
設定Astra控制中心	74
新增Astra Control Center授權	74
啟用 Astra Control Provisioner	75
使用Astra Control為環境做好叢集管理準備	85
（技術預覽）安裝 Astra Connector for Managed 叢集	96
新增叢集	99
在 ONTAP 儲存後端上啟用驗證	100
新增儲存後端	106
新增儲存庫	107

開始使用

瞭解Astra Control

Astra Control是Kubernetes應用程式資料生命週期管理解決方案、可簡化狀態應用程式的作業。輕鬆保護、備份、複寫及移轉Kubernetes工作負載、並即時建立運作中的應用程式複本。

功能

Astra Control為Kubernetes應用程式資料生命週期管理提供關鍵功能：

- 自動管理持續儲存
- 建立應用程式感知的隨需快照與備份
- 自動化原則導向的快照與備份作業
- 將應用程式和資料從一個Kubernetes叢集移轉到另一個叢集
- 使用 NetApp SnapMirror 技術（ Astra Control Center ）將應用程式複寫到遠端系統
- 將應用程式從接移複製到正式作業
- 視覺化應用程式健全狀況與保護狀態
- 使用Web UI或API來實作備份與移轉工作流程

部署模式

Astra Control提供兩種部署模式：

- *** Astra Control Service***：NetApp管理的服務、可在多個雲端供應商環境中、以及自行管理的Kubernetes叢集、提供Kubernetes叢集的應用程式感知資料管理功能。
- *** Astra Control Center***：自我管理的軟體、可針對在內部部署環境中執行的Kubernetes叢集、提供應用程式感知資料管理功能。Astra Control Center也可安裝在多個雲端供應商環境中、採用NetApp Cloud Volumes ONTAP 的解決方案儲存後端。

	Astra控制服務	Astra控制中心
如何提供？	是NetApp提供的完整託管雲端服務	可下載、安裝及管理的軟體
它的代管位置為何？	在NetApp首選的公有雲上	在您自己的Kubernetes叢集上
如何更新？	由NetApp管理	您可以管理任何更新

	Astra控制服務	Astra控制中心
支援的 Kubernetes 發佈內容為何？	• * 雲端供應商 * ◦ Amazon Web Services ▪ Amazon Elastic Kubernetes Service (EKS) ◦ Google Cloud ▪ Google Kubernetes Engine (GKE) ◦ Microsoft Azure ▪ Azure Kubernetes服務 (KS) • * 自我管理叢集 * ◦ Kubernetes (上游) ◦ Rancher Kubernetes引擎 (RKE) ◦ Red Hat OpenShift Container Platform • * 內部部署叢集 * ◦ 內部部署 Red Hat OpenShift Container Platform	• Azure Stack HCI 上的 Azure Kubernetes 服務 • Google Anthos • Kubernetes (上游) • Rancher Kubernetes引擎 (RKE) • Red Hat OpenShift Container Platform

	Astra控制服務	Astra控制中心
支援的儲存後端有哪些？	• * 雲端供應商 * ◦ Amazon Web Services ▪ Amazon EBS ▪ Amazon FSX for NetApp ONTAP 產品 ▪ "Cloud Volumes ONTAP" ◦ Google Cloud ▪ Google持續磁碟 ▪ NetApp Cloud Volumes Service ▪ "Cloud Volumes ONTAP" ◦ Microsoft Azure ▪ Azure託管磁碟 ▪ Azure NetApp Files ▪ "Cloud Volumes ONTAP" • * 自我管理叢集 * ◦ Amazon EBS ◦ Azure託管磁碟 ◦ Google持續磁碟 ◦ "Cloud Volumes ONTAP" ◦ NetApp MetroCluster ◦ "Longhorn" • * 內部部署叢集 * ◦ NetApp MetroCluster ◦ NetApp ONTAP AFF 的功能與FAS 功能 ◦ NetApp ONTAP Select ◦ "Cloud Volumes ONTAP" ◦ "Longhorn"	• NetApp ONTAP AFF 的功能與FAS 功能 • NetApp ONTAP Select • "Cloud Volumes ONTAP" • "Longhorn"

Astra Control Service的運作方式

Astra Control Service是NetApp託管的雲端服務、隨時可用最新功能進行更新。它利用多個元件來實現應用程式資料生命週期管理。

Astra Control Service的高層級運作方式如下：

- 您可以設定雲端供應商並註冊Astra帳戶、開始使用Astra Control Service。

- 對於GKE叢集、Astra Control Service使用 "[適用於Cloud Volumes Service Google Cloud的NetApp解決方案](#)" 或Google持續磁碟做為持續磁碟區的儲存後端。
- 對於高峰叢集、Astra Control Service使用 "[Azure NetApp Files](#)" 或Azure託管磁碟做為持續磁碟區的儲存後端。
- 對於Amazon EKS叢集、Astra Control Service使用 "[Amazon彈性區塊存放區](#)" 或 "[Amazon FSx for NetApp ONTAP 產品](#)" 作為持續磁碟區的儲存後端。
- 您將第一部Kubernetes運算新增至Astra Control Service。Astra Control Service接著會執行下列作業：
 - 在雲端供應商帳戶中建立物件存放區、以儲存備份複本。

在Azure中、Astra Control Service也會為Blob容器建立資源群組、儲存帳戶和金鑰。

 - 在叢集上建立新的管理員角色和Kubernetes服務帳戶。
 - 使用該新的管理員角色在叢集上安裝 `link./inics/architecture #Astra 控制元件 [Astra Control Provisioner^]`、並建立一或多個儲存類別。
 - 如果您使用 NetApp 雲端服務儲存產品作為儲存後端、Astra Control Service 會使用 Astra Control Provisioner 為應用程式配置持續容量。如果您使用Amazon EBS或Azure託管磁碟做為儲存後端、則需要安裝供應商專屬的SCSI驅動程式。安裝說明請參閱 "[設定Amazon Web Services](#)" 和 "[使用Azure託管磁碟來設定Microsoft Azure](#)"。- 此時、您可以將應用程式新增至叢集。將在新的預設儲存類別上配置持續磁碟區。
- 然後使用Astra Control Service來管理這些應用程式、並開始建立快照、備份和複製。

Astra Control的免費方案可讓您管理帳戶中最多10個命名空間。如果您想要管理10多個項目、則必須將「免費方案」升級為「優質方案」、以設定帳單。

Astra控制中心的運作方式

Astra Control Center可在您自己的私有雲端本機執行。

Astra Control Center 支援 Kubernetes 叢集、其中包含 Astra Control Provisioner 設定的儲存類別、以及 ONTAP 儲存後端。

Astra Control Center 提供有限（7 天計量）的監控和遙測功能、並透過開放式度量端點匯出至 Kubernetes 原生監控工具（例如 Prometheus 和 Grafana）。

Astra Control Center 已完全整合至 AutoSupport 和 Active IQ 數位顧問（也稱為數位顧問）生態系統、為使用者和 NetApp 支援提供疑難排解和使用資訊。

您可以使用 90 天內嵌評估授權、試用 Astra Control Center。在評估 Astra Control Center 時、您可以透過電子郵件和社群選項獲得支援。此外、您也可以從產品內的支援儀表板存取知識庫文章和文件。

若要安裝及使用Astra Control Center、您必須符合特定需求 "[需求](#)"。

Astra Control Center的高層級運作方式如下：

- 您可以在本機環境中安裝Astra Control Center。深入瞭解如何操作 "[安裝Astra Control Center](#)"。
- 您可以完成以下設定工作：
 - 設定授權。

- 新增第一個叢集。
- 新增新增叢集時發現的儲存後端。
- 新增物件存放區儲存應用程式備份。

深入瞭解如何操作 ["設定Astra控制中心"](#)。

您可以將應用程式新增至叢集。或者、如果叢集中已有一些應用程式正在管理中、您可以使用Astra Control Center來管理這些應用程式。然後、使用Astra Control Center建立快照、備份、複製及複寫關係。

以取得更多資訊

- ["Astra Control Service文件"](#)
- ["Astra Control Center文件"](#)
- ["Astra Trident文件"](#)
- ["Astra Control API 文件"](#)
- ["本文檔 ONTAP"](#)

Astra Control Center需求

開始驗證作業環境、應用程式叢集、應用程式、授權和網頁瀏覽器的整備度。確保您的環境符合這些需求、以部署和操作 Astra Control Center。

支援的主機叢集 **Kubernetes** 環境

Astra Control Center 已通過下列 Kubernetes 主機環境的驗證：



確保您選擇架設 Astra Control Center 的 Kubernetes 環境符合環境正式文件中所述的基本資源需求。

Kubernetes 在主機叢集上的發佈	支援的版本
Azure Stack HCI 上的 Azure Kubernetes 服務	Azure Stack HCI 21H2 和 22H2 、含 1.24.11 至 1.26.6
Google Anthos	1.15 至 1.16 （請參閱 Google Anthos 入口要求 ）
Kubernetes （上游）	1.27 至 1.29
Rancher Kubernetes引擎（RKE）	RKE 1 ：採用 Rancher Manager 2.7.9 的版本 1.24.17 、 1.25.13 、 1.26.8 RKE 2 ：版本 1.23.16 和 1.24.13 搭配 Rancher Manager 2.6.13 RKE 2 ：採用 Rancher Manager 2.7.9 的版本 1.24.17 、 1.25.14 、 1.26.9
Red Hat OpenShift Container Platform	4.12 至 4.14

主機叢集資源需求

除了環境的資源需求之外、Astra Control Center還需要下列資源：



這些需求假設Astra Control Center是營運環境中唯一執行的應用程式。如果環境正在執行其他應用程式、請相應調整這些最低需求。

- ***CPU 擴充 ***：主機環境中所有節點的 CPU 都必須啟用 AVX 擴充功能。
- *** 工作節點 ***：總計至少 3 個工作節點、每個節點有 4 個 CPU 核心和 12GB RAM
- *** VMware Tanzu Kubernetes Grid 叢集需求 ***：在 VMware Tanzu Kubernetes Grid（TKG）或 Tanzu Kubernetes Grid 整合版（TKGi）叢集上代管 Astra Control Center 時、請謹記下列考量事項。
 - 預設的VMware TKG和TKGi組態檔案權杖會在部署後10小時內過期。如果您使用Tanzu產品組合產品、則必須產生一個含有非過期權杖的Tanzu Kubernetes叢集組態檔、以避免Astra Control Center與託管應用程式叢集之間發生連線問題。如需相關指示、請造訪 ["VMware NSxT-T資料中心產品文件。"](#)
 - 使用 `kubectl get nsxlbmonitors -A` 命令以查看您是否已設定服務監視器以接受入口流量。如果存在、則不應安裝MetalLB、因為現有的服務監視器將會覆寫任何新的負載平衡器組態。
 - 停用要由Astra Control管理的任何應用程式叢集上的TKG或TKGi預設儲存類別強制。您可以編輯來執行此作業 `TanzuKubernetesCluster` 命名空間叢集上的資源。
 - 在 TKG 或 TKGi 環境中部署 Astra Control Center 時、請注意 Astra Control Provisioner 的特定需求：
 - 叢集必須支援特殊權限的工作負載。
 - `--kubelet-dir` 旗標應設定為kubelet目錄的位置。依預設、這是 `/var/vcap/data/kubelet`。
 - 使用指定kubelet位置 `--kubelet-dir` 已知適用於Trident運算子、Helm和 `tridentctl` 部署：

服務網狀網路需求

強烈建議您在 Astra Control Center 主機叢集上安裝支援的 Istio 服務網格香草版本。請參閱 ["支援版本"](#) 適用於受支援版本的 Istio。Istio 服務網格（例如 OpenShift Service Mesh）的品牌版本未通過 Astra Control Center 驗證。

若要將 Astra Control Center 與主機叢集上安裝的 Istio 服務網格整合、您必須將其整合為 Astra Control Center 的一部分 **"安裝"** 而且不受此程序的影響。



在主機叢集上安裝和使用 Astra Control Center 而不設定服務網格、可能會對安全性造成嚴重影響。

Astra Trident

如果您打算在此版本中使用 Astra Trident 而非 Astra Control Provisioner、則支援 Astra Trident 23.04 及更新版本。Astra Control Center 需要 [Astra Control 資源配置程式](#) 在未來版本中。

Astra Control 資源配置程式

若要使用 Astra Control Provisioner 進階儲存功能、您必須安裝 Astra Trident 23.10 或更新版本並啟用 ["Astra Control Provisioner 功能"](#)。若要使用最新的 Astra Control Provisioner 功能、您需要最新版本的 Astra Trident 和 Astra Control Center。

- * 與 Astra Control Center 搭配使用的最低 Astra Control Provisioner 版本 * : Astra Control Provisioner 23.10 或更新版本已安裝及設定。

採用 **Astra Trident** 的 **ONTAP** 組態

- * 儲存類別 * : 在叢集上至少設定一個儲存類別。如果已設定預設儲存類別、請確定它是唯一具有預設指定的儲存類別。
- * 儲存驅動程式與工作節點 * : 請務必使用適當的儲存驅動程式來設定叢集中的工作節點、以便 Pod 與後端儲存設備互動。Astra Control Center支援ONTAP Astra Trident提供的下列支援資訊驅動程式：
 - `ontap-nas`
 - `ontap-san`
 - `ontap-san-economy` (此儲存類別類型無法使用應用程式複寫)
 - `ontap-nas-economy` (此儲存類別類型無法使用快照和應用程式複寫原則)

儲存設備後端

確保您擁有支援的後端、且具有足夠的容量。

- * 所需的儲存後端容量 * : 可用容量至少 500 GB
- * 支援的後端 * : Astra Control Center 支援下列儲存後端：
 - NetApp ONTAP 9.9.1 或更新版本的 AFF、FAS 和 ASA 系統
 - NetApp ONTAP Select 9.9.1 或更新版本
 - NetApp Cloud Volumes ONTAP 9.9.1 或更新版本
 - (適用於 Astra Control Center 技術預覽) NetApp ONTAP 9.10.1 或更新版本、適用於技術預覽所提供的資料保護作業
 - Longhorn 1.5.0 或更新版本
 - 需要手動建立 Volume SnapshotClass 物件。請參閱 ["Longhorn 文件"](#) 以取得相關指示。
 - NetApp MetroCluster
 - 託管 Kubernetes 叢集必須採用彈性組態。
 - 支援的雲端供應商可提供儲存設備後端

不需要授權 **ONTAP**

若要使用Astra Control Center、請視ONTAP 您需要完成的工作而定、確認您擁有下列各項的版次授權：

- FlexClone
- SnapMirror：選用。僅使用SnapMirror技術複寫至遠端系統時才需要。請參閱 ["SnapMirror授權資訊"](#)。
- S3授權：選用。僅適用於SS3鏟斗ONTAP

若要檢查ONTAP 您的不實系統是否有必要的授權、請參閱 ["管理ONTAP 不需購買的授權"](#)。

NetApp MetroCluster

當您使用 NetApp MetroCluster 作為儲存後端時、必須執行下列動作：

- 在您使用的 Astra Trident 驅動程式中、將 SVM 管理 LIF 指定為後端選項
- 請確定您擁有適當的 ONTAP 授權

若要設定 MetroCluster LIF 、請參閱每個驅動程式的下列選項和範例：

- "SAN"
- "NAS"

Astra Control Center 授權

Astra Control Center 需要 Astra Control Center 授權。安裝 Astra Control Center 時、已啟動內嵌式 90 天試用版授權、可用於 4 、 800 個 CPU 單元。如果您需要更多容量或不同的評估條款、或想要升級至完整授權、您可以向 NetApp 取得不同的評估授權或完整授權。您需要授權來保護應用程式和資料。

您可以報名免費試用 Astra Control Center 。您可以註冊註冊 ["請按這裡"](#) 。

若要設定授權、請參閱 ["使用90天試用版授權"](#) 。

若要深入瞭解授權的運作方式、請參閱 ["授權"](#) 。

網路需求

設定您的營運環境、確保 Astra Control Center 能夠正常通訊。需要下列網路組態：

- * FQDN 位址 * ：您必須擁有 Astra Control Center 的 FQDN 位址。
- * 存取網際網路 * ：您應該判斷是否有外部存取網際網路的權限。如果您不這麼做、某些功能可能會受到限制、例如將支援服務組合傳送至 ["NetApp 支援網站"](#) 。
- * 連接埠存取 * ：裝載 Astra Control Center 的作業環境使用下列 TCP 連接埠進行通訊。您應確保這些連接埠可透過任何防火牆、並設定防火牆、以允許來自Astra網路的任何HTTPS輸出流量。有些連接埠需要在裝載Astra Control Center的環境與每個託管叢集之間進行連線（視情況而定）。



您可以在雙堆疊Kubernetes叢集中部署Astra Control Center、Astra Control Center則可管理已設定為雙堆疊作業的應用程式和儲存後端。如需雙堆疊叢集需求的詳細資訊、請參閱 ["Kubernetes 文件"](#) 。

來源	目的地	連接埠	傳輸協定	目的
用戶端PC	Astra控制中心	443..	HTTPS	UI / API 存取 - 確保 Astra Control Center 和用於存取 Astra Control Center 的系統之間雙向開啟此連接埠

來源	目的地	連接埠	傳輸協定	目的
度量使用者	Astra Control Center 工作節點	9090	HTTPS	度量資料通訊：確保每個託管叢集都能存取裝載Astra Control Center的叢集上的此連接埠（需要雙向通訊）
Astra控制中心	Amazon S3儲存貯體供應商	443..	HTTPS	Amazon S3儲存通訊
Astra控制中心	NetApp AutoSupport	443..	HTTPS	NetApp AutoSupport 通訊
Astra控制中心	託管 Kubernetes 叢集	443/643. * 注意 *：受管理叢集使用的連接埠可能會因叢集而異。請參閱叢集軟體廠商提供的文件。	HTTPS	與託管叢集通訊：確保此連接埠在託管 Astra Control Center 的叢集和每個託管叢集之間都是開放的

內部部署Kubernetes叢集的入口

您可以選擇網路入侵Astra控制中心的用途類型。依預設、Astra Control Center會將Astra Control Center閘道（服務/網路）部署為整個叢集的資源。Astra Control Center也支援使用服務負載平衡器（如果環境允許）。如果您想要使用服務負載平衡器、但尚未設定一個、則可以使用MetalLB負載平衡器自動將外部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。



