aws

使用者指南

Amazon EKS

Amazon EKS: 使用者指南

Table of Contents

什麼是 Amazon EKS？

Amazon Elastic Kubernetes Service (Amazon EKS) 是一項受管服務，無需在 Amazon Web Services (AWS) 上安裝、操作和維護您自己的 Kubernetes 控制平面。Kubernetes 是一套開放原始碼系統，可自動化容器化應用程式的管理、擴展和部署。

Amazon EKS 的功能

以下是 Amazon EKS 的關鍵功能：

安全聯網與驗證

Amazon EKS 將您的Kubernetes工作負載與 AWS 網路和安全服務整合。它還與 AWS Identity and Access Management (IAM) 整合，為您的Kubernetes叢集提供身份驗證。

輕鬆擴展叢集

Amazon EKS 能讓您根據工作負載的需求，輕鬆向上和向下擴展 Kubernetes 叢集。Amazon EKS 支援根據 CPU 或自訂指標的水平 Pod 自動擴展，以及根據整個工作負載需求的叢集自動擴展。

受管 Kubernetes 體驗

您可以使用 `eksctl`、AWS Management Console、AWS Command Line Interface (AWS CLI)、API、`kubectl`，以及 Terraform 來變更您的 Kubernetes 叢集。

高可用性

Amazon EKS 在多個可用區域中為您的控制平面提供高可用性。

與 AWS 服務整合

Amazon EKS 與其他 AWS 服務整合，提供部署和管理容器化應用程式的全方位平台。您也可以更輕鬆地使用各種可觀測性工具來疑難排解您的 Kubernetes 工作負載。

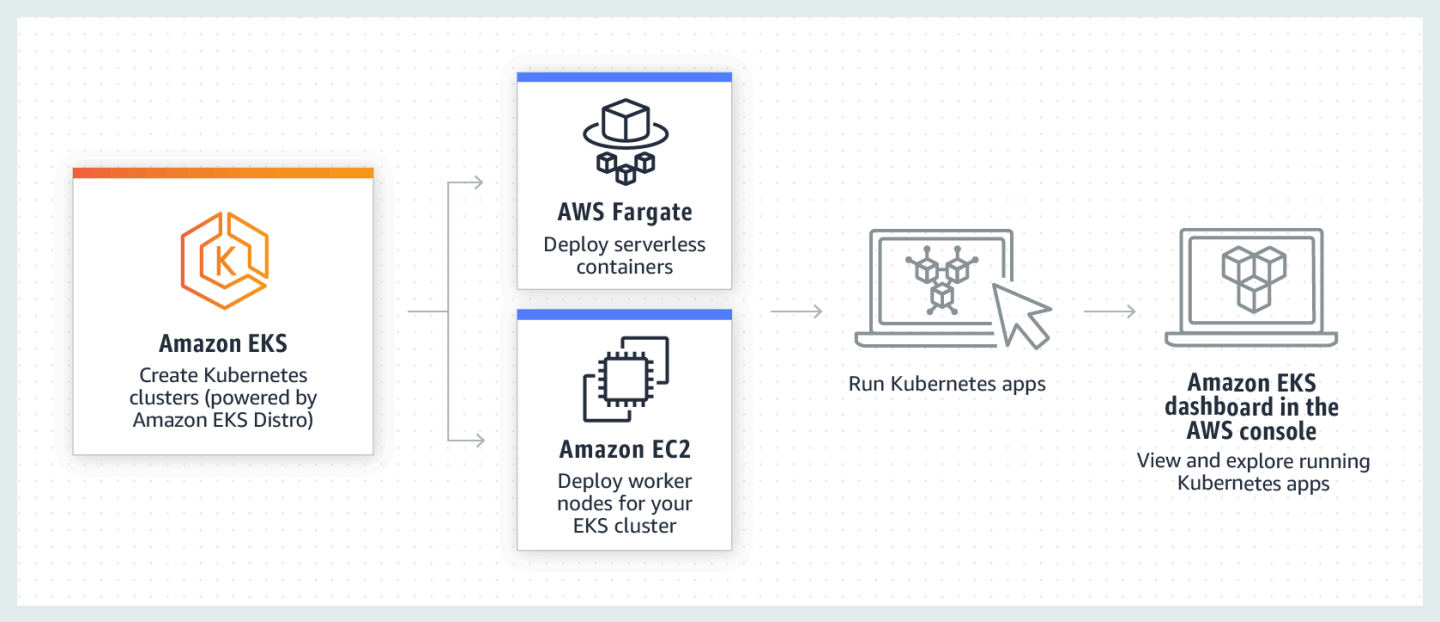
如需 Amazon EKS 其他功能的詳細資訊，請參閱 Amazon EKS 功能。

開始使用 Amazon EKS

若要建立您的第一個叢集及其相關資源，請參閱 Amazon EKS 入門。一般來說，開始使用 Amazon EKS 會需要下列步驟。

1. 建立叢集 — 首先使用`eksctl` AWS Management Console AWS CLI、或其中一個 AWS SDK 建立叢集。
2. 選擇您的運算資源方式 — 決定受控節點群組和自我管理節點。 AWS FargateKarpenter
3. 設定 – 設定必要的控制器、驅動程式和服務。
4. 部署工作負載 – 量身打造您的 Kubernetes 工作負載來充分利用所選節點類型的資源和功能。
5. 管理 – 監督您的工作負載、整合 AWS 服務來簡化操作，並強化工作負載效能。您可以使用檢視工作負載的相關資訊 AWS Management Console。

下列圖表顯示執行雲端 Amazon EKS 的基本流程。若要了解其他 Kubernetes 部署選項，請參閱 部署選項。



Amazon EKS 的定價

Amazon EKS 叢集由一個控制平面和您在其中執行 Pods 的 Amazon Elastic Compute Cloud (Amazon EC2) 或 Fargate 運算組成。如需控制平面定價的詳細資訊，請參閱 Amazon EKS 定價。Amazon EC2 和 Fargate 都提供：

隨需執行個體

> 您只需要按秒支付使用執行個體的費用，無須長期購買或預付款。如需詳細資訊，請參閱 Amazon EC2 隨需定價 和 AWS Fargate 定價。

, Savings Plans

您可以透過對一致的使用量設定綁約來降低成本，每小時以美金計價，為期一年或三年。如需詳細資訊，請參閱透過 Savings Plans 定價。

Amazon EKS 中的常用案例

Amazon EKS 在 AWS 上提供強大的受管 Kubernetes 服務，專為最佳化容器化應用程式而設計。以下是一些 Amazon EKS 最常見的使用案例，協助您充分利用其強項來滿足您的特定需求。

部署高可用性應用程式

使用 Elastic Load Balancing，您可以確保您的應用程式在多個可用區域之中均高度可用。

建置微服務架構

搭配 AWS Cloud Map 或 Amazon VPC Lattice 使用 Kubernetes 服務探索功能來建置彈性系統。

自動化軟體發行程序

管理持續整合和持續部署 (CICD) 管道，這些管道可簡化應用程式的自動化建置、測試和部署程序。

執行無伺服器應用程式

搭配 Amazon EKS 使用 AWS Fargate 來執行無伺服器應用程式。這表示您可以專注於應用程式開發，而 Amazon EKS 和 Fargate 則可以處理基礎結構。

執行機器學習工作負載

Amazon EKS 與熱門的機器學習架構相容，例如 TensorFlow、MXNet，以及 PyTorch。有了 GPU 支援，您甚至可以有效地處理複雜的機器學習任務。

在內部部署和雲端以一致的方式部署

使用 Amazon EKS Anywhere 在您自己的基礎結構上使用與雲端 Amazon EKS 一致的工具來操作 Kubernetes 叢集。

執行符合成本效益的批次處理和大數據工作負載

運用 Spot 執行個體來執行批次處理和大數據工作負載，例如 Apache Hadoop 和 Spark，僅需極少的成本。這可讓您以折扣價格充分利用未使用的 Amazon EC2 容量。

保護應用程式並確保合規性

使用 Amazon EKS 實作強大的安全實務並維持合規性，此功能與 AWS 安全服務整合，例如 AWS Identity and Access Management (IAM)、Amazon Virtual Private Cloud (Amazon VPC) 和 AWS Key Management Service (AWS KMS)。這確保了資料隱私權和保護均依照產業標準實行。

Amazon EKS 架構

Amazon EKS 與 Kubernetes 的一般叢集架構一致。如需詳細資訊，請參閱 Kubernetes 文件中的 Kubernetes 元件。下列各節摘要了 Amazon EKS 一些額外架構的詳細資訊。

控制平台

Amazon EKS 能確保每個叢集都有自己獨一無二的 Kubernetes 控制平面。這種設計能讓每個叢集的基礎結構保持分隔，叢集或 AWS 帳戶之間沒有重疊。設定包含：

分散式元件

控制平面會放置至少兩個 API 伺服器執行個體和三個 `etcd` 執行個體，這些執行個體會在 AWS 區域 內的三個 AWS 可用區域中執行。

理想效能

Amazon EKS 會主動監控和調整控制平面執行個體，以維持尖峰效能。

復原能力

如果控制平面執行個體動搖，如有需要，Amazon EKS 會使用其他可用區域迅速取代它。

一致的執行時間

透過在多個可用區域之中執行叢集的方式來達成可靠的 API 伺服器端點可用性服務水準協議 (SLA)。

Amazon EKS 會使用 Amazon Virtual Private Cloud (Amazon VPC) 來限制單一叢集內控制平面元件之間的流量。叢集元件無法檢視或接收來自其他叢集或 AWS 帳戶的通訊，唯當元件已獲得 Kubernetes 角色型存取控制 (RBAC) 政策的授權時例外。

運算

除了控制平面之外，Amazon EKS 叢集還有一組稱為節點的工作者機器。選取適當的 Amazon EKS 叢集節點類型，對於滿足您特定的需求與最佳化資源使用率而言十分關鍵。Amazon EKS 提供下列主節點類型：

AWS Fargate

Fargate 是適用於容器的無伺服器運算引擎，無需管理基礎執行個體。使用 Fargate，您可以指定應用程式的資源需求，且 AWS 會自動佈建、擴展和維護基礎結構。此選項非常適合優先考慮簡單易用的使用者，也相當適合希望專注於應用程式開發和部署，而不是管理基礎結構的使用者。

Karpenter

Karpenter 是靈活、高效能的 Kubernetes 叢集自動擴展工具，有助於提高應用程式可用性和叢集效率。Karpenter 會啟動大小適中的運算資源，以便回應不斷變化的應用程式負載。此選項可佈建符合工作負載要求的即時運算資源。

受管節點群組

受管節點群組融合了可用來管理 Amazon EKS 叢集中一系列 Amazon EC2 執行個體的自動化和自訂功能。AWS 負責修補、更新和擴展節點之類的任務，大幅簡化操作層面的複雜性。同時，也支援自訂 `kubelet` 引數，為進階 CPU 和記憶體管理政策開啟了可能性。此外，它們會透過服務帳戶的 AWS Identity and Access Management (IAM) 角色來增強安全性，同時減少每個叢集對個別許可的需求。

自我管理的節點

自我管理節點提供對 Amazon EKS 叢集中的 Amazon EC2 執行個體的完全控制權。您負責管理、擴展和維護節點，讓您完全掌握基礎結構。此選項適合需要精細控制和自訂其節點的使用者，也適合已經準備好投入時間來管理和維護其基礎架構的使用者。

Kubernetes 概念

Amazon Elastic Kubernetes Service（Amazon EKS）是一種基於開源項目的 AWS 託管服務。Kubernetes雖然 Amazon EKS 服務如何與 AWS 雲端整合 (特別是當您第一次建立 Amazon EKS 叢集時)，您需要瞭解一些事項，但是一旦啟動並執行，就像使用任何其他叢集相同的方式使用 Amazon EKS 叢集。Kubernetes因此，若要開始管理Kubernetes叢集和部署工作負載，您至少需要對Kubernetes概念有基本的了解。

此頁面將Kubernetes概念劃分為三個部分：原因Kubernetes、叢集和工作負載。第一節說明執行Kubernetes服務的價值，特別是 Amazon EKS 等受管服務。「工作負載」區段涵蓋Kubernetes應用程式的建置、儲存、執行和管理方式。[叢集] 區段會列出組成Kubernetes叢集的不同元件，以及建立和維護Kubernetes叢集的責任。

主題

當您瀏覽此內容時，連結將引導您進一步瞭解 Amazon EKS 和Kubernetes文件中Kubernetes概念的詳細說明，以防您想深入探討我們在此處介紹的任何主題。如需 Amazon EKS 如何實作Kubernetes控制平面和運算功能的詳細資訊，請參閱 Amazon EKS 架構。

為什麼選擇 Kubernetes？

Kubernetes在執行關鍵任務、生產品質的容器化應用程式時，可提升可用性和可擴充性。不只是在單一機器Kubernetes上執行 (儘管這是可能的)，而是讓您跨可擴充或收縮以滿足需求的電腦組執行應用程式，藉此Kubernetes達成這些目標。 Kubernetes包含的功能可讓您更輕鬆地執行以下操作：

- 在多部機器上部署應用程式 (使用 Pod 中部署的容器)
- 監控容器健康狀況並重新啟動失敗
- 根據負載擴展和縮減容器
- 使用新版本更新容器
- 在容器之間分配資源
- 平衡機器之間的流量

將這些類型的複雜工作Kubernetes自動化後，應用程式開發人員可以專注於建置和改善其應用程式工作負載，而不必擔心基礎結構。開發人員通常會建立描述應用程式所需狀態的設定檔 (格式為 YAML 檔案)。這可能包括要執行的容器、資源限制、Pod 複本數目、CPU/ 記憶體配置、相似性規則等。

的屬性 Kubernetes

為了實現其目標，Kubernetes具有以下屬性：

- 容器化-Kubernetes 是一種容器協調工具。若要使用Kubernetes，您必須先將應用程式容器化。視應用程式類型而定，這可能是一組微服務、批次工作或其他形式。然後，您的應用程式可以利用包含龐大工具生態系統的工Kubernetes作流程，讓容器可以作為映像儲存在容器登錄中、部署到Kubernetes叢集，並在可用節點上執行。您可以使用Docker或其他容器執行階段在本機電腦上建置和測試個別容器，然後再將它們部署到Kubernetes叢集。
- 可擴充-如果您的應用程式需求超過這些應用程式執行中執行個體的容量，Kubernetes就能夠擴充。視需要，Kubernetes可判斷應用程式是否需要更多 CPU 或記憶體，並透過自動擴充可用容量或使用更多現有容量來回應。如果有足夠的運算可用於只執行更多應用程式執行個體 (水平 Pod 自動調度資源)，或者在節點層級 (如果需要啟動更多節點來處理增加的容量 (叢集自動配置器或 Karpenter)，則可以在 Pod 層級完成擴展。由於不再需要容量，這些服務可以刪除不必要的 Pod 並關閉不需要的節點。
- 可用-如果應用程式或節點運作狀態不佳或無法使用，Kubernetes可以將執行中的工作負載移至另一個可用節點。您只需刪除正在執行工作負載的工作負載或節點的執行個體，即可強制執行此問題。這裡的底線是，如果工作負載無法再在其他位置執行，就可以在其他位置提升。
- 宣告式-Kubernetes 使用主動式調解來持續檢查您為叢集宣告的狀態是否符合實際狀態。例如，透過將Kubernetes物件套用至叢集 (通常是透過 YAML 格式的組態檔)，您可以要求啟動要在叢集上執行的工作負載。您可以稍後更改配置以執行某些操作，例如使用更高版本的容器或分配更多內存。Kubernetes將做它需要做的事情來建立所需的狀態。這可能包括啟動或關閉節點、停止和重新啟動工作負載，或提取更新的容器。
- 組合式-因為應用程式通常由多個元件組成，因此您希望能夠同時管理一組這些元件 (通常由多個容器表示)。儘管Docker撰寫提供了一種直接執行此操作的方法Docker，但 Kubernetes Kompose 命令可以幫助您使用。Kubernetes有關如何執行此操作的範例，請參閱將Docker構成檔案 Translate 為Kubernetes資源。
- 可擴展-與專有軟件不同，開源Kubernetes項目旨在向您開放擴展Kubernetes任何您喜歡的方式以滿足您的需求。API 和配置文件是開放的，以直接修改。鼓勵第三方編寫自己的控制器，以擴展基礎架構和最終用戶Kubernetes功能。Webhook 可讓您設定叢集規則以強制執行原則並適應不斷變化的條件。如需如何擴充Kubernetes叢集的詳細想法，請參閱延伸Kubernetes。
- 可攜式-許多組織已將其作業標準化，Kubernetes因為它可以讓他們以相同的方式管理所有應用程式需求。開發人員可以使用相同的管道來建置和儲存容器化應用程式。然後，這些應用程式可以部署到執行內部部署、雲端、餐廳的 point-of-sales 終端機或分散在公司遠端站台的 IOT 裝置上的Kubernetes叢集。它的開源性質使人們有可能開發這些特殊的發Kubernetes行版，以及管理它們所需的意志工具。

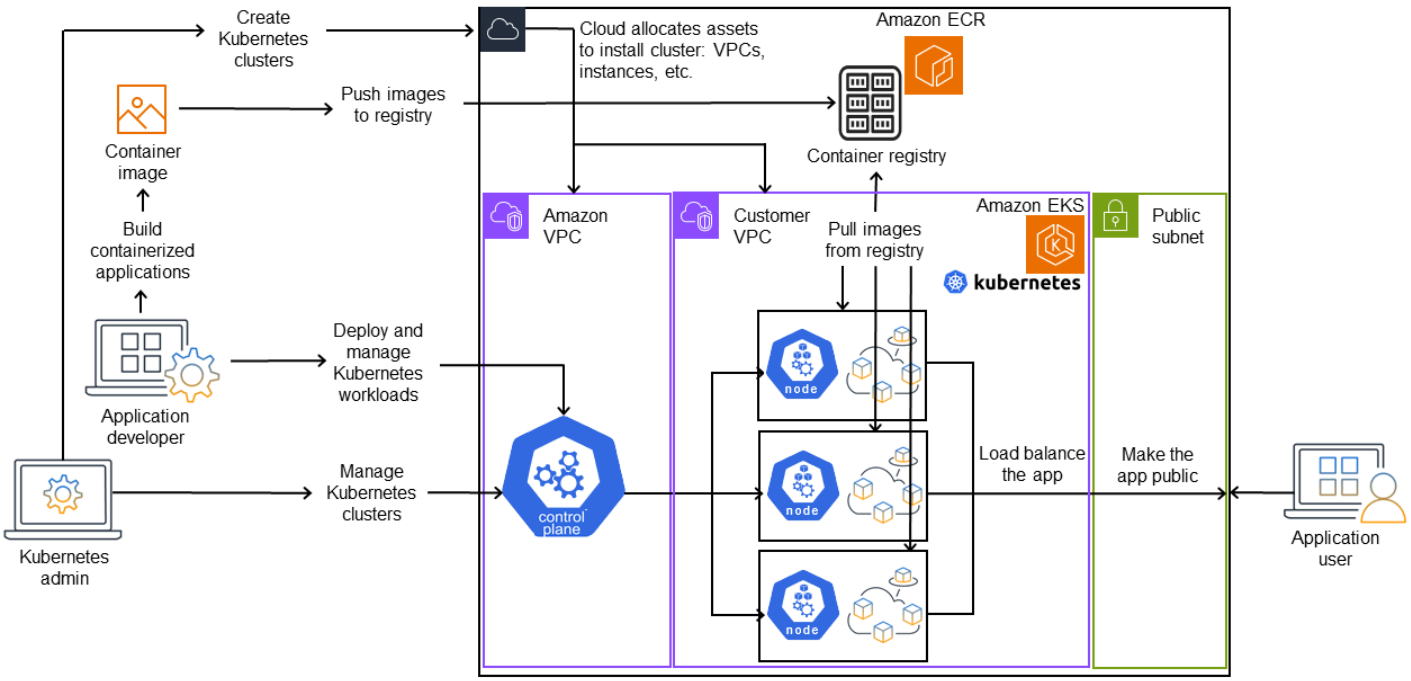
管理 Kubernetes

Kubernetes源代碼是免費提供的，因此您可以使用自己的設備進行安裝和管理Kubernetes。但是，自我管理Kubernetes需要深厚的運營專業知識，並且需要時間和精力來維護。基於這些原因，大多數部署生產工作負載的人會選擇雲端供應商 (例如 Amazon EKS) 或現場部署供應商 (例如 Amazon EKS Anywhere)，並提供自己經過測試的KubernetesKubernetes分配和專家支援。這可讓您卸載維護叢集所需的大部分無差別繁重工作，包括：

- 硬體-如果您沒有可Kubernetes根據需求執行的硬體， AWS Amazon EKS 之類的雲端供應商可以為您節省前期成本。使用 Amazon EKS，這表示您可以使用所提供的最佳雲端資源 AWS，包括電腦執行個體 (Amazon 彈性運算雲端)、您自己的私有環境 (Amazon VPC)、中央身分和許可管理 (IAM) 以及儲存 (Amazon EBS)。 AWS 管理運行所需的計算機，網絡，數據中心和所有其他物理組件Kubernetes。同樣地，您不必規劃資料中心處理最高需求日期的最大容量。對於 Amazon EKS Anywhere 或其他內部部署Kubernetes叢集，您必須負責管理Kubernetes部署中使用的基礎設施，但仍然可以依靠協 AWS 助您保持Kubernetes最新狀態。
- 控制平面管理-Amazon EKS 可管理 AWS託管Kubernetes控制平面的安全性和可用性，該平面負責排程容器、管理應用程式的可用性和其他關鍵任務，讓您可以專注於應用程式工作負載。如果叢集中斷， AWS 應該可以將叢集還原至執行中狀態。對於 Amazon EKS Anywhere，您可以自己管理控制平面。
- 經過測試的升級-升級叢集時，您可以仰賴 Amazon EKS 或 Amazon EKS Anywhere 來提供其Kubernetes發行版的測試版本。
- 附加元件-有數百個專案可擴充和使用，您可以新增至叢集的基礎架構，或使用Kubernetes這些專案來協助您執行工作負載。 AWS 提供可與叢集搭配使用的 Amazon EKS 附加元件，而不是自行建置和管理這些附加元件。Amazon EKS Anywhere 提供精選套件，其中包含許多熱門開放原始碼專案的組建。因此，您不必自行建置軟體或管理重要的安全性修補程式、錯誤修正或升級。同樣，如果默認值滿足您的需求，那麼通常只需對這些附加組件進行非常少的配置。如需使用附加元件擴充叢集的詳細資訊，請參閱延伸叢集

Kubernetes在行動

下圖顯示您作為「Kubernetes管理員」或「應用程式開發人員」建立和使用Kubernetes叢集時可執行的重要活動。在此過程中，它說明了Kubernetes組件如何相互交互，以 AWS 雲為基礎雲提供者的示例。