負載平衡器應使用與Astra Control Center工作節點IP位址位於同一子網路中的IP位址。

如需詳細資訊、請參閱 ["設定入口以進行負載平衡"](#)。

Google Anthos 入口要求

在 Google Anthos 叢集上代管 Astra Control Center 時、請注意 Google Antos 預設包含 MetalLB 負載平衡器和 Istio 入口服務、讓您在安裝期間只需使用 Astra Control Center 的一般入口功能即可。請參閱 ["Astra Control Center 安裝文件"](#) 以取得詳細資料。

支援的網頁瀏覽器

Astra Control Center支援最新版本的Firefox、Safari和Chrome、最低解析度為1280 x 720。

應用程式叢集的其他需求

如果您打算使用這些Astra Control Center功能、請謹記以下要求：

- 應用程式叢集需求：["叢集管理需求"](#)
 - 受管理的應用程式需求：["應用程式管理需求"](#)
 - * 應用程式複寫的其他需求 *：["複寫先決條件"](#)

下一步

檢視 "[快速入門](#)" 總覽：

Astra Control Center快速入門

以下是使用Astra Control Center所需的步驟總覽。每個步驟中的連結都會帶您前往提供更多詳細資料的頁面。

1

檢閱Kubernetes叢集需求

確保您的環境符合下列需求：

- [Kubernetes叢集*](#)
- ["確保您的主機叢集符合作業環境需求"](#)
- ["設定內部部署Kubernetes叢集的負載平衡入口"](#)

儲存整合

- ["確保您的環境包含 Astra Control Provisioner"](#)
- ["啟用 Astra Control Provisioner 進階管理和儲存資源配置功能"](#)
- ["準備叢集工作者節點"](#)
- ["設定儲存設備後端"](#)
- ["設定儲存類別"](#)
- ["安裝 Volume Snapshot 控制器"](#)
- ["建立Volume Snapshot類別"](#)

不包含認證資料 ONTAP

- ["設定ONTAP 驗證資料"](#)

2

下載並安裝Astra Control Center

完成下列安裝工作：

- ["從 NetApp 支援網站 下載頁面下載 Astra 控制中心"](#)
- 取得NetApp授權檔案：
 - 如果您正在評估 Astra Control Center 、則已包含內嵌評估授權
 - ["如果您已購買Astra Control Center、請產生授權檔案"](#)
- ["安裝Astra Control Center"](#)
- ["執行其他選用的組態步驟"](#)

3

完成一些初始設定工作

完成一些基本工作以開始：

- ["新增授權"](#)
- ["為叢集管理做好準備"](#)
- ["新增叢集"](#)
- ["新增儲存後端"](#)
- ["新增儲存庫"](#)

4

使用Astra控制中心

完成 Astra Control Center 設定後、請使用 Astra Control UI 或 ["Astra Control API"](#) 若要開始管理及保護應用程式：

- ["管理帳戶"](#)：使用者、角色、LDAP、認證等。
- ["管理通知"](#)
- ["管理應用程式"](#)：定義要管理的資源。
- ["保護應用程式"](#)：設定保護原則、並複寫、複製及移轉應用程式。

以取得更多資訊

- ["使用Astra Control API"](#)
- ["升級Astra Control Center"](#)
- ["取得Astra Control的協助"](#)

安裝總覽

選擇並完成下列其中一個Astra Control Center安裝程序：

- ["使用標準程序安裝Astra Control Center"](#)
- ["（如果您使用Red Hat OpenShift）使用OpenShift作業系統集線器安裝Astra Control Center"](#)
- ["安裝Astra Control Center搭配Cloud Volumes ONTAP 一套功能性儲存後端"](#)

視您的環境而定、安裝Astra Control Center之後可能需要額外的組態：

- ["安裝後設定Astra Control Center"](#)

使用標準程序安裝Astra Control Center

若要安裝 Astra Control Center、請下載安裝映像並執行下列步驟。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。

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

開始之前

- * 符合環境先決條件 * : ["開始安裝之前、請先準備好環境以進行Astra Control Center部署"](#)。



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。

- * 確保健康服務 * : 檢查所有 API 服務是否均處於健全狀態且可用：

```
kubectl get apiservices
```

- * 確保可路由的 FQDN* : 您打算使用的 Astra FQDN 可路由至叢集。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。
- * 設定憑證管理員 * : 如果叢集中已存在憑證管理員、您需要執行一些動作 ["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。
- * (僅限 ONTAP SAN 驅動程式) 啟用多重路徑 * : 如果您使用的是 ONTAP SAN 驅動程式、請務必在所有 Kubernetes 叢集上啟用多重路徑。

您也應該考慮下列事項：

- * 取得 NetApp Astra 控制影像登錄的存取權限 * :

您可以選擇從 NetApp 映像登錄取得 Astra Control 的安裝映像和功能增強功能、例如 Astra Control Provisioner。

- a. 記錄您登入登錄所需的 Astra Control 帳戶 ID。

您可以在 Astra Control Service 網頁 UI 中看到您的帳戶 ID。選取頁面右上角的圖示、選取 * API access*、然後寫下您的帳戶 ID。

- b. 從同一頁面選取 * 產生 API 權杖 *、然後將 API 權杖字串複製到剪貼簿、並將其儲存在編輯器中。

- c. 登入 Astra Control 登錄：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

- * 安裝服務網格以進行安全通訊 * : 強烈建議使用來保護 Astra Control 主機叢集通訊通道 ["支援的服務網格"](#)。



只有在 Astra Control Center 期間、才能將 Astra Control Center 與服務網格整合 ["安裝"](#) 而且不受此程序的影響。不支援從網格環境變更回無網格環境。

若要使用 Istio 服務網格、您必須執行下列動作：

- 新增 `istio-injection:enabled` [標籤](#) 部署 Astra Control Center 之前先移至 Astra 命名空間。

- 使用 Generic [入口設定](#) 並為提供替代入口 [外部負載平衡](#)。
- 對於 Red Hat OpenShift 叢集、您需要定義 NetworkAttachmentDefinition 在所有相關的 Astra Control Center 命名空間上 (netapp-acc-operator、netapp-acc、netapp-monitoring 應用程式叢集或任何已取代的自訂命名空間)。

```
cat <<EOF | oc -n netapp-acc-operator create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-acc create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-monitoring create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF
```

步驟

若要安裝Astra Control Center、請執行下列步驟：

- [下載並擷取Astra Control Center](#)
- [\[如果您使用本機登錄、請完成其他步驟\]](#)
- [\[設定具有驗證需求之登錄的命名空間和機密\]](#)
- [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

從下列其中一個位置下載 Astra Control Center 映像：

- ***Astra Control Service 映像登錄 ***：如果您不將本機登錄與 Astra Control Center 映像搭配使用、或是您偏好從 NetApp 支援網站 下載套件時使用此方法、請使用此選項。
- *** NetApp 支援網站 ***：如果您將本機登錄與 Astra 控制中心映像搭配使用、請使用此選項。

Astra Control 映像登錄

1. 登入 Astra Control Service 。
2. 在儀表板上、選取 * 部署自動管理的 Astra Control* 執行個體。
3. 依照指示登入 Astra Control 映像登錄、拉出 Astra Control Center 安裝映像、並擷取映像。

NetApp 支援網站

1. 下載包含Astra Control Center的套裝組合 (astra-control-center-[version].tar.gz) 從 ["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。

```
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
```

如果您使用本機登錄、請完成其他步驟

如果您打算將 Astra Control Center 套裝軟體推送至本機登錄、則需要使用 NetApp Astra kubectl 命令列外掛程式。

安裝NetApp Astra kubectl外掛程式

完成這些步驟以安裝最新的 NetApp Astra Kubectl 命令列外掛程式。

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
```

將映像新增至登錄

1. 如果您打算將 Astra Control Center 套件推送至本機登錄、請為您的容器引擎完成適當的步驟順序：

Docker

- a. 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
```

- b. 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

- a. 切換到tar檔案的根目錄。您應該會看到這個檔案和目錄：

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

- b. 登入您的登錄：

```
podman login <YOUR_REGISTRY>
```

- c. 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=24.02.0-69
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=24.02.0-69
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://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version

```

2. 變更目錄：

```
cd manifests
```

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 Kubeconfig：

```
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：

- a. 建立「NetApp-acc operator」：

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

- b. 建立「NetApp-acc operator」命名空間的秘密。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/24.02.0-69`）。

```
kubectl create secret docker-registry astra-registry-cred -n netapp-acc-operator --docker-server=cr.astra.netapp.io --docker-username=[astra_account_id] --docker-password=[astra_api_token]
```

+

```
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。

- a. 建立 `netapp-acc`（或自訂命名）命名空間。

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

- b. 為建立秘密 `netapp-acc`（或自訂命名）命名空間。根據您的登錄偏好設定、新增 Docker 資訊並執行其中一個適當的命令：

```
kubectl create secret docker-registry astra-registry-cred -n [netapp-acc or custom namespace] --docker-server=cr.astra.netapp.io --docker-username=[astra_account_id] --docker-password=[astra_api_token]
```

```
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. (僅限本機登錄) 如果您使用本機登錄、請完成下列步驟：

a. 開啟 Astra Control Center 營運商部署 YAML：

```
vim astra_control_center_operator_deploy.yaml
```



附註的Y反 洗錢範例遵循下列步驟。

b. 如果您使用需要驗證的登錄、請將預設行「imagePullSecrets：[]」改為：

```
imagePullSecrets: [{name: astra-registry-cred}]
```

c. 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](#)。

d. 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:24.02.68
        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
```

2. 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
```

3. 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：

帳戶名稱

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

astraVersion

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	24.02.0-69

astraAddress

設定	指導	類型	範例
<code>astraAddress</code>	變更 <code>astraAddress</code> 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"。 附註：請勿使用 <code>http://</code> 或 <code>https://</code> 地址中。複製此FQDN以供在中使用 後續步驟。	字串	<code>astra.example.com</code>

AutoSupport

您在本節中所做的選擇、決定您是否要參與 NetApp 的主動式支援應用程式、數位顧問、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> for 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

電子郵件

設定	指導	類型	範例
email	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 。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	admin@example.com

名字

設定	指導	類型	範例
firstName	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

姓氏

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

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
<code>imageRegistry.name</code>	必要	Astra Control 映像登錄的名稱、主控部署 Astra Control Center 所需的所有映像。此值將預先填入、除非您已設定本機登錄、否則不需要採取任何行動。如果是本機登錄、請將此現有值取代為您在其中推入影像的映像登錄名稱 上一步 。請勿使用 <code>http://</code> 或 <code>https://</code> 在登錄名稱中。	字串	<code>cr.astra.netapp.io</code> (預設) <code>example.registry.com/astra</code> (本機登錄範例)
<code>imageRegistry.secret</code>	選用	用來驗證映像登錄的Kubernetes機密名稱。此值會預先填入、除非您已設定本機登錄和為該登錄輸入的字串、否則不需要採取任何行動 <code>imageRegistry.name</code> 需要密碼。 重要事項：如果您使用的本機登錄不需要授權、則必須刪除此項目 <code>secret</code> 行內 <code>imageRegistry</code> 否則安裝將會失敗。	字串	<code>astra-registry-cred</code>

storageClass

設定	指導	類型	範例
storageClass	變更 storageClass 價值來源 <code>ontap-gold</code> 視安裝需求而定、移轉至其他 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 Control Provisioner 設定的儲存類別 (<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。 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。	字串	<code>ontap-gold</code>

Volume ReclaimPolicy

設定	指導	類型	選項
volumeReclaimPolicy	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。	字串	• <code>Retain</code> (這是預設值) • <code>Delete</code>



設定	指導	類型	選項
ingressType	使用下列其中一種入口類型： 通用 (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。Astra Control Center 部署完成後、您需要設定 "入口控制器" 使用URL公開Astra Control Center。 重要事項：如果您打算搭配 Astra Control Center 使用服務網狀網路、則必須選取 Generic 進入類型、自行設定 "入口控制器"。 * 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。 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。 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	字串	• Generic (這是預設值) • AccTraefik

scaleSize

設定	指導	類型	選項
scaleSize	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。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。	字串	• Small • Medium（這是預設值）

astraResources2200

設定	指導	類型	選項
astraResourcesScaler	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。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。	字串	• Default（這是預設值） • Off



在 Astra Control Center CR 中新增下列其他值、以避免安裝中出現已知問題：

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

客戶需求日

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>	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。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。	布林值	<code>False</code> （此值為預設值）
<code>crds.externalTraefik</code>	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。	布林值	<code>False</code> （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。

Astra 控制中心 .yaml 範例

```
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: "[cr.astra.netapp.io or your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
  volumeReclaimPolicy: "Retain"
  ingressType: "Generic"
  scaleSize: "Medium"
  astraResourcesScaler: "Default"
  additionalValues:
    keycloak-operator:
      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操作、請建立「NetApp-acc」（或自訂）命名空間：

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

2. 如果您是搭配 Astra Control Center 使用服務網格、請將下列標籤新增至 netapp-acc 或自訂命名空間：



您的入口類型 (ingressType) 必須設為 Generic 在 Astra Control Center CR 中執行此命令之前。

```
kubectl label ns [netapp-acc or custom namespace] istio-  
injection:enabled
```

3. (建議) "啟用嚴格的 MTLS" 對於 Istio 服務網格：

```
kubectl apply -n istio-system -f - <<EOF  
apiVersion: security.istio.io/v1beta1  
kind: PeerAuthentication  
metadata:  
  name: default  
spec:  
  mtls:  
    mode: STRICT  
EOF
```

4. 在「NetApp-acc」（或您的自訂）命名空間中安裝Astra Control Center：

```
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「執行中」。部署系統Pod可能需要幾分鐘的時間。

acc-helm-repo-5bd77c9ddd-8wxm2 1h	1/1	Running	0
activity-5bb474dc67-8l9ss 1h	1/1	Running	0
activity-5bb474dc67-qbrtq 1h	1/1	Running	0
api-token-authentication-6wbj2 1h	1/1	Running	0
api-token-authentication-9pgw6 1h	1/1	Running	0
api-token-authentication-tqf6d 1h	1/1	Running	0
asup-5495f44dbd-z4kft 1h	1/1	Running	0
authentication-6fdd899858-5x45s 1h	1/1	Running	0
bucket-service-84d47487d-n9xgp 1h	1/1	Running	0
bucket-service-84d47487d-t5jhm 1h	1/1	Running	0
cert-manager-5dcb7648c4-hbldc 1h	1/1	Running	0
cert-manager-5dcb7648c4-nr9qf 1h	1/1	Running	0
cert-manager-cainjector-59b666fb75-bk2tf 1h	1/1	Running	0
cert-manager-cainjector-59b666fb75-pfnck 1h	1/1	Running	0
cert-manager-webhook-c6f9b6796-ngz2x 1h	1/1	Running	0
cert-manager-webhook-c6f9b6796-rwtbn 1h	1/1	Running	0
certificates-5f5b7b4dd-52tnj 1h	1/1	Running	0
certificates-5f5b7b4dd-gtjbx 1h	1/1	Running	0
certificates-expiry-check-28477260-dz5vw 1h	0/1	Completed	0
cloud-extension-6f58cc579c-lzfmv 1h	1/1	Running	0
cloud-extension-6f58cc579c-zw2km 1h	1/1	Running	0
cluster-orchestrator-79dd5c8d95-qjg92	1/1	Running	0

1h			
composite-compute-85dc84579c-nz82f	1/1	Running	0
1h			
composite-compute-85dc84579c-wx2z2	1/1	Running	0
1h			
composite-volume-bff6f4f76-789nj	1/1	Running	0
1h			
composite-volume-bff6f4f76-kwnd4	1/1	Running	0
1h			
credentials-79fd64f788-m7m8f	1/1	Running	0
1h			
credentials-79fd64f788-qnc6c	1/1	Running	0
1h			
entitlement-f69cdbd77-4p2kn	1/1	Running	0
1h			
entitlement-f69cdbd77-hswm6	1/1	Running	0
1h			
features-7b9585444c-7xd7m	1/1	Running	0
1h			
features-7b9585444c-dcqwc	1/1	Running	0
1h			
fluent-bit-ds-crq8m	1/1	Running	0
1h			
fluent-bit-ds-gmgq8	1/1	Running	0
1h			
fluent-bit-ds-gzr4f	1/1	Running	0
1h			
fluent-bit-ds-j6sf6	1/1	Running	0
1h			
fluent-bit-ds-v4t9f	1/1	Running	0
1h			
fluent-bit-ds-x7j59	1/1	Running	0
1h			
graphql-server-6cc684fb46-2x8lr	1/1	Running	0
1h			
graphql-server-6cc684fb46-bshbd	1/1	Running	0
1h			
hybridauth-84599f79fd-fjc7k	1/1	Running	0
1h			
hybridauth-84599f79fd-s9pmn	1/1	Running	0
1h			
identity-95df98cb5-dvlmz	1/1	Running	0
1h			
identity-95df98cb5-krf59	1/1	Running	0
1h			
influxdb2-0	1/1	Running	0