Kubernetes管理員使用特定於將在其上建立Kubernetes叢集之提供者類型的工具來建立叢集。此範例使用 AWS 雲端做為供應商，該供應商提供稱為 Amazon EKS 的受管Kubernetes服務。受管服務會自動配置建立叢集所需的資源，包括為叢集建立兩個新的虛擬私有雲端 (Amazon VPC)、設定聯網、將許可映射到雲端中管理資產的Kubernetes許可，看到控制平面服務有可執行的地方，以及將零個或多個 Amazon EC2 執行個體配置為執行工作負載的Kubernetes節點。 AWS 為控制平面管理一個 Amazon VPC 本身，而另一個 Amazon VPC 則包含執行工作負載的客戶節點。

許多Kubernetes管理員未來的任務都是使用諸如 kubectl 之類的Kubernetes工具完成的。該工具會直接向叢集的控制平面提出服務要求。對叢集進行查詢和變更的方式與您在任何Kubernetes叢集上執行這些變更的方式非常相似。

想要將工作負載部署到此叢集的應用程式開發人員可以執行多項工作。開發人員需要將應用程式建置到一或多個容器映像中，然後將這些映像推送至Kubernetes叢集可存取的容器登錄。 AWS 為此提供 Amazon Elastic Container Registry (Amazon ECR)。

若要執行應用程式，開發人員可以建立 YAML 格式的組態檔，告知叢集如何執行應用程式，包括要從登錄中提取的容器，以及如何將這些容器包裝在 Pod 中。控制平面（調度程序）將容器排程到一個或多個節點，並且每個節點上的容器運行時實際上提取並運行所需的容器。開發人員也可以設定應用程式負載平衡器，以平衡每個節點上執行之可用容器的流量，並將應用程式公開，以便在公用網路上提供給外部世界。完成這一切後，想要使用該應用程序的人可以連接到應用程序端點進行訪問。

從Kubernetes叢集和工作負載的角度來看，下一節將介紹這些功能的詳細資訊。

叢集

如果您的工作是啟動和管理Kubernetes叢集，您應該知道Kubernetes叢集的建立、增強、管理和刪除方式。您也應該知道組成叢集的元件是什麼，以及維護這些元件所需要採取的動作。

管理叢集的工具可處理Kubernetes服務與基礎硬體提供者之間的重疊。因此，這些任務的自動化通常由Kubernetes供應商 (例如 Amazon EKS 或 Amazon EKS Anywhere 不在) 使用供應商專屬的工具來完成。例如，要啟動 Amazon EKS 集群，您可以使用`eksctl create cluster`，而對於 Amazon EKS Anywhere，您可以使用。`eksctl anywhere create cluster`請注意，雖然這些命令會建立Kubernetes叢集，但它們是提供者特定的，而不是Kubernetes專案本身的一部分。

叢集建立和管理工具

該Kubernetes項目提供了用於手動創建Kubernetes集群的工具。所以，如果你想Kubernetes在一台機器上安裝, 或運行控制平面上的機器上，並手動添加節點, 您可以使用 CLI 工具，如種, 縮小, 或安裝工具下列出的 kubeadm. Kubernetes 為了簡化和自動化叢集建立和管理的完整生命週期，使用既有Kubernetes供應商 (例如 Amazon EKS 或 Amazon EKS Anywhere) 所支援的工具變得更加容易。

在 AWS 雲端中，您可以使用 CLI 工具 (例如 eksctl) 或更多宣告式工具 (如地形表單) 建立 Amazon EKS 叢集 (請參閱地形表的 Amazon EKS 藍圖)。您也可以從 AWS 管理主控台建立叢集。請參閱 Amazon EKS 功能，了解您使用 Amazon EKS 獲得的列表。 KubernetesAmazon EKS 為您承擔的責任包括：

- 受管控制平面-確保 Amazon EKS 叢集可供 AWS 使用且可擴展，因為它會為您管理控制平面，並使其跨可 AWS 用區域提供。
- 節點管理-您可以讓 Amazon EKS 視需要使用受管節點群組或 Kar penter 自動建立節點，而不是手動新增節點。受管節點群組與Kubernetes叢集自動調度資源整合。使用節點管理工具，您可以利用節省成本的優勢，例如 Spot 執行個體和節點合併以及可用性，並使用排程功能來設定工作負載的部署方式和選取節點。
- 叢集網路-使用 CloudFormation 範本，在Kubernetes叢集中的控制平面和資料平面 (節點) 元件之間`eksctl`設定網路。它還設置可以通過該端點進行內部和外部通信。如需詳細資訊，請參閱 Amazon EKS 工作者節點的消除神秘叢集聯網。Amazon EKS 中的網繭之間的通訊是使用 Amazon EKS Pod 身分來完成的，這可讓網繭利用 AWS 雲端方法來管理登入資料和許可。
- 附加元件-Amazon EKS 讓您不必建置和新增常用於支援Kubernetes叢集的軟體元件。例如，當您從 AWS 管理主控台建立 Amazon EKS 叢集時，它會自動新增 Amazon EKS 庫貝代理、用於的 Amazon VPC CNI 外掛程式以及 CoreDNS 附加元件。Kubernetes如需這些附加元件的詳細資訊，請參閱 Amazon EKS 附加元件，包括可用的附加元件清單。

若要在您自己的現場部署電腦和網路上執行叢集，亞馬遜提供 Amazon EKS 隨處可用。您可以選擇使用自己的設備在 VMware vSphere，裸機（小叮噹提供商），S now 或 Nutanix 平台上運行 Amazon EKS Anywhere CloudStack，而不是 AWS 雲端提供商。

Amazon EKS Anywhere 是基於 Amazon EKS 所使用的相同的 Amazon EKS 發行版軟件。不過，Amazon EKS Anywhere 不在依賴Kubernetes叢集 API (CAPI) 界面的不同實作來管理 Amazon EKS Anywhere 不在叢集中機器的完整生命週期 (例如 vSphere 的 CAPV 和用於的 CAP C)。CloudStack由於整個叢集都在您的設備上執行，因此您負擔管理控制平面及備份其資料的額外責任 (請參閱本文件稍後的 etcd)。

叢集元件

Kubernetes群集組件分為兩個主要區域：控制平面和工作節點。控制平面元件可管理叢集並提供對其 API 的存取權。背景工作者節點 (有時也稱為節點) 提供執行實際工作負載的位置。節點元件包含在每個節點上執行以與控制平面通訊並執行容器的服務。叢集的工作節點集稱為「資料平面」。

控制平台

控制平面由一組管理叢集的服務組成。這些服務可能全部在單一電腦上執行，也可以分散在多部電腦上。在內部，這些例證稱為「控制平面例證」(CPI)。CPI 的執行方式取決於叢集的大小和高可用性的需求。隨著叢集的需求增加，控制平面服務可以擴展以提供更多該服務的執行個體，並在執行個體之間進行負載平衡的要求。

Kubernetes控制平面元件執行的工作包括：

- 與群集組件（API 服務器）通信-API 服務器（kube-apiserver）公開了 Kubernetes API，因此可以從集群內部和外部發出對集群的請求。換句話說，新增或變更叢集物件 (Pod、服務、節點等) 的要求可能來自外部命令，例如執行 Pod 的要求。`kubectl`同樣地，也可以從 API 伺服器向叢集內的元件發出要求，例如查詢`kubelet`服務的 Pod 狀態。
- 儲存叢集的相關資料 (etcd 金鑰值存放區)-etcd 服務提供追蹤叢集目前狀態的關鍵作用。如果 etcd 服務無法存取，您將無法更新或查詢叢集的狀態，儘管工作負載會持續執行一段時間。基於這個原因，重要叢集通常會有多個 etcd 服務的負載平衡執行個體一次執行，並在資料遺失或損毀時定期備份 etcd 金鑰值存放區。請記住，在 Amazon EKS 中，預設情況下這一切都會自動為您處理。Amazon EKS Anywhere 不在提供 etcd 備份和還原的說明。請參閱 etcd 資料模型以瞭解 etcd 如何管理資料。
- 將網繭排程至節點 (排程器)-啟動或停止中網繭的要求Kubernetes會導向至Kubernetes排程器 (kube-Scheduler)。由於叢集可能有多個能夠執行 Pod 的節點，因此排程器必須由排程器選擇 Pod 應該在哪個節點 (或多個節點) 上執行。如果沒有足夠的可用容量無法在現有節點上執行要求的 Pod，除非

您已進行其他規定，否則要求將失敗。這些規定可能包括啟用可自動啟動新節點以處理工作負載的服務，例如受管節點群組或 Karpenter。

- 將元件保持在所需狀態 (Controller Manager)-Kubernetes 控制器管理員會以精靈處理程序 (kube-controller-manager) 的形式執行，以監視叢集的狀態並變更叢集以重新建立預期的狀態。特別是，有幾個控制器可以監視不同的對Kubernetes象，其中包括一個，狀態控制器 node-lifecycle-controller，端點控制器，cronjob-控制器等。
- 管理雲端資源 (Cloud Controller Manager)-執行基礎資料中心資源要求的雲端供應商Kubernetes與雲端供應商之間的互動，會由雲端控制器管理員 (cloud-controller-manager) 處理。由 Cloud Controller Manager 管理的控制器可以包括路由控制器（用於設置雲網絡路由），服務控制器（用於使用雲負載平衡服務）和節點控制器（用於使用雲端 API 保持Kubernetes節點與雲節點同步）。

工作者節點 (資料平面)

對於單一節點Kubernetes叢集，工作負載會在與控制平面相同的機器上執行。不過，較正常的組態是擁有一個或多個專用於執行Kubernetes工作負載的獨立電腦系統 (Nodes)。

當您第一次建立Kubernetes叢集時，某些叢集建立工具可讓您設定要新增至叢集的特定數目節點 (透過識別現有的電腦系統或讓提供者建立新的電腦系統)。在將任何工作負載新增至這些系統之前，服務會新增至每個節點以實作下列功能：

- 管理每個節點 (kubelet)-API 伺服器會與每個節點上執行的 kubelet 服務通訊，以確保節點已正確註冊，且排程器要求的 Pod 正在執行中。kubelet 可以讀取 Pod 資訊清單，並設定儲存磁碟區或本機系統上 Pod 所需的其他功能。它也可以檢查本機執行中容器的健全狀況。
- 在節點上執行容器 (容器執行階段)-每個節點上的容器執行階段會管理指派給節點的每個 Pod 要求的容器。這意味著它可以從適當的註冊表中提取容器映像，運行容器，停止它，並響應有關容器的查詢。預設 containerd 執行階段為容器。從 Kubernetes 1.24 開始，可以用作容器運行時的 Docker（Dockershim）的特殊集成已從中刪除。Kubernetes雖然您仍然可Docker以使用在本機系統上測試和執行容器，但若要Docker搭配Kubernetes使用，您現在必須在每個節點上安裝Docker引擎才能搭配使用Kubernetes。
- 管理容器之間的網路 (kube-proxy)-為了能夠使用服務支援 Pod 之間的通訊，Kubernetes需要一種方法來設定 Pod 網路來追蹤與這些網繭相關聯的 IP 位址和連接埠。kube-proxy 服務會在每個節點上執行，以允許 Pod 之間進行通訊。

擴充叢集

您可以新增一些服務Kubernetes來支援叢集，但不會在控制平面中執行。這些服務通常直接在 kube-system 命名空間中的節點上或其自己的命名空間中運行（通常與第三方服務提供者一樣）。一個常見

的例子是 CoreDNS 服務，它為叢集提供 DNS 服務。如需如何查看叢集上的 kube-system 中執行哪些叢集服務的詳細資訊，請參閱探查內建服務。

您可以考慮將不同類型的附加元件新增至叢集。為了保持叢集的健康狀態，您可以新增可觀察性功能，讓您執行記錄、稽核和指標等操作。有了這些資訊，您就可以疑難排解發生的問題，通常是透過相同的可觀察性介面。這些類型的服務的例子包括 Amazon GuardDuty, CloudWatch, AWS 發行版 OpenTelemetry, Amazon VPC CNI 插件Kubernetes, 和 Grafana 監控. Kubernetes 對於儲存，Amazon EKS 的附加元件包括 Amazon 彈性區塊存放區 CSI 驅動程式 (用於新增區塊儲存裝置)、Amazon Elastic File System CSI 驅動程式 (用於新增檔案系統儲存) 和數個第三方儲存附加元件 (例如用於 NetApp ONTAP CSI 驅動程式的 Amazon FSx)。

有關可用 Amazon EKS 附加組件的更完整列表，請參閱 Amazon EKS 附加組件。

工作負載

Kubernetes將工作負載定義為「在上執行的應用程式」Kubernetes。該應用程式可以包含一組以 Pod 中的容器形式執行的微服務，也可以作為批次工作或其他類型的應用程式執行。的工作Kubernetes是確定已執行您對要設定或部署的物件所發出的要求。在部署應用程式時，您應該瞭解如何建置容器、如何定義 Pod，以及可以使用哪些方法來部署它們。

容器

您在中部署和管理的應用程式工作負載中最基本的元素Kubernetes是 Pod。Pod 代表保存應用程式元件以及定義描述 Pod 屬性之規格的方式。請與 RPM 或 Deb 套件做比較，這些套件會將 Linux 系統的軟體封裝在一起，但本身並不會當做實體執行。

由於 Pod 是最小的可部署單位，因此它通常會保留單一容器。但是，在容器緊密耦合的情況下，多個容器可以位於 Pod 中。例如，Web 伺服器容器可能會封裝在具有附屬容器類型的 Pod 中，該容器可能會提供記錄、監視或其他與 Web 伺服器容器緊密相關的服務。在此情況下，位於同一個 Pod 中，可確保對於 Pod 的每個執行個體，兩個容器一律在同一個節點上執行。同樣地，Pod 中的所有容器共用相同的環境，Pod 中的容器在執行時，就像它們位於相同的隔離主機中一樣。這樣做的效果是容器共用單一 IP 位址，該 IP 位址可提供對 Pod 的存取，而且容器可以彼此通訊，就像它們在自己的本機主機上執行一樣。

網繭規格 (PodSpec) 定義所需的網繭狀態。您可以使用工作負載資源管理網繭範本來部署個別網繭或多個網繭。工作負載資源包括部署 (用於管理多個網繭複本)、StatefulSets(用於部署需要唯一的網繭，例如資料庫網繭)，以及 DaemonSets(網繭需要在每個節點上持續執行)。更多關於那些以後。

雖然 Pod 是您部署的最小單位，但容器是您建置和管理的最小單位。

建築容器

Pod 實際上只是圍繞一個或多個容器的結構，每個容器本身都保存文件系統，可執行文件，配置文件，庫和其他組件來實際運行應用程序。因為一家名為 Docker Inc. 的公司首先普及的容器，有些人稱容器為Docker容器。然而，「開放容器計劃」已經定義了產業的容器執行階段、映像和散發方法。除此之外，容器是從許多現有 Linux 功能創建的，其他功能通常將容器稱為 OCI 容器，Linux 容器或僅容器。

當您建置容器時，通常會從Docker檔案開始 (字面上命名為該檔案)。在該碼頭文件中，您確定：

- 基本映像檔-基本容器映像檔是一種容器，通常是以最小版本的作業系統檔案系統 (例如 Red Hat Enterprise Linux 或 Ubuntu) 所建立，或是經過強化的最小系統，可提供軟體來執行特定類型的應用程式 (例如 nodejs 或 python 應用程式)。
- 應用程式軟體-您可以將應用程式軟體新增至容器，方式與將其新增至 Linux 系統的方式相同。例如，在您的 Docker 文件中，您可以運行`npm`並`yarn`安裝 Java 應用程序或`yum`安裝 RPM 軟件包。`dnf`換句話說，在 Dockerfile 中使用 RUN 命令，您可以運行基本映像文件系統中可用的任何命令，以在生成的容器映像內安裝軟件或配置軟件。
- 指示-Dockerfile 參考描述了您在配置文件時可以添加到 Docker 文件中的說明。這些指令包括用來建置容器本身中的內容 (`ADD`或來自本機系統的`COPY`檔案)、識別容器執行時要執行的命令 (`CMD`或`ENTRYPOINT`)，以及將容器連接至其執行的系統 (藉由識別`USER`要執行的身分、`VOLUME`要掛載的本機或連接埠`EXPOSE`)。

雖然命`docker`令和服務傳統上是用來建立容器 (`docker build`)，但其他可用來建立容器映像檔的工具包括 podman 和 ner dctl。請參閱建置更好的容器映像檔或使用建置Docker來瞭解如何建置容器。

儲存容器

建立容器映像之後，您可以將其儲存在工作站的容器散發登錄中，或是公用容器登錄中。在工作站上執行私人容器登錄可讓您將容器映像儲存在本機，讓您隨時可用。

若要以更公開的方式儲存容器映像檔，您可以將它們推送至公用容器登錄。公用容器登錄提供儲存和散發容器映像的中央位置。公用容器登錄的範例包括 Amazon 彈性容器登錄、Red Hat Quay 登錄和Docker集線器登錄。

在 Amazon 彈性 Kubernetes 服務 (Amazon EKS) 上執行容器化工作負載時，我們建議您提取存放在 Amazon 彈性容器登錄中的Docker官方映像副本。 AWS 自 2021 年以來，Amazon ECR 一直在存儲這些圖像。您可以在 Amazon ECR 公共圖庫中搜索流行的容器映像，特別是對於Docker集線器映像，您可以搜索 Amazon ECR Docker 圖庫。

執行容器

由於容器是以標準格式建置，因此容器可以在任何可執行容器執行階段 (例如Docker) 且其內容符合本機電腦架構 (例如x86_64或arm) 的機器上執行。要測試容器或僅在本地桌面上運行它，您可以使用docker run或podman run命令在本地主機上啟動容器。但是Kubernetes，對於每個 Worker 節點都已部署容器執行階段，並且由Kubernetes要求節點執行容器。

指派要在節點上執行的容器之後，節點會查看要求的容器映像版本是否已存在於節點上。如果沒有，請Kubernetes告知容器執行階段從適當的容器登錄中提取該容器，然後在本機執行該容器。請記住，容器映像檔是指在筆記型電腦、容器登錄和Kubernetes節點之間移動的軟體套件。容器是指該映像檔的執行中執行個體。

豆莢

容器準備就緒後，使用 Pod 就會包括設定、部署和使 Pod 可供存取。

設定網繭

定義網繭時，您可以為其指派一組屬性。這些屬性至少必須包含要執行的 Pod 名稱和容器映像。但是，您還需要使用 Pod 定義配置許多其他內容（有關可以進入 Pod 的詳細信息，請參閱PodSpec頁面）。其中包含：

- 儲存-當執行中的容器停止並刪除時，除非您設定更多永久儲存空間，否則該容器中的資料儲存將會消失。 Kubernetes支持許多不同的存儲類型，並在 Volumes 的保護傘下對它們進行抽象化。儲存區類型包括虛擬主機、NFS、iSCSI 及其他儲存裝置。您甚至可以使用本地計算機上的本地塊設備。透過叢集提供的其中一種儲存體類型，您就可以將儲存磁碟區掛載到容器檔案系統中選取的掛載點。持續性磁碟區是在刪除網繭後繼續存在的磁碟區，而暫時磁碟區會在刪除網繭時刪除。如果您的叢集管理員StorageClasses為叢集建立了不同的選項，您可以選擇所使用的儲存屬性，例如磁碟區是否在使用後刪除還是回收磁碟區、在需要更多空間時是否會擴充磁碟區，以及是否符合特定的效能需求。
- 秘密-透過將 Secret 提供給 Pod 規格中的容器，您可以提供這些容器存取檔案系統、資料庫或其他受保護資產所需的權限。密鑰，密碼和令牌是可以存儲為秘密的項目之一。使用秘密可以讓您不必將此資訊儲存在容器映像檔中，而只需要將密碼提供給執行容器。類似於秘密ConfigMaps。A ConfigMap 傾向於保存較不重要的信息，例如用於配置服務的鍵值對。
- 容器資源-用於進一步配置容器的對象可以採用資源配置的形式。對於每個容器，您可以請求其可以使用的內存和 CPU 數量，以及容器可以使用的這些資源的總數限制。如需範例，請參閱網繭和容器的資源管理。
- 中斷-Pod 可能會非自願中斷 (節點當機) 或自願中斷 (需要升級)。透過設定 Pod 中斷預算，您可以對發生中斷時應用程式保留的可用程式進行一些控制。如需範例，請參閱指定應用程式的中斷預算。

- 命名空間-Kubernetes 提供不同的方法來隔離Kubernetes元件和工作負載彼此。針對相同命名空間中的特定應用程式執行所有 Pod，是一起保護和管理這些 Pod 的常用方式。您可以創建自己的命名空間來使用或選擇不指示命名空間（這會導Kubernetes致使用`default`命名空間）。 Kubernetes控制平面元件通常在 kube- system 命名空間中執行。

剛才描述的組態通常會收集在 YAML 檔案中，以套用至Kubernetes叢集。對於個人Kubernetes叢集，您可能只是將這些 YAML 檔案儲存在本機系統上。但是，對於更重要的叢集和工作負載，GitOps是一種自動化工作負載和Kubernetes基礎架構資源的儲存和更新的熱門方式。

用來收集和部署 Pod 資訊的物件是由下列其中一種部署方法所定義。

部署 Pod

您選擇用於部署 Pod 的方法取決於您打算使用這些 Pod 執行的應用程式類型。以下是您的一些選擇：

- 無狀態應用程序-無狀態應用程序不會保存客戶端的會話數據，因此另一個會話不需要引用以前的會話發生的事情。如果 Pod 變得不健康，或者在不保存狀態的情況下四處移動它們，則可以更輕鬆地用新的 Pod 替換。如果您正在執行無狀態應用程式 (例如 Web 伺服器)，則可以使用部署來部署 Pod 和 ReplicaSets. A ReplicaSet 定義您要同時執行的 Pod 執行個體數目。雖然您可以 ReplicaSet 直接執行，但通常是直接在部署中執行複本，以定義一次應該執行多少 Pod 複本。
- 可設定狀態的應用程式-可設定狀態的應用程式是 Pod 的身分識別以及 Pod 啟動的順序很重要的應用程式。這些應用程式需要穩定且需要以一致的方式部署和擴充的持續性儲存。若要在中部署可設定狀態應用程式Kubernetes，您可以使用 StatefulSets. 通常以資料庫形式執行的應用 StatefulSet 程式範例。在 a 中 StatefulSet，您可以定義複本、Pod 及其容器、要掛接的儲存磁碟區，以及儲存資料的容器中的位置。如需部署為資料庫的範例，請參閱執行複寫的可設定狀態應用程式 ReplicaSet。
- 每節點應用程式-有時您會想要在Kubernetes叢集中的每個節點上執行應用程式。例如，您的資料中心可能需要每台電腦執行監視應用程式或特定的遠端存取服務。對於Kubernetes，您可以使用 a DaemonSet來確保選取的應用程式在叢集中的每個節點上執行。
- 應用程序運行到完成-有一些應用程序要運行來完成特定任務。這可能包括執行每月狀態報告或清除舊資料的報告。Job 物件可用來設定應用程式以啟動和執行，然後在工作完成時結束。CronJob物件可讓您使用 Linux crontab 格式定義的結構，將應用程式設定為在特定的小時、分鐘、月份、月份或星期幾執行。