1h			
keycloak-operator-6d4d688697-cfq8b	1/1	Running	0
1h			
krakend-5d5c8f4668-7bq8g	1/1	Running	0
1h			
krakend-5d5c8f4668-t8hbn	1/1	Running	0
1h			
license-689cdd4595-2gsc8	1/1	Running	0
1h			
license-689cdd4595-g6vwk	1/1	Running	0
1h			
login-ui-57bb599956-4fwgz	1/1	Running	0
1h			
login-ui-57bb599956-rhztb	1/1	Running	0
1h			
loki-0	1/1	Running	0
1h			
metrics-facade-846999bdd4-f7jdm	1/1	Running	0
1h			
metrics-facade-846999bdd4-lnsxl	1/1	Running	0
1h			
monitoring-operator-6c9d6c4b8c-ggkrl	2/2	Running	0
1h			
nats-0	1/1	Running	0
1h			
nats-1	1/1	Running	0
1h			
nats-2	1/1	Running	0
1h			
natssync-server-6df7d6cc68-9v2gd	1/1	Running	0
1h			
nautilus-64b7fbdd98-bsgwb	1/1	Running	0
1h			
nautilus-64b7fbdd98-djlhw	1/1	Running	0
1h			
openapi-864584bccc-75nlv	1/1	Running	0
1h			
openapi-864584bccc-zh6bx	1/1	Running	0
1h			
polaris-consul-consul-server-0	1/1	Running	0
1h			
polaris-consul-consul-server-1	1/1	Running	0
1h			
polaris-consul-consul-server-2	1/1	Running	0
1h			
polaris-keycloak-0	1/1	Running	2 (1h

ago)	1h			
polaris-keycloak-1		1/1	Running	0
1h				
polaris-keycloak-db-0		1/1	Running	0
1h				
polaris-keycloak-db-1		1/1	Running	0
1h				
polaris-keycloak-db-2		1/1	Running	0
1h				
polaris-mongodb-0		1/1	Running	0
1h				
polaris-mongodb-1		1/1	Running	0
1h				
polaris-mongodb-2		1/1	Running	0
1h				
polaris-ui-66476dcf87-f6s8j		1/1	Running	0
1h				
polaris-ui-66476dcf87-ztjk7		1/1	Running	0
1h				
polaris-vault-0		1/1	Running	0
1h				
polaris-vault-1		1/1	Running	0
1h				
polaris-vault-2		1/1	Running	0
1h				
public-metrics-bfc4fc964-x4m79		1/1	Running	0
1h				
storage-backend-metrics-7dbb88d4bc-g78cj		1/1	Running	0
1h				
storage-provider-5969b5df5-hjvcm		1/1	Running	0
1h				
storage-provider-5969b5df5-r79ld		1/1	Running	0
1h				
task-service-5fc9dc8d99-4q4f4		1/1	Running	0
1h				
task-service-5fc9dc8d99-8l5zl		1/1	Running	0
1h				
task-service-task-purge-28485735-fdzkd		1/1	Running	0
12m				
telegraf-ds-2rgm4		1/1	Running	0
1h				
telegraf-ds-4qp6r		1/1	Running	0
1h				
telegraf-ds-77frs		1/1	Running	0
1h				
telegraf-ds-bc725		1/1	Running	0

1h			
telegraf-ds-cvmxf	1/1	Running	0
1h			
telegraf-ds-tqzgj	1/1	Running	0
1h			
telegraf-rs-5wtd8	1/1	Running	0
1h			
telemetry-service-6747866474-5djnc	1/1	Running	0
1h			
telemetry-service-6747866474-thb7r	1/1	Running	1 (1h
ago)			
1h			
tenancy-5669854fb6-gzdzf	1/1	Running	0
1h			
tenancy-5669854fb6-xvsm2	1/1	Running	0
1h			
traefik-8f55f7d5d-4lgfw	1/1	Running	0
1h			
traefik-8f55f7d5d-j4wt6	1/1	Running	0
1h			
traefik-8f55f7d5d-p6gcq	1/1	Running	0
1h			
trident-svc-7cb5bb4685-54cnq	1/1	Running	0
1h			
trident-svc-7cb5bb4685-b28xh	1/1	Running	0
1h			
vault-controller-777b9bbf88-b5bqt	1/1	Running	0
1h			
vault-controller-777b9bbf88-fdfd8	1/1	Running	0
1h			

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	24.02.0-69	
10.111.111.111	True		



複製UUID值。密碼為「ACC-」、後面接著UUID值（「ACC-[UUUID]」、或是在本範例中為「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。

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「istio-inple.yaml」檔案中提供的「pec.tls.secretName」相符。

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Nginvin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 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

如果有任何服務處於「錯誤」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
```

替代安裝程序

- * 安裝 Red Hat OpenShift OperatorHub *：請使用此選項 ["替代程序"](#) 使用 OperatorHub 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。

下一步

- (選用) 視您的環境而定、請在安裝後完成 ["組態步驟"](#)。
- ["安裝 Astra Control Center、登入 UI 並變更密碼之後、您將需要設定授權、新增叢集、啟用驗證、管理儲存設備及新增儲存區"](#)。

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-66dc99cc56-9ldmt	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。

使用OpenShift作業系統集線器安裝Astra Control Center

如果您使用Red Hat OpenShift、可以使用Red Hat認證的操作員來安裝Astra Control Center。請使用此程序從安裝Astra Control Center "[Red Hat生態系統目錄](#)" 或使用Red Hat OpenShift Container Platform。

完成此程序之後、您必須返回安裝程序、才能完成 "[剩餘步驟](#)" 以驗證安裝是否成功並登入。

開始之前

- * 符合環境先決條件 * : "[開始安裝之前、請先準備好環境以進行Astra Control Center部署](#)"。



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。

- * 確保健全的叢集操作員和 API 服務 * :
 - 從OpenShift叢集確保所有叢集操作員都處於健全狀態：

```
oc get clusteroperators
```

- 從OpenShift叢集、確保所有API服務都處於健全狀態：

```
oc get apiservices
```

- * 確保可路由的 FQDN* : 您打算使用的 Astra FQDN 可路由至叢集。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。
- * 取得 OpenShift 權限 * : 您將需要所有必要的權限、以及對 Red Hat OpenShift Container Platform 的存取權限、才能執行所述的安裝步驟。
- * 設定憑證管理員 * : 如果叢集中已存在憑證管理員、您需要執行一些動作 "[必要步驟](#)" 因此Astra Control Center不會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。
- * 設定 Kubernetes 入口控制器 * : 如果您的 Kubernetes 入口控制器負責管理外部服務存取、例如叢集中的

負載平衡、則您必須將其設定為與 Astra Control Center 搭配使用：

- a. 建立運算子命名空間：

```
oc create namespace netapp-acc-operator
```

- b. **"完成設定"** 適用於您的入口控制器類型。

- *（僅限 ONTAP SAN 驅動程式）啟用多重路徑*：如果您使用的是 ONTAP SAN 驅動程式、請務必在所有 Kubernetes 叢集上啟用多重路徑。

您也應該考慮下列事項：

- *取得 NetApp Astra 控制影像登錄的存取權限*：

您可以選擇從 NetApp 映像登錄取得 Astra Control 的安裝映像和功能增強功能、例如 Astra Control Provisioner。

- a. 記錄您登入登錄所需的 Astra Control 帳戶 ID。

您可以在 Astra Control Service 網頁 UI 中看到您的帳戶 ID。選取頁面右上角的圖示、選取 *API access*、然後寫下您的帳戶 ID。

- b. 從同一頁面選取 *產生 API 權杖*、然後將 API 權杖字串複製到剪貼簿、並將其儲存在編輯器中。

- c. 登入 Astra Control 登錄：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

- *安裝服務網格以進行安全通訊*：強烈建議使用來保護 Astra Control 主機叢集通訊通道 **"支援的服務網格"**。



只有在 Astra Control Center 期間、才能將 Astra Control Center 與服務網格整合 **"安裝"** 而且不受此程序的影響。不支援從網格環境變更回無網格環境。

若要使用 Istio 服務網格、您必須執行下列動作：

- 新增 `istio-injection:enabled` 在部署 Astra Control Center 之前、請先將標籤貼到 Astra 命名空間。
- 使用 Generic **入口設定** 並為提供替代入口 **"外部負載平衡"**。
- 對於 Red Hat OpenShift 叢集、您需要定義 `NetworkAttachmentDefinition` 在所有相關的 Astra Control Center 命名空間上 (`netapp-acc-operator`、`netapp-acc`、`netapp-monitoring` 應用程式叢集或任何已取代的自訂命名空間)。

```
cat <<EOF | oc -n netapp-acc-operator create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-acc create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-monitoring create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF
```

步驟

- [下載並擷取Astra Control Center](#)
- [\[如果您使用本機登錄、請完成其他步驟\]](#)
- [\[尋找操作員安裝頁面\]](#)
- [\[安裝操作員\]](#)
- [安裝Astra Control Center](#)



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

從下列其中一個位置下載 Astra Control Center 影像：

- ***Astra Control Service 映像登錄***：如果您不將本機登錄與 Astra Control Center 映像搭配使用、或是您偏好從 NetApp 支援網站 下載套件時使用此方法、請使用此選項。
- *** NetApp 支援網站 ***：如果您將本機登錄與 Astra 控制中心影像搭配使用、請使用此選項。

Astra Control 影像登錄

1. 登入 Astra Control Service 。
2. 在儀表板上、選取 * 部署自動管理的 Astra Control* 執行個體。
3. 依照指示登入 Astra Control 影像登錄、拉出 Astra Control Center 安裝映像、並擷取映像。

NetApp 支援網站

1. 下載包含Astra Control Center的套裝組合 (astra-control-center-[version].tar.gz) 從 "[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。

```
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
```

如果您使用本機登錄、請完成其他步驟

如果您打算將 Astra Control Center 套裝軟體推送至本機登錄、則需要使用 NetApp Astra kubectl 命令列外掛程式。

安裝NetApp Astra kubectl外掛程式

完成這些步驟以安裝最新的 NetApp Astra Kubectl 命令列外掛程式。

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、並記下作業系統和CPU架構所需的檔案名稱：



KECBECTI外掛程式庫是tar套件的一部分、會擷取到資料夾中 `kubect1-astra`。

```
ls kubect1-astra/
```

2. 將正確的二進位檔移至目前路徑、並將其重新命名為 `kubect1-astra`：

```
cp kubect1-astra/<binary-name> /usr/local/bin/kubect1-astra
```

將映像新增至登錄

1. 如果您打算將 Astra Control Center 套件推送至本機登錄、請為您的容器引擎完成適當的步驟順序：

Docker

- a. 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
```

- b. 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

- a. 切換到tar檔案的根目錄。您應該會看到這個檔案和目錄：

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

- b. 登入您的登錄：

```
podman login <YOUR_REGISTRY>
```

- c. 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=24.02.0-69
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=24.02.0-69
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://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version

```

2. 變更目錄：

```
cd manifests
```

尋找操作員安裝頁面

1. 請完成下列其中一個程序、以存取操作員安裝頁面：

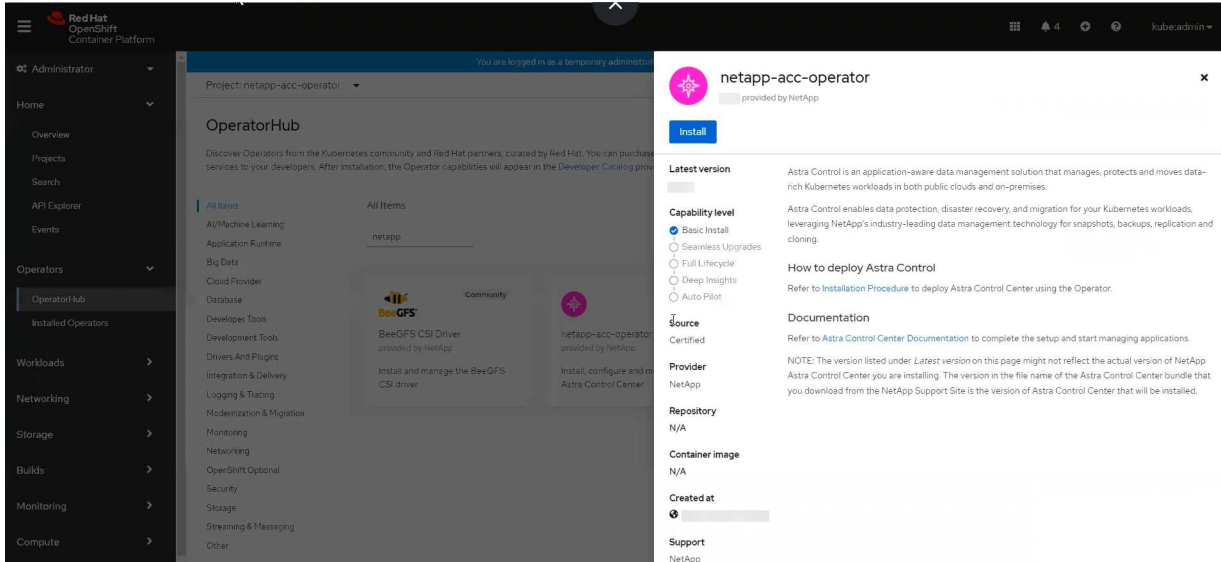
Red Hat OpenShift Web 主控台

- 登入 OpenShift Container Platform UI。
- 從側功能表中、選取*運算子>運算子中樞*。



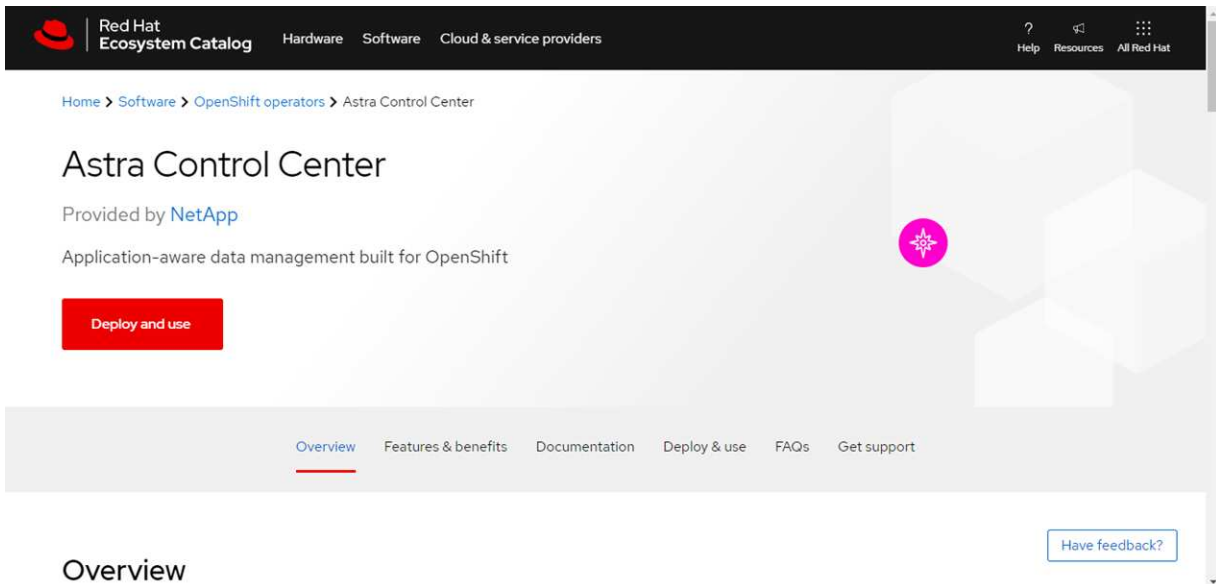
您只能使用此運算子升級至 Astra Control Center 的目前版本。

- 搜尋 netapp-acc 然後選取 NetApp Astra 控制中心操作員。



Red Hat生態系統目錄

- 選擇 NetApp Astra Control Center "營運者"。
- 選擇 * 部署與使用 *。



安裝操作員

1. 完成*安裝操作員*頁面並安裝操作員：



此運算子可用於所有叢集命名空間。

- a. 在操作員安裝過程中、系統會自動建立運算子命名空間或「NetApp-acc operator」命名空間。
- b. 選取手動或自動核准策略。



建議手動核准。每個叢集只能執行單一運算子執行個體。

- c. 選擇*安裝*。

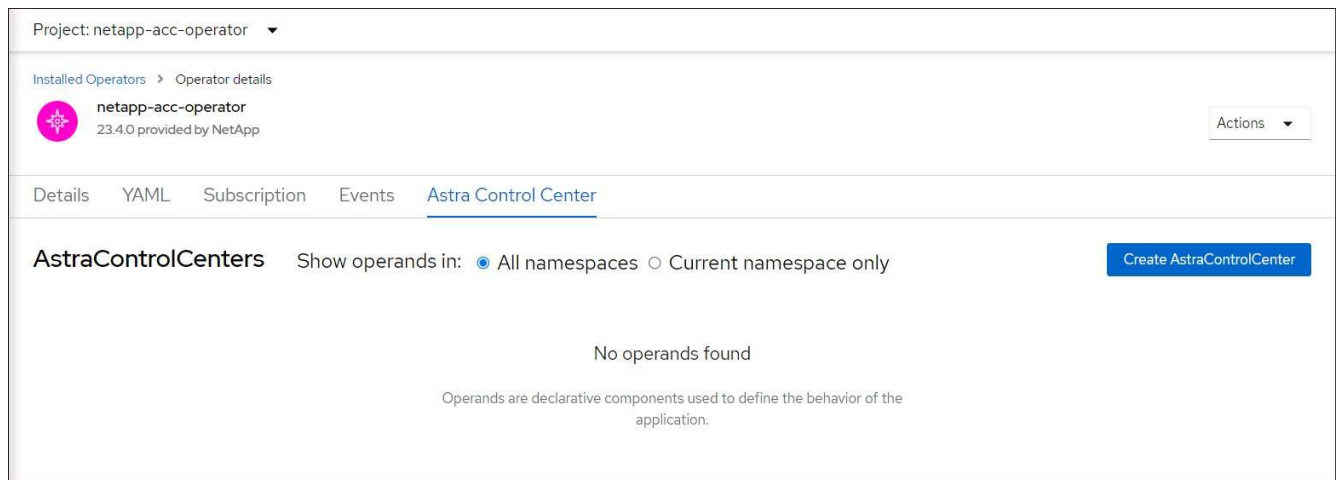


如果您選擇手動核准策略、系統會提示您核准此操作員的手動安裝計畫。

2. 從主控台移至「作業系統集線器」功能表、確認操作員已成功安裝。

安裝Astra Control Center

1. 從Astra控制中心操作員* Astra控制中心*索引標籤內的主控台、選取*建立適用的*



2. 填寫「Create適用的」表單欄位：

- a. 保留或調整Astra Control Center名稱。
- b. 新增Astra Control Center的標籤。
- c. 啟用或停用自動支援。建議保留「自動支援」功能。
- d. 輸入Astra Control Center FQDN或IP位址。請勿進入 http:// 或 https:// 在「地址」欄位中。
- e. 輸入 Astra Control Center 版本、例如 24.02.0-69。
- f. 輸入帳戶名稱、電子郵件地址和管理員姓氏。
- g. 選擇的Volume回收原則 Retain、Recycle、或 Delete。預設值為 Retain。
- h. 選取安裝的規模大小。



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。

i. 選取入口類型：

- 通用 (`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。Astra Control Center 部署完成後、您需要設定 ["入口控制器"](#) 使用URL公開Astra Control Center。

- `* 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` 閘道即 Kubernetes 「負載平衡器」類型服務。

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「負載平衡器」和入口服務類型的詳細資訊、請參閱 ["需求"](#)。

- 在 `* 映像登錄 *` 中，除非您設定本機登錄，否則請使用預設值。如果是本機登錄、請將此值取代為您在上一步驟中推入影像的本機映像登錄路徑。請勿進入 `http://` 或 `https://` 在「地址」欄位中。
- 如果您使用需要驗證的映像登錄、請輸入映像秘密。



如果您使用需要驗證的登錄、[在叢集上建立秘密](#)。

- 輸入管理員名字。
- 設定資源擴充。
- 提供預設的儲存類別。



如果已設定預設儲存類別、請確定它是唯一具有預設註釋的儲存類別。

- 定義客戶需求日處理偏好設定。

- 選取「Yaml」檢視以檢閱您所選的設定。
- 選取「Create」（建立）。

建立登錄機密

如果您使用需要驗證的登錄、請在 OpenShift 叢集上建立密碼、然後在中輸入密碼名稱 `Create AstraControlCenter` 表單欄位。

- 為Astra Control Center運算子建立命名空間：