讓應用程式可從網路存取

由於應用程式通常部署為一組微服務，並移動到不同的地方，因此Kubernetes需要一種方法讓這些微服務能夠彼此找到。此外，若要讓其他人存取Kubernetes叢集外部的應用程式，則Kubernetes需要在

外部位址和連接埠上公開該應用程式的方法。這些與網路相關的功能分別是透過 Service 和 Ingress 物件完成的：

- 服務-由於 Pod 可以四處移動到不同的節點和位址，因此需要與第一個 Pod 通訊的另一個 Pod 可能會發現很難找到它所在的位置。若要解決這個問題，Kubernetes可讓您將應用程式表示為服務。使用服務，您可以使用特定名稱識別網繭或一組網繭，然後指出從 Pod 公開該應用程式服務的連接埠，以及其他應用程式可用來連絡該服務的連接埠。叢集中的另一個 Pod 可以直接依名稱要求服務，並Kubernetes將該要求導向至執行該服務之 Pod 執行個體的適當連接埠。
- Ingress-Ingress 可以讓「Kubernetes服務」代表的應用程式提供給叢集外部的用戶端使用。Ingress 的基本功能包括負載平衡器 (由 Ingress 管理)、Ingress 控制器，以及將要求從控制器路由到服務的規則。您可以選擇多種入口控制器。Kubernetes

後續步驟

了解基本Kubernetes概念以及它們與 Amazon EKS 的關係將協助您瀏覽 Amazon EKS 文件和Kubernetes文件，以尋找管理 Amazon EKS 叢集和將工作負載部署到這些叢集所需的資訊。要開始使用 Amazon EKS，請從以下選項中進行選擇：

- 建立簡單叢集
- 建立更複雜的叢集
- 部署範例應用程式
- 探索管理叢集的方式

部署選項

您可以使用下列任一選項來部署 Amazon EKS：

雲端 Amazon EKS

> 您可以在 AWS 雲端中執行 Kubernetes，無需安裝、操作和維護自己的 Kubernetes 控制平面或節點。此選項是本指南中涵蓋的內容。

Amazon EKS on Outposts

> AWS Outposts 會在您的內部部署設施中啟用原生 AWS 服務、基礎設施和操作模型。在 Outpost 上使用 Amazon EKS 時，您可以選擇執行擴充叢集或本機叢集。使用擴充叢集時，Kubernetes 控制平面會在 AWS 區域 中執行，且節點會在 Outpost 上執行。使用本機叢集時，整個 Kubernetes 叢集會在 Outpost 上本機執行，包括 Kubernetes 控制平面和節點。如需更多詳細資訊，請參閱 AWS Outposts 上的 Amazon EKS。

Amazon EKS Anywhere

Amazon EKS Anywhere 是 Amazon EKS 的部署選項，可讓您輕鬆地在內部部署建立和操作 Kubernetes 叢集。Amazon EKS 和 Amazon EKS Anywhere 均建置在 Amazon EKS Distro 上。若要進一步了解有關 Amazon EKS Anywhere 以及其與 Amazon EKS 之間的差異，請參閱 Amazon EKS Anywhere 文件中的概觀以及比較 Amazon EKS Anywhere 和 Amazon EKS。如需一些常見問題的答案，請參閱 Amazon EKS Anywhere 常見問答集。

Amazon EKS Distro

Amazon EKS Distro 是由雲端 Amazon EKS 部署的相同開放原始碼 Kubernetes 軟體和相依性的發行版本。Amazon EKS Distro 遵循與 Amazon EKS 相同的 Kubernetes 版本發行週期，並以開放原始碼專案提供。如需進一步了解，請參閱 Amazon EKS Distro。您也可以在 GitHub 上檢視和下載 Amazon EKS Distro 的原始程式碼。

在選擇用於 Kubernetes 叢集的部署選項時，請考慮以下事項：

功能	Amazon EKS	Amazon EKS on Outposts	Amazon EKS Anywhere	Amazon EKS Distro
硬體	由 AWS 提供	由 AWS 提供	由您提供	由您提供
部署位置	AWS 雲端	您的資料中心	您的資料中心	您的資料中心
Kubernetes 控制平面位置	AWS 雲端	AWS 雲端或您的資料中心	您的資料中心	您的資料中心
Kubernetes 資料平面位置	AWS 雲端	您的資料中心	您的資料中心	您的資料中心
支援	AWS Support	AWS Support	AWS Support	OSS 社群支援

設定以使用 Amazon EKS。

AWS 資源通常具有存取限制，限制對建立其之 AWS 實體的存取。因此，從一開始就在 AWS Command Line Interface 中建立正確的使用者組態是至關重要的。此外，您需要為本機電腦配備必要的工具，以便能夠使用命令列高效管理 Amazon EKS 叢集。本主題將協助您為使用命令列管理叢集做好準備。

步驟 1：設定 AWS CLI

AWS CLI 命令列工具用於管理 AWS 服務，包括 Amazon EKS。其也用於從本機電腦對 IAM 使用者或角色進行身分驗證，以便存取 Amazon EKS 叢集和其他 AWS 資源。若要從命令列佈建 AWS 中的資源，您需要取得要在命令列中使用的 AWS 存取金鑰 ID 和私密金鑰。然後您需要在 AWS CLI 中對這些憑證進行設定。若您尚未安裝 AWS CLI，請參閱《AWS Command Line Interface 使用者指南》中的安裝或更新至 AWS CLI 的最新版本 一節。

建立存取金鑰

1. 登入 AWS Management Console。
2. 在右上角，選擇 AWS 使用者名稱，以開啟導覽選單。例如，選擇 **webadmin**。然後選擇安全憑證。
3. 在存取金鑰之下選擇建立存取金鑰。
4. 選擇命令列介面 (CLI)，然後選擇下一步。
5. 選擇 Create access key (建立新的存取金鑰)。
6. 選擇下載 .csv 檔案。

設定AWS CLI

安裝 AWS CLI 之後，請遵循這些步驟安裝以完成設定：如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的設定 AWS CLI。

1. 在終端機視窗中，輸入以下命令：

   ```
   aws configure
   ```

   您也可以選擇設定具名設定檔，例如 **--profile cluster-admin**。如果您在 AWS CLI 中設定了具名設定檔，則必須永遠在後續命令中傳遞此旗標。

2. 輸入您的 AWS 憑證。例如：

```
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: region-code
Default output format [None]: json
```

取得安全字符

如有需要，請執行以下命令以取得 AWS CLI 的新安全字符。如需詳細資訊，請參閱 AWS CLI 命令參考中的 get-session-token。

依預設，該字符的有效期為 15 分鐘。若要變更預設工作階段逾時，請傳遞 **--duration-seconds** 旗標。例如：

```
aws sts get-session-token --duration-seconds 3600
```

此命令會傳回 AWS CLI 工作階段的暫時安全憑證。您應該會看到下列回應輸出：

```
{
    "Credentials": {
        "AccessKeyId": "ASIA5FTRU3LOEXAMPLE",
        "SecretAccessKey": "JnKgvwfqUD9mNsPoi9IbxAYEXAMPLE",
        "SessionToken": "VERYLONGSESSIONTOKENSTRING",
        "Expiration": "2023-02-17T03:14:24+00:00"
    }
}
```

若要驗證使用者身分

如有需要，請執行以下命令來驗證終端工作階段的 IAM 使用者身分 (例如 *ClusterAdmin*) 的 AWS 憑證。

```
aws sts get-caller-identity
```

此命令會傳回為 AWS CLI 設定的 IAM 實體的 Amazon Resource Name (ARN)。您應該會看到類似以下內容的回應輸出：

```
{
```

```
    "UserId": "AKIAIOSFODNN7EXAMPLE",
    "Account": "01234567890",
    "Arn": "arn:aws:iam::01234567890:user/ClusterAdmin"
}
```

步驟 2：安裝 Kubernetes 工具

若要與 Kubernetes 叢集進行通訊，您需要一個與 Kubernetes API 互動的工具。此外，您還需要一些其他工具，例如用於在本機電腦上管理 Kubernetes 環境的工具。

建立 AWS 資源

- Amazon EKS 叢集資源：如果您不熟悉 AWS，則建議您安裝 `eksctl`。`eksctl` 是一種基礎設施即程式碼 (IaC) 公用程式，它讓您能夠使用 AWS CloudFormation 輕鬆建立 Amazon EKS 叢集。它還可用於建立其他 Kubernetes 資源，例如服務帳戶。如需有關安裝 `eksctl` 的指示，請參閱 `eksctl` 文件中的 Installation 一節。
- AWS 資源：如果您習慣於自動佈建和部署 AWS 基礎設施，則建議您安裝 Terraform。Terraform 是 HashiCorp 開發的一種開放原始碼基礎設施即程式碼 (IaC) 工具。它讓您可以使用高階佈建語言 (例如 HashiCorp Configuration Language (HCL) 或 JSON) 定義和佈建基礎設施。如需有關安裝 Terraform 的指示，請參閱 Terraform 文件中的 Install Terraform 一節。

安裝 `kubectl`

`kubectl` 是開放原始碼命令列工具，用於與 Amazon EKS 叢集上的 Kubernetes API 伺服器進行通訊。如果您尚未在本機上安裝它，請選擇下列選項之一。

- AWS 版本：若要安裝 Amazon EKS 支援的 `kubectl` 版本，請參閱 安裝或更新 `kubectl`。
- 社群版本：若要安裝 `kubectl` 的最新社群版本，請參閱 Kubernetes 文件中的 Install tools 頁面。

設定開發環境

- 本機部署工具：如果您不熟悉 Kubernetes，請考慮安裝本機部署工具，例如 `minikube` 或 `kind`。這些工具可讓您在本機電腦上管理 Amazon EKS 叢集。
- 套件管理器：Helm 是適用於 Kubernetes 的熱門套件管理器，可用於簡化複雜套件的安裝和管理。藉助 Helm，您可以更輕鬆地在 Amazon EKS 叢集上安裝和管理 AWS Load Balancer Controller 等套件。

後續步驟

- Amazon EKS 入門

安裝或更新 kubectl

Kubectl 是一種命令列工具，可用來與 Kubernetes API 伺服器進行通訊。kubectl 二進位檔案可在眾多作業系統套件軟體管理工具中使用。在安裝時使用套件管理工具，時常會比手動下載與安裝程序更加簡便。

本主題將協助您在裝置上下載並安裝或更新 kubectl 二進位檔案。二進位檔案與上游社群版本相同。二進製文件不是唯一的 Amazon EKS 或 AWS.

> **Note**
>
> 您所使用的 kubectl 版本，必須與 Amazon EKS 叢集控制平面的版本差距在一個版本以內。例如，1.29 kubectl 用戶端搭配 Kubernetes、1.28、1.29 和 1.30 叢集運作。

安裝或更新 kubectl

1. 判斷您是否已在裝置上安裝 kubectl。

   ```
   kubectl version --client
   ```

   若您已在裝置路徑中安裝 kubectl，則範例輸出包含下列類似資訊。若您要更新目前使用較新版本安裝的版本，請完成下一個步驟，確定將新版本安裝在目前版本所處的位置。

   ```
   Client Version: v1.30.X-eks-1234567
   ```

   若您未收到輸出，表示您尚未安裝 kubectl，或其未安裝在裝置路徑中的位置。
2. 在 macOS、Linux 和 Windows 作業系統上安裝或更新 kubectl。

   macOS

   #### 在 macOS 上安裝或更新 kubectl

   1. 從 Amazon S3 下載適用您叢集 Kubernetes 版本的二進位檔案。

- Kubernetes 1.30

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/darwin/amd64/kubectl
```

- Kubernetes 1.29

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.28

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.27

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.26

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.25

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.23

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.22

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes 1.21

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/darwin/amd64/kubectl
```

2. (選用) 使用您的二進位檔案 SHA-256 檢查總和，驗證下載的二進位檔案。

   a. 下載適用於您叢集 Kubernetes 版本的 SHA-256 檢查總和。

   - Kubernetes 1.30

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/darwin/amd64/kubectl.sha256
   ```

   - Kubernetes 1.29

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/darwin/amd64/kubectl.sha256
   ```

   - Kubernetes 1.28

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/darwin/amd64/kubectl.sha256
   ```

   - Kubernetes 1.27

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/darwin/amd64/kubectl.sha256
   ```

   - Kubernetes 1.26

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/darwin/amd64/kubectl.sha256
   ```

   - Kubernetes 1.25

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/darwin/amd64/kubectl.sha256
   ```

- Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.23

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.22

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.21

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

b. 檢查適用您下載之二進位檔案的 SHA-256 檢查總和。

```
openssl sha1 -sha256 kubectl
```

c. 確保輸出中產生的檢查總和與下載 kubectl.sha256 檔案中的檢查總和相符。

3. 將執行許可套用至二進位檔。

```
chmod +x ./kubectl
```

4. 將二進位檔複製到 PATH 中的資料夾。如果您已安裝某一版本的 kubectl，建議您建立 $HOME/bin/kubectl 並確認您的 $PATH 中會先出現 $HOME/bin。

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (選用) 將 $HOME/bin 路徑新增到 Shell 初始化檔案，因此當您開啟 shell 時，該組態已設定完畢。

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bash_profile
```

Linux (amd64)

在 Linux (**amd64**) 上安裝或更新 **kubectl**

1. 從 Amazon S3 下載適用您叢集 Kubernetes 版本的 `kubectl` 二進位檔案。
   - Kubernetes `1.30`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.30`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.29`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.28`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.27`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.26`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.25`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/amd64/kubectl
     ```

- Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes 1.23

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes 1.22

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes 1.21

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/amd64/kubectl
```

2. (選用) 使用您的二進位檔案 SHA-256 檢查總和，驗證下載的二進位檔案。

   a. 使用適用於您裝置硬體平台的命令，從 Amazon S3下載 Amazon EKS 提供之適用您叢集 Kubernetes 版本的 SHA-256 檢查總和。每個版本的第一個連結適用於 amd64，第二個連結則適用於 arm64。

   - Kubernetes 1.30

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl.sha256
   ```

   - Kubernetes 1.30

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl.sha256
   ```

   - Kubernetes 1.29

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/amd64/kubectl.sha256
   ```

   - Kubernetes 1.28

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes 1.27

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes 1.26

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes 1.25

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes 1.23

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes 1.22

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes 1.21

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/amd64/kubectl.sha256
```

b. 執行下列其中一個命令，檢查您下載的二進製檔案的 SHA-256 檢查總和。

-

```
sha256sum -c kubectl.sha256
```

使用此命令時，請確保看到下列輸出：

```
kubectl: OK
```

- ```
  openssl sha1 -sha256 kubectl
  ```

  使用此命令時，確保輸出中產生的檢查總和與下載 `kubectl.sha256` 檔案中的檢查總和相符。

3. 將執行許可套用至二進位檔。

```
chmod +x ./kubectl
```

4. 將二進位檔複製到 PATH 中的資料夾。如果您已安裝某一版本的 `kubectl`，建議您建立 `$HOME/bin/kubectl` 並確認您的 `$PATH` 中會先出現 `$HOME/bin`。

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (選用) 將 `$HOME/bin` 路徑新增到 Shell 初始化檔案，因此當您開啟 shell 時，該組態已設定完畢。

> **Note**
>
> 本步驟假設您是使用 Bash Shell；若您使用其他 Shell，請將命令更改為使用具體的 Shell 初始化檔案。

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
```

Linux (arm64)

在 Linux (**arm64**) 上安裝或更新 **kubectl**

1. 從 Amazon S3 下載適用您叢集 Kubernetes 版本的 `kubectl` 二進位檔案。
   - Kubernetes `1.30`
```

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/
bin/linux/arm64/kubectl
```

- Kubernetes 1.22

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/
bin/linux/arm64/kubectl
```

- Kubernetes 1.21

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/
bin/linux/arm64/kubectl
```

2. (選用) 使用您的二進位檔案 SHA-256 檢查總和，驗證下載的二進位檔案。

   a. 使用適用於您裝置硬體平台的命令，從 Amazon S3下載 Amazon EKS 提供之適用您叢集 Kubernetes 版本的 SHA-256 檢查總和。每個版本的第一個連結適用於 amd64，第二個連結則適用於 arm64。

   - Kubernetes 1.30

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/
   bin/linux/arm64/kubectl.sha256
   ```

   - Kubernetes 1.30

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/
   bin/linux/arm64/kubectl.sha256
   ```

   - Kubernetes 1.29

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/
   bin/linux/arm64/kubectl.sha256
   ```

   - Kubernetes 1.28

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/
   bin/linux/arm64/kubectl.sha256
   ```

   - Kubernetes 1.27

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes 1.26

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes 1.25

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes 1.23

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes 1.22

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes 1.21

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/arm64/kubectl.sha256
```

b. 執行下列其中一個命令，檢查您下載的二進製檔案的 SHA-256 檢查總和。

- ```
  sha256sum -c kubectl.sha256
  ```

  使用此命令時，請確保看到下列輸出：

  ```
  kubectl: OK
  ```
  ```

- ```
  openssl sha1 -sha256 kubectl
  ```

  使用此命令時，確保輸出中產生的檢查總和與下載 kubectl.sha256 檔案中的檢查總和相符。

3. 將執行許可套用至二進位檔。

   ```
   chmod +x ./kubectl
   ```

4. 將二進位檔複製到 PATH 中的資料夾。如果您已安裝某一版本的 kubectl，建議您建立 $HOME/bin/kubectl 並確認您的 $PATH 中會先出現 $HOME/bin。

   ```
   mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
   ```

5. (選用) 將 $HOME/bin 路徑新增到 Shell 初始化檔案，因此當您開啟 shell 時，該組態已設定完畢。

   > **Note**
   >
   > 本步驟假設您是使用 Bash Shell；若您使用其他 Shell，請將命令更改為使用具體的 Shell 初始化檔案。

   ```
   echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
   ```

Windows

在 Windows 上安裝或更新 **kubectl**

1. 開啟 PowerShell 終端機。
2. 從 Amazon S3 下載適用您叢集 Kubernetes 版本的 kubectl 二進位檔案。
   - Kubernetes 1.30

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes 1.29
     ```

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/
bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.28

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/
bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.27

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.27.12/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.26

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.26.15/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.25

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.25.16/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.24

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.24.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.23

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.23.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.22

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.22.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes 1.21

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.21.14/2024-04-19/bin/windows/amd64/kubectl.exe
```

3. (選用) 使用您的二進位檔案 SHA-256 檢查總和，驗證下載的二進位檔案。

   a. 若您使用 Windows，請下載適用於您叢集 Kubernetes 版本的 SHA-256 檢查總和。

   - Kubernetes 1.30

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
     eks/1.30.0/2024-05-12/bin/windows/amd64/kubectl.exe.sha256
     ```

   - Kubernetes 1.29

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
     eks/1.29.3/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
     ```

   - Kubernetes 1.28

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
     eks/1.28.8/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
     ```

   - Kubernetes 1.27

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
     eks/1.27.12/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
     ```

   - Kubernetes 1.26

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
     eks/1.26.15/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
     ```

   - Kubernetes 1.25

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
     eks/1.25.16/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
     ```

   - Kubernetes 1.24

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
     eks/1.24.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
     ```

- Kubernetes 1.23

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.23.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes 1.22

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.22.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes 1.21

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.21.14/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

b. 檢查適用您下載之二進位檔案的 SHA-256 檢查總和。

```
Get-FileHash kubectl.exe
```

c. 確保輸出中產生的檢查總和與下載 kubectl.sha256 檔案中的檢查總和相符。PowerShell輸出應該是字符的大寫等效字符串。

4. 將二進位檔複製到 PATH 中的資料夾。若在 PATH 中已有命令列公用程式專用的現存目錄，請將二進位檔案複製至該目錄。否則，請完成下列步驟。

a. 建立新目錄以供存放命令列二進位檔，例如 C:\bin。

b. 將 kubectl.exe 二進位檔複製到該新目錄。

c. 編輯您的使用者或系統 PATH 環境變數，將新目錄新增至 PATH。

d. 關閉 PowerShell 終端機再重新開啟，以挑選新的 PATH 變數。

3. 安裝 kubectl 後，您可以驗證其版本。

```
kubectl version --client
```

第一次安裝 kubectl 時，它並未設定可與任何伺服器通訊。我們會視需要在其他程序中涵蓋此組態。如果您需要更新組態來與特定叢集進行通訊，那麼您可以執行下列命令。*region-code*以叢集所 AWS 區域 在的位置取代。使用您叢集的名稱取代 *my-cluster*。

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

Amazon EKS 入門

在閱讀此入門指南之前，請確保您已完成 Amazon EKS 的設定並可開始使用。如需詳細資訊，請參閱設定以使用 Amazon EKS。。

對於在 Amazon EKS 中建立含節點的新 Kubernetes 叢集，我們提供兩種入門指南：

- Amazon EKS 入門：eksctl：這個入門指南可協助您安裝在將 Amazon EKS 和 eksctl 搭配使用所需的所有資源，後者是簡單的命令列公用程式，可用於在 Amazon EKS 上建立和管理 Kubernetes 叢集。教學結束時，您將擁有一個執行中的 Amazon EKS 叢集，而您可以在其中部署應用程式。這是 Amazon EKS 入門的最快且最簡易的方式。
- 開始使用 Amazon EKS — AWS Management Console 和 AWS CLI— 本入門指南可協助您建立所有必要的資源，以便使用 AWS Management Console 和 AWS CLI開始使用 Amazon EKS。教學結束時，您將擁有一個執行中的 Amazon EKS 叢集，而您可以在其中部署應用程式。在本指南中，您可以手動建立 Amazon EKS 叢集所需的各個資源。這些程序可讓您完全了解每個資源的建立方式，以及它們彼此之間的互動方式。

我們還提供以下參考資料：

- 如需精選的實作教學課程集合，請參閱在AWS 社群上導覽 Amazon EKS。
- 如需程式碼範例，請參閱使用 AWS 開發套件的 Amazon EKS 程式碼範例。

Amazon EKS 入門：eksctl

本指南可協助您安裝在將 Amazon Elastic Kubernetes Service (Amazon EKS) 和 eksctl 搭配使用所需的所有資源，後者是簡單的命令列公用程式，可用於在 Amazon EKS 上建立和管理 Kubernetes 叢集。此教學結束時，您將擁有一個執行中的 Amazon EKS 叢集，而您可以在其中部署應用程式。