```
oc create ns [netapp-acc-operator or custom namespace]
```

2. 在此命名空間中建立秘密：

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



Astra Control僅支援Docker登錄機密。

3. 填寫中的其餘欄位「[Create](#)」（[建立](#)）「[吧！Control Center](#)」表單欄位。

下一步

完成 "[剩餘步驟](#)" 若要驗證Astra Control Center是否安裝成功、請設定入口控制器（選用）、然後登入UI。此外、您還需要執行 "[設定工作](#)" 安裝完成後。

安裝Astra Control Center搭配Cloud Volumes ONTAP 一套功能性儲存後端

有了Astra Control Center、您就能在混合雲環境中使用自我管理的Kubernetes叢集和Cloud Volumes ONTAP 實例來管理應用程式。您可以在內部部署的 Kubernetes 叢集或雲端環境中的自我管理 Kubernetes 叢集中部署 Astra Control Center。

有了其中一項部署、您就能使用Cloud Volumes ONTAP 下列其中一項部署、以下列方式執行應用程式資料管理作業：將NetApp當成儲存後端。您也可以將S3儲存區設定為備份目標。

若要在Amazon Web Services（AWS）、Google Cloud Platform（GCP）和Microsoft Azure中安裝Astra Control Center、並搭配Cloud Volumes ONTAP 使用整套儲存後端、請視您的雲端環境而定、執行下列步驟。

- [在Amazon Web Services中部署Astra Control Center](#)
- [在Google Cloud Platform中部署Astra Control Center](#)
- [在Microsoft Azure中部署Astra Control Center](#)

您可以使用自我管理的Kubernetes叢集、例如OpenShift Container Platform（OCP）、在發佈版本中管理應用程式。只有自我管理的OCP叢集已通過驗證、可用於部署Astra Control Center。

在Amazon Web Services中部署Astra Control Center

您可以在Amazon Web Services（AWS）公有雲上的自我管理Kubernetes叢集上部署Astra Control Center。

AWS所需的功能

在 AWS 中部署 Astra Control Center 之前、您需要下列項目：

- Astra Control Center授權。請參閱 "[Astra Control Center授權要求](#)"。

- ["符合Astra Control Center的要求"](#)。
- NetApp Cloud Central帳戶
- 如果使用OCP、則Red Hat OpenShift Container Platform (OCP) 權限（位於命名空間層級以建立Pod）
- AWS認證資料、存取ID和秘密金鑰、具備可讓您建立儲存區和連接器的權限
- AWS帳戶彈性容器登錄（ECR）存取與登入
- 存取 Astra Control UI 需要 AWS 託管區域和 Amazon Route 53 項目

AWS的作業環境需求

Astra Control Center需要下列AWS作業環境：

- Red Hat OpenShift Container Platform 4.11 至 4.13

確保您選擇裝載Astra Control Center的作業環境符合環境正式文件中所述的基本資源需求。

Astra Control Center 除了需要環境的資源需求外、還需要特定資源。請參閱 ["Astra Control Center營運環境需求"](#)。



AWS 登錄權杖會在 12 小時內過期、之後您必須更新 Docker 映像登錄機密。

AWS部署總覽

以下是安裝Astra Control Center for AWS的程序總覽、Cloud Volumes ONTAP 其中包含以作為儲存後端的功能。

以下將詳細說明每個步驟。

1. [確保您擁有足夠的IAM權限](#)。
2. [在AWS上安裝RedHat OpenShift叢集](#)。
3. [設定AWS](#)。
4. [設定適用於AWS的NetApp BlueXP](#)。
5. [安裝AWS的Astra Control Center](#)。

確保您擁有足夠的IAM權限

確保您擁有足夠的IAM角色和權限、可讓您安裝RedHat OpenShift叢集和NetApp BlueXP（前身為Cloud Manager）Connector。

請參閱 ["初始 AWS 認證資料"](#)。

在AWS上安裝RedHat OpenShift叢集

在AWS上安裝RedHat OpenShift Container Platform叢集。

如需安裝指示、請參閱 ["在OpenShift Container Platform的AWS上安裝叢集"](#)。

設定AWS

接下來、設定 AWS 來建立虛擬網路、設定 EC2 運算執行個體、以及建立 AWS S3 儲存區。如果您無法存取 NetApp Astra Control Center 映像登錄、也需要建立彈性容器登錄（ECR）來主控 Astra Control Center 映像、並將映像推入此登錄。

請遵循AWS文件完成下列步驟。請參閱 ["AWS安裝文件"](#)。

1. 建立AWS虛擬網路。
2. 檢閱EC2運算執行個體。這可以是AWS中的裸機伺服器或VM。
3. 如果執行個體類型尚未符合主節點和工作節點的Astra最低資源需求、請在AWS中變更執行個體類型以符合Astra需求。請參閱 ["Astra Control Center需求"](#)。
4. 建立至少一個AWS S3儲存區來儲存備份。
5. （選用）如果您無法存取 NetApp 映像登錄、請執行下列步驟：
 - a. 建立 AWS 彈性容器登錄（ECR）以裝載所有 Astra Control Center 影像。



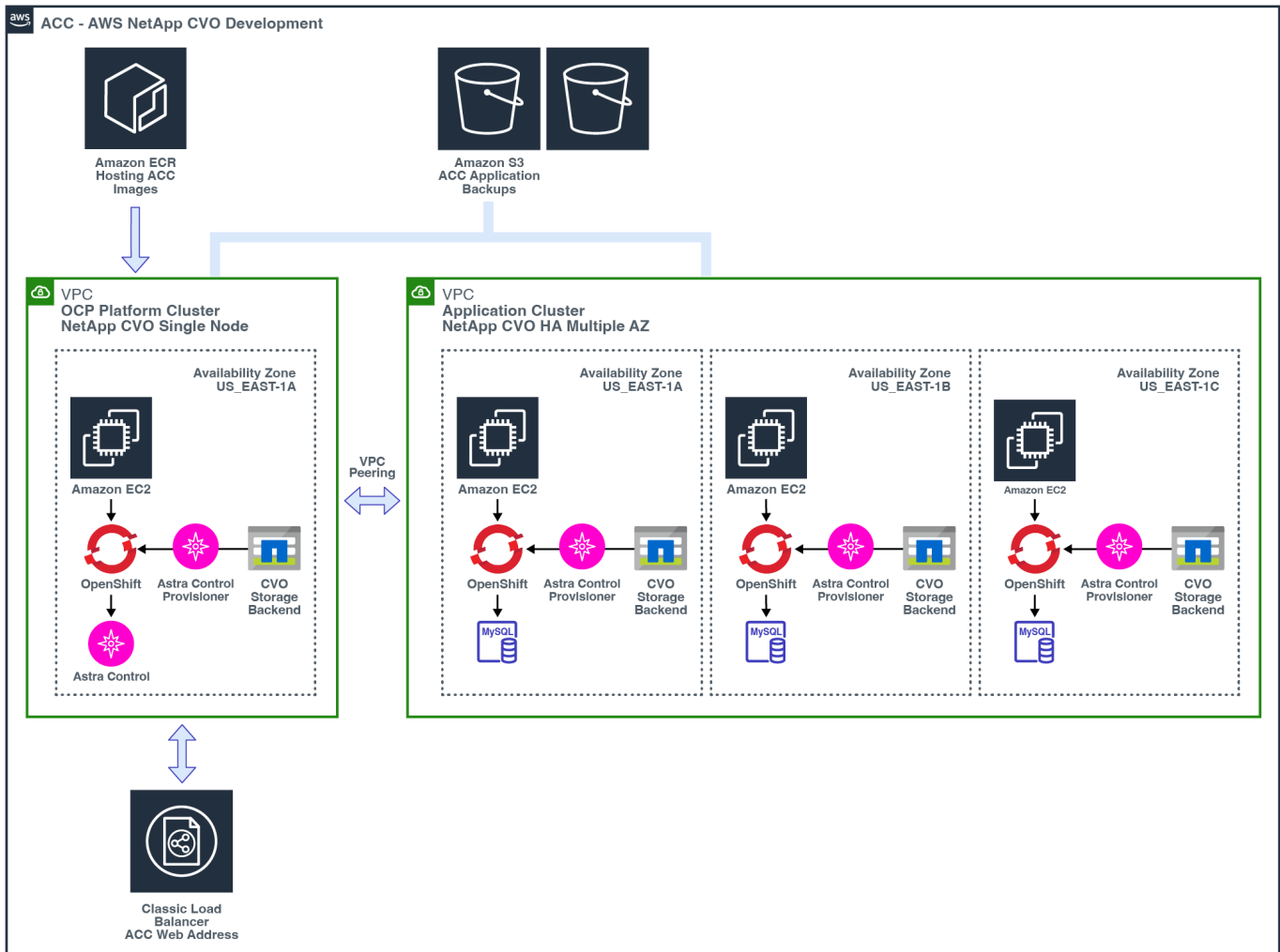
如果您未建立ECR、Astra Control Center將無法從含有Cloud Volumes ONTAP AWS後端的支援的叢集存取監控資料。此問題是因為您嘗試使用Astra Control Center探索及管理的叢集無法存取AWS ECR。

- b. 將 Astra Control Center 影像推送到您定義的登錄。



AWS Elastic Container登錄（ECR）權杖會在12小時後過期、導致跨叢集複製作業失敗。從Cloud Volumes ONTAP 針對AWS設定的功能進行的功能區管理儲存後端時、就會發生此問題。若要修正此問題、請再次向ECR驗證、並產生新的秘密、讓複製作業順利恢復。

以下是AWS部署範例：



設定適用於AWS的NetApp BlueXP

使用NetApp BlueXP（前身為Cloud Manager）建立工作區、新增AWS連接器、建立工作環境、以及匯入叢集。

請遵循BlueXP文件完成下列步驟。請參閱下列內容：

- "開始使用Cloud Volumes ONTAP AWS的功能"。
- "使用BlueXP在AWS中建立連接器"

步驟

1. 將您的認證資料新增至BlueXP。
2. 建立工作區。
3. 新增AWS的連接器。選擇AWS做為供應商。
4. 為您的雲端環境建立工作環境。
 - a. 位置：「Amazon Web Services (AWS)」
 - b. 類型：Cloud Volumes ONTAP「EHA」
5. 匯入OpenShift叢集。叢集將連線至您剛建立的工作環境。
 - a. 選擇「K8s」>「叢集清單」>「叢集詳細資料」，即可檢視NetApp叢集詳細資料。

- b. 在右上角記下 Astra Control Provisioner 版本。
- c. 請注意Cloud Volumes ONTAP、顯示NetApp為資源配置程式的叢集儲存類別。

這會匯入您的Red Hat OpenShift叢集、並將其指派為預設儲存類別。您可以選取儲存類別。
Astra Control Provisioner 會自動安裝為匯入和探索程序的一部分。

- 6. 請注意此Cloud Volumes ONTAP 功能部署中的所有持續磁碟區和磁碟區。



可作為單一節點或高可用度運作。Cloud Volumes ONTAP如果已啟用HA、請記下在AWS中執行的HA狀態和節點部署狀態。

安裝AWS的Astra Control Center

遵循標準 "[Astra Control Center安裝說明](#)"。



AWS使用一般S3儲存區類型。

在Google Cloud Platform中部署Astra Control Center

您可以在Google Cloud Platform（GCP）公有雲上的自我管理Kubernetes叢集上部署Astra Control Center。

GCP的必備功能

在 GCP 中部署 Astra Control Center 之前、您需要下列項目：

- Astra Control Center授權。請參閱 "[Astra Control Center授權要求](#)"。
- "[符合Astra Control Center的要求](#)"。
- NetApp Cloud Central帳戶
- 如果使用的是 OCP、Red Hat OpenShift Container Platform（OCP）4.11 至 4.13
- 如果使用OCP、則Red Hat OpenShift Container Platform（OCP）權限（位於命名空間層級以建立Pod）
- GCP服務帳戶具備權限、可讓您建立貯體和連接器

GCP的作業環境需求

確保您選擇裝載Astra Control Center的作業環境符合環境正式文件中所述的基本資源需求。

Astra Control Center 除了需要環境的資源需求外、還需要特定資源。請參閱 "[Astra Control Center營運環境需求](#)"。

GCP部署總覽

以下是將Astra Control Center安裝在GCP的自我管理OCP叢集上的程序總覽、Cloud Volumes ONTAP 其中包含以作儲存後端的功能。

以下將詳細說明每個步驟。

1. [在GCP上安裝RedHat OpenShift叢集](#)。
2. [建立GCP專案和虛擬私有雲端](#)。

3. 確保您擁有足夠的IAM權限。
4. 設定GCP。
5. 設定適用於GCP的NetApp BlueXP。
6. 安裝Astra Control Center for GCP。

在GCP上安裝RedHat OpenShift叢集

第一步是在GCP上安裝RedHat OpenShift叢集。

如需安裝指示、請參閱下列內容：

- ["在GCP中安裝OpenShift叢集"](#)
- ["建立GCP服務帳戶"](#)

建立GCP專案和虛擬私有雲端

建立至少一個GCP專案和虛擬私有雲端（VPC）。



OpenShift可能會建立自己的資源群組。此外、您也應該定義GCP VPC。請參閱OpenShift文件。

您可能想要建立平台叢集資源群組和目標應用程式OpenShift叢集資源群組。

確保您擁有足夠的IAM權限

確保您擁有足夠的IAM角色和權限、可讓您安裝RedHat OpenShift叢集和NetApp BlueXP（前身為Cloud Manager）Connector。

請參閱 ["初始GCP認證與權限"](#)。

設定GCP

接下來、設定 GCP 以建立 VPC、設定運算執行個體、以及建立 Google Cloud Object Storage。如果您無法存取 NetApp Astra Control Center 映像登錄、您也需要建立 Google Container 登錄來主控 Astra Control Center 映像、並將映像推入此登錄。

請遵循GCP文件完成下列步驟。請參閱在GCP中安裝OpenShift叢集。

1. 在您計畫用於具有CVO後端的OCP叢集的GCP中建立GCP專案和VPC。
2. 檢閱運算執行個體。這可以是GCP中的裸機伺服器或VM。
3. 如果執行個體類型尚未符合主要節點和工作節點的 Astra 最低資源需求、請在 GCP 中變更執行個體類型、以符合 Astra 需求。請參閱 ["Astra Control Center需求"](#)。
4. 建立至少一個GCP雲端儲存庫來儲存備份。
5. 建立儲存貯體存取所需的機密。
6. （選用）如果您無法存取 NetApp 映像登錄、請執行下列步驟：
 - a. 建立 Google Container 登錄以裝載 Astra Control Center 映像。
 - b. 設定所有Astra Control Center映像的Google Container登錄存取權、以供Docker推/拉。

範例：Astra Control Center 影像可透過輸入下列指令碼、推送至此登錄：

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

此指令碼需要Astra Control Center資訊清單檔案和Google Image登錄位置。範例：

```
manifestfile=acc.manifest.bundle.yaml
GCP_CR_REGISTRY=<target GCP image registry>
ASTRA_REGISTRY=<source Astra Control Center image registry>

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 < acc.manifest.bundle.yaml
```

7. 設定DNS區域。

設定適用於**GCP**的**NetApp BlueXP**

使用NetApp BlueXP（前身為Cloud Manager）建立工作區、將連接器新增至GCP、建立工作環境、以及匯入叢集。

請遵循BlueXP文件完成下列步驟。請參閱 ["從GCP開始使用Cloud Volumes ONTAP"](#)。

開始之前

- 以所需的IAM權限和角色存取GCP服務帳戶

步驟

1. 將您的認證資料新增至BlueXP。請參閱 ["新增GCP帳戶"](#)。
2. 新增GCP的連接器。
 - a. 選擇「GCP」作為供應商。
 - b. 輸入GCP認證。請參閱 ["從BlueXP在GCP中建立連接器"](#)。
 - c. 確認連接器正在執行、並切換至該連接器。
3. 為您的雲端環境建立工作環境。
 - a. 地點：「GCP」
 - b. 類型：Cloud Volumes ONTAP「EHA」
4. 匯入OpenShift叢集。叢集將連線至您剛建立的工作環境。

- a. 選擇 `*K8s*>*叢集清單*>*叢集詳細資料*`、即可檢視NetApp叢集詳細資料。
- b. 在右上角記下 Astra Control Provisioner 版本。
- c. 請注意Cloud Volumes ONTAP、顯示「NetApp」為資源配置程式的叢集儲存類別。

這會匯入您的Red Hat OpenShift叢集、並將其指派為預設儲存類別。您可以選取儲存類別。
Astra Control Provisioner 會自動安裝為匯入和探索程序的一部分。

5. 請注意此Cloud Volumes ONTAP 功能部署中的所有持續磁碟區和磁碟區。



可作為單一節點或高可用度（HA）運作。Cloud Volumes ONTAP如果已啟用HA、請記下在GCP中執行的HA狀態和節點部署狀態。

安裝Astra Control Center for GCP

遵循標準 "[Astra Control Center安裝說明](#)"。



GCP使用通用S3儲存區類型。

1. 產生Docker祕密以擷取Astra Control Center安裝的映像：

```
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
```

在Microsoft Azure中部署Astra Control Center

您可以將Astra Control Center部署在Microsoft Azure公有雲上的自我管理Kubernetes叢集上。

Azure的必備功能

在 Azure 中部署 Astra Control Center 之前、您需要下列項目：

- Astra Control Center授權。請參閱 "[Astra Control Center授權要求](#)"。
- "[符合Astra Control Center的要求](#)"。
- NetApp Cloud Central帳戶
- 如果使用的是 OCP、Red Hat OpenShift Container Platform（OCP）4.11 至 4.13
- 如果使用OCP、則Red Hat OpenShift Container Platform（OCP）權限（位於命名空間層級以建立Pod）
- Azure認證、具備可讓您建立儲存區和連接器的權限

Azure的營運環境需求

確保您選擇裝載Astra Control Center的作業環境符合環境正式文件中所述的基本資源需求。

Astra Control Center 除了需要環境的資源需求外、還需要特定資源。請參閱 "[Astra Control Center營運環境需求](#)"。

Azure部署總覽

以下是安裝Astra Control Center for Azure的程序總覽。

以下將詳細說明每個步驟。

1. [在Azure上安裝RedHat OpenShift叢集](#)。
2. [建立Azure資源群組](#)。
3. [確保您擁有足夠的IAM權限](#)。
4. [設定Azure](#)。
5. [設定適用於Azure的NetApp BlueXP（前身為Cloud Manager）](#)。
6. [安裝及設定Azure的Astra Control Center](#)。

在Azure上安裝RedHat OpenShift叢集

第一步是在Azure上安裝RedHat OpenShift叢集。

如需安裝指示、請參閱下列內容：

- ["在Azure上安裝OpenShift叢集"](#)。
- ["安裝Azure帳戶"](#)。

建立Azure資源群組

建立至少一個Azure資源群組。



OpenShift可能會建立自己的資源群組。此外、您也應該定義Azure資源群組。請參閱OpenShift文件。

您可能想要建立平台叢集資源群組和目標應用程式OpenShift叢集資源群組。

確保您擁有足夠的IAM權限

確保您擁有足夠的IAM角色和權限、可讓您安裝RedHat OpenShift叢集和NetApp BlueXP Connector。

請參閱 ["Azure 認證與權限"](#)。

設定Azure

接下來、將 Azure 設定為建立虛擬網路、設定運算執行個體、以及建立 Azure Blob 容器。如果您無法存取 NetApp Astra Control Center 映像登錄、也需要建立 Azure Container 登錄（ACR）來主控 Astra Control Center 映像、並將映像推入此登錄。

請依照Azure文件完成下列步驟。請參閱 ["在Azure上安裝OpenShift叢集"](#)。

1. 建立Azure虛擬網路。
2. 檢閱運算執行個體。這可以是Azure中的裸機伺服器或VM。
3. 如果執行個體類型尚未符合主節點和工作節點的Astra最低資源需求、請變更Azure中的執行個體類型以符

合Astra要求。請參閱 ["Astra Control Center需求"](#)。

4. 建立至少一個Azure Blob容器來儲存備份。
5. 建立儲存帳戶。您將需要一個儲存帳戶、才能在 Astra Control Center 中建立一個容器作為貯體使用。
6. 建立儲存貯體存取所需的機密。
7. (選用) 如果您無法存取 NetApp 映像登錄、請執行下列步驟：
 - a. 建立 Azure Container 登錄 (ACR) 以裝載 Astra Control Center 映像。
 - b. 為所有 Astra Control Center 影像設定 Docker 推 / 拉存取。
 - c. 使用下列指令碼將 Astra Control Center 影像推入此登錄：

```
az acr login -n <AZ ACR URL/Location>
This script requires the Astra Control Center manifest file and your
Azure ACR location.
```

範例：

```
manifestfile=acc.manifest.bundle.yaml
AZ_ACR_REGISTRY=<target Azure ACR image registry>
ASTRA_REGISTRY=<source Astra Control Center image registry>

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 < acc.manifest.bundle.yaml
```

8. 設定DNS區域。

設定適用於**Azure**的**NetApp BlueXP** (前身為**Cloud Manager**)

使用BlueXP (前身為Cloud Manager) 建立工作區、將連接器新增至Azure、建立工作環境、以及匯入叢集。

請遵循BlueXP文件完成下列步驟。請參閱 ["Azure中的BlueXP入門指南"](#)。

開始之前

以所需的IAM權限和角色存取Azure帳戶

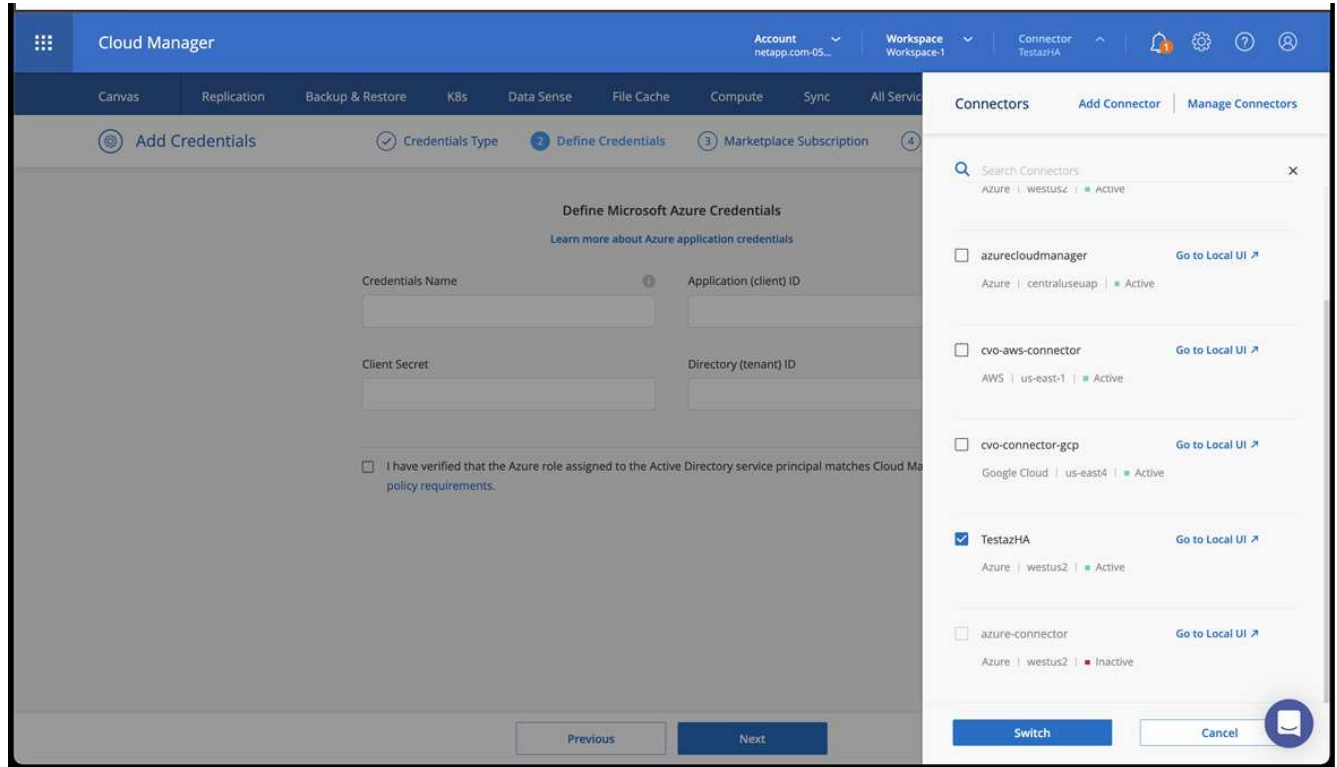
步驟

1. 將您的認證資料新增至BlueXP。
2. 新增Azure連接器。請參閱 ["BlueXP原則"](#)。

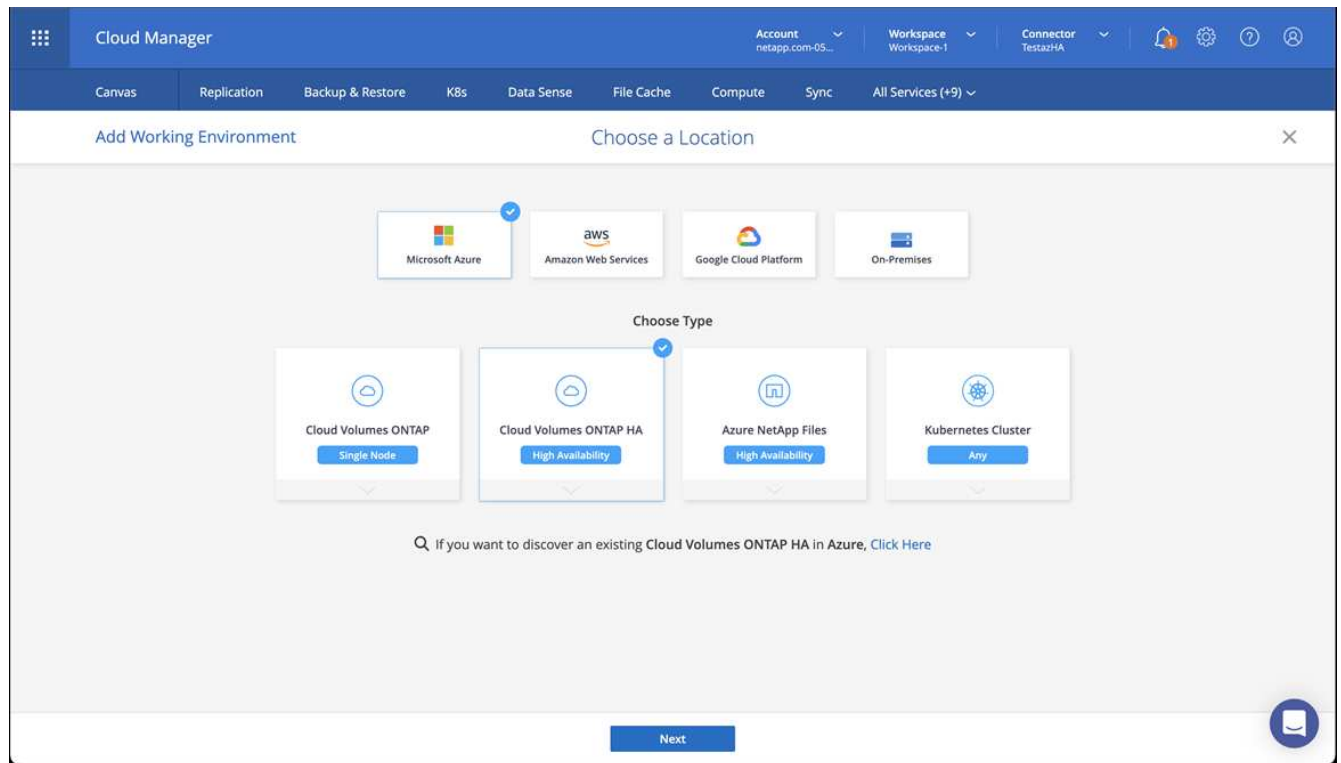
- a. 選擇* Azure *作為供應商。
- b. 輸入Azure認證資料、包括應用程式ID、用戶端機密和目錄（租戶）ID。

請參閱 "從BlueXPr在Azure中建立連接器"。

3. 確認連接器正在執行、並切換至該連接器。

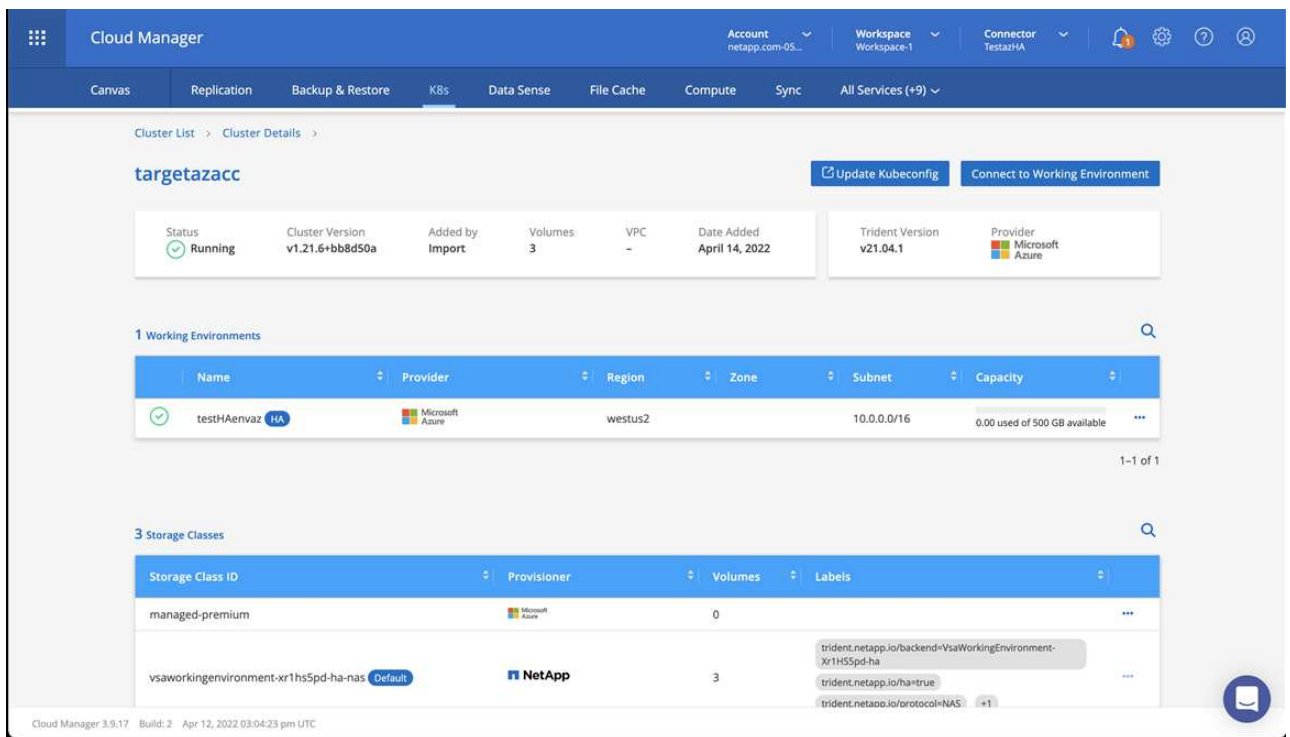


4. 為您的雲端環境建立工作環境。
 - a. 位置：「Microsoft Azure」。
 - b. 輸入：Cloud Volumes ONTAP 「EHA」。



5. 匯入OpenShift叢集。叢集將連線至您剛建立的工作環境。

a. 選擇* K8s*>*叢集清單*>*叢集詳細資料*、即可檢視NetApp叢集詳細資料。



b. 在右上角記下 Astra Control Provisioner 版本。

c. 請注意Cloud Volumes ONTAP、顯示NetApp為資源配置程式的叢集儲存類別。

這會匯入您的Red Hat OpenShift叢集、並指派預設的儲存類別。您可以選取儲存類別。

Astra Control Provisioner 會自動安裝為匯入和探索程序的一部分。

6. 請注意此Cloud Volumes ONTAP 功能部署中的所有持續磁碟區和磁碟區。
7. 可作為單一節點或高可用度運作。Cloud Volumes ONTAP如果已啟用HA、請記下Azure中執行的HA狀態和節點部署狀態。

安裝及設定Azure的Astra Control Center

使用標準安裝Astra Control Center "[安裝說明](#)"。

使用Astra Control Center新增Azure儲存庫。請參閱 "[設定Astra Control Center並新增鏟斗](#)"。

安裝後設定Astra Control Center

視您的環境而定、安裝Astra Control Center之後可能需要額外的組態。

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。如果您的環境是以這種方式設定、則需要從您打算安裝Astra Control Center的命名空間中移除這些資源。

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. 取得中的資源配額 netapp-acc （或自訂命名）命名空間：

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

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 or custom namespace]
```

```
kubectl delete resourcequota pods-low -n [netapp-acc or custom namespace]
```

```
kubectl delete resourcequota pods-medium -n [netapp-acc or custom namespace]
```

3. 取得中的限制範圍 netapp-acc （或自訂命名）命名空間：

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

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 or custom namespace]
```

新增自訂TLS憑證

Astra Control Center預設使用自我簽署的TLS憑證來處理入站控制器流量（僅適用於特定組態）、以及使用網頁瀏覽器進行Web UI驗證。若要正式作業、您應該移除現有的自我簽署 TLS 憑證、並以由憑證授權單位（CA）簽署的 TLS 憑證來取代。



預設的自我簽署憑證可用於兩種類型的連線：

- HTTPS連線至Astra Control Center網路UI
- 入口控制器流量（僅當 ingressType: "AccTraefik" 內容已在中設定 astra_control_center.yaml 安裝Astra Control Center期間的檔案）

取代預設TLS憑證會取代用於驗證這些連線的憑證。

開始之前

- Kubernetes叢集已安裝Astra Control Center
- 管理存取叢集上的命令Shell以執行「kubectl」命令
- 來自CA的私密金鑰和憑證檔案

移除自我簽署的憑證

移除現有的自我簽署TLS憑證。

1. 使用SSH、以管理使用者身分登入裝載Astra Control Center的Kubernetes叢集。
2. 使用下列命令尋找與目前憑證相關的TLS密碼、並以Astra Control Center部署命名空間取代「<ACC-deployment-namespace>」：

```
kubectl get certificate -n <ACC-deployment-namespace>
```

3. 使用下列命令刪除目前安裝的機密與憑證：

```
kubectl delete cert cert-manager-certificates -n <ACC-deployment-namespace>
```

```
kubectl delete secret secure-testing-cert -n <ACC-deployment-namespace>
```

使用命令列新增憑證

新增由CA簽署的TLS憑證。

1. 使用下列命令以CA的私密金鑰和憑證檔案建立新的TLS秘密，並以適當的資訊取代括弧<>中的引數：