本指南中的程序會為您自動建立數個資源，而您必須在使用 AWS Management Console建立叢集時手動建立這些資源。如果您想要手動建立大部分的資源，以便更好地瞭解它們彼此之間的互動方式，請使用 AWS Management Console 來建立叢集和計算。如需詳細資訊，請參閱 開始使用 Amazon EKS — AWS Management Console 和 AWS CLI。

必要條件

開始此教學之前，您必須先安裝和設定下列在建立和管理 Amazon EKS 叢集所需的工具和資源。

- **kubectl**：命令列工具，適用於使用 Kubernetes 叢集。如需詳細資訊，請參閱 安裝或更新 kubectl。
- **eksctl**：命令列工具，適用於使用 EKS 叢集，可自動執行許多個別任務。如需詳細資訊，請參閱 eksctl 文件中的 Installation 一節。
- 必要的 IAM 許可 — 您使用的 IAM 安全主體必須具有使用 Amazon EKS IAM 角色、服務連結角色 AWS CloudFormation、VPC 和相關資源的許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 Amazon Elastic Container Service for Kubernetes 的動作、資源和條件索引鍵和使用服務連結角色。您必須以同一位使用者的身分完成本指南中的所有步驟。若要檢查目前使用者，請執行以下命令：

```
aws sts get-caller-identity
```

步驟 1：建立 Amazon EKS 叢集和節點

> **Important**
>
> 為了盡可能簡單快速地開始使用，本主題包含使用預設設定建立叢集和節點的步驟。建立叢集和節點供生產使用之前，推薦您先熟悉所有設定，並使用符合您需求的設定來部署叢集和節點。如需詳細資訊，請參閱 建立 Amazon EKS 叢集 及 Amazon EKS 節點。只有在建立叢集和節點時，才能夠啟用某些設定。

您可以使用下列其中一種節點類型來建立叢集。若要進一步了解每種類型，請參閱 Amazon EKS 節點。叢集部署完成後，您可以新增其他節點類型。

- Fargate – Linux：如果您想要在 AWS Fargate 上執行 Linux 應用程式，請選取此節點類型。Fargate 是無伺服器運算引擎，可讓您部署 Kubernetes Pods 而無須管理 Amazon EC2 執行個體。
- 受管節點 – Linux：如果您想要在 Amazon EC2 執行個體上執行 Amazon Linux 應用程式，請選取此節點類型。雖然未涵蓋在本指南中，但您也可以將 Windows 自我管理節點和 Bottlerocket 節點新增至您的叢集。

透過以下命令建立 Amazon EKS 叢集。您可以使用自己的值取代 *my-cluster*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。替換為 Amazon EKS 支持*region-*

*code*的任何 AWS 區域 產品。如需清單 AWS 區域，請參閱 AWS 一般參考指南中的 Amazon EKS 端點和配額。

Fargate – Linux

```
eksctl create cluster --name my-cluster --region region-code --fargate
```

Managed nodes – Linux

```
eksctl create cluster --name my-cluster --region region-code
```

叢集建立需要幾分鐘的時間。在建立期間，您會看到數行輸出。輸出的最後一行類似於下面的範例行。

```
[...]
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

eksctl 已在 ~/.kube 建立了 kubectl config 檔案，或將新叢集的組態新增至電腦上 ~/.kube 中的現有 config 檔案。

叢集建立完成後，請檢視 AWS CloudFormation 主控台eksctl-*my-cluster*-cluster中名為的 AWS CloudFormation 堆疊 https://console.aws.amazon.com/cloudformation，以查看已建立的所有資源。

步驟 2：檢視 Kubernetes 資源

1. 檢視叢集節點。

   ```
   kubectl get nodes -o wide
   ```

   範例輸出如下。

   Fargate – Linux

   ```
   NAME                                                STATUS   ROLES    AGE
      VERSION              INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-
   VERSION                   CONTAINER-RUNTIME
   fargate-ip-192-0-2-0.region-code.compute.internal   Ready    <none>
     8m3s    v1.2.3-eks-1234567   192.0.2.0     <none>        Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   ```

```
fargate-ip-192-0-2-1.region-code.compute.internal   Ready    <none>
  7m30s   v1.2.3-eks-1234567   192-0-2-1     <none>        Amazon Linux 2
  1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
```

Managed nodes – Linux

```
NAME                                        STATUS   ROLES    AGE    VERSION
          INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-VERSION
       CONTAINER-RUNTIME
ip-192-0-2-0.region-code.compute.internal   Ready    <none>   6m7s
   v1.2.3-eks-1234567   192.0.2.0     192.0.2.2     Amazon Linux 2
  1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
ip-192-0-2-1.region-code.compute.internal   Ready    <none>   6m4s
   v1.2.3-eks-1234567   192.0.2.1     192.0.2.3     Amazon Linux 2
  1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
```

如需您在輸出中看到之內容的詳細資訊，請參閱 檢視 Kubernetes 資源。

2. 檢視叢集上執行的工作負載。

```
kubectl get pods -A -o wide
```

範例輸出如下。

Fargate – Linux

```
NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE   IP
          NODE                                                NOMINATED NODE
 READINESS GATES
kube-system   coredns-1234567890-abcde   1/1     Running   0          18m
  192.0.2.0   fargate-ip-192-0-2-0.region-code.compute.internal   <none>
   <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          18m
  192.0.2.1   fargate-ip-192-0-2-1.region-code.compute.internal   <none>
   <none>
```

Managed nodes – Linux

```
NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE     IP
          NODE                                          NOMINATED NODE   READINESS
 GATES
```

```
kube-system   aws-node-12345              1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
kube-system   aws-node-67890              1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-abcde    1/1     Running   0          14m
   192.0.2.3   ip-192-0-2-3.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-12345    1/1     Running   0          14m
   192.0.2.4   ip-192-0-2-4.region-code.compute.internal   <none>
 <none>
kube-system   kube-proxy-12345            1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   kube-proxy-67890            1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
```

如需您在輸出中看到之內容的詳細資訊，請參閱 檢視 Kubernetes 資源。

步驟 3：刪除您的叢集和節點

在完成為本教學建立的叢集和節點後，您應該使用以下命令刪除叢集和節點來完成清理。如果想在清理前使用此叢集執行更多作業，請參閱 後續步驟。

```
eksctl delete cluster --name my-cluster --region region-code
```

後續步驟

以下文件主題可協助您擴展叢集的功能。

- 將範例應用程式部署至叢集。
- 建立叢集的 IAM 主體是唯一可以使用 kubectl 或 AWS Management Console對 Kubernetes API 伺服器進行呼叫的主體。如果要其他 IAM 主體能夠存取叢集，則需新增這些主體。如需詳細資訊，請參閱 授予 Kubernetes API 的存取權 及 所需的許可。
- 部署叢集以供生產使用之前，建議您先熟悉叢集和節點的所有設定。建立叢集時，必須進行某些設定 (例如啟用對 Amazon EC2 節點的 SSH 存取)。

- 若要提高叢集的安全性，可將 Amazon VPC 容器聯網介面外掛程式設定為使用服務帳戶的 IAM 角色。

開始使用 Amazon EKS — AWS Management Console 和 AWS CLI

本指南可協助您建立所有必要的資源，以開始使用 Amazon Elastic Kubernetes Service (Amazon EKS) 使用和. AWS Management Console AWS CLI在本指南中，您可以手動建立每個資源。此教學結束時，您將擁有一個執行中的 Amazon EKS 叢集，而您可以在其中部署應用程式。

本指南中的程序可讓您完全瞭解每個資源的建立方式，以及資源彼此之間的互動方式。如果您希望自動為您建立大部分資源，請使用 `eksctl` CLI 建立叢集和節點。如需詳細資訊，請參閱 Amazon EKS 入門：`eksctl`。

必要條件

開始此教學之前，您必須先安裝和設定下列在建立和管理 Amazon EKS 叢集所需的工具和資源。

- AWS CLI— 用於處理包括 Amazon EKS 在內的 AWS 服務的命令行工具。如需詳細資訊，請參閱《 AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI。安裝之後 AWS CLI，我們建議您也對其進行配置。如需詳細資訊，請參閱《 AWS Command Line Interface 使用者指南》中的使用 `aws configure` 進行快速組態設定。
- **`kubectl`**：命令列工具，適用於使用 Kubernetes 叢集。如需詳細資訊，請參閱 安裝或更新 `kubectl`。
- 必要的 IAM 許可 — 您使用的 IAM 安全主體必須具有使用 Amazon EKS IAM 角色、服務連結角色 AWS CloudFormation、VPC 和相關資源的許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 Amazon Elastic Kubernetes Service 的動作、資源和條件索引鍵以及使用服務連結角色。您必須以同一位使用者的身分完成本指南中的所有步驟。若要檢查目前使用者，請執行以下命令：

  ```
  aws sts get-caller-identity
  ```

- 建議您在 Bash shell 中完成本主題中的步驟。如果您不使用 Bash shell，則某些指令碼命令 (如行接續字元以及設定和使用變數的方式) 需要針對 shell 進行調整。此外，您的 Shell 的引用及轉義規則可能會有所不同。若要取得更多資訊，請參閱《使用指南》中的〈 AWS CLI中的〈使用引號搭配字串〉。 AWS Command Line Interface

步驟 1：建立您的 Amazon EKS 叢集

> **Important**
>
> 為了盡可能簡單快速地開始使用，本主題包含使用預設設定建立叢集的步驟。建立叢集供生產使用之前，建議您先熟悉所有設定，再使用符合需求的設定來部署叢集。如需詳細資訊，請參閱 建立 Amazon EKS 叢集。只有在建立叢集時，才能夠啟用某些設定。

若要建立叢集

1. 使用符合 Amazon EKS 要求的公有和私有子網路建立 Amazon VPC。使用 Amazon EKS 支援的任何 AWS 區域 取代 *region-code*。如需清單 AWS 區域，請參閱 AWS 一般參考指南中的 Amazon EKS 端點和配額。您可以使用您選擇的任何名稱取代 *my-eks-vpc-stack*。

   ```
   aws cloudformation create-stack \
     --region region-code \
     --stack-name my-eks-vpc-stack \
     --template-url https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
   ```

   > **Tip**
   >
   > 如需上一個命令建立的所有資源清單，請開啟位於 https://console.aws.amazon.com/cloudformation 的 AWS CloudFormation 主控台。選擇 *my-eks-vpc-stack* 堆疊，然後選擇 Resources (資源) 索引標籤。

2. 建立叢集 IAM 角色，並將必要的 Amazon EKS IAM 受管政策附加到該角色。 KubernetesAmazon EKS 管理的叢集代表您呼叫其他 AWS 服務，以管理您搭配服務使用的資源。

   a. 將下列內容複製到名為 *eks-cluster-role-trust-policy.json* 的檔案。

      ```
      {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Principal": {
              "Service": "eks.amazonaws.com"
            },
      ```

```
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. 建立角色。

```
aws iam create-role \
  --role-name myAmazonEKSClusterRole \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
```

c. 將必要的 Amazon EKS 受管 IAM 政策連接到角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name myAmazonEKSClusterRole
```

3. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

   請確定主控台右上角 AWS 區域 顯示的是您 AWS 區域 要在其中建立叢集的。如果不是，請選擇 AWS 區域 名稱旁邊的下拉菜單，然後選擇 AWS 區域 要使用的名稱。

4. 選擇 Add cluster (新增叢集)，然後選擇 Create (建立)。如果沒有看到這個選項，請先在左側導覽窗格中選擇 Clusters (叢集)。

5. 在 Configure cluster (設定叢集) 頁面上，執行下列操作：

   a. 輸入叢集的 Name (名稱)，例如 **my-cluster**。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。

   b. 對於叢集服務角色，請選擇*## Amazoneks ClusterRole*。

   c. 將其餘設定維持在其預設值，然後選取 Next (下一步)。

6. 在 Specify networking (指定聯網) 頁面中，執行下列操作：

   a. 從 VPC 下拉式清單中選擇在先前步驟中建立的 VPC ID。這是類似 *vpc-00x0000x000x0x000* | *my-eks-vpc-stack-VPC* 的文字。

   b. 將其餘設定維持在其預設值，然後選取 Next (下一步)。

7. 在設定可觀測性頁面上，選擇下一步。

8. 在選取附加元件頁面上，選擇下一步。

   如需附加元件的詳細資訊，請參閱 Amazon EKS 附加元件。

9. 在設定選取的附加元件設定頁面中，選取下一步：
10. 在 Review and create (檢閱和建立) 頁面上，選取 Create (建立)。

    叢集名稱右側的叢集狀態會顯示 Creating (建立中) 幾分鐘，直到叢集佈建程序完成為止。在狀態為 Active (作用中) 之前不要進行下一個步驟。

    > **Note**
    >
    > 您可能會收到錯誤，表示在請求中的其中一個可用區域沒有足夠的容量可建立 Amazon EKS 叢集。如果發生這種情況，錯誤輸出包含的可用區域可支援新的叢集。使用至少兩個位於帳戶的支援可用區域子網路來建立您的叢集。如需詳細資訊，請參閱 容量不足。

步驟 2：將您的電腦設為與叢集進行通訊

在此區段中，您會為叢集建立 kubeconfig 檔案。此檔案中的設定會啟用 kubectl CLI 與您的叢集進行通訊。

若要設定您的電腦與叢集通訊

1. 為您的叢集建立或更新 kubeconfig 檔案。請使用您在其中建立叢集的 AWS 區域 取代 *region-code*。使用您叢集的名稱取代 *my-cluster*。

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   根據預設，已在 ~/.kube 建立了 config 檔案，或者已將新叢集的組態新增至現有 ~/.kube 中的 config 檔案。
2. 測試組態。

   ```
   kubectl get svc
   ```

   > **Note**
   >
   > 如果您收到任何授權或資源類型錯誤，請參閱故障診斷主題中的未經授權或存取遭拒 (kubectl)。

   範例輸出如下。

```
NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
```

步驟 3：建立節點

> **Important**
>
> 為了盡可能簡單快速地開始使用，本主題包含使用預設設定建立節點的步驟。建立節點供生產使用之前，建議您先熟悉所有設定，再使用符合需求的設定來部署節點。如需詳細資訊，請參閱 Amazon EKS 節點。只有在建立節點時，才能夠啟用某些設定。

您可以使用下列其中一種節點類型來建立叢集。若要進一步了解每種類型，請參閱 Amazon EKS 節點。叢集部署完成後，您可以新增其他節點類型。

- Fargate – Linux：如果您想要在 AWS Fargate 上執行 Linux 應用程式，請選取此節點類型。Fargate 是無伺服器運算引擎，可讓您部署 Kubernetes Pods 而無須管理 Amazon EC2 執行個體。
- 受管節點 – Linux：如果您想要執行，請選擇此節點類型 Amazon Linux 應用程式。雖然未涵蓋在本指南中，但您也可以將 Windows 自我管理節點和 Bottlerocket 節點新增至您的叢集。

Fargate – Linux

建立 Fargate 設定檔。當 Kubernetes Pods 的部署條件符合設定檔中定義的條件時，Pods 會部署至 Fargate。

建立 Fargate 設定檔

1. 建立 IAM 角色，並將所需的 Amazon EKS IAM 受管政策連接到該角色。當您的叢集Pods在 Fargate 基礎架構上建立時，在 Fargate 基礎架構上執行的元件必須代表您呼叫 AWS API。這樣他們就可以執行動作，例如從 Amazon ECR 提取容器映像或將日誌路由到其他 AWS 服務。Amazon EKS Pod 執行角色提供進行此類工作的 IAM 許可。

   a. 將下列內容複製到名為 *pod-execution-role-trust-policy*.json 的檔案。*region-code*以叢集所 AWS 區域 在的位置取代。如果您想在所有帳戶中使用相同 AWS 區域 的角色，請*region-code*替換為*。使用您的帳戶 ID 取代 *111122223333*，

再以您的叢集名稱取代 *my-cluster*。若要在您的帳戶中的所有叢集使用相同角色，請使用 * 取代 *my-cluster*。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. 建立 Pod 執行 IAM 角色。

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-
policy.json"
```

c. 將必要的 Amazon EKS 受管 IAM 政策連接到角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/
AmazonEKSFargatePodExecutionRolePolicy \
  --role-name AmazonEKSFargatePodExecutionRole
```

2. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
3. 在 Clusters (叢集) 頁面上，選擇 *my-cluster* 叢集。
4. 在 ***my-cluster*** 頁面上，執行下列操作：

   a. 選擇 Compute (運算) 索引標籤。

b. 在 Fargate profiles (Fargate 設定檔) 下，選擇 Add Fargate profile (新增 Fargate 設定檔)。

5. 在 Configure Fargate Profile (設定 Fargate 設定檔) 頁面上，執行以下操作：

   a. 對於 Name (名稱)，輸入 Fargate 設定檔的唯一名稱，例如 ***my-profile***。

   b. 對於網繭執行角色，請選擇您在上一FargatePodExecutionRole個步驟中建立的 AmazonEks。

   c. 選擇 Subnets (子網路) 下拉式清單並取消選取名稱中含有 `Public` 的任何子網路。只有私有子網路支援在 Fargate 上執行的 Pods。

   d. 選擇下一步。

6. 在 Configure Pod selection (設定 選擇) 頁面上，執行以下操作：

   a. 針對 Namespace (命名空間)，輸入 **`default`**。

   b. 選擇下一步。

7. 在 Review and create (檢閱和建立) 頁面，檢閱您 Fargate 設定檔的資訊並選擇 Create (建立)。

8. 數分鐘後，Fargate Profile configuration (Fargate 設定檔組態) 區段中的 Status (狀態) 將從 Creating (建立中) 變更為 Active (作用中)。在狀態為 Active (作用中) 之前不要進行下一個步驟。

9. 如果打算將所有 Pods 部署至 Fargate (而非 Amazon EC2 節點)，請完成以下步驟，以建立另一個 Fargate 設定檔並在 Fargate 上執行預設名稱解析程式 (CoreDNS)。

> **Note**
>
> 如果您未執行此操作，則此時不會有任何節點。

   a. 在 Fargate Profile (Fargate 設定檔) 頁面上，選擇 *`my-profile`*。

   b. 在 Fargate profiles (Fargate 設定檔) 下，選擇 Add Fargate Profile (新增 Fargate 設定檔)。

   c. 針對名稱，輸入 ***CoreDNS***。

   d. 對於網繭執行角色，請選擇您在上一FargatePodExecutionRole個步驟中建立的 AmazonEks。

   e. 選擇 Subnets (子網路) 下拉式清單並取消選取名稱中含有 `Public` 的任何子網路。只有私有子網路支援在 Fargate 上執行的 Pods。

f. 選擇下一步。

g. 針對 Namespace (命名空間)，輸入 **kube-system**。

h. 選擇 Match labels (比對標籤)，然後選擇 Add label (新增標籤)。

i. 對於 **k8s-app**Key (索引鍵)，輸入 ，對於其值輸入 **kube-dns**。這是將預設名稱解析程式 (CoreDNS) 部署到 Fargate 的必要操作。

j. 選擇下一步。

k. 在 Review and create (檢閱和建立) 頁面，檢閱您 Fargate 設定檔的資訊並選擇 Create (建立)。

l. 執行以下命令，將預設 `eks.amazonaws.com/compute-type : ec2` 註釋從 CoreDNS Pods 中移除。

```
kubectl patch deployment coredns \
    -n kube-system \
    --type json \
    -p='[{"op": "remove", "path": "/spec/template/metadata/annotations/
eks.amazonaws.com~1compute-type"}]'
```

> **Note**
>
> 系統根據您新增的 Fargate 設定檔標籤建立和部署兩個節點。您不會看到 Node groups (節點群組) 中列出任何節點，因為它們不適用於 Fargate 節點，但您將看到 Overview (概觀) 索引標籤中列出的新節點。