```
kubectl create secret tls <secret-name> --key <private-key-filename>  
--cert <certificate-filename> -n <ACC-deployment-namespace>
```

2. 使用以下命令和範例編輯叢集自訂資源定義（CRD）檔案、並將「pec.selfSigned」值變更為「spec.ca.secretName」、以參照您先前建立的TLS密碼：

```
kubectl edit clusterissuers.cert-manager.io/cert-manager-certificates -n  
<ACC-deployment-namespace>
```

客戶需求日：

```
#spec:  
#  selfSigned: {}  
  
spec:  
  ca:  
    secretName: <secret-name>
```

3. 使用下列命令和範例輸出來驗證變更是否正確、而且叢集已準備好驗證憑證、並以Astra Control Center部署命名空間取代「<ACC-deployment-namespaces>」：

```
kubectl describe clusterissuers.cert-manager.io/cert-manager-  
certificates -n <ACC-deployment-namespaces>
```

回應：

```
Status:  
  Conditions:  
    Last Transition Time:  2021-07-01T23:50:27Z  
    Message:               Signing CA verified  
    Reason:                KeyPairVerified  
    Status:                True  
    Type:                  Ready  
  Events:                  <none>
```

4. 使用下列範例建立「create.yaml」檔案、並以適當的資訊取代括弧<>中的預留位置值：



此範例使用 `dnsNames` 用於指定 Astra Control Center DNS 位址的內容。Astra Control Center 不支援使用 Common Name (CN) 屬性來指定 DNS 位址。

```
apiVersion: cert-manager.io/v1  
kind: Certificate  
metadata:  
  <strong>name: <certificate-name></strong>  
  namespace: <ACC-deployment-namespaces>  
spec:  
  <strong>secretName: <certificate-secret-name></strong>  
  duration: 2160h # 90d  
  renewBefore: 360h # 15d  
  dnsNames:  
    <strong>- <astra.dnsname.example.com></strong> #Replace with the  
    correct Astra Control Center DNS address  
  issuerRef:  
    kind: ClusterIssuer  
    name: cert-manager-certificates
```

5. 使用下列命令建立憑證：

```
kubectl apply -f certificate.yaml
```

6. 使用下列命令和範例輸出來驗證憑證是否已正確建立、以及是否已使用您在建立期間所指定的引數（例如名稱、持續時間、續約期限及DNS名稱）。

```
kubectl describe certificate -n <ACC-deployment-namespace>
```

回應：

```
Spec:
  Dns Names:
    astra.example.com
  Duration: 125h0m0s
  Issuer Ref:
    Kind: ClusterIssuer
    Name: cert-manager-certificates
  Renew Before: 61h0m0s
  Secret Name: <certificate-secret-name>
Status:
  Conditions:
    Last Transition Time: 2021-07-02T00:45:41Z
    Message: Certificate is up to date and has not expired
    Reason: Ready
    Status: True
    Type: Ready
  Not After: 2021-07-07T05:45:41Z
  Not Before: 2021-07-02T00:45:41Z
  Renewal Time: 2021-07-04T16:45:41Z
  Revision: 1
  Events: <none>
```

7. 編輯 TLS 儲存 CRD 以使用下列命令和範例指向您的新憑證密碼名稱、以適當的資訊取代括弧 <> 中的預留位置值

```
kubectl edit tlsstores.traefik.io -n <ACC-deployment-namespace>
```

客戶需求日：

```
...
spec:
  defaultCertificate:
    secretName: <certificate-secret-name>
```

8. 編輯「入口CRD TLS」選項、使用下列命令和範例指向新的憑證密碼、並以適當的資訊取代方括弧<>中的預留位置值：

```
kubectl edit ingressroutes.traefik.io -n <ACC-deployment-namespace>
```

客戶需求日：

```
...
tls:
  secretName: <certificate-secret-name>
```

9. 使用網頁瀏覽器瀏覽至Astra Control Center的部署IP位址。
10. 確認憑證詳細資料與您安裝的憑證詳細資料相符。
11. 匯出憑證並將結果匯入網頁瀏覽器中的憑證管理程式。

設定Astra控制中心

新增Astra Control Center授權

安裝 Astra Control Center 時、已安裝內嵌評估授權。如果您正在評估 Astra Control Center 、可以跳過此步驟。

您可以使用Astra Control UI或新增授權 ["Astra Control API"](#) 。

Astra Control Center授權會使用Kubernetes CPU單元來測量CPU資源、並計算指派給所有受管理Kubernetes叢集之工作節點的CPU資源。授權是根據vCPU使用率而定。如需如何計算授權的詳細資訊、請參閱 ["授權"](#) 。



如果您的安裝量成長到超過授權的CPU單元數量、Astra Control Center會防止您管理新的應用程式。超過容量時會顯示警示。



若要更新現有的評估或完整授權、請參閱 ["更新現有授權"](#) 。

開始之前

- 存取新安裝的Astra Control Center執行個體。
- 系統管理員角色權限。
- 答 ["NetApp授權檔案"](#) (If) 。

步驟

1. 登入Astra Control Center UI 。
2. 選擇*帳戶*>*授權* 。
3. 選擇*新增授權* 。
4. 瀏覽至您下載的授權檔案 (NLF) 。
5. 選擇*新增授權* 。

「帳戶>*授權*」頁面會顯示授權資訊、到期日、授權序號、帳戶ID及使用的CPU單位。



如果您擁有評估授權、但並未將資料傳送AutoSupport至效益分析系統、請務必儲存您的帳戶ID、以免發生Astra Control Center故障時發生資料遺失。

啟用 **Astra Control Provisioner**

Astra Trident 版本 23.10 及更新版本包含使用 Astra Control Provisioner 的選項、可讓獲授權的 Astra Control 使用者存取進階儲存資源配置功能。Astra Control Provisioner 除了提供標準 Astra Trident CSI 型功能之外、還提供這項延伸功能。

Astra Control 的更新將取代 Astra Trident 做為儲存資源配置程式和 Orchestrator、並成為 Astra Control 使用的必要項目。因此、強烈建議 Astra Control 使用者啟用 Astra Control Provisioner。Astra Trident 將繼續保持開放原始碼、並以 NetApp 的新 CSI 和其他功能來發行、維護、支援及更新。

關於這項工作

如果您是獲得授權的 Astra Control Center 使用者、且您想要使用 Astra Control Provisioner 功能、則應遵循此程序。如果您是 Astra Trident 使用者、而且想要使用 Astra Control Provisioner 所提供的額外功能、而不同時使用 Astra Control、則也應遵循此程序。

在每種情況下、Astra Trident 24.02 預設不會啟用資源配置程式功能、而且必須啟用。

開始之前

如果您要啟用 Astra Control Provisioner、請先執行下列步驟：

Astra Control Provisioners 使用 Astra Control Center 的使用者

- * 取得 Astra Control Center 授權 * : 您需要 "[Astra Control Center 授權](#)" 啟用 Astra Control Provisioner 並存取其所提供的功能。
- * 安裝或升級至 Astra Control Center 23.10 或更新版本 * : 如果您打算使用最新的 Astra Control Provisioner 功能 (24.02) 搭配 Astra Control 、您將需要最新的 Astra Control Center 版本 (24.02) 。
- * 確認您的叢集具有 AMD64 系統架構 * : Astra Control Provisioner 影像同時在 AMD64 和 ARM64 CPU 架構中提供、但 Astra Control Center 僅支援 AMD64 。
- * 取得登錄存取的 Astra 控制服務帳戶 * : 如果您打算使用 Astra 控制登錄而非 NetApp 支援網站 來下載 Astra 控制項資源配置程式映像、請完成的登錄 "[Astra Control Service 帳戶](#)"。完成並提交表單並建立 BlueXP 帳戶後、您會收到 Astra Control Service 歡迎電子郵件。
- * 如果您已安裝 Astra Trident 、請確認其版本位於四個版本的視窗 * : 如果 Astra Trident 位於 24.02 版的四個版本視窗內、您可以使用 Astra Control Provisioner 直接升級至 Astra Trident 24.02 。例如、您可以直接從 Astra Trident 23.04 升級至 24.02 。

Astra Control Provisioner 僅限使用者

- * 取得 Astra Control Center 授權 * : 您需要 "[Astra Control Center 授權](#)" 啟用 Astra Control Provisioner 並存取其所提供的功能。
- * 如果您已安裝 Astra Trident 、請確認其版本位於四個版本的視窗 * : 如果 Astra Trident 位於 24.02 版的四個版本視窗內、您可以使用 Astra Control Provisioner 直接升級至 Astra Trident 24.02 。例如、您可以直接從 Astra Trident 23.04 升級至 24.02 。
- * 取得登錄存取的 Astra Control Service 帳戶 * : 您需要存取登錄才能下載 Astra Control Provisioner 映像。若要開始使用、請完成的註冊 "[Astra Control Service 帳戶](#)"。完成並提交表單並建立 BlueXP 帳戶後、您會收到 Astra Control Service 歡迎電子郵件。

(步驟 1) 取得 Astra Control Provisioner 映像

Astra Control Center 使用者可以使用 Astra Control 登錄或 NetApp 支援網站 方法取得 Astra Control Provisioner 映像。想要在不使用 Astra Control 的情況下使用 Astra Control Provisioner 的 Astra Trident 使用者、應使用登錄方法。

Astra Control 影像登錄



您可以在本程序中使用 Podman 而非 Docker 來執行命令。如果您使用的是 Windows 環境、建議您使用 PowerShell。

1. 存取 NetApp Astra Control 影像登錄：

- 登入 Astra Control Service Web UI、然後選取頁面右上角的圖示。
- 選擇* API存取*。
- 記下您的帳戶 ID。
- 從同一頁面選取 * 產生 API 權杖 *、然後將 API 權杖字串複製到剪貼簿、並將其儲存在編輯器中。
- 使用您偏好的方法登入 Astra Control 登錄：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

```
crane auth login cr.astra.netapp.io -u <account-id> -p <api-token>
```

2. (僅限自訂登錄) 請依照下列步驟將映像移至自訂登錄。如果您未使用登錄，請遵循中的 Trident 運算子步驟 ["下一節"](#)。

- 從登錄中拉出 Astra Control Provisioner 映像：



所擷取的映像不支援多個平台、而且僅支援與擷取映像的主機相同的平台、例如 Linux AMD64。

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0  
--platform <cluster platform>
```

範例：

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0 --platform  
linux/amd64
```

- 標記影像：

```
docker tag cr.astra.netapp.io/astra/trident-acp:24.02.0  
<my_custom_registry>/trident-acp:24.02.0
```

- 將映像推送至自訂登錄：

```
docker push <my_custom_registry>/trident-acp:24.02.0
```



您可以使用 Crane 複本來取代執行下列 Docker 命令：

```
crane copy cr.astra.netapp.io/astra/trident-acp:24.02.0  
<my_custom_registry>/trident-acp:24.02.0
```

NetApp 支援網站

1. 下載 Astra Control Provisioner 套件 (trident-acp-[version].tar) 從 "[Astra Control Center 下載頁面](#)"。
2. (建議但為選用) 下載 Astra Control Center (Astra 控制中心) 的憑證與簽名套件 (Astra 控制中心 - certs-[version].tar.gz) 、以驗證 Trident -ACP-[version] tar 套件的簽名。

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

```
openssl dgst -sha256 -verify certs/AstraControlCenterDockerImages-  
public.pub -signature certs/trident-acp-[version].tar.sig trident-  
acp-[version].tar
```

3. 載入 Astra Control Provisioner 映像：

```
docker load < trident-acp-24.02.0.tar
```

回應：

```
Loaded image: trident-acp:24.02.0-linux-amd64
```

4. 標記影像：

```
docker tag trident-acp:24.02.0-linux-amd64  
<my_custom_registry>/trident-acp:24.02.0
```

5. 將映像推送至自訂登錄：

```
docker push <my_custom_registry>/trident-acp:24.02.0
```


(步驟 2) 在 **Astra Trident** 中啟用 **Astra Control Provisioner**

判斷原始安裝方法是否使用 "運算子 (手動或使用 Helm) 或 `tridentctl`" 並根據您的原始方法完成適當的步驟。

Astra Trident 運算子

1. "下載 [Astra Trident 安裝程式並將其解壓縮](#)"。
2. 如果您尚未安裝 Astra Trident 、或是從原始 Astra Trident 部署中移除運算子、請完成下列步驟：
 - a. 建立客戶需求日：

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.y
aml
```

- b. 建立 Trident 命名空間 (kubectl create namespace trident) 或確認 Trident 命名空間仍然存在 (kubectl get all -n trident) 。如果已移除命名空間、請重新建立。
3. 將 Astra Trident 更新為 24.02.0 ：



對於執行 Kubernetes 1.24 或更早版本的叢集、請使用 bundle_pre_1_25.yaml 。對於執行 Kubernetes 1.25 或更新版本的叢集、請使用 bundle_post_1_25.yaml 。

```
kubectl -n trident apply -f trident-installer/deploy/<bundle-
name.yaml>
```

4. 確認 Astra Trident 正在執行：

```
kubectl get torc -n trident
```

回應：

NAME	AGE
trident	21m

5. [\[\[Pull 機密\]\]](#) 如果您有使用機密的登錄、請建立秘密來拉出 Astra Control Provisioner 映像：

```
kubectl create secret docker-registry <secret_name> -n trident
--docker-server=<my_custom_registry> --docker-username=<username>
--docker-password=<token>
```

6. 編輯 TridentOrchestrator CR 並進行下列編輯：

```
kubectl edit torc trident -n trident
```

- a. 為 Astra Trident 映像設定自訂登錄位置、或從 Astra Control 登錄中拉出 (tridentImage: <my_custom_registry>/trident:24.02.0 或 tridentImage: netapp/trident:24.02.0) 。
- b. 啟用 Astra Control Provisioner (enableACP: true) 。
- c. 設定 Astra Control Provisioner 映像的自訂登錄位置、或從 Astra Control 登錄中拉出映像 (acpImage: <my_custom_registry>/trident-acp:24.02.0 或 acpImage: cr.astra.netapp.io/astra/trident-acp:24.02.0) 。
- d. 如果您已建立 [影像拉出秘密](#) 在本程序稍早的部分、您可以在這裡設定 (imagePullSecrets: - <secret_name>) 。

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  tridentImage: <registry>/trident:24.02.0
  enableACP: true
  acpImage: <registry>/trident-acp:24.02.0
  imagePullSecrets:
    - <secret_name>
```

7. 儲存並結束檔案。部署程序將會自動開始。
8. 確認已建立運算子、部署和複本集。

```
kubectl get all -n trident
```



Kubernetes 叢集中只應有*一個運算子執行個體*。請勿建立 Astra Trident 運算子的多個部署。

9. 驗證 trident-acp 容器正在執行 acpVersion 是 24.02.0 狀態為 Installed:

```
kubectl get torc -o yaml
```

回應：

```
status:
  acpVersion: 24.02.0
  currentInstallationParams:
    ...
    acpImage: <registry>/trident-acp:24.02.0
    enableACP: "true"
    ...
  ...
status: Installed
```

試用

1. "下載 Astra Trident 安裝程式並將其解壓縮"。
2. "如果您有現有的 Astra Trident 、請將其從裝載它的叢集上解除安裝"。
3. 在啟用 Astra Control Provisioner 的情況下安裝 Astra Trident (`--enable-acp=true`) ：

```
./tridentctl -n trident install --enable-acp=true --acp
-image=mycustomregistry/trident-acp:24.02
```

4. 確認 Astra Control Provisioner 已啟用：

```
./tridentctl -n trident version
```

回應：

```
+-----+-----+-----+ | SERVER VERSION |
CLIENT VERSION | ACP VERSION | +-----+-----+
+-----+ | 24.02.0 | 24.02.0 | 24.02.0. | +-----+
+-----+-----+-----+
```

掌舵

1. 如果您已安裝 Astra Trident 23.07.1 或更早版本、"解除安裝" 操作員和其他元件。
2. 如果 Kubernetes 叢集執行 1.24 或更早版本、請刪除 PSP ：

```
kubectl delete psp tridentoperatorpod
```

3. 新增Astra Trident Helm儲存庫：

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

4. 更新 Helm 圖表：

```
helm repo update netapp-trident
```

回應：

```
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "netapp-trident" chart
repository
Update Complete. ☐Happy Helming!☐
```

5. 列出影像：

```
./tridentctl images -n trident
```

回應：

```
| v1.28.0           | netapp/trident:24.02.0|
|                   | docker.io/netapp/trident-autosupport:24.02|
|                   | registry.k8s.io/sig-storage/csi-
provisioner:v4.0.0|
|                   | registry.k8s.io/sig-storage/csi-
attacher:v4.5.0|
|                   | registry.k8s.io/sig-storage/csi-
resizer:v1.9.3|
|                   | registry.k8s.io/sig-storage/csi-
snapshotter:v6.3.3|
|                   | registry.k8s.io/sig-storage/csi-node-driver-
registrar:v2.10.0 |
|                   | netapp/trident-operator:24.02.0 (optional)
```

6. 確保 Trident 操作員 24.02.0 可用：

```
helm search repo netapp-trident/trident-operator --versions
```

回應：

NAME	CHART VERSION	APP VERSION	
DESCRIPTION			
netapp-trident/trident-operator	100.2402.0	24.02.0	A

7. 使用 `helm install` 然後執行下列其中一個選項、其中包括這些設定：

- 部署位置的名稱
- Astra Trident 版本
- Astra Control Provisioner 映像的名稱
- 啟用資源配置程式的旗標
- (選用) 本機登錄路徑。如果您使用的是本機登錄 "Trident 影像" 可以位於一個登錄或不同的登錄中、但所有的SCSI映像都必須位於同一個登錄中。
- Trident 命名空間

選項

- 沒有登錄的映像

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=cr.astra.netapp.io/astra/trident-acp:24.02.0
--set enableACP=true --set operatorImage=netapp/trident-
operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

- 一個或多個登錄中的影像

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=<your-registry>:<acp image> --set
enableACP=true --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=netapp/trident-operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

您可以使用 `helm list` 若要檢閱安裝詳細資料、例如名稱、命名空間、圖表、狀態、應用程式版本、和修訂編號。

如果您在使用 Helm 部署 Trident 時遇到任何問題、請執行此命令以完全解除安裝 Astra Trident：

```
./tridentctl uninstall -n trident
```

- 請勿 * "完全移除 Astra Trident 客戶需求日" 在嘗試再次啟用 Astra Control Provisioner 之前、請先將其作為

解除安裝的一部分。

結果

Astra Control Provisioner 功能已啟用、您可以使用任何適用於所執行版本的功能。

(僅適用於 Astra Control Center 使用者) 安裝 Astra Control Provisioner 之後、在 Astra Control Center UI 中裝載置備程式的叢集將會顯示 ACP version 而非 Trident version 欄位和目前安裝的版本號碼。

CLUSTER STATUS

Available

Version v1.24.9+rke2r2	Managed 2024/03/15 17:32 UTC	Kube-system namespace UID [Progress Bar]	ACP Version [Progress Bar]
Private route identifier [Progress Bar]	Cloud instance private	Default bucket astra-bucket1 (inherited)	

Overview

Namespaces

Storage

Activity

以取得更多資訊

- ["Astra Trident 升級文件"](#)

使用Astra Control為環境做好叢集管理準備

在新增叢集之前、您應確保符合下列先決條件。您也應該執行資格檢查、以確保叢集已準備好新增至 Astra Control Center、並視需要建立 kubeconfig 叢集角色。

Astra Control 可讓您根據您的環境和偏好、新增由自訂資源（CR）或 Kubeconfig 管理的叢集。

開始之前

- * 符合環境先決條件 *：您的環境符合要求 ["營運環境需求"](#) 適用於 Astra Control Center。
- * 設定工作節點 *：確保您 ["設定工作節點"](#) 在具有適當儲存驅動程式的叢集中、Pod 可以與後端儲存設備互動。
- `[[enable - psa]]`*Enable PSA restrictions *：如果叢集已啟用 Pod 安全許可強制（Kubernetes 1.25 及更新叢集的標準）、則您必須啟用 PSA 對這些命名空間的限制：
 - netapp-acc-operator 命名空間：

```
kubectl label --overwrite ns netapp-acc-operator pod-security.kubernetes.io/enforce=privileged
```

- netapp monitoring 命名空間：

```
kubectl label --overwrite ns netapp-monitoring pod-  
security.kubernetes.io/enforce=privileged
```

- 《支援》認證：您需要使用支援版的支援版支援系統上設定的支援認證和超級使用者與使用者ID、才能使用Astra Control Center來備份及還原應用程式ONTAP ONTAP ONTAP。

在flexf2命令列中執行下列命令ONTAP：

```
export-policy rule modify -vserver <storage virtual machine name>  
-policyname <policy name> -ruleindex 1 -superuser sys  
export-policy rule modify -vserver <storage virtual machine name>  
-policyname <policy name> -ruleindex 1 -anon 65534
```

- * KUBUBeconfig 託管叢集需求 *：這些需求是專為由 KUBeconfig 管理的應用程式叢集所設計。
 - * 讓 kubeconfig 可存取 *：您可以存取 ["預設叢集 kubeconfig"](#) 那 ["您已在安裝期間進行設定"](#)。
 - * 憑證授權單位考量 *：如果您要使用參考私有憑證授權單位（CA）的 kubeconfig 檔案來新增叢集、請在中新增下列一行 cluster kubeconfig 檔案的一節。這可讓 Astra Control 新增叢集：
- ```
insecure-skip-tls-verify: true
```
- \* 僅限Rancher \*：在Rancher環境中管理應用程式叢集時、請在Rancher提供的Kubeconfig檔案中修改應用程式叢集的預設內容、以使用控制面內容而非Rancher API伺服器內容。如此可減少Rancher API伺服器的負載、並改善效能。
  - \* Astra Control Provisioner 需求 \*：您應該擁有正確設定的 Astra Control 資源配置程式、包括其 Astra Trident 元件、來管理叢集。
    - \* 檢閱 Astra Trident 環境需求 \*：在安裝或升級 Astra Control Provisioner 之前、請檢閱 ["支援的前端、後端及主機組態"](#)。
    - \* 啟用 Astra Control Provisioner 功能 \*：強烈建議您安裝 Astra Trident 23.10 或更新版本並啟用 ["Astra Control Provisioner 進階儲存功能"](#)。在後續版本中、如果 Astra Control 資源配置程式未啟用、Astra Control 將不支援 Astra Trident。
    - \* 設定儲存後端 \*：至少必須有一個儲存後端 ["在 Astra Trident 中設定"](#) 在叢集上。
    - \* 設定儲存類別 \*：至少必須有一個儲存類別 ["在 Astra Trident 中設定"](#) 在叢集上。如果已設定預設儲存類別、請確定它是具有預設註釋的 \* 僅 \* 儲存類別。
    - \* 設定 Volume Snapshot 控制器並安裝 Volume Snapshot 類別 \*：["安裝 Volume Snapshot 控制器"](#) 以便在Astra Control中建立快照。["建立"](#) 至少一個 VolumeSnapshotClass 使用 Astra Trident。

## 執行資格檢查

執行下列資格檢查、確保您的叢集已準備好新增至Astra控制中心。

### 步驟

1. 判斷您正在執行的 Astra Trident 版本：



```
kubectl get tridentversion -n trident
```

如果 Astra Trident 存在、您會看到類似下列的輸出：

| NAME    | VERSION |
|---------|---------|
| trident | 24.02.0 |

如果 Astra Trident 不存在、您會看到類似下列的輸出：

```
error: the server doesn't have a resource type "tridentversions"
```

## 2. 執行下列其中一項：

- 如果您執行的是 Astra Trident 23.01 或更早版本、請使用這些項目 ["說明"](#) 在升級至 Astra Control Provisioner 之前、先升級至較新版本的 Astra Trident。您可以 ["執行直接升級"](#) 如果您的 Astra Trident 位於 24.02 版的四個版本視窗內、則為 Astra Control Provisioner 24.02。例如、您可以直接從 Astra Trident 23.04 升級至 Astra Control Provisioner 24.02。
- 如果您執行的是 Astra Trident 23.10 或更新版本、請確認 Astra Control Provisioner 已完成 ["已啟用"](#)
  - Astra Control Provisioner 無法搭配 23.10 之前的 Astra Control Center 版本使用。["升級 Astra Control Provisioner"](#) 因此它與您要升級以存取最新功能的 Astra Control Center 版本相同。

## 3. 確保所有 Pod（包括 trident-acp）正在執行：

```
kubectl get pods -n trident
```

## 4. 判斷儲存類別是否使用支援的 Astra Trident 驅動程式。置備程式名稱應為 `csi.trident.netapp.io`。請參閱下列範例：

```
kubectl get sc
```

回應範例：

| NAME                    | PROVISIONER                 | RECLAIMPOLICY |
|-------------------------|-----------------------------|---------------|
| ontap-gold (default)    | csi.trident.netapp.io       | Delete        |
| VolumeBindingMode: true | ALLOWVOLUMEEXPANSION: 5d23h | Immediate     |

## 建立叢集角色庫比諾圖

對於使用 kubeconfig 管理的叢集、您可以選擇性地為 Astra Control Center 建立有限權限或擴充權限管理員角色。這不是 Astra Control Center 設定的必要程序、因為您已將 Kubeconfig 設定為的一部分 ["安裝程序"](#)。

如果下列任一情況適用於您的環境、本程序可協助您建立個別的 Kubeconfig：

- 您想要限制其管理叢集的 Astra Control 權限
- 您使用多個內容範圍、無法使用安裝期間設定的預設 Astra Control Kubeconfig、或是具有單一內容的受限角色、都無法在您的環境中運作

開始之前

在完成程序步驟之前、請確定您要管理的叢集具備下列項目：

- 已安裝KECV1.23或更新版本
- 利用Astra Control Center來存取您要新增及管理的叢集



在此程序中、您不需要透過KECBECVL存取執行Astra Control Center的叢集。

- 使用叢集管理權限來管理作用中內容的叢集的作用中KECBEConfig

步驟

1. 建立服務帳戶：

a. 建立名為「astracontrol-service-account.yaml」的服務帳戶檔案。

```
astracontrol-service-account.yaml
```

```
apiVersion: v1
kind: ServiceAccount
metadata:
 name: astracontrol-service-account
 namespace: default
```

b. 套用服務帳戶：

```
kubectl apply -f astracontrol-service-account.yaml
```

2. 為要由 Astra Control 管理的叢集建立具有足夠權限的下列叢集角色之一：

## 有限的叢集角色

此角色包含 Astra Control 管理叢集所需的最低權限：

- a. 建立 ClusterRole 例如、astra-admin-account.yaml。

```
astra-admin-account.yaml
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
 name: astra-admin-account
rules:

Get, List, Create, and Update all resources
Necessary to backup and restore all resources in an app
- apiGroups:
 - '*'
 resources:
 - '*'
 verbs:
 - get
 - list
 - create
 - patch

Delete Resources
Necessary for in-place restore and AppMirror failover
- apiGroups:
 - ""
 - apps
 - autoscaling
 - batch
 - crd.projectcalico.org
 - extensions
 - networking.k8s.io
 - policy
 - rbac.authorization.k8s.io
 - snapshot.storage.k8s.io
 - trident.netapp.io
 resources:
 - configmaps
 - cronjobs
 - daemonsets
 - deployments
```

```

- horizontalpodautoscalers
- ingresses
- jobs
- namespaces
- networkpolicies
- persistentvolumeclaims
- poddisruptionbudgets
- pods
- podtemplates
- replicaset
- replicationcontrollers
- replicationcontrollers/scale
- rolebindings
- roles
- secrets
- serviceaccounts
- services
- statefulsets
- tridentmirrorrelationships
- tridentnapshotinfos
- volumesnapshots
- volumesnapshotcontents
verbs:
- delete

Watch resources
Necessary to monitor progress
- apiGroups:
 - ""
 resources:
 - pods
 - replicationcontrollers
 - replicationcontrollers/scale
 verbs:
 - watch

Update resources
- apiGroups:
 - ""
 - build.openshift.io
 - image.openshift.io
 resources:
 - builds/details
 - replicationcontrollers
 - replicationcontrollers/scale
 - imagestreams/layers

```

```
- imagestreamtags
- imagetags
verbs:
- update
```

- b. (僅限 OpenShift 叢集) 在的結尾處附加下列項目 `astra-admin-account.yaml` 檔案：

```
OpenShift security
- apiGroups:
 - security.openshift.io
 resources:
 - securitycontextconstraints
 verbs:
 - use
 - update
```

- c. 套用叢集角色：

```
kubectl apply -f astra-admin-account.yaml
```

#### 擴充叢集角色

此角色包含將由 Astra Control 管理之叢集的擴充權限。如果您使用多個內容範圍、且無法使用安裝期間設定的預設 Astra Control Kbeconfig、或是具有單一內容的有限角色無法在您的環境中運作、則可以使用此角色：



以下內容 ClusterRole 步驟是 Kubernetes 的一般範例。請參閱 Kubernetes 散佈文件、以取得特定於您環境的指示。

- a. 建立 ClusterRole 例如、`astra-admin-account.yaml`。

```
astra-admin-account.yaml
```

```

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
 name: astra-admin-account
rules:
- apiGroups:
 - '*'
 resources:
 - '*'
 verbs:
 - '*'
- nonResourceURLs:
 - '*'
 verbs:
 - '*'

```

b. 套用叢集角色：

```
kubectl apply -f astra-admin-account.yaml
```

3. 建立叢集角色與服務帳戶的叢集角色繫結：

a. 建立名為「astracontrol-clusterrolebinding.yaml」的「ClusterRoleBinding」檔案。

```
astracontrol-clusterrolebinding.yaml
```

```

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
 name: astracontrol-admin
roleRef:
 apiGroup: rbac.authorization.k8s.io
 kind: ClusterRole
 name: astra-admin-account
subjects:
- kind: ServiceAccount
 name: astracontrol-service-account
 namespace: default

```

b. 套用叢集角色繫結：

```
kubectl apply -f astracontrol-clusterrolebinding.yaml
```

4. 建立並套用權杖密碼：

- a. 建立一個稱為的權杖秘密檔案 `secret-astracontrol-service-account.yaml`。

```
secret-astracontrol-service-account.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
 name: secret-astracontrol-service-account
 namespace: default
 annotations:
 kubernetes.io/service-account.name: "astracontrol-service-
account"
type: kubernetes.io/service-account-token
```

- b. 套用權杖密碼：

```
kubectl apply -f secret-astracontrol-service-account.yaml
```

5. 將權杖密碼新增至服務帳戶、將其名稱新增至 `secrets Array`（以下範例中的最後一行）：

```
kubectl edit sa astracontrol-service-account
```

```

apiVersion: v1
imagePullSecrets:
- name: astracontrol-service-account-dockercfg-48xhx
kind: ServiceAccount
metadata:
 annotations:
 kubectl.kubernetes.io/last-applied-configuration: |

{"apiVersion":"v1","kind":"ServiceAccount","metadata":{"annotations":{},"name":"astracontrol-service-account","namespace":"default"},"creationTimestamp":"2023-06-14T15:25:45Z","name":"astracontrol-service-account","namespace":"default","resourceVersion":"2767069","uid":"2ce068c4-810e-4a96-ada3-49cbf9ec3f89"}
secrets:
- name: astracontrol-service-account-dockercfg-48xhx
- name: secret-astracontrol-service-account

```

## 6. 列出服務帳戶機密、將「<內容>」取代為正確的安裝內容：

```

kubectl get serviceaccount astracontrol-service-account --context
<context> --namespace default -o json

```

輸出的結尾應類似於下列內容：

```

"secrets": [
{ "name": "astracontrol-service-account-dockercfg-48xhx"},
{ "name": "secret-astracontrol-service-account"}
]

```

中每個元素的索引 secrets 陣列開頭為0。在上述範例中、索引為 astracontrol-service-account-dockercfg-48xhx 將為0、索引則為 secret-astracontrol-service-account 應該是1。在輸出中、記下服務帳戶密碼的索引編號。您在下一個步驟中需要此索引編號。

## 7. 產生以下的Kbeconfig：

- a. 建立 create-kubeconfig.sh 檔案：
- b. 更換 TOKEN\_INDEX 在下列指令碼開頭、使用正確的值。

```

create-kubeconfig.sh

```



```

Update these to match your environment.
Replace TOKEN_INDEX with the correct value
from the output in the previous step. If you
didn't change anything else above, don't change
anything else here.

SERVICE_ACCOUNT_NAME=astraccontrol-service-account
NAMESPACE=default
NEW_CONTEXT=astraccontrol
KUBECONFIG_FILE='kubeconfig-sa'

CONTEXT=$(kubectl config current-context)

SECRET_NAME=$(kubectl get serviceaccount ${SERVICE_ACCOUNT_NAME} \
 --context ${CONTEXT} \
 --namespace ${NAMESPACE} \
 -o jsonpath='{.secrets[TOKEN_INDEX].name}')
TOKEN_DATA=$(kubectl get secret ${SECRET_NAME} \
 --context ${CONTEXT} \
 --namespace ${NAMESPACE} \
 -o jsonpath='{.data.token}')

TOKEN=$(echo ${TOKEN_DATA} | base64 -d)

Create dedicated kubeconfig
Create a full copy
kubectl config view --raw > ${KUBECONFIG_FILE}.full.tmp

Switch working context to correct context
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp config use-context
${CONTEXT}

Minify
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp \
 config view --flatten --minify > ${KUBECONFIG_FILE}.tmp

Rename context
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
 rename-context ${CONTEXT} ${NEW_CONTEXT}

Create token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
 set-credentials ${CONTEXT}-${NAMESPACE}-token-user \
 --token ${TOKEN}

Set context to use token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \

```

```

set-context ${NEW_CONTEXT} --user ${CONTEXT}-${NAMESPACE}-token-
user

Set context to correct namespace
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
 set-context ${NEW_CONTEXT} --namespace ${NAMESPACE}

Flatten/minify kubeconfig
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
 view --flatten --minify > ${KUBECONFIG_FILE}

Remove tmp
rm ${KUBECONFIG_FILE}.full.tmp
rm ${KUBECONFIG_FILE}.tmp

```

c. 請輸入命令以將其套用至Kubernetes叢集。

```
source create-kubeconfig.sh
```

8. (選用) 將Kubeconfig重新命名為有意義的叢集名稱。

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

## (技術預覽) 安裝 **Astra Connector for Managed** 叢集

由 Astra Control Center 管理的叢集使用 Astra Connector 來啟用託管叢集與 Astra Control Center 之間的通訊。您需要在所有想要管理的叢集上安裝 Astra Connector。

### 安裝 **Astra Connector**

您可以使用 Kubernetes 命令和自訂資源（CR）檔案來安裝 Astra Connector。

#### 關於這項工作

- 執行這些步驟時、請在您要使用 Astra Control 管理的叢集上執行這些命令。
- 如果您使用的是堡壘主機、請從堡壘主機的命令列發出這些命令。

#### 開始之前

- 您需要存取想要使用 Astra Control 管理的叢集。
- 您需要 Kubernetes 管理員權限、才能在叢集上安裝 Astra Connector 運算子。



如果叢集已設定 Pod 安全許可強制（Kubernetes 1.25 及更新叢集的預設值）、則您需要針對適當的命名空間啟用 PSA 限制。請參閱 ["使用Astra Control為環境做好叢集管理準備"](#) 以取得相關指示。

## 步驟

1. 在您要使用 Astra Control 管理的叢集上安裝 Astra Connector 運算子。當您執行此命令時、命名空間 `astra-connector-operator` 已建立並套用組態至命名空間：

```
kubectl apply -f https://github.com/NetApp/astra-connector-operator/releases/download/24.02.0-202403151353/astraconnector_operator.yaml
```

2. 確認操作員已安裝就緒：

```
kubectl get all -n astra-connector-operator
```

3. 從 Astra Control 取得 API 權杖。請參閱 "[Astra Automation文件](#)" 以取得相關指示。
4. 使用權杖建立秘密。將 `<API_TOKEN>` 取代為您從 Astra Control 收到的權杖：

```
kubectl create secret generic astra-token \
--from-literal=apiToken=<API_TOKEN> \
-n astra-connector
```

5. 建立 Docker 秘密以用於拉出 Astra Connector 映像。以您環境的資訊取代括弧 `<>` 中的值：



您可以在 Astra 控制網路 UI 中找到 `<ASTRA_CONTROL_ACCOUNT_ID>`。在 Web UI 中、選取頁面右上角的圖示、然後選取 \*API 存取\*。