Managed nodes – Linux

建立受管節點群組，指定您在先前步驟中建立的子網路和節點 IAM 角色。

建立 Amazon EC2 Linux 受管節點群組

1. 建立節點 IAM 角色，並將所需的 Amazon EKS IAM 受管政策連接到該角色。Amazon EKS 節點`kubelet`常駐程式會代表您呼叫 AWS API。節點透過 IAM 執行個體描述檔和關聯的政策，取得這些 API 呼叫的許可。

   a. 將下列內容複製到名為 *`node-role-trust-policy.json`* 的檔案。

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. 建立節點 IAM 角色。

```
aws iam create-role \
  --role-name myAmazonEKSNodeRole \
  --assume-role-policy-document file://"node-role-trust-policy.json"
```

c. 將所需的受管 IAM 政策連接到角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name myAmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name myAmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name myAmazonEKSNodeRole
```

2. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
3. 選擇您在 步驟 1：建立您的 Amazon EKS 叢集 中建立之叢集的名稱，例如 *my-cluster*。
4. 在 ***my-cluster*** 頁面上，執行下列操作：
   a. 選擇 Compute (運算) 索引標籤。
   b. 選擇 Add Node Group (新增節點群組)。
5. 在 Configure Node Group (設定節點群組) 頁面上，執行以下操作：
   a. 對於 Name (名稱)，輸入受管節點群組的唯一名稱，例如 ***my-nodegroup***。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。

b. 對於節點 IAM 角色名稱，請選擇您在上一步中建立的 *MyAmazonks NodeRole* 角色。建議每個節點群組使用其唯一的 IAM 角色。

c. 選擇下一步。

6. 在 Set compute and scaling configuration (設定運算和擴展組態) 頁面上，接受預設值，然後選取 Next (下一步)。

7. 在 Specify networking (指定聯網) 頁面上，接受預設值，然後選取 Next (下一步)。

8. 在 Review and create (檢閱並建立) 頁面上，檢閱您的受管節點群組組態，然後選擇 Create (建立)。

9. 數分鐘後，Node Group configuration (節點群組組態) 區段中的 Status (狀態) 將從 Creating (建立中) 變更為 Active (作用中)。在狀態為 Active (作用中) 之前不要進行下一個步驟。

步驟 4：檢視資源

您可以檢視節點和 Kubernetes 工作負載。

檢視節點和工作負載

1. 在左側導覽窗格中選擇 Clusters (叢集)。在 Clusters (叢集) 清單中，選擇您建立之叢集的名稱，例如 *my-cluster*。

2. 在 ***my-cluster*** 頁面上，選擇以下事項：

a. Compute (運算) 標籤 – 您可以看到為叢集部署的 Nodes (節點) 清單。您可以選取節點的名稱，以查看有關節點的詳細資訊。

b. Resources (資源) 標籤：您會看到預設部署至 Amazon EKS 叢集的所有 Kubernetes 資源。在主控台中選取任何資源類型，以進一步了解其詳細資訊。

步驟 5：刪除資源

在完成為本教學建立的叢集和節點後，您應該刪除所建立的資源。如果想在刪除資源前使用此叢集執行更多作業，請參閱 後續步驟。

刪除在本指南中建立的資源

1. 刪除您建立的任何節點群組或 Fargate 設定檔。

a. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。

b. 在左側導覽窗格中選擇 Clusters (叢集)。在叢集清單中，選取 *my-cluster*。

c. 選擇 Compute (運算) 索引標籤。

d. 如果建立了節點群組，請選擇 *my-nodegroup* 節點群組，然後選擇 Delete (刪除)。輸入 ***my-nodegroup***，然後選擇 Delete (刪除)。

e. 對於您建立的每個 Fargate 設定檔，選擇此設定檔，然後選擇 Delete (刪除)。輸入設定檔的名稱，然後選取 Delete (刪除)。

> **Note**
>
> 刪除第二個 Fargate 設定檔時，您可能需要等待第一個設定檔完成刪除。

f. 請刪除節點群組或 Fargate 設定檔後再繼續。

2. 刪除叢集。

a. 在左側導覽窗格中選擇 Clusters (叢集)。在叢集清單中，選取 *my-cluster*。

b. 選擇 Delete cluster (刪除叢集)。

c. 輸入 ***my-cluster***，然後選擇 Delete (刪除)。刪除叢集後再繼續。

3. 刪除您建立的 VPC AWS CloudFormation 堆疊。

a. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation

b. 選擇 *my-eks-vpc-stack* 堆疊，然後選擇 Delete (刪除)。

c. 在 Delete ***my-eks-vpc-stack*** (刪除 my-eks-vpc-stack) 確認對話方塊中，選擇 Delete stack (刪除堆疊)。

4. 刪除您建立的 IAM 角色。

a. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。

b. 在左側導覽窗格中，選擇 Roles (角色)。

c. ***################ AmazonEKS ClusterRole############ FargatePod ExecutionRole***##*NodeRole*選擇 Delete (刪除)，輸入請求的確認文字，然後選擇 Delete (刪除)。

後續步驟

以下文件主題可協助您擴展叢集的功能。

- 建立叢集的 IAM 主體是唯一可以使用 `kubectl` 或 AWS Management Console對 Kubernetes API 伺服器進行呼叫的主體。如果要其他 IAM 主體能夠存取叢集，則需新增這些主體。如需更多詳細資訊，請參閱 授予 Kubernetes API 的存取權 及 所需的許可。
- 將範例應用程式部署至叢集。
- 部署叢集以供生產使用之前，建議您先熟悉叢集和節點的所有設定。建立叢集時，必須進行某些設定 (例如啟用對 Amazon EC2 節點的 SSH 存取)。
- 若要提高叢集的安全性，可將 Amazon VPC 容器聯網介面外掛程式設定為使用服務帳戶的 IAM 角色。

Amazon EKS 叢集

Amazon EKS 叢集包含兩個主要元件：

- Amazon EKS 控制平面
- 使用控制平面註冊的 Amazon EKS 節點

Amazon EKS 控制平面由執行 Kubernetes 軟體 (例如 `etcd` 和 Kubernetes API 伺服器) 的控制平面節點所組成。控制平面在由管理的帳戶中執行 AWS，而 Kubernetes API 會透過與叢集關聯的 Amazon EKS 端點公開。每一個 Amazon EKS 叢集控制平面都是單一租用戶且是唯一的，並在自己的一組 Amazon EC2 執行個體上執行。

`etcd`節點和相關聯的 Amazon EBS 磁碟區存放的所有資料均使用 AWS KMS加密。叢集控制平面會跨多個可用區域佈建，並會由 Elastic Load Balancing Network Load Balancer 作為前端。Amazon EKS 也會在您的 VPC 子網中佈建彈性網路介面來提供從控制平面執行個體到節點間的連接能力 (例如，支援 `kubectl exec logs proxy` 資料流程)。

Important

在 Amazon EKS 環境中，根據上游指導，`etcd` 儲存空間限制為 8 GiB。您可以執行以下命令，以監控目前資料庫大小的指標。如果您叢集的 Kubernetes 版本低於 `1.28`，請用以下列項目取代 *`apiserver_storage_size_bytes`*：

- Kubernetes 版本 `1.27` 和 `1.26`：**`apiserver_storage_db_total_size_in_bytes`**
- Kubernetes 版本 `1.25` 及以下：**`etcd_db_total_size_in_bytes`**

```
kubectl get --raw=/metrics | grep "apiserver_storage_size_bytes"
```

Amazon EKS 節點在您的 AWS 帳戶中執行，並透過 API 伺服器端點和為叢集建立的憑證檔案連接到叢集的控制平面。

Note

- 您可以在 Amazon EKS 聯網 了解 Amazon EKS 的不同元件的運作方式。

- 如需已連接叢集的相關資訊，請參閱 Amazon EKS 連接器。

主題

建立 Amazon EKS 叢集

本主題提供可用選項的概觀，並介紹您建立 Amazon EKS 叢集時需要考慮的問題。如果您需要在 AWS Outpost 上建立叢集，請參閱AWS Outposts 上的 Amazon EKS 本機叢集。如果這是您第一次建立 Amazon EKS 叢集，建議您按照其中一個Amazon EKS 入門指南的步驟進行。這些指南可協助您建立一個簡單的預設叢集，而無需擴展到所有可用選項。

必要條件

- 現有 VPC 和子網符合 Amazon EKS 要求。部署叢集以供生產使用之前，建議您先徹底了解 VPC 和子網要求。如果您沒有 VPC 和子網路，您可以使用 Amazon EKS 提供的範本建立它們。 AWS CloudFormation
- 已在裝置或 AWS CloudShell上安裝 `kubectl` 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 `1.29`，則可以搭配使用 `kubectl` `1.28`、`1.29` 或 `1.30` 版。若要安裝或升級 `kubectl`，請參閱 安裝或更新 `kubectl`。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本`1.27.160`或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。`2.12.3`若要檢查您目前的版本，請使

用 **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到您的主目錄〉。

- 具有對 Amazon EKS 叢集 `create` 和 `describe` 的許可的 IAM 主體。如需詳細資訊，請參閱 在 Outpost 上建立本機 Kubernetes 叢集 及 所有叢集的清單或描述。

建立 Amazon EKS 叢集

1. 如果您已擁有叢集 IAM 角色，或者您要使用 `eksctl` 來建立叢集，則可以略過此步驟。根據預設，`eksctl` 會為您建立角色。

   建立 Amazon EKS 叢集 IAM 角色

   1. 執行下列命令以建立 IAM 信任政策 JSON 檔案。

      ```
      cat >eks-cluster-role-trust-policy.json <<EOF
      {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Principal": {
              "Service": "eks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      }
      EOF
      ```

   2. 建立 Amazon EKS 叢集 IAM 角色。如有必要，請在 *`eks-cluster-role-trust-policy.json`* 前面加上您在上一步中寫入檔案的電腦的路徑。命令會將您在上一步驟中建立的信任策略與角色相關聯。若要建立 IAM 角色，必須為建立角色的 IAM 主體指派以下 `iam:CreateRole` 動作 (許可)。

```
aws iam create-role --role-name myAmazonEKSClusterRole --assume-role-policy-
document file://"eks-cluster-role-trust-policy.json"
```

3. 您可以指派 Amazon EKS 受管政策或建立自己的自訂政策。如需了解在自訂政策中必須使用的最低許可，請參閱 Amazon EKS 叢集 IAM 角色。

   將名為 AmazonEKSClusterPolicy 的 Amazon EKS 受管 IAM 政策連接到角色。若要將 IAM 政策連接至 IAM 主體，必須為連接政策的 IAM 實體指派以下 IAM 動作之一 (許可)：iam:AttachUserPolicy 或 iam:AttachRolePolicy。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name myAmazonEKSClusterRole
```

2. 建立 Amazon EKS 叢集。

   您可以使用eksctl AWS Management Console、或建立叢集 AWS CLI。

   eksctl

   先決條件

   已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

   若要建立叢集

   在您的預設 AWS 區域中，使用 Amazon EKS 預設 Kubernetes 版本建立 Amazon EKS IPv4 叢集。執行命令之前，請執行下列替換：

   - 取*region-code*代為您 AWS 區域 要在其中建立叢集的。
   - 使用您的叢集名稱取代 *my-cluster*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - 使用任何 Amazon EKS 支援的版本取代 *1.29*。

   > **Note**
   > 若要在此時部署 1.30 叢集，您必須使用 AWS Management Console 或. AWS CLI

- 變更 vpc-private-subnets 的值以符合您的要求。您也可以新增其他 ID。您必須指定至少兩個子網路 ID。如果您希望指定公有子網，則可以將 --vpc-private-subnets 變更為 --vpc-public-subnets。公有子網具有與網際網路閘道路由相關的路由表，但私有子網沒有關聯的路由表。我們建議盡可能使用私有子網。

  您選擇的子網必須符合 Amazon EKS 子網要求。選擇子網之前，建議您先熟悉 Amazon EKS VPC 和子網要求和考量事項。

```
eksctl create cluster --name my-cluster --region region-code --version 1.29 --
vpc-private-subnets subnet-ExampleID1,subnet-ExampleID2 --without-nodegroup
```

叢集佈建需要幾分鐘的時間。建立叢集時，會出現幾行輸出。輸出的最後一行類似於下面的範例行。

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

> **Tip**
>
> 若要查看使用 eksctl 建立叢集時可指定的大部分選項，請使用 **eksctl create cluster --help** 命令。若要查看所有可用的選項，您可使用 config 檔案。如需詳細資訊，請參閱 eksctl 文件中的使用組態檔與組態檔結構描述。您可以在 GitHub 上找到組態檔範例。

選項設定

以下是選用設定，如有需要，必須將其新增至上一個命令中。您只能在建立叢集時啟用這些選項，而不能在建立叢集之後啟用。如果需要指定這些選項，您必須使用 eksctl 組態檔案建立叢集並指定設定，而不是使用上一個命令。

- 如果您要指定一或多個 Amazon EKS 指派給其建立的網路介面的安全群組，請指定 securityGroup 選項。

  無論您是否選擇任何安全群組，Amazon EKS 都會建立一個安全群組，以支援您的叢集和 VPC 之間的通訊。Amazon EKS 將此安全群組及您選擇的任何群組與其建立的網路介面相關聯。如需有關 Amazon EKS 建立的叢集安全群組的詳細資訊，請參閱 the section called “安全群組要求”。您可以修改 Amazon EKS 建立的叢集安全群組中的規則。

- 如果您想指定 Kubernetes 要從哪個 IPv4 無類別域間路由 (CIDR) 區塊中指派服務 IP 地址，請指定 serviceIPv4CIDR 選項。

  指定您自己的範圍有助於防止 Kubernetes 服務與對等或連接到您的 VPC 的其他網路之間發生衝突。在 CIDR 標記法中輸入範圍。例如：10.2.0.0/16。

  CIDR 區塊必須符合下列需求：

  - 處於以下範圍之一：10.0.0.0/8、172.16.0.0/12 或 192.168.0.0/16。
  - 具有最小尺寸的 /24 和最大尺寸的 /12。
  - 與您的 Amazon EKS 資源的 VPC 範圍不重疊。

  您只能在使用 IPv4 地址系列時指定此選項，並且只能在建立叢集時指定。如果您未指定此項，則 Kubernetes 會從 10.100.0.0/16 或 172.20.0.0/16 CIDR 區塊指派服務 IP 地址。

- 如果您正在建立叢集，並希望叢集將 IPv6 地址 (而不是 IPv4 地址) 指派給 Pods 和服務，請指定 ipFamily 選項。

  預設情況下，Kubernetes 會為 Pods 和服務指派 IPv4 地址。在決定使用 IPv6 系列之前，請確認您已熟悉 the section called “VPC 要求和注意事項”、the section called “子網需求和注意事項”、the section called “安全群組要求” 和 the section called “IPv6” 主題中的所有考量事項和要求。如果選擇 IPv6 系列，則無法像為 IPv4 系列一樣為 Kubernetes 指定地址範圍來指派 IPv6 服務地址。Kubernetes 從唯一的本機地址範圍 (fc00::/7) 指派服務地址。

AWS Management Console

若要建立叢集

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選取 Add cluster (新增叢集)，然後選取 Create (建立)。
3. 在 Configure cluster (設定叢集) 頁面上，輸入下列欄位：
   - Name (名稱)：叢集的名稱。名稱只能包含英數字元 (區分大小寫)、連字號和底線。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - Kubernetes version (Kubernetes 版本) – 您的叢集使用的 版本。我們建議選取最新版本，除非您需要較早版本。

- 叢集服務角色 — 選擇您建立的 Amazon EKS 叢集 IAM 角色，以允許Kubernetes控制平面代表您管理 AWS 資源。
- Secrets encryption (秘密加密) – (選用) 選擇使用 KMS 金鑰啟用 Kubernetes 秘密的秘密加密。您也可以在建立叢集後啟用此功能。啟用此功能之前，請確定您已熟悉 在現有叢集上啟用密碼加密 中的資訊。
- Tags (標籤) – (選用) 將任何標籤新增到您的叢集。如需詳細資訊，請參閱 為您的 Amazon EKS 資源加上標籤。

  完成此頁面後，請選擇下一步。

4. 在 Specify networking (指定網路) 頁面上，選取下列欄位的值：
   - VPC：選擇符合 Amazon EKS VPC 要求的現有 VPC 來建立您的叢集。選擇 VPC 之前，建議您先熟悉 Amazon EKS VPC 與子網要求和注意事項 中的所有要求和考量事項。建立叢集後，您無法變更要使用的 VPC。如果沒有列出 VPC，則您需要先建立一個。如需詳細資訊，請參閱 為 Amazon EKS 叢集建立 VPC。
   - Subnets (子網路)：根據預設，前一個欄位指定的 VPC 中的所有可用子網路會預先選取。您必須選取至少兩個。

     您選擇的子網必須符合 Amazon EKS 子網要求。選擇子網之前，建議您先熟悉 Amazon EKS VPC 和子網要求和考量事項。

     安全群組:(選用) 指定一或多個您希望 Amazon EKS 與其建立的網路介面相關聯的安全群組。

     無論您是否選擇任何安全群組，Amazon EKS 都會建立一個安全群組，以支援您的叢集和 VPC 之間的通訊。Amazon EKS 將此安全群組及您選擇的任何群組與其建立的網路介面相關聯。如需有關 Amazon EKS 建立的叢集安全群組的詳細資訊，請參閱 the section called “安全群組要求”。您可以修改 Amazon EKS 建立的叢集安全群組中的規則。
   - 選擇叢集 IP 地址系列：您可以選擇 IPv4 或 IPv6。

     預設情況下，Kubernetes 會為 Pods 和服務指派 IPv4 地址。在決定使用 IPv6 系列之前，請確認您已熟悉 the section called “VPC 要求和注意事項”、the section called “子網需求和注意事項”、the section called “安全群組要求” 和 the section called “IPv6” 主題中的所有考量事項和要求。如果選擇 IPv6 系列，則無法像為 IPv4 系列一樣為 Kubernetes 指定地址範圍來指派 IPv6 服務地址。Kubernetes 從唯一的本機地址範圍 (fc00::/7) 指派服務地址。

- (選用) 選擇 Configure Kubernetes Service IP address range (設定 服務 IP 地址範圍)，並指定 Service range (服務 IPv4 範圍)。

  指定您自己的範圍有助於防止 Kubernetes 服務與對等或連接到您的 VPC 的其他網路之間發生衝突。在 CIDR 標記法中輸入範圍。例如：10.2.0.0/16。

  CIDR 區塊必須符合下列需求：

  - 處於以下範圍之一：10.0.0.0/8、172.16.0.0/12 或 192.168.0.0/16。
  - 具有最小尺寸的 /24 和最大尺寸的 /12。
  - 與您的 Amazon EKS 資源的 VPC 範圍不重疊。

  您只能在使用 IPv4 地址系列時指定此選項，並且只能在建立叢集時指定。如果您未指定此項，則 Kubernetes 會從 10.100.0.0/16 或 172.20.0.0/16 CIDR 區塊指派服務 IP 地址。

- 針對 Cluster endpoint access (叢集端點存取)，選取一個選項。建立叢集後，您可以變更此選項。在選取非預設選項之前，請務必熟悉這些選項及其含義。如需詳細資訊，請參閱 Amazon EKS 叢集端點存取控制。

  完成此頁面後，請選擇下一步。

5. (選用) 在設定可觀測性頁面上，選擇要開啟的指標和控制平面日誌記錄選項。根據預設，系統會關閉每個日誌類型。
   - 如需有關 Prometheus 指標選項的詳細資訊，請參閱 建立叢集時開啟 Prometheus 指標。
   - 如需控制平面日誌記錄選項的詳細資訊，請參閱 Amazon EKS 控制平面記錄。

   完成此頁面後，請選擇下一步。
6. 在 Select add-ons (選取附加元件) 頁面上，選擇您要新增至叢集的附加元件。您可以根據需要選擇任意多個 Amazon EKS 附加元件和 AWS Marketplace 附加元件。如果未列出您要安裝的 AWS Marketplace 附加元件，則您可以在搜尋方框中輸入文字來搜尋可用的 AWS Marketplace 附加元件。您也可以依 category (類別)、vendor (廠商) 或 pricing model (定價模式) 進行搜尋，然後從搜尋結果中選擇附加元件。完成此頁面後，請選擇下一步。
7. 在設定選取的附加元件設定頁面上，選取您要安裝的版本。建立叢集後，您隨時皆可更新至更新版本。您可以在建立叢集後更新每個附加元件的組態。如需有關設定附加元件的詳細資訊，請參閱更新附加元件。完成此頁面後，請選擇下一步。
8. 在 Review and create (檢閱並建立) 頁面上，檢閱您在先前頁面上輸入或選取的資訊。如需變更，請選擇 Edit (編輯)。當您感到滿意時，請選擇 Create (建立)。在佈建叢集時，Status (狀態) 欄位顯示 CREATING (正在建立)。

> **Note**
>
> 您可能會收到錯誤，表示在請求中的其中一個可用區域沒有足夠的容量可建立 Amazon EKS 叢集。如果發生這種情況，錯誤輸出包含的可用區域可支援新的叢集。使用至少兩個位於帳戶的支援可用區域子網路來建立您的叢集。如需詳細資訊，請參閱 容量不足。

叢集佈建需要幾分鐘的時間。

AWS CLI

若要建立叢集

1. 使用下列命令建立您的叢集。執行命令之前，請執行下列替換：
   - 取*region-code*代為您 AWS 區域 要在其中建立叢集的。
   - 使用您的叢集名稱取代 *my-cluster*。名稱只能包含英數字元 (區分大小寫)、連字號和底線。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - 使用任何 Amazon EKS 支援的版本取代 *1.30*。
   - 使用您的帳戶 ID 取代 *111122223333*，再以您的叢集 IAM 角色名稱取代 *myAmazonEKSClusterRole*。
   - 以您自己的值取代 `subnetIds` 值。您也可以新增其他 ID。您必須指定至少兩個子網路 ID。

     您選擇的子網必須符合 Amazon EKS 子網要求。選擇子網之前，建議您先熟悉 Amazon EKS VPC 和子網要求和考量事項。
   - 如果您不想要指定安全群組 ID，請從命令中移除 `,securityGroupIds=sg-`*ExampleID1*。如果要指定一或多個安全群組 ID，請將 `securityGroupIds` 取代為您自己的值。您也可以新增其他 ID。

     無論您是否選擇任何安全群組，Amazon EKS 都會建立一個安全群組，以支援您的叢集和 VPC 之間的通訊。Amazon EKS 將此安全群組及您選擇的任何群組與其建立的網路介面相關聯。如需有關 Amazon EKS 建立的叢集安全群組的詳細資訊，請參閱 the section called “安全群組要求”。您可以修改 Amazon EKS 建立的叢集安全群組中的規則。

```
aws eks create-cluster --region region-code --name my-cluster --kubernetes-
version 1.30 \
   --role-arn arn:aws:iam::111122223333:role/myAmazonEKSClusterRole \
   --resources-vpc-config
 subnetIds=subnet-ExampleID1,subnet-ExampleID2,securityGroupIds=sg-ExampleID1
```

> **Note**
>
> 您可能會收到錯誤，表示在請求中的其中一個可用區域沒有足夠的容量可建立 Amazon EKS 叢集。如果發生這種情況，錯誤輸出包含的可用區域可支援新的叢集。使用至少兩個位於帳戶的支援可用區域子網路來建立您的叢集。如需詳細資訊，請參閱 容量不足。

選項設定

以下是選用設定，如有需要，必須將其新增至上一個命令中。您只能在建立叢集時啟用這些選項，而不能在建立叢集之後啟用。

- 如果您想指定 Kubernetes 要從哪個 IPv4 無類別域間路由 (CIDR) 區塊中指派服務 IP 地址，您必須將 **--kubernetes-network-config serviceIpv4Cidr=*CIDR block*** 新增至下列命令來指定。

  指定您自己的範圍有助於防止 Kubernetes 服務與對等或連接到您的 VPC 的其他網路之間發生衝突。在 CIDR 標記法中輸入範圍。例如：10.2.0.0/16。

  CIDR 區塊必須符合下列需求：

  - 處於以下範圍之一：10.0.0.0/8、172.16.0.0/12 或 192.168.0.0/16。
  - 具有最小尺寸的 /24 和最大尺寸的 /12。
  - 與您的 Amazon EKS 資源的 VPC 範圍不重疊。

  您只能在使用 IPv4 地址系列時指定此選項，並且只能在建立叢集時指定。如果您未指定此項，則 Kubernetes 會從 10.100.0.0/16 或 172.20.0.0/16 CIDR 區塊指派服務 IP 地址。
- 如果您正在建立叢集，並希望叢集將 IPv6 地址 (而不是 IPv4 地址) 指派給 Pods 和服務，請將 **--kubernetes-network-config ipFamily=ipv6** 新增至以下命令。

預設情況下，Kubernetes 會為 Pods 和服務指派 IPv4 地址。在決定使用 IPv6 系列之前，請確認您已熟悉 the section called “VPC 要求和注意事項”、the section called “子網需求和注意事項”、the section called “安全群組要求” 和 the section called “IPv6” 主題中的所有考量事項和要求。如果選擇 IPv6 系列，則無法像為 IPv4 系列一樣為 Kubernetes 指定地址範圍來指派 IPv6 服務地址。Kubernetes 從唯一的本機地址範圍 (fc00::/7) 指派服務地址。

2. 佈建叢集需要幾分鐘才能完成。您可以使用下列命令來查詢叢集的狀態。

```
aws eks describe-cluster --region region-code --name my-cluster --query
 "cluster.status"
```

在傳回的輸出為 ACTIVE 之前，請勿進行下一個步驟。

3. 如果您使用 eksctl 建立叢集，則可以略過此步驟。這是因為 eksctl 已為您完成此步驟。透過向 kubectl config 檔案新增內容，使 kubectl 能夠與您的叢集通訊。如需建立和更新檔案的詳細資訊，請參閱 建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

範例輸出如下。

```
Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/
username/.kube/config
```

4. 透過執行以下命令確認與叢集的通訊。

```
kubectl get svc
```

範例輸出如下。

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

5. (建議) 若要使用某些 Amazon EKS 附加元件，或讓個別Kubernetes工作負載具有特定 AWS Identity and Access Management (IAM) 許可，請為叢集建立 IAM OpenID Connect (OIDC) 提供者。您只需要為叢集建立 IAM OIDC 提供商一次。若要進一步了解 Amazon EKS 附加元件，請參

閱 Amazon EKS 附加元件。若要進一步了解如何將特定 IAM 許可指派給您的工作負載，請參閱服務帳戶的 IAM 角色。

6. (建議) 將叢集設定為 Amazon VPC CNI plugin for Kubernetes 外掛程式，然後再將 Amazon EC2 節點部署到叢集。預設情況下，外掛程式已隨叢集一起安裝。將 Amazon EC2 節點新增到叢集時，外掛程式會自動部署至您新增的每個 Amazon EC2 節點。外掛程式要求您將以下 IAM 政策之一連接至 IAM 角色：

   `AmazonEKS_CNI_Policy` 受管 IAM 政策

   如果您的叢集使用 `IPv4` 系列

   您建立的 IAM 政策

   如果您的叢集使用 `IPv6` 系列

   您連接政策的 IAM 角色可以是節點 IAM 角色，也可以是僅用於外掛程式的專用角色。我們建議將政策連接至此角色。如需建立角色的詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA) 或 Amazon EKS 節點 IAM 角色。

7. 如果您使用部署叢集 AWS Management Console，則可以略過此步驟。在預設情況下， AWS Management Console 會部署 Amazon VPC CNI plugin for Kubernetes、CoreDNS 和 `kube-proxy` Amazon EKS 附加元件。

   如果您使用 `eksctl` 或 AWS CLI來部署叢集，則會部署 Amazon VPC CNI plugin for Kubernetes、CoreDNS 和 `kube-proxy` 自我管理附加元件。您可以將隨叢集一起部署的 Amazon VPC CNI plugin for Kubernetes、CoreDNS 和 `kube-proxy` 自我管理附加元件遷移到 Amazon EKS 附加元件。如需詳細資訊，請參閱 Amazon EKS 附加元件。

8. (選用) 如果您尚未執行此操作，則可啟用叢集的 Prometheus 指標。如需詳細資訊，請參閱《Amazon Managed Service for Prometheus 使用者指南》中的建立湊集器。

9. 如果啟用了 Prometheus 指標，則必須設定 `aws-auth ConfigMap` 來授予湊集器叢集內許可。如需詳細資訊，請參閱《Amazon Managed Service for Prometheus 使用者指南》中的設定 Amazon EKS 叢集。

10. 如果您計劃將工作負載部署到使用 Amazon EBS 磁碟區的叢集，並建立了 `1.23` 或更新版本的叢集，則必須在部署工作負載之前先為您的叢集安裝 Amazon EBS CSI 驅動程式。

建議的後續步驟：

- 建立叢集的 IAM 主體是唯一可以存取叢集的 IAM 主體。將許可授予其他 IAM 主體，這樣他們就可以存取您的叢集。
- 如果建立叢集的 IAM 主體僅具有先決條件中所參考的最低 IAM 許可，那麼您可能需要為該主體新增其他 Amazon EKS 許可。如需將 Amazon EKS 許可授予 IAM 主體的詳細資訊，請參閱 Amazon EKS 的 Identity and Access Management。
- 如果您希望建立叢集的 IAM 主體或任何其他主體在 Amazon EKS 主控台中檢視 Kubernetes 資源，請對實體授予 所需的許可。
- 如果您希望節點和 IAM 主體從 VPC 內存取您的叢集，請啟用叢集的私有端點。根據預設，公有端點為啟用狀態。如有需要，您可以在啟用私有端點後停用公有端點。如需詳細資訊，請參閱 Amazon EKS 叢集端點存取控制。
- 為叢集啟用密鑰加密。
- 設定叢集的日誌。
- 將節點新增至叢集。

叢集洞察

Amazon EKS 叢集洞察提供建議，協助您遵循 Amazon EKS 和 Kubernetes 最佳實務。每個 Amazon EKS 叢集都會根據 Amazon EKS 精選洞察清單定期進行自動檢查。這些洞察檢查完全由 Amazon EKS 管理，並提供有關如何處理調查結果的建議。

叢集見解的建議使用方式：

- 更新叢集Kubernetes版本之前，請檢查 EKS 主控台中的叢集見解。
- 如果您的叢集發現問題，請檢閱問題並進行適當的修正。這些問題包括鏈接到 Amazon EKS 和 Kubernetes.
- 修正問題後，請等待叢集深入解析重新整理。如果所有問題都已解決，請更新叢集。

目前，Amazon EKS 只傳回與 Kubernetes 版本升級準備相關的洞察。

升級洞察提供有關可能影響 Kubernetes 叢集升級問題的資訊。這可將管理員準備升級所需的工作量降至最低，並提高較新 Kubernetes 版本上應用程式的可靠性。Amazon EKS 會根據可能影響 Kubernetes 版本升級的問題清單自動掃描叢集。Amazon EKS 會經常透過檢閱每個 Kubernetes 版本發佈中所做的變更來更新洞察檢查清單。

Amazon EKS 升級洞察有助於加快新版本的測試和驗證流程。這些洞察還透過突出問題並提供補救建議，讓叢集管理員和應用程式開發人員能夠來利用 Kubernetes 最新的功能。若要查看 Amazon EKS

執行的洞察檢查清單以及識別的任何相關問題，您可以呼叫 Amazon EKS `ListInsights` API 操作或在 Amazon EKS 主控台中查看。

叢集見解會定期更新。您無法手動重新整理叢集見解。如果您修正叢集問題，叢集深入解析需要一些時間才能更新。若要判斷修正是否成功，請將部署變更的時間與叢集分析的「上次重新整理時間」進行比較。

檢視叢集見解 (主控台)

若要檢視 Amazon EKS 叢集的見解，請執行下列動作：

a. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

b. 從叢集清單中，選擇您想要查看其相關洞察之 Amazon EKS 叢集的名稱。

c. 選擇升級洞察索引標籤。

d. 在升級洞察頁面上，您將看到下列欄位：

- 名稱：Amazon EKS 對叢集執行的檢查。
- 洞察狀態：狀態為「錯誤」的洞察通常意味著受影響的 Kubernetes 版本是當前叢集版本的 N+1，而「警告」狀態意味著該洞察適用於未來 Kubernetes 版本 N+2 或更高。「通過」狀態意味著 Amazon EKS 在此洞察檢查中未發現叢集中存在任何問題。「不明」狀態意味著 Amazon EKS 無法確定叢集是否受到此洞察檢查的影響。
- 版本：洞察檢查對之進行可能問題檢查的 Kubernetes 版本。
- 上次重新整理時間 (UTC-5:00)：上次重新整理此叢集的洞察狀態的時間。
- 上次轉換時間 (UTC-5:00)：此洞察的狀態上次變更的時間。
- 描述：來自洞察檢查的資訊，其中包括提醒和建議的補救措施。

檢視叢集見解 (AWS CLI)

若要檢視 Amazon EKS 叢集的見解，請執行下列動作：

a. 確定您想要檢查哪個叢集以取得洞察。以下命令會列出針對指定叢集的洞察。視需要對命令進行下列修改，然後執行修改後的命令：

- 用相應 AWS 區域代碼取代 *region-code*。
- 使用您叢集的名稱取代 *my-cluster*。

```
aws eks list-insights --region region-code --cluster-name my-cluster
```

範例輸出如下。

```
{
    "insights": [
        {
            "category": "UPGRADE_READINESS",
            "name": "Deprecated APIs removed in Kubernetes v1.29",
            "insightStatus": {
                "status": "PASSING",
                "reason": "No deprecated API usage detected within the last 30 days."
            },
            "kubernetesVersion": "1.29",
            "lastTransitionTime": 1698774710.0,
            "lastRefreshTime": 1700157422.0,
            "id": "123e4567-e89b-42d3-a456-579642341238",
            "description": "Checks for usage of deprecated APIs that are scheduled
 for removal in Kubernetes v1.29. Upgrading your cluster before migrating to the
 updated APIs supported by v1.29 could cause application impact."
        }
    ]
}
```

b. 如需洞察的描述性資訊，請執行以下命令。視需要對命令進行下列修改，然後執行修改後的命令：

- 用相應 AWS 區域代碼取代 *region-code*。
- 用從列出的叢集洞察中擷取到的洞察 ID 取代 *123e4567-e89b-42d3-a456-579642341238*。
- 使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-insight --region region-code --id 123e4567-e89b-42d3-
a456-579642341238 --cluster-name my-cluster
```

範例輸出如下。

```
{
    "insight": {
        "category": "UPGRADE_READINESS",
        "additionalInfo": {
            "EKS update cluster documentation": "https://docs.aws.amazon.com/eks/
latest/userguide/update-cluster.html",
            "Kubernetes v1.29 deprecation guide": "https://kubernetes.io/docs/
reference/using-api/deprecation-guide/#v1-29"
```

```
        },
        "name": "Deprecated APIs removed in Kubernetes v1.29",
        "insightStatus": {
            "status": "PASSING",
            "reason": "No deprecated API usage detected within the last 30 days."
        },
        "kubernetesVersion": "1.29",
        "recommendation": "Update manifests and API clients to use newer Kubernetes
 APIs if applicable before upgrading to Kubernetes v1.29.",
        "lastTransitionTime": 1698774710.0,
        "lastRefreshTime": 1700157422.0,
        "categorySpecificSummary": {
            "deprecationDetails": [
                {
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
flowschemas",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
flowschemas",
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                },
                {
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
prioritylevelconfigurations",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
prioritylevelconfigurations",
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                }
            ]
        },
        "id": "f6a11fe4-77f7-48c6-8326-9a13f022ecb3",
        "resources": [],
        "description": "Checks for usage of deprecated APIs that are scheduled for
 removal in Kubernetes v1.29. Upgrading your cluster before migrating to the updated
 APIs supported by v1.29 could cause application impact."
    }
}
```

更新 Amazon EKS 叢集 Kubernetes 版本

Amazon EKS 中提供新的 Kubernetes 版本時，您可以將 Amazon EKS 叢集更新至最新版本。

> **Important**
>
> 一旦升級叢集，您就無法降級至先前的版本。我們建議您先檢閱 Amazon EKS Kubernetes 版本 中的資訊及檢閱本主題中的更新步驟，再更新至新的 Kubernetes 版本。

新的 Kubernetes 版本有時會加入重大變更。因此，推薦您在執行生產叢集更新之前，先針對新的 Kubernetes 版本測試您應用程式的行為。若要達成此目標，您可以在移至新的 Kubernetes 版本之前，先建置持續的整合工作流程來測試應用程式行為。

更新程序包括 Amazon EKS 啟動新的 API 伺服器節點，並使用更新的 Kubernetes 版本取代現有的節點。Amazon EKS 會針對這些新節點的網路流量執行標準的基礎設施和整備運作狀態檢查，以確認新節點如同預期順利運作。然而，一旦開始叢集升級，您就無法暫停或停止升級。如果檢查有失敗，Amazon EKS 會還原基礎設施部署，之前的 Kubernetes 版本仍然保留您的叢集。執行中的應用程式不會受到影響，您的叢集絕對不會處於非確定性或無法恢復狀態。Amazon EKS 會定期備份所有受管叢集，並且具備在必要時復原叢集的機制。我們會持續評估和改善 Kubernetes 基礎設施管理程序。

若要更新叢集，Amazon EKS 最多需要五個來自子網的可用 IP 地址，子網是在您建立叢集時指定的。Amazon EKS 會在您指定的任何子網中建立新的叢集彈性網路介面（網路介面）。網路介面可能會在與您現有網路介面所在的子網不同的子網中建立，因此請確認您的安全群組規則針對您建立叢集時指定的任何子網，允許所需的叢集通訊 如果您建立叢集時指定的任何子網皆不存在、沒有足夠的可用 IP 地址，或者沒有允許必要的叢集通訊的安全群組規則，則更新可能會失敗。

> **Note**
>
> 為了確保叢集的 API 伺服器端點一律可供存取，Amazon EKS 提供了高可用性的 Kubernetes 控制平面，並在更新作業期間執行 API 伺服器執行個體的滾動式更新。為了說明更改支援您的 Kubernetes API 伺服器端點的 API 伺服器執行個體之 IP 地址，您必須確保您的 API 伺服器用戶端會有效管理重新連線。最新版本的 `kubectl` 和 Kubernetes 用戶端程式庫已正式獲得支援，可透明地執行此重新連線程序。

更新 Amazon EKS 叢集的 Kubernetes 版本

更新叢集的 Kubernetes 版本

1. 將叢集控制平面的 Kubernetes 版本與節點的 Kubernetes 版本進行比較。

   - 取得叢集控制平面的 Kubernetes 版本。

     ```
     kubectl version
     ```

   - 取得節點的 Kubernetes 版本。此命令會傳回所有自我管理和受管 Amazon EC2 和 Fargate 節點。每個 Fargate Pod 會被列為其自己的節點。

     ```
     kubectl get nodes
     ```

   在將控制平面更新為新的 Kubernetes 版本之前，您叢集中的受管節點和 Fargate 節點的 Kubernetes 次要版本，必須要與控制平面目前版本的版本相同。例如，如果您的控制平面正在執行版本，`1.29`而其中一個節點正在執行版本`1.28`，則必須在將控制平面更新為 1.30 `1.29` 之前，將節點更新為版本。我們也推薦您在更新控制平面之前，先將自我管理的節點更新為與控制平面相同的版本。如需更多詳細資訊，請參閱 更新受管節點群組 及 自我管理的節點更新。若 Fargate 節點的次要版本低於控制平面版本，請先刪除該節點表示的 Pod。接著更新您的控制平面。重新部署後，任何剩餘的 Pods 都將更新為新版本。

2. 如果您最初部署叢集時使用的 Kubernetes 版本是 Kubernetes `1.25` 或更新版本，請跳過此步驟。

   根據預設，Amazon EKS 叢集上已啟用 Pod 安全政策許可控制器。在您更新叢集之前，請先確認已具有適當的 Pod 安全政策。這是為了避免可能的安全問題。您可以使用 **`kubectl get psp eks.privileged`** 命令來檢查預設政策。

   ```
   kubectl get psp eks.privileged
   ```

   如果您收到下列錯誤，請參閱Amazon EKS 預設 Pod 安全政策然後再繼續。

   ```
   Error from server (NotFound): podsecuritypolicies.extensions "eks.privileged" not
    found
   ```

3. 如果您最初部署叢集時使用的 Kubernetes 版本是 Kubernetes `1.18` 或更新版本，請跳過此步驟。

您可能需要從 CoreDNS 清單檔案中移除已終止的期限。

a. 檢查您的 CoreDNS 清單檔案是否有一行只有文字 upstream 的行。

```
kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
 grep upstream
```

如果沒有傳回任何輸出，意味著您的清單檔案沒有該行。如果是這種情況，請跳至下一個步驟。如果傳回文字 upstream，請移除該行。

b. 移除 Configmap 檔案中檔案頂部附近只有文字 upstream 的行。不要變更檔案中的任何其他內容。移除該行之後，儲存變更。

```
kubectl edit configmap coredns -n kube-system -o yaml
```

4. 使用eksctl、或更新叢集 AWS CLI。 AWS Management Console

> **Important**
>
> - 如果您正在更新至版本 1.23，並在叢集中使用 Amazon EBS 磁碟區，則必須在叢集中安裝 Amazon EBS CSI 驅動程式，才能將叢集更新為版本 1.23 以避免工作負載中斷。如需詳細資訊，請參閱 Kubernetes1.23 及 Amazon EBS CSI 驅動程式。
> - Kubernetes 1.24 和更新版本使用 containerd 作為預設容器執行期。如果您正切換至 containerd 執行期且已為 Container Insights 設定了 Fluentd，則必須先將 Fluentd 遷移至 Fluent Bit，才能更新叢集。系統會將 Fluentd 剖析器設定為僅剖析 JSON 格式的日誌訊息。與 dockerd 不同的是，containerd 容器執行期具有不是 JSON 格式的日誌訊息。如果您未遷移到 Fluent Bit，則某些已設定的 Fluentd's 剖析器將在 Fluentd 容器內生成大量錯誤。如需移轉的詳細資訊，請參閱設定Fluent Bit為將記錄檔傳送 DaemonSet 至記 CloudWatch 錄檔。
> - 由於 Amazon EKS 執行高可用性的控制平面，因此您一次只能更新一個次要版本。如需此要求的詳細資訊，請參閱 Kubernetes 版本和版本差異支援政策。假設您目前的叢集版本是版本 1.28，並且您想要將其更新為版本 1.30。您必須先將您的版本 1.28 叢集更新為版本 1.29，再將版本 1.29 叢集更新為版本 1.30。
> - 檢查節點上 Kubernetes kube-apiserver 和 kubelet 之間的版本差異。
>   - 從 Kubernetes 版本 1.28 開始，kubelet 最多可能比 kube-apiserver 低三個次要版本。請參閱 Kubernetes upstream version skew policy 一節。

- 如果受管節點和 Fargate 節點上的 kubelet 為 Kubernetes 版本 1.25 或更高版本，則您可以將叢集的版本更新到最多超前三個版本，而無需更新 kubelet 版本。例如，如果 kubelet 的版本是 1.25，在保持 kubelet 的版本為 1.25 的情況下，您可以將您的 Amazon EKS 叢集版本從 1.25 更新為 1.26、1.27 或 1.28
- 如果受管節點和 Fargate 節點上的 kubelet 為 Kubernetes 版本 1.24 或更低版本，則它最多只能比 kube-apiserver 低兩個次要版本。換言之，如果 kubelet 的版本是 1.24 或更低版本，則您最多只能將叢集的版本更新到超前兩個版本。例如，如果 kubelet 的版本是 1.21，在保持 kubelet 的版本為 1.21 的情況下，您可以將您的 Amazon EKS 叢集版本從 1.21 更新為 1.22 或 1.23，而不能更新為 1.24。

- 作為最佳實務，在開始更新之前，請確認節點上的 kubelet 與控制平面的 Kubernetes 版本相同。
- 如果您的叢集設定的 Amazon VPC CNI plugin for Kubernetes 版本早於 1.8.0，建議您先將外掛程式更新為最新版本，然後再更新叢集。若要更新外掛程式，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。
- 如果您要將叢集更新為版本 1.25 或更新版本，並且已在叢集中部署 AWS Load Balancer Controller，請先將控制器更新為版本 2.4.7 或更新版本，然後再將叢集版本更新為 1.25。如需詳細資訊，請參閱 Kubernetes1.25 版本備註。

eksctl

此程序需要 eksctl 版本 0.183.0 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝和更新 eksctl 的指示，請參閱 eksctl 文件中的 Installation 一節。

更新 Amazon EKS 控制平面的 Kubernetes 版本。使用您的叢集名稱取代 *my-cluster*。將 *1.30* 取代為您要將叢集更新到的目標 Amazon EKS 支援的版本號碼。如需支援的版本編號清單，請參閱 Amazon EKS Kubernetes 版本。

```
eksctl upgrade cluster --name my-cluster --version 1.30 --approve
```

此更新需要幾分鐘的時間來完成。

AWS Management Console

a. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
b. 選擇要更新的 Amazon EKS 叢集的名稱，並選擇 Update cluster version (更新叢集版本)。
c. 針對 Kubernetes 版本，選取叢集要更新的版本並選擇 Update (更新)。
d. 針對 Cluster name (叢集名稱)，輸入叢集的名稱並選擇 Confirm (確認)。

   此更新需要幾分鐘的時間來完成。

AWS CLI

a. 透過以下 AWS CLI 命令更新 Amazon EKS 叢集。使用自己的取代 *example values*。將 *1.30* 取代為您要將叢集更新到的目標 Amazon EKS 支援的版本號碼。如需支援的版本編號清單，請參閱 Amazon EKS Kubernetes 版本。

```
aws eks update-cluster-version --region region-code --name my-cluster --
kubernetes-version 1.30
```

   範例輸出如下。

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.30"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
```

```
}
```

b. 使用下列命令來監控叢集更新的狀態。使用先前命令傳回的叢集名稱和更新 ID。顯示 `Successful` 狀態時，即表示更新已完成。此更新需要幾分鐘的時間來完成。

```
aws eks describe-update --region region-code --name my-cluster --update-
id b5f0ba18-9a87-4450-b5a0-825e6e84496f
```

範例輸出如下。

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "Successful",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.30"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
}
```

5. 叢集更新完成之後，請將您的節點更新至與已更新叢集相同的 Kubernetes 最低版本。如需詳細資訊，請參閱 自我管理的節點更新 及 更新受管節點群組。在 Fargate 上啟動的任何新 Pods 都會有符合您叢集版本的 `kubelet` 版本。現有的 Fargate Pods 不會變更。

6. (選用) 如果您在更新叢集之前已將 Kubernetes Cluster Autoscaler 部署至叢集，則請將 Cluster Autoscaler 更新為符合您最後更新之 Kubernetes 主要和次要版本的最新版本。

   a. 在網頁瀏覽器中開啟 Cluster Autoscaler 版本頁面，並尋找符合叢集 Kubernetes 主要和次要版本的最新 Cluster Autoscaler 版本。例如，如果叢集的 Kubernetes 版本是 `1.30`，請尋找開頭為 `1.30` 的 Cluster Autoscaler 最新版本。記錄該版本的語意版本編號 (例如，`1.30.n`)，以在下一步中使用。

b. 使用下列命令，將 Cluster Autoscaler 映像標籤設定為您在前一個步驟中記錄的版本。如有必要，請使用您自己的值取代 *1.30.n*。

```
kubectl -n kube-system set image deployment.apps/cluster-autoscaler cluster-
autoscaler=registry.k8s.io/autoscaling/cluster-autoscaler:v1.30.n
```

7. (僅限含 GPU 節點的叢集) 如果您的叢集擁有含 GPU 支援的節點群組 (例如，p3.2xlarge)，則必須在叢集上更新 Kubernetes 專用 NVIDIA 裝置外掛程式的 DaemonSet。請先以您想要的 NVIDIA/k8s-device-plugin 版本來取代 *vX.X.X*，再執行下列命令。

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
nvidia-device-plugin.yml
```

8. 更新 Amazon VPC CNI plugin for Kubernetes、CoreDNS 以及 kube-proxy 附加元件。建議您將附加元件更新為服務帳戶字符中列出的最低版本。

   - 如果您正在使用 Amazon EKS 附加元件，請在 Amazon EKS 主控台中選取 Clusters (叢集)，然後在左導覽窗格中選取您已更新的叢集名稱。通知會顯示在主控台。它們會通知您每個有可用更新的附加元件有新的可用版本。若要更新附加元件，請選擇 Add-ons (附加元件) 標籤。在具有可用更新之附加元件的其中一個方塊中，選取 Update now (立即更新)，選取可用的版本，然後選取 Update (更新)。
   - 或者，您也可以使用 AWS CLI 或eksctl更新附加元件。如需詳細資訊，請參閱 更新附加元件。

9. 如有必要，請更新您的 kubectl 版本。您所使用的 kubectl 版本，必須與 Amazon EKS 叢集控制平面的版本差距在一個版本以內。例如，1.29 kubectl 用戶端搭配 Kubernetes、1.28、1.29 和 1.30 叢集運作。您可以使用下列命令檢查目前安裝的版本。

```
kubectl version --client
```

刪除 Amazon EKS 叢集

當您完成使用 Amazon EKS 叢集後，應刪除與之相關聯的資源，才不會產生任何不必要的成本。

若要移除連接的叢集，請參閱 取消註冊叢集

> **Important**
>
> - 如果您的叢集中有使用中的服務與負載平衡器相關，必須在刪除叢集前先刪除這些服務，才能夠正確刪除負載平衡器。否則，您可以孤立 VPC 中的資源，以防止自己刪除 VPC。
> - 如果由於已移除叢集建立者而收到錯誤，請參閱此篇文章來解決問題。
> - Prometheus 資源的 Amazon 受管服務不在叢集生命週期之外，需要獨立於叢集進行維護。刪除叢集時，請務必同時刪除任何適用的抓取工具，以停止適用的成本。如需詳細資訊，請參閱 Amazon Prometheus 受管服務使用者指南中的尋找和刪除抓取工具。

您可以使用eksctl AWS Management Console、或刪除叢集 AWS CLI。

eksctl

若要使用 **eksctl** 刪除 Amazon EKS 叢集與節點

此程序需要 eksctl 版本 0.183.0 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的 Installation 一節。

1. 列出所有在叢集中執行的服務。

   ```
   kubectl get svc --all-namespaces
   ```

2. 刪除任何與 EXTERNAL-IP 值相關的服務。這些服務皆是由 Elastic Load Balancing 負載平衡器做為前端，且您必須在 Kubernetes 中刪除它們，以允許正確釋出負載平衡器與相關資源。

   ```
   kubectl delete svc service-name
   ```

3. 使用下列命令，將 *prod* 取代為您的叢集名稱，來刪除叢集及其關聯的節點。

   ```
   eksctl delete cluster --name prod
   ```

   輸出：

   ```
   [#]  using region region-code
   ```

```
[#]  deleting EKS cluster "prod"
[#]  will delete stack "eksctl-prod-nodegroup-standard-nodes"
[#]  waiting for stack "eksctl-prod-nodegroup-standard-nodes" to get deleted
[#]  will delete stack "eksctl-prod-cluster"
[#]  the following EKS cluster resource(s) for "prod" will be deleted: cluster.
 If in doubt, check CloudFormation console
```

AWS Management Console

若要 Amazon 除具有 AWS Management Console

1. 列出所有在叢集中執行的服務。

```
kubectl get svc --all-namespaces
```

2. 刪除任何與 EXTERNAL-IP 值相關的服務。這些服務皆是由 Elastic Load Balancing 負載平衡器做為前端，且您必須在 Kubernetes 中刪除它們，以允許正確釋出負載平衡器與相關資源。

```
kubectl delete svc service-name
```

3. 刪除所有節點群組和 Fargate 描述檔。

   a. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

   b. 在左側導覽窗格中，選擇 Amazon EKS Clusters (叢集)，然後在叢集的標籤式清單中，選擇您要刪除的叢集名稱。

   c. 選擇 Compute (運算) 索引標籤，然後選取要刪除的節點群組。選擇 Delete (刪除)，輸入節點群組的名稱，然後選擇 Delete (刪除)。刪除叢集中的所有節點群組。

   > **Note**
   >
   > 列出的節點群組僅為受管節點群組。

   d. 選擇要刪除的 Fargate Profile (Fargate 描述檔)，選取 Delete (刪除)，輸入描述檔的名稱，然後選擇 Delete (刪除)。刪除叢集中的所有 Fargate 描述檔。

4. 刪除所有自我管理的節點 AWS CloudFormation 堆疊。

   a. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation

b. 選擇要刪除的節點堆疊，然後選擇 Delete (刪除)。

c. 在 Delete stack (刪除堆疊) 確認對話方塊中，選擇 Delete stack (刪除堆疊)。刪除叢集中的所有自我管理節點堆疊。

5. 刪除叢集。

a. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

b. 選擇要刪除的叢集並選擇 Delete (刪除)。

c. 在刪除叢集確認畫面上，選擇 Delete (刪除)。

6. (選擇性) 刪除 VPC AWS CloudFormation 堆疊。

a. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation

b. 選取要刪除的 VPC 堆疊，然後選擇 Delete (刪除)。

c. 在 Delete stack (s刪除堆疊) 確認對話方塊中，選擇 Delete stack (刪除堆疊)。

AWS CLI

若要 Amazon 除具有 AWS CLI

1. 列出所有在叢集中執行的服務。

```
kubectl get svc --all-namespaces
```

2. 刪除任何與 EXTERNAL-IP 值相關的服務。這些服務皆是由 Elastic Load Balancing 負載平衡器做為前端，且您必須在 Kubernetes 中刪除它們，以允許正確釋出負載平衡器與相關資源。

```
kubectl delete svc service-name
```

3. 刪除所有節點群組和 Fargate 描述檔。

a. 使用下列命令列出叢集中的節點群組。

```
aws eks list-nodegroups --cluster-name my-cluster
```

> **Note**
>
> 列出的節點群組僅為受管節點群組。

b. 使用下列命令來刪除每個節點群組。刪除叢集中的所有節點群組。

```
aws eks delete-nodegroup --nodegroup-name my-nodegroup --cluster-name my-cluster
```

c. 使用下列命令列出您叢集中的 Fargate 描述檔。

```
aws eks list-fargate-profiles --cluster-name my-cluster
```

d. 使用下列命令來刪除每個 Fargate 描述檔。刪除叢集中的所有 Fargate 描述檔。

```
aws eks delete-fargate-profile --fargate-profile-name my-fargate-profile --cluster-name my-cluster
```

4. 刪除所有自我管理的節點 AWS CloudFormation 堆疊。

a. 使用以下命令列出可用的 AWS CloudFormation 堆棧。在產生輸出中尋找節點範本名稱。

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

b. 使用以下命令，將 *node-stack* 取代為您節點堆疊的名稱，來刪除工作節點堆疊。刪除叢集中的所有自我管理節點堆疊。

```
aws cloudformation delete-stack --stack-name node-stack
```

5. 使用以下命令，將 *my-cluster* 取代為您的叢集名稱，來刪除叢集。

```
aws eks delete-cluster --name my-cluster
```

6. (選擇性) 刪除 VPC AWS CloudFormation 堆疊。

a. 使用以下命令列出可用的 AWS CloudFormation 堆棧。在產生輸出中尋找 VPC 範本名稱。

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

b. 使用以下命令，將 *my-vpc-stack* 取代為您的 VPC 堆疊名稱，以刪除 VPC 堆疊。

```
aws cloudformation delete-stack --stack-name my-vpc-stack
```

Amazon EKS 叢集端點存取控制

本主題可協助您啟用 Amazon EKS 叢集 Kubernetes API 伺服器端點和限制的私有存取，或完全停用從網際網路的公有存取。

建立新的叢集時，Amazon EKS 會為您用來與叢集通訊的受管 Kubernetes API 伺服器建立端點 (使用 Kubernetes 管理工具，例如 `kubectl`)。預設情況下，此 API 伺服器端點對網際網路是公開的，而 API 伺服器的存取是使用 AWS Identity and Access Management (IAM) 和原生Kubernetes角色型存取控制 (RBAC) 的組合來保護對 API 伺服器的存取。

您可啟用 Kubernetes API 伺服器的私有存取，讓節點和 API 伺服器間的所有通訊都不會離開 VPC。您可以限制可以從網際網路存取 API 伺服器的 IP 地址，或完全停用對 API 伺服器的網際網路存取。

> **Note**
>
> 由於此端點適用於 Kubernetes API 伺服器，而不是用於與 AWS API 通訊的傳統 AWS PrivateLink 端點，因此在 Amazon VPC 主控台中不會顯示為端點。

啟用叢集的端點私有存取時，Amazon EKS 會代表您建立 Route 53 私有託管區域，並將其與您叢集的 VPC 建立關聯。此私有託管區域由 Amazon EKS 管理，不會顯示在帳戶的 Route 53 資源中。若要讓私有託管區域正確將流量路由到 API 伺服器，您的 VPC 必須將 `enableDnsHostnames` 和 `enableDnsSupport` 設為 `true`，且 VPC 設定的 DHCP 選項必須在網域名稱伺服器清單中包含 `AmazonProvidedDNS`。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的更新 VPC 的 DNS 支援。

建立新的叢集時，可定義您 API 伺服器端點的存取要求，您也可隨時更新叢集的 API 伺服器端點存取。

修改叢集端點存取

使用本節所述程序來修改現有叢集的端點存取。下表說明支援的 API 伺服器端點存取組合及其相關的行為。

API 伺服器端點存取選項

端點公有存取	端點私有存取	Behavior (行為)
已啟用	已停用	• 這是新 Amazon EKS 叢集的預設行為。 • 來自您叢集 VPC 內 (如節點與控制平面間的通訊) 的 Kubernetes API 請求會離開 VPC，但不會離開 Amazon 的網路。 • 您的叢集 API 伺服器可從網際網路上存取。您可以選擇性地限制可存取公有端點的 CIDR 區塊。如果您限制對特定 CIDR 區塊的存取權，則建議您同時啟用私有端點，或確保您指定的 CIDR 區塊包含節點和 Fargate Pods (如果您使用它們) 用來存取公有端點的地址。
已啟用	已啟用	• 來自您叢集 VPC 內 (如節點與控制平面間的通訊) 的 Kubernetes API 請求會使用私有 VPC 端點。 • 您的叢集 API 伺服器可從網際網路上存取。您可以選擇性地限制可存取公有端點的 CIDR 區塊。
已停用	已啟用	• 前往您叢集 API 伺服器的所有流量必須來自叢集的 VPC 或連接的網路。 • 您的 API 伺服器無法從網際網路上公有存取。任何 `kubectl` 命令都必須來自

端點公有存取	端點私有存取	Behavior (行為)
		VPC 或連接的網路內。如需連接選項，請參閱 存取僅限私有 API 伺服器。 • 公有 DNS 伺服器會將叢集的 API 伺服器端點解析為 VPC 的私有 IP 地址。在過去，端點只能從 VPC 內解析。 如果您的端點未針對現有叢集解析為 VPC 內的私有 IP 地址，您可以： • 啟用公有存取，然後再次停用它。您只需對叢集執行一次，從此以後，端點就會解析為私有 IP 地址。 • 更新您的叢集。

您可以使用或修改叢集 API 伺服器端點存 AWS Management Console 取 AWS CLI。

AWS Management Console

若要修改叢集 API 伺服器端點存取 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇叢集名稱以顯示您叢集的資訊。
3. 選擇 Networking (聯網) 索引標籤，選擇 Update (更新)。
4. 對於 Private access (私有存取)，選擇是否啟用或停用您叢集的 Kubernetes API 伺服器端點的私有存取。如果您啟用私有存取，源自於您叢集的 VPC 的 Kubernetes API 將使用私有 VPC 端點。您必須啟用私有存取，才能停用公有存取。
5. 對於 Public access (公有存取)，選擇是否啟用或停用您叢集的 Kubernetes API 伺服器端點的公有存取。若您停用公有存取，您叢集的 Kubernetes API 伺服器僅可接收來自叢集 VPC 內的請求。

6. (選用) 如果您已啟用 Public access (公開存取)，則可以指定網際網路可用來與公有端點通訊的地址。選取 Advanced Settings (進階設定)。輸入 CIDR 區塊，例如 *203.0.113.5/32*。區塊不能包含預留地址。您可以透過選取 Add source (新增來源) 來輸入其他區塊。您可以指定的 CIDR 區塊有數量上限。如需詳細資訊，請參閱 Amazon EKS 服務配額。如果不指定區塊，則公有 API 伺服器端點會接收來自所有 (`0.0.0.0/0`) IP 地址的要求。如果您使用 CIDR 區塊限制對公有端點的存取，則建議您也啟用私有端點存取，以便節點和 Fargate Pods (如果您使用它們) 可以與叢集通訊。若不啟用私有端點，您的公有存取端點 CIDR 來源必須包含來自 VPC 的輸出來源。例如，如果您的私有子網路中有節點，透過 NAT 閘道與網際網路通訊，則您必須將 NAT 閘道的對外 IP 地址新增至公有端點上的白名單 CIDR 區塊。
7. 選擇 Update (更新) 以完成操作。

AWS CLI

運用 AWS CLI修改您叢集 API 伺服器端點的存取

使用 AWS CLI 版本`1.27.160`或更新版本完成下列步驟。您可以使用 `aws --version` 來檢查您的目前版本。若要安裝或升級 AWS CLI，請參閱安裝 AWS CLI.

1. 透過下列 AWS CLI 命令，更新叢集 API 伺服器端點的存取。替換叢集名稱和所需的端點存取值。如果您設定了 `endpointPublicAccess=true`，則可以 (選擇性地) 為 `publicAccessCidrs` 輸入單一 CIDR 區塊或以逗號分隔的 CIDR 區塊清單。區塊不能包含預留地址。如果您指定 CIDR 區塊，則公有 API 伺服器端點只會接收來自所列出區塊的要求。您可以指定的 CIDR 區塊有數量上限。如需詳細資訊，請參閱 Amazon EKS 服務配額。如果您使用 CIDR 區塊限制對公有端點的存取，則建議您也啟用私有端點存取，以便節點和 Fargate Pods (如果您使用它們) 可以與叢集通訊。若不啟用私有端點，您的公有存取端點 CIDR 來源必須包含來自 VPC 的輸出來源。例如，如果您的私有子網路中有節點，透過 NAT 閘道與網際網路通訊，則您必須將 NAT 閘道的對外 IP 地址新增至公有端點上的白名單 CIDR 區塊。如果不指定 CIDR 區塊，則公有 API 伺服器端點會接收來自所有 (0.0.0.0/0) IP 地址的要求。

   > **Note**
   >
   > 下列命令會啟用 API 伺服器端點的私有存取以及來自單一 IP 地址的公有存取。使用單一 CIDR 區塊或您想要限制網路存取的 CIDR 區塊清單取代 *203.0.113.5/32*。

```
aws eks update-cluster-config \
    --region region-code \
```

```
    --name my-cluster \
    --resources-vpc-config
 endpointPublicAccess=true,publicAccessCidrs="203.0.113.5/32",endpointPrivateAccess=true
```

範例輸出如下。

```
{
    "update": {
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "InProgress",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\203.0.113.5/32\"]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
    }
}
```

2. 使用叢集名稱和前述命令傳回的更新 ID，藉由以下命令監控端點存取的更新狀態，當狀態顯示為 Successful，您的更新就完成了。

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000
```

範例輸出如下。

```
{
    "update": {
```

```
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "Successful",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\203.0.113.5/32\]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
    }
}
```

存取僅限私有 API 伺服器

若您已停用叢集 Kubernetes API 伺服器端點的公有存取，將只能在 VPC 內或連接的網路存取 API 伺服器。以下是存取 Kubernetes API 伺服器端點的幾種可能方法：

連線網路

使用 AWS 傳輸閘道或其他連線選項，將您的網路連線到 VPC，然後使用連線網路中的電腦。您必須確認 Amazon EKS 控制平面安全群組包含的規則允許連接埠 443 上來自連接網路的傳入流量。

Amazon EC2 堡壘主機

您可以在叢集 VPC 中的公有子網路啟動 Amazon EC2 執行個體，然後透過 SSH 登入該執行個體，以執行 `kubectl` 命令。如需詳細資訊，請參閱 AWS上的 Linux 堡壘主機。您必須確認 Amazon EKS 控制平面安全群組包含的規則允許連接埠 443 上來自堡壘主機的傳入流量。如需詳細資訊，請參閱 Amazon EKS 安全群組與考量。

設定堡壘主機的 `kubectl` 時，請確認您使用的 AWS 憑證已映射至叢集的 RBAC 組態，或者將堡壘會使用的 IAM 主體新增至 RBAC 組態，之後再移除端點的公有存取。如需詳細資訊，請參閱 the section called “授與庫伯內特 API 的存取權” 及 未經授權或存取遭拒 (`kubectl`)。

AWS Cloud9 IDE

AWS Cloud9 是雲端整合式開發環境 (IDE)，讓您只要使用瀏覽器即可撰寫、執行和偵錯程式碼。您可以在叢集的 VPC 中建立 AWS Cloud9 IDE，並使用 IDE 與叢集進行通訊。如需詳細資訊，請參閱在 AWS Cloud9中建立環境。您必須確認 Amazon EKS 控制平面安全群組包含的規則允許連接埠 443 上來自 IDE 安全群組的傳入流量。如需詳細資訊，請參閱 Amazon EKS 安全群組與考量。

`kubectl`為 AWS Cloud9 IDE 設定時，請務必使用已對應至叢集 RBAC 組態的 AWS 認證，或在移除端點公用存取權之前，將 IDE 將使用的 IAM 主體新增至 RBAC 組態。如需更多詳細資訊，請參閱 授予 Kubernetes API 的存取權 及 未經授權或存取遭拒 (`kubectl`)。

在現有叢集上啟用密碼加密

若您啟用密碼加密，系統會使用您選取的 AWS KMS key 加密 Kubernetes 密碼。KMS 金鑰必須滿足以下條件：

- 對稱
- 可加密和解密資料
- 在與叢集相同的 AWS 區域 中建立
- 如果 KMS 金鑰是在其他帳戶中建立，則 IAM p主體必須具有 KMS 金鑰的存取權。

如需詳細資訊，請參閱 AWS Key Management Service 開發人員指南中的允許其他帳戶中的 IAM 主體使用 KMS 金鑰。

> **Warning**
>
> 啟用密碼加密後，您無法予以停用。此動作不可復原。

eksctl

您可以在以兩種方式啟用加密：

- 使用單一命令將加密新增至叢集。

  若要自動重新加密您的秘密，請執行下列命令。

```
eksctl utils enable-secrets-encryption \
```

```
    --cluster my-cluster \
    --key-arn arn:aws:kms:region-code:account:key/key
```

若要退出自動重新加密您的密碼，請執行下列命令。

```
eksctl utils enable-secrets-encryption
    --cluster my-cluster \
    --key-arn arn:aws:kms:region-code:account:key/key \
    --encrypt-existing-secrets=false
```

- 使用 `kms-cluster.yaml` 檔案將加密新增到您的叢集中。

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code

secretsEncryption:
  keyARN: arn:aws:kms:region-code:account:key/key
```

若要讓您的秘密自動重新加密，請執行下列命令。

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml
```

若要退出自動重新加密您的密碼，請執行下列命令。

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml --encrypt-existing-
secrets=false
```

AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇您要向其新增 KMS 加密的叢集。
3. 選擇 Overview (概觀) 索引標籤 (預設選取此項)。
4. 向下捲動到 Secrets encryption (私密加密) 區段，然後選擇 Enable (啟用)。

5. 從下拉式清單中選取金鑰，然後選擇 Enable (啟用) 按鈕。如果未列出任何金鑰，您必須先建立一個。如需詳細資訊，請參閱建立金鑰。
6. 選擇 Confirm (確認) 按鈕以使用選擇的金鑰。

AWS CLI

1. 使用下列 AWS CLI 命令，將私密加密組態與叢集建立關聯。使用自己的取代 *example values*。

```
aws eks associate-encryption-config \
    --cluster-name my-cluster \
    --encryption-config '[{"resources":["secrets"],"provider":
{"keyArn":"arn:aws:kms:region-code:account:key/key"}}]'
```

範例輸出如下。

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "InProgress",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734,
    "errors": []
  }
}
```

2. 您可以使用下列命令來監控您的加密更新的狀態。使用之前輸出傳回的特定 `cluster name` 和 `update ID`。顯示 `Successful` 狀態時，即表示更新已完成。

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id 3141b835-8103-423a-8e68-12c2521ffa4d
```

範例輸出如下。

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "Successful",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734>,
    "errors": []
  }
}
```

3. 若要驗證您的叢集中是否已啟用加密，請執行 describe-cluster 命令。回應包含 EncryptionConfig 字串。

```
aws eks describe-cluster --region region-code --name my-cluster
```

在叢集上啟用加密之後，您須使用新金鑰來加密所有現有的密碼：

> **Note**
>
> 如果您使用 eksctl，則僅當您選擇退出自動重新加密秘密時，才需要執行以下命令。

```
kubectl get secrets --all-namespaces -o json | kubectl annotate --overwrite -f - kms-
encryption-timestamp="time value"
```

> **Warning**
>
> 如果針對現有叢集啟用密碼加密，且使用的 KMS 金鑰會被刪除，則沒有復原叢集的路徑。若刪除 KMS 金鑰，即會永久使叢集處於降級狀態。如需詳細資訊，請參閱刪除 AWS KMS 金鑰。

> **Note**
>
> 根據預設，`create-key` 命令會建立具有金鑰政策的對稱加密 KMS 金鑰，該政策賦予該帳戶對 AWS KMS 動作和資源的根管理員存取權。如果您想要縮小許可範圍，請確保要呼叫 `create-cluster` API 之主體的政策上允許 `kms:DescribeKey` 和 `kms:CreateGrant` 動作。
>
> 對於使用 KMS 封套加密的叢集，需要 `kms:CreateGrant` 許可。此 CreateCluster動作`kms:GrantIsForAWSResource`不支援此條件，且不應在 KMS 原則中使用此條件來控制執行使用者的`kms:CreateGrant`權限 CreateCluster。

為您的 Amazon EKS 叢集啟用 Windows 支援

部署 Windows 節點之前，請注意以下考量。

考量事項

- 您可以透過 `HostProcess` Pod 在 Windows 節點上使用主機聯網。如需詳細資訊，請參閱 Kubernetes 文件中的建立 Windows `HostProcess`Pod。
- Amazon EKS 叢集必須包含一或多個 Linux 或 Fargate 節點，才能執行只在 Linux 上執行的核心系統 Pods，例如：CoreDNS。
- `kubelet` 和 `kube-proxy` 事件日誌會重新導向至 EKS Windows 事件日誌，且其限制會設定為 200 MB。
- 您無法將 Pods 的安全群組 和在 Windows 節點上執行的 Pods 搭配使用。
- 您無法將自訂聯網與 Windows 節點搭配使用。
- 您無法將 `IPv6` 與 Windows 節點搭配使用。
- Windows 節點為每個節點支援一個彈性網路介面。依預設，每個 Windows 節點可以執行的 Pods 數量，等於節點執行個體類型的每個彈性網路介面可用的 IP 地址數量減去一。如需詳細資訊，請參閱 Amazon EC2 使用者指南中每個執行個體類型的每個網路界面的 IP 地址。

- 在 Amazon EKS 叢集中，具有負載平衡器的單一服務最多可支援 1024 個後端 Pods。每個 Pod 都有自己的唯一 IP 地址。自 OS Build 17763.2746 起的 Windows Server 更新後，過去的 64 個 Pods 限制已不存在。
- Windows 容器不支援在 Fargate 上使用 Amazon EKS Pods。
- 無法從 `vpc-resource-controller` Pod 擷取日誌。您先前在將控制器部署到資料平面時可以這麼做。
- 在將 IPv4 地址指派給新的 Pod 之前，有一段冷卻期間。這可以防止流量因 `kube-proxy` 規則過時而流動到具有相同 IPv4 地址的較舊 Pod。
- 在 GitHub 上可管理控制器的來源。若要對控制器做出貢獻或提出問題，請造訪 GitHub 上的專案。
- 為Windows受管節點群組指定自訂 AMI ID 時，請新增`eks:kube-proxy-windows`至 AWS IAM 驗證器組態對應。如需詳細資訊，請參閱 指定 AMI ID 時的限制和條件。

必要條件

- 現有的叢集。叢集必須執行下表所列的其中一種 Kubernetes 版本和平台版本。所列版本之後推出的所有 Kubernetes 和平台版本亦受支援。若叢集或平台版本比下列其中一個版本還舊，則需要在叢集的資料平面上啟用舊版 Windows 支援。當叢集為以下 Kubernetes 和平台版本之一或更新的版本時，您可以在控制平面上移除舊版 Windows 支援和啟用 Windows 支援。

Kubernetes 版本	平台版本
1.30	eks.2
1.29	eks.1
1.28	eks.1
1.27	eks.1
1.26	eks.1
1.25	eks.1
1.24	eks.2

- 叢集必須至少有一個 (建議至少有兩個) Linux 節點或 Fargate Pod，才能執行 CoreDNS。若啟用舊版 Windows 支援，則必須使用 Linux 節點 (不能使用 Fargate Pod) 來執行 CoreDNS。

- 現有 Amazon EKS 叢集 IAM 角色。

啟用 Windows 支援

若叢集不屬於先決條件中所列出的 Kubernetes 與平台版本其中的一個版本或更新的版本，則必須改為啟用舊版 Windows 支援。如需詳細資訊，請參閱 啟用舊版 Windows 支援。

若您從未在叢集上啟用 Windows 支援，請跳至下一個步驟。

若在早於先決條件中所列 Kubernetes 或平台版本的叢集上啟用 Windows 支援，則必須先從資料平面中移除 vpc-resource-controller 和 vpc-admission-webhook。其將遭取代，且未來不再需要。

為叢集啟用 Windows 支援

1. 若叢集中沒有 Amazon Linux 節點，並且使用 Pods 的安全群組，請跳至下一個步驟。否則，請確認 AmazonEKSVPCResourceController 受管政策會連接至叢集角色。使用您的叢集角色名稱取代 *eksClusterRole*。

   ```
   aws iam list-attached-role-policies --role-name eksClusterRole
   ```

   範例輸出如下。

   ```
   {
       "AttachedPolicies": [
           {
               "PolicyName": "AmazonEKSClusterPolicy",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSClusterPolicy"
           },
           {
               "PolicyName": "AmazonEKSVPCResourceController",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSVPCResourceController"
           }
       ]
   }
   ```

   如果已連接政策 (如上一個輸出所示)，請略過下一個步驟。

2. 將亞馬遜 VPC ResourceController 託管策略附加到您的. Amazon EKS 叢集 IAM 角色 使用您的叢集角色名稱取代 *eksClusterRole*。

```
aws iam attach-role-policy \
  --role-name eksClusterRole \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
```

3. 使用下列內容建立名為 *vpc-resource-controller-configmap.yaml* 的檔案。

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: amazon-vpc-cni
  namespace: kube-system
data:
  enable-windows-ipam: "true"
```

4. 將 ConfigMap 套用至您的叢集。

```
kubectl apply -f vpc-resource-controller-configmap.yaml
```

5. 確認您的 aws-auth ConfigMap 包含 Windows 節點執行個體角色的對應，以包含 eks:kube-proxy-windows RBAC 許可群組。您可以透過執行以下命令來驗證。

```
kubectl get configmap aws-auth -n kube-system -o yaml
```

範例輸出如下。

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      - eks:kube-proxy-windows # This group is required for Windows DNS resolution
 to work
      rolearn: arn:aws:iam::111122223333:role/eksNodeRole
      username: system:node:{{EC2PrivateDNSName}}
[...]
```

您應該會在群組下看到 eks:kube-proxy-windows 列出。如果未指定群組，您需要更新 ConfigMap 或建立群組以包含所需的群組。如需 aws-auth ConfigMap 的詳細資訊，請參閱 將 aws-authConfigMap 套用至您的叢集。

從資料平面中移除舊版 Windows 支援

若在早於先決條件中所列 Kubernetes 或平台版本的叢集上啟用 Windows 支援，則必須先從資料平面中移除 vpc-resource-controller 和 vpc-admission-webhook。其將遭取代，且未來不再需要，因為現已在控制平面上啟用其所提供之功能。

1. 可使用以下命令解除安裝 vpc-resource-controller。不論最初使用哪個工具進行安裝，都請使用這個命令。使用叢集所在的 AWS 區域 取代 *region-code* (只有在 /manifests/ 之後的該文字其執行個體)。

```
kubectl delete -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
```

2. 按照安裝時所用工具的說明來解除安裝 vpc-admission-webhook。

   eksctl

   執行下列命令。

```
kubectl delete deployment -n kube-system vpc-admission-webhook
kubectl delete service -n kube-system vpc-admission-webhook
kubectl delete mutatingwebhookconfigurations.admissionregistration.k8s.io vpc-admission-webhook-cfg
```

   kubectl on macOS or Windows

   執行下列命令。將*region-code*（僅在之後的文本實例/manifests/）替換為您 AWS 區域的集群所在的位置。

```
kubectl delete -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml
```

3. 為在控制平面上的叢集啟用 Windows 支援。

停用 Windows 支援

停用叢集上的 Windows 支援

1. 若叢集包含 Amazon Linux 節點，並且搭配使用 Pods 的安全群組，則請跳過這個步驟。

   從叢集角色中移除 AmazonVPCResourceController 受管 IAM 政策。使用您的叢集角色名稱取代 *eksClusterRole*，再以帳戶 ID 取代 *111122223333*。

   ```
   aws iam detach-role-policy \
       --role-name eksClusterRole \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
   ```

2. 在中停用 Windows IPAM。amazon-vpc-cni ConfigMap

   ```
   kubectl patch configmap/amazon-vpc-cni \
                       -n kube-system \
                       --type merge \
                       -p '{"data":{"enable-windows-ipam":"false"}}'
   ```

部署 Pod

在將 Pod 部署到叢集時，您需要指定在執行混合節點類型時所使用的作業系統。

對於 Linux Pods，請在清單檔案中使用以下節點選取器文字。

```
nodeSelector:
        kubernetes.io/os: linux
        kubernetes.io/arch: amd64
```

對於 Windows Pods，請在清單檔案中使用以下節點選取器文字。

```
nodeSelector:
        kubernetes.io/os: windows
        kubernetes.io/arch: amd64
```

您可以部署範例應用程式來查看正在使用的節點選擇器。

啟用舊版 Windows 支援

若叢集不屬於先決條件所列出的 Kubernetes 與平台版本其中的一個版本或更新的版本，則我們建議您改為在控制平面上啟用 Windows 支援。如需詳細資訊，請參閱 啟用 Windows 支援。

若叢集或平台版本早於先決條件所列的版本時，您可以參考以下步驟以協助您為 Amazon EKS 叢集的資料平面啟用舊版 Windows 支援。當叢集和平台版本為先決條件所列的版本或更新的版本時，我們建議您移除舊版 Windows 支援並為控制平面啟用 Windows 支援。

您可以使用 `eksctl`、Windows 用戶端、macOS 或 Linux 用戶端來為叢集啟用舊版 Windows 支援。

eksctl

為搭配使用 **`eksctl`** 的叢集啟用舊版 Windows 支援

先決條件

此程序需要 `eksctl` 版本 `0.183.0` 或更新版本。您可使用以下命令檢查您的版本。

```
eksctl version
```

如需安裝或升級 `eksctl` 的詳細資訊，請參閱 `eksctl` 文件中的 Installation 一節。

1. 使用以下 `eksctl` 命令以為 Amazon EKS 叢集啟用 Windows 支援。使用您叢集的名稱取代 *`my-cluster`*。此命令會在 Amazon EKS 叢集上部署執行 Windows 工作負載所需的 VPC 資源控制器和 VPC 許可控制器 Webhook。

   ```
   eksctl utils install-vpc-controllers --cluster my-cluster --approve
   ```

   > **Important**
   >
   > VPC 許可控制器 Webhook 使用憑證簽署，該憑證在發行日期後一年到期。若要避免停機時間，請務必在憑證到期之前續約憑證。如需詳細資訊，請參閱 續約 VPC 許可 Webhook 憑證。

2. 啟用 Windows 支援以後，您可以在叢集中啟動 Windows 節點群組。如需詳細資訊，請參閱 正在啟動自我管理的 Windows 節點。

Windows

使用 Windows 用戶端為叢集啟用舊版 Windows 支援

在以下步驟中，以您叢集所在的 AWS 區域 取代 *region-code*。

1. 將 VPC 資源控制器部署到您的叢集。

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
```

2. 將 VPC 許可控制器 Webhook 部署到您的叢集。

   a. 下載必要的指令碼和部署檔案。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/Setup-VPCAdmissionWebhook.ps1;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.ps1;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-patch-ca-bundle.ps1;
```

   b. 安裝 OpenSSL 和 jq。

   c. 設定和部署 VPC 許可 Webhook。

```
./Setup-VPCAdmissionWebhook.ps1 -DeploymentTemplate ".\vpc-admission-webhook-deployment.yaml"
```

> **Important**
>
> VPC 許可控制器 Webhook 使用憑證簽署，該憑證在發行日期後一年到期。若要避免停機時間，請務必在憑證到期之前續約憑證。如需詳細資訊，請參閱 續約 VPC 許可 Webhook 憑證。

3. 判斷您的叢集是否具有必要的叢集角色繫結。

```
kubectl get clusterrolebinding eks:kube-proxy-windows
```

如果傳回類似下列範例輸出的輸出，則叢集具有必要的角色繫結。

```
NAME                        AGE
eks:kube-proxy-windows      10d
```

如果輸出包含 `Error from server (NotFound)`，則叢集沒有必要的叢集角色繫結。建立具有下列內容且名為 *eks-kube-proxy-windows-crb.yaml* 的檔案，以新增繫結。

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: eks:kube-proxy-windows
  labels:
    k8s-app: kube-proxy
    eks.amazonaws.com/component: kube-proxy
subjects:
  - kind: Group
    name: "eks:kube-proxy-windows"
roleRef:
  kind: ClusterRole
  name: system:node-proxier
  apiGroup: rbac.authorization.k8s.io
```

將組態套用至叢集。

```
kubectl apply -f eks-kube-proxy-windows-crb.yaml
```

4. 啟用 Windows 支援以後，您可以在叢集中啟動 Windows 節點群組。如需詳細資訊，請參閱 正在啟動自我管理的 Windows 節點。

macOS and Linux

使用 macOS 或 Linux 用戶端為叢集啟用舊版 Windows 支援

此程序需要將 `openssl` 程式庫和 `jq` JSON 處理器安裝在用戶端系統上。

在以下步驟中，以您叢集所在的 AWS 區域 取代 *region-code*。

1. 將 VPC 資源控制器部署到您的叢集。

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
```

2. 為您的叢集建立 VPC 許可控制器 Webhook 清單檔案。

   a. 下載必要的指令碼和部署檔案。

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.sh
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-patch-ca-bundle.sh
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml
   ```

   b. 將許可新增到 Shell 指令碼，以便可以執行它們。

   ```
   chmod +x webhook-create-signed-cert.sh webhook-patch-ca-bundle.sh
   ```

   c. 建立安全通訊的密碼。

   ```
   ./webhook-create-signed-cert.sh
   ```

   d. 驗證密碼。

   ```
   kubectl get secret -n kube-system vpc-admission-webhook-certs
   ```

   e. 設定 Webhook 並建立部署檔案。

   ```
   cat ./vpc-admission-webhook-deployment.yaml | ./webhook-patch-ca-bundle.sh > vpc-admission-webhook.yaml
   ```

3. 部署 VPC 許可 Webhook。

```
kubectl apply -f vpc-admission-webhook.yaml
```

> **Important**
>
> VPC 許可控制器 Webhook 使用憑證簽署，該憑證在發行日期後一年到期。若要避免停機時間，請務必在憑證到期之前續約憑證。如需詳細資訊，請參閱 續約 VPC 許可 Webhook 憑證。

4. 判斷您的叢集是否具有必要的叢集角色繫結。

```
kubectl get clusterrolebinding eks:kube-proxy-windows
```

如果傳回類似下列範例輸出的輸出，則叢集具有必要的角色繫結。

```
NAME                     ROLE                                AGE
eks:kube-proxy-windows   ClusterRole/system:node-proxier   19h
```

如果輸出包含 `Error from server (NotFound)`，則叢集沒有必要的叢集角色繫結。建立具有下列內容且名為 *eks-kube-proxy-windows-crb.yaml* 的檔案，以新增繫結。

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: eks:kube-proxy-windows
  labels:
    k8s-app: kube-proxy
    eks.amazonaws.com/component: kube-proxy
subjects:
  - kind: Group
    name: "eks:kube-proxy-windows"
roleRef:
  kind: ClusterRole
  name: system:node-proxier
  apiGroup: rbac.authorization.k8s.io
```

將組態套用至叢集。

```
kubectl apply -f eks-kube-proxy-windows-crb.yaml
```

5. 啟用 Windows 支援以後，您可以在叢集中啟動 Windows 節點群組。如需詳細資訊，請參閱 正在啟動自我管理的 Windows 節點。

續約 VPC 許可 Webhook 憑證

VPC 許可 Webhook 使用的憑證會在發行後一年到期。若要避免停機時間，請務必在其過期之前續約憑證。您可以使用以下命令檢查目前憑證的過期日期。

```
kubectl get secret \
    -n kube-system \
    vpc-admission-webhook-certs -o json | \
    jq -r '.data."cert.pem"' | \
    base64 -decode | \
    openssl x509 \
    -noout \
    -enddate | \
    cut -d= -f2
```

範例輸出如下。

```
May 28 14:23:00 2022 GMT
```

您可以使用 `eksctl` 或 Windows，或者是 Linux/macOS 電腦來續約憑證。請依照您原本用於裝 VPC 許可 Webhook 的工具指示進行操作。例如，如果您最初使用 `eksctl` 安裝 VPC 許可 Webhook，那麼您應該使用 `eksctl` 標籤上的指示續約憑證。

eksctl

1. 重新安裝憑證。使用您叢集的名稱取代 *my-cluster*。

   ```
   eksctl utils install-vpc-controllers -cluster my-cluster -approve
   ```

2. 確認您收到下列輸出。

   ```
   2021/05/28 05:24:59 [INFO] generate received request
   2021/05/28 05:24:59 [INFO] received CSR
   2021/05/28 05:24:59 [INFO] generating key: rsa-2048
   2021/05/28 05:24:59 [INFO] encoded CSR
   ```

3. 重新啟動 Webhook 部署。

   ```
   kubectl rollout restart deployment -n kube-system vpc-admission-webhook
   ```

4. 若續約的憑證已過期，且您有陷於 `Container creating` 狀態的 Windows Pods，則您必須刪除並重新部署這些 Pods。

Windows

1. 取得指令碼以產生新憑證。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.ps1;
```

2. 準備指令碼的參數。

```
./webhook-create-signed-cert.ps1 -ServiceName vpc-admission-webhook-svc -SecretName vpc-admission-webhook-certs -Namespace kube-system
```

3. 重新啟動 Webhook 部署。

```
kubectl rollout restart deployment -n kube-system vpc-admission-webhook-deployment
```

4. 若續約的憑證已過期，且您有陷於 `Container creating` 狀態的 Windows Pods，則您必須刪除並重新部署這些 Pods。

Linux and macOS

先決條件

您必須在電腦上安裝 OpenSSL 和 `jq`。

1. 取得指令碼以產生新憑證。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.sh
```

2. 變更許可。

```
chmod +x webhook-create-signed-cert.sh
```

3. 執行指令碼。

```
./webhook-create-signed-cert.sh
```

4. 重新啟動 Webhook。

```
kubectl rollout restart deployment -n kube-system vpc-admission-webhook-deployment
```

5. 若續約的憑證已過期，且您有陷於 `Container creating` 狀態的 Windows Pods，則您必須刪除並重新部署這些 Pods。

在 Windows 節點上支援更多數量的 Pod

在 Amazon EKS 中，每個 Pod 都會從您的 VPC 配置一個 IPv4 地址。因此，即使有足夠的資源可以在節點上執行更多 Pods，您能部署到節點的 Pods 數量也會受到可用 IP 地址的限制。由於 Windows 節點僅支援一個彈性網路介面，因此依預設，Windows 節點上可用 IP 地址的數量上限為：

```
Number of private IPv4 addresses for each interface on the node - 1
```

一個 IP 地址用作網路介面的主要 IP 地址，因此無法將其配置給 Pods。

您可以啟用 IP 字首委派，在 Windows 節點上啟用更多數量的 Pod。此功能可讓您將 /28 IPv4 字首指派給主要網路介面，而不是指派次要 IPv4 位址。而指派 IP 字首會將節點上可用 IPv4 地址的數量上限增加到：

```
(Number of private IPv4 addresses assigned to the interface attached to the node - 1) *
 16
```

由於可用 IP 地址的數量大幅增加，可用的 IP 地址不應限制您擴展節點上 Pods 數量的能力。如需更多詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。

私有叢集要求

本主題說明如何部署在上部署但無法存取輸出網際網路的 Amazon EKS 叢集。 AWS 雲端如果您已開啟本機叢集 AWS Outposts，請參閱啟動 Outpost 上自我管理的 Amazon Linux 節點，而非本主題。

如果您不熟悉 Amazon EKS 聯網，請參閱探究 Amazon EKS 工作節點之叢集網路的奧秘。如果您的叢集沒有對外網際網路存取，則其必須符合下列需求：

- 您的叢集必須從 VPC 中的容器登錄檔中提取映像。您可以在 VPC 中建立 Amazon Elastic Container Registry，並將容器映像複製到其中，以供節點提取。如需詳細資訊，請參閱 將容器映像從一個儲存庫複製到另一個儲存庫。
- 您的叢集必須啟用端點私有存取。這對節點向叢集端點註冊而言是必要的。端點公有存取權限並非必要。如需詳細資訊，請參閱 Amazon EKS 叢集端點存取控制。
- 自我管理 Linux 和 Windows 節點在啟動之前必須包含下列引導引數。這些引數會略過 Amazon EKS 自我檢查，並且不需要從 VPC 內存取 Amazon EKS API。
  1. 使用下列命令判斷叢集端點的值。使用您叢集的名稱取代 *my-cluster*。

     ```
     aws eks describe-cluster --name my-cluster --query cluster.endpoint --output text
     ```

     範例輸出如下。

     ```
     https://EXAMPLE108C897D9B2F1B21D5EXAMPLE.sk1.region-code.eks.amazonaws.com
     ```

  2. 使用下列命令判斷叢集憑證授權單位的值。使用您叢集的名稱取代 *my-cluster*。

     ```
     aws eks describe-cluster --name my-cluster --query cluster.certificateAuthority --output text
     ```

     傳回的輸出是長字串。

  3. 將下列命令中的 *cluster-endpoint* 和 *certificate-authority* 取代為先前命令輸出中傳回的值。如需有關在啟動自我管理節點時指定引導參數的詳細資訊，請參閱 啟動自我管理的 Amazon Linux 節點 和 正在啟動自我管理的 Windows 節點。
     - 適用於 Linux 節點：

       ```
       --apiserver-endpoint cluster-endpoint --b64-cluster-ca certificate-authority
       ```

       如需其他引數，請參閱 GitHub 上的 Bootstrap 指令碼。
     - 適用於 Windows 節點：

       > **Note**
       >
       > 如果您使用的是自訂服務 CIDR，則需要使用 `-ServiceCIDR` 參數來對其進行指定。否則，叢集中 Pods 的 DNS 解析將會失敗。

```
-APIServerEndpoint cluster-endpoint -Base64ClusterCA certificate-authority
```

如需其他引數，請參閱 引導指令碼組態參數。

- 您的叢集 aws-auth ConfigMap 必須從 VPC 建立。若要進一步了解如何建立項目並將項目新增至 aws-auth ConfigMap，請在終端機中輸入 **eksctl create iamidentitymapping --help**。如果您的伺服器上不存在 ConfigMap，則 eksctl 會在您使用命令新增身分映射時建立它。
- 為服務帳戶設定 IAM 角色的 Pods 會從 AWS Security Token Service (AWS STS) API 呼叫取得憑證。如果沒有輸出網際網路存取，則必須在 VPC 中建立並使用 AWS STS VPC 端點。預設情況下，大多數 AWS v1 SDK 都使用全域 AWS STS 端點 (sts.amazonaws.com)，該端點不使用 AWS STS VPC 端點。若要使用 AWS STS VPC 端點，您可能需要將 SDK 設定為使用地區 AWS STS 端點 (sts.*region-code*.amazonaws.com)。如需詳細資訊，請參閱 設定服務帳戶的 AWS Security Token Service 端點。
- 您叢集的 VPC 子網路必須具有您的 AWS 服務 需要存取的任何 Pods 的 VPC 介面端點。如需詳細資訊，請參閱使用介面 VPC 端點存取 AWS 服務。下表列出了一些常用的服務和端點。如需完整的端點清單，請參閱《AWS PrivateLink 指南》https://docs.aws.amazon.com/vpc/latest/privatelink/中的與 AWS PrivateLink整合的AWS 服務。

服務	端點
Amazon EC2	com.amazonaws.*region-code* .ec2
Amazon Elastic Container Registry (用於提取容器映像)	com.amazonaws.*region-code* .ecr.api、com.amazonaws.*region-code* .ecr.dkr 和 com.amazonaws.*region-code* .s3
Application Load Balancer 與 Network Load Balancer	com.amazonaws.*region-code* .elasticloadbalancing
AWS X-Ray