```
kubectl create secret docker-registry regcred \
--docker-username=<ASTRA_CONTROL_ACCOUNT_ID> \
--docker-password=<API_TOKEN> \
-n astra-connector \
--docker-server=cr.astra.netapp.io
```

6. 建立 Astra Connector CR 檔案並命名 `astra-connector-cr.yaml`。更新括弧 `<>` 中的值以符合 Astra Control 環境和叢集組態：
  - `<ASTRA_CONTROL_ACCOUNT_ID>`：在前一個步驟中從 Astra Control 網路 UI 取得。
  - `<CLUSTER_NAME>`：應在 Astra Control 中指派此叢集的名稱。
  - `<ASTRA_CONTROL_URL>`：Astra Control 的網路 UI URL。例如：

```
https://astra.control.url
```

```

apiVersion: astra.netapp.io/v1
kind: AstraConnector
metadata:
 name: astra-connector
 namespace: astra-connector
spec:
 astra:
 accountId: <ASTRA_CONTROL_ACCOUNT_ID>
 clusterName: <CLUSTER_NAME>
 #Only set `skipTLSValidation` to `true` when using the default
 self-signed
 #certificate in a proof-of-concept environment.
 skipTLSValidation: false #Should be set to false in production
 environments
 tokenRef: astra-token
 natsSyncClient:
 cloudBridgeURL: <ASTRA_CONTROL_HOST_URL>
 imageRegistry:
 name: cr.astra.netapp.io
 secret: regcred

```

7. 填入之後 astra-connector-cr.yaml 使用正確值的檔案、請套用 CR：

```
kubectl apply -n astra-connector -f astra-connector-cr.yaml
```

8. 確認 Astra Connector 已完全部署：

```
kubectl get all -n astra-connector
```

9. 確認叢集已向 Astra Control 註冊：

```
kubectl get astraconnectors.astra.netapp.io -A
```

您應該會看到類似下列的輸出：

| NAMESPACE       | NAME                  | REGISTERED | ASTRACONNECTORID             |
|-----------------|-----------------------|------------|------------------------------|
| STATUS          |                       |            |                              |
| astra-connector | astra-connector       | true       | 00ac8-2cef-41ac-8777-ed0583e |
|                 | Registered with Astra |            |                              |

10. 驗證叢集是否出現在 Astra Control 網路 UI 的 \* Clusters\* 頁面上的受管理叢集清單中。

## 新增叢集

若要開始管理應用程式、請新增Kubernetes叢集、並將其當作運算資源來管理。您必須為Astra Control Center新增叢集、才能探索Kubernetes應用程式。



我們建議Astra Control Center先管理部署於上的叢集、再將其他叢集新增至Astra Control Center進行管理。需要管理初始叢集、才能傳送Kubmetrics資料和叢集相關資料、以供進行度量和疑難排解。

### 開始之前

- 新增叢集之前、請先檢閱並執行必要的 ["必要工作"](#)。
- 如果您使用的是 ONTAP SAN 驅動程式、請務必在所有 Kubernetes 叢集上啟用多重路徑。

### 步驟

1. 從儀表板或叢集功能表瀏覽：
  - 從「資源摘要」的\*「儀表板」中、從「叢集」窗格中選取「新增\*」。
  - 在左側導覽區域中、選取\*叢集\*、然後從「叢集」頁面選取\*新增叢集\*。
2. 在打開的\* Add Cluster.yaml視窗中、上傳「kubeconfig · yaml」檔案、或貼上「kubeconfig · yaml」檔案的內容。



「kubeconfig · yaml」檔案應包含\*一個叢集\*的叢集認證資料。



如果您自行建立 kubeconfig 檔案中、您應該只定義\*一個\*內容元素。請參閱 ["Kubernetes 文件"](#) 以取得有關建立的資訊 kubeconfig 檔案：如果您使用為有限的叢集角色建立了Kbeconfig ["此程序"](#)請務必在本步驟中上傳或貼上該KECBConnfig。

3. 提供認證名稱。根據預設、認證名稱會自動填入為叢集名稱。
4. 選擇\*下一步\*。
5. 選取要用於此Kubernetes叢集的預設儲存類別、然後選取\* Next\*。



您應該選取在 Astra Control Provisioner 中設定並由 ONTAP 儲存設備備份的儲存類別。

6. 檢閱資訊、如果一切看起來都很好、請選取\*新增\*。

### 結果

叢集進入\*探索\*狀態、然後變更為\*健全\*。您現在正使用Astra Control Center來管理叢集。



在Astra Control Center中新增要管理的叢集之後、可能需要幾分鐘的時間來部署監控操作員。在此之前、通知圖示會變成紅色、並記錄\*監控代理程式狀態檢查失敗\*事件。您可以忽略這一點、因為當Astra Control Center取得正確狀態時、問題就能解決。如果幾分鐘內仍無法解決問題、請前往叢集並執行 `oc get pods -n netapp-monitoring` 做為起點。您需要查看監控操作員記錄、以偵錯問題。

## 在 ONTAP 儲存後端上啟用驗證

### Astra Control Center 提供兩種驗證 ONTAP 後端的模式：

- **\* 認證型驗證 \***：具有必要權限的 ONTAP 使用者的使用者名稱和密碼。您應該使用預先定義的安全登入角色、例如 admin 或 vsadmin、以確保與 ONTAP 版本的最大相容性。
- **\* 憑證型驗證 \***：Astra 控制中心也可以使用安裝在後端的憑證與 ONTAP 叢集通訊。您應該使用用戶端憑證、金鑰和信任的 CA 憑證（如果使用）（建議使用）。

您可以稍後更新現有的後端、將某種驗證類型移至另一種方法。一次只支援一種驗證方法。

#### 啟用認證型驗證

Astra Control Center 需要具備叢集範圍的認證 admin 與 ONTAP 後端通訊。您應該使用預先定義的標準角色、例如 admin。這可確保與未來 ONTAP 版本的前移相容性、這些版本可能會公開未來 Astra 控制中心版本所使用的功能 API。



您可以建立自訂安全登入角色、並與 Astra Control Center 搭配使用、但不建議使用。

後端定義範例如下：

```
{
 "version": 1,
 "backendName": "ExampleBackend",
 "storageDriverName": "ontap-nas",
 "managementLIF": "10.0.0.1",
 "dataLIF": "10.0.0.2",
 "svm": "svm_nfs",
 "username": "admin",
 "password": "secret"
}
```

後端定義是唯一以純文字儲存認證的地方。建立或更新後端是唯一需要具備認證知識的步驟。因此、這是僅供管理員使用的操作、由 Kubernetes 或儲存管理員執行。

#### 啟用憑證型驗證

Astra 控制中心可以使用憑證與新的和現有的 ONTAP 後端通訊。您應該在後端定義中輸入下列資訊。

- `clientCertificate`：用戶端憑證。
- `clientPrivateKey`：關聯的私鑰。
- `trustedCACertificate`：可信 CA 證書。如果使用信任的CA、則必須提供此參數。如果未使用信任的CA、則可忽略此問題。

您可以使用下列其中一種類型的憑證：

- 自我簽署的憑證

- 協力廠商憑證

## 啟用自我簽署憑證的驗證

典型的工作流程包括下列步驟。

### 步驟

1. 產生用戶端憑證和金鑰。產生時、請將一般名稱（CN）設定為 ONTAP 使用者、以驗證為。

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=<common-name>"
```

2. 安裝用戶端類型的憑證 client-ca 以及 ONTAP 叢集上的金鑰。

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

3. 確認 ONTAP 安全登入角色支援憑證驗證方法。

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

4. 使用產生的憑證測試驗證。以管理 LIF IP 和 SVM 名稱取代 ONTAP Management LIF> 和 <vserver name>。  
◦ 您必須確保LIF的服務原則設定為 default-data-management。

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns=http://www.netapp.com/filer/admin version="1.21" vfiler="<vserver-
name>"><vserver-get></vserver-get></netapp>
```

5. 使用從上一步取得的值、在 Astra Control Center UI 中新增儲存後端。

## 啟用協力廠商憑證的驗證

如果您有協力廠商憑證、您可以使用這些步驟來設定憑證型驗證。

### 步驟

1. 產生私密金鑰和 CSR：

```
openssl req -new -newkey rsa:4096 -nodes -sha256 -subj "/" -outform pem
-out ontap_cert_request.csr -keyout ontap_cert_request.key -addext
"subjectAltName = DNS:<ONTAP_CLUSTER_FQDN_NAME>,IP:<ONTAP_MGMT_IP>"
```

2. 將 CSR 傳遞至 Windows CA（協力廠商 CA）、然後核發簽署的憑證。
3. 下載已簽署的憑證、並將其命名為「ontap\_signed\_cert.crt」
4. 從 Windows CA（協力廠商 CA）匯出根憑證。
5. 命名此檔案 ca\_root.crt

您現在有下列三個檔案：

- \* 私密金鑰 \*：ontap\_signed\_request.key（這是 ONTAP 中伺服器憑證的對應金鑰。安裝伺服器憑證時需要此功能。）
  - \* 簽署憑證 \*：ontap\_signed\_cert.crt（這在 ONTAP 中也稱為伺服器憑證 \_。）
  - \* 根 CA 憑證 \*：ca\_root.crt（這在 ONTAP 中也稱為 \_server-ca 憑證 \_。）
6. 在 ONTAP 中安裝這些憑證。產生及安裝 server 和 server-ca ONTAP 上的憑證。



```
Copy the contents of ca_root.crt and use it here.
```

```
security certificate install -type server-ca
```

Please enter Certificate: Press <Enter> when done

```
-----BEGIN CERTIFICATE-----
```

```
<certificate details>
```

```
-----END CERTIFICATE-----
```

You should keep a copy of the CA-signed digital certificate for future reference.

The installed certificate's CA and serial number for reference:

CA:

serial:

The certificate's generated name for reference:

```
===
```

```
Copy the contents of ontap_signed_cert.crt and use it here. For
key, use the contents of ontap_cert_request.key file.
```

```
security certificate install -type server
```

Please enter Certificate: Press <Enter> when done

```
-----BEGIN CERTIFICATE-----
```

```
<certificate details>
```

```
-----END CERTIFICATE-----
```

Please enter Private Key: Press <Enter> when done

```
-----BEGIN PRIVATE KEY-----
```

```
<private key details>
```

```
-----END PRIVATE KEY-----
```

Enter certificates of certification authorities (CA) which form the certificate chain of the server certificate. This starts with the issuing CA certificate of the server certificate and can range up to the root CA certificate.

Do you want to continue entering root and/or intermediate

```
certificates {y|n}: n
```

The provided certificate does not have a common name in the subject field.

Enter a valid common name to continue installation of the certificate: <ONTAP\_CLUSTER\_FQDN\_NAME>

You should keep a copy of the private key and the CA-signed digital certificate for future reference.

The installed certificate's CA and serial number for reference:

CA:

serial:

The certificate's generated name for reference:

```
==
```

```
Modify the vsrver settings to enable SSL for the installed certificate
```

```
ssl modify -vsrver <vsrver_name> -ca <CA> -server-enabled true
-serial <serial number> (security ssl modify)
```

```
==
```

```
Verify if the certificate works fine:
```

```
openssl s_client -CAfile ca_root.crt -showcerts -servername server
-connect <ONTAP_CLUSTER_FQDN_NAME>:443
```

```
CONNECTED(00000005)
```

```
depth=1 DC = local, DC = umca, CN = <CA>
```

```
verify return:1
```

```
depth=0
```

```
verify return:1
```

```
write W BLOCK
```

```

```

```
Certificate chain
```

```
0 s:
```

```
 i:/DC=local/DC=umca/<CA>
```

```
-----BEGIN CERTIFICATE-----
```

```
<Certificate details>
```

7. 為同一主機建立用戶端憑證、以進行無密碼通訊。Astra 控制中心使用此程序與 ONTAP 通訊。
8. 在 ONTAP 上產生及安裝用戶端憑證：

```
Use /CN=admin or use some other account which has privileges.
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout
ontap_test_client.key -out ontap_test_client.pem -subj "/CN=admin"

Copy the content of ontap_test_client.pem file and use it in the
below command:
security certificate install -type client-ca -vserver <vserver_name>

Please enter Certificate: Press <Enter> when done

-----BEGIN CERTIFICATE-----
<Certificate details>
-----END CERTIFICATE-----

You should keep a copy of the CA-signed digital certificate for
future reference.
The installed certificate's CA and serial number for reference:

CA:
serial:
The certificate's generated name for reference:

==

ssl modify -vserver <vserver_name> -client-enabled true
(security ssl modify)

Setting permissions for certificates
security login create -user-or-group-name admin -application ontapi
-authentication-method cert -role admin -vserver <vserver_name>

security login create -user-or-group-name admin -application http
-authentication-method cert -role admin -vserver <vserver_name>

==

#Verify passwordless communication works fine with the use of only
certificates:

curl --cacert ontap_signed_cert.crt --key ontap_test_client.key
--cert ontap_test_client.pem
https://<ONTAP_CLUSTER_FQDN_NAME>/api/storage/aggregates
{
```

```

"records": [
{
"uuid": "f84e0a9b-e72f-4431-88c4-4bf5378b41bd",
"name": "<aggr_name>",
"node": {
"uuid": "7835876c-3484-11ed-97bb-d039ea50375c",
"name": "<node_name>",
"_links": {
"self": {
"href": "/api/cluster/nodes/7835876c-3484-11ed-97bb-d039ea50375c"
}
}
},
"_links": {
"self": {
"href": "/api/storage/aggregates/f84e0a9b-e72f-4431-88c4-4bf5378b41bd"
}
}
},
],
"num_records": 1,
"_links": {
"self": {
"href": "/api/storage/aggregates"
}
}
}%

```

9. 在 Astra Control Center UI 中新增儲存後端、並提供下列值：

- \* 用戶端憑證 \*：ONTAP 測試用戶端 .pem
- \* 私密金鑰 \*：ontap\_test\_client.key
- \* 可信 CA 證書 \*：ONTAP 簽署的 \_cert.crt

## 新增儲存後端

設定認證或憑證驗證資訊之後、您可以將現有的 ONTAP 儲存後端新增至 Astra 控制中心、以管理其資源。

將Astra Control中的儲存叢集管理為儲存後端、可讓您在持續磁碟區（PV）與儲存後端之間建立連結、以及取得額外的儲存指標。

如果您已啟用 Astra 控制資源配置程式、在 Astra 控制中心新增及管理 ONTAP 儲存設備後端、則使用 NetApp SnapMirror 技術時為選用功能。

## 步驟

1. 從左側導覽區域的儀表板中、選取\*後端\*。
2. 選取\*「Add\*」。
3. 在「新增儲存設備後端」頁面的「使用現有的」區段中、選取 \* ONTAP \*。
4. 選取下列其中一項：
  - \* 使用管理員認證 \*：輸入 ONTAP 叢集管理 IP 位址和管理認證。認證資料必須是整個叢集的認證資料。



您在此處輸入認證的使用者必須擁有 `ontapi` 使用者登入存取方法已在ONTAP 支援的叢集上的「支援系統管理程式」中啟用ONTAP。如果您打算使用SnapMirror複寫、請套用具有「admin」角色的使用者認證、該角色具有存取方法 `ontapi` 和 `http`、在來源ONTAP 和目的地等叢集上。請參閱 ["管理ONTAP 使用者帳戶、請參閱本文檔"](#) 以取得更多資訊。

- \* 使用憑證 \*：上傳憑證 `.pem` 檔案、憑證金鑰 `.key` 檔案、以及選擇性的憑證授權單位檔案。
5. 選擇\*下一步\*。
  6. 確認後端詳細資料、然後選取\*管理\*。

## 結果

後端隨即出現在中 online 列出摘要資訊。



您可能需要重新整理頁面、以便顯示後端。

## 新增儲存庫

您可以使用Astra Control UI或來新增儲存區 ["Astra Control API"](#)。如果您想要備份應用程式和持續儲存設備、或是想要跨叢集複製應用程式、則必須新增物件存放區資源庫供應商。Astra Control會將這些備份或複製儲存在您定義的物件存放區中。

如果您要將應用程式組態和持續儲存設備複製到同一個叢集、則無需使用Astra Control中的儲存庫。應用程式快照功能不需要儲存庫。

## 開始之前

- 確保您擁有一個可從 Astra Control Center 所管理的叢集存取的貯體。
- 確保您擁有貯體的認證。
- 確認貯體為下列其中一種類型：
  - NetApp ONTAP 產品S3
  - NetApp StorageGRID 產品S3
  - Microsoft Azure
  - 一般S3



Amazon Web Services (AWS) 和Google Cloud Platform (GCP) 使用通用S3儲存區類型。



雖然Astra Control Center支援Amazon S3做為通用S3儲存區供應商、但Astra Control Center可能不支援所有聲稱Amazon S3支援的物件儲存區廠商。

#### 步驟

1. 在左側導覽區域中、選取\*鏟斗\*。
2. 選取\*「Add\*」。
3. 選取貯體類型。



新增儲存庫時、請選擇正確的儲存庫供應商、並提供該供應商的適當認證資料。例如、UI接受NetApp ONTAP S3作為類型並接受StorageGRID 驗證、但這將導致所有未來使用此儲存庫的應用程式備份與還原失敗。

4. 輸入現有的庫位名稱和選用說明。



庫位名稱和說明會顯示為備份位置、您可以在建立備份時稍後選擇。此名稱也會在保護原則組態期間顯示。

5. 輸入S3端點的名稱或IP位址。
6. 在「選取認證」下、選擇「新增」或「使用現有」索引標籤。
  - 如果您選擇\*新增\*：
    - i. 在Astra Control中輸入認證與其他認證不同的名稱。
    - ii. 從剪貼簿貼上內容、輸入存取ID和秘密金鑰。
  - 如果您選擇\*使用現有\*：
    - i. 選取您要搭配儲存區使用的現有認證資料。
7. 選取 Add。



當您新增貯體時、Astra Control會使用預設的貯體指標來標記一個貯體。您建立的第一個儲存區會成為預設儲存區。當您新增儲存庫時、可以稍後決定 ["設定另一個預設儲存區"](#)。

## 版權資訊

Copyright © 2025 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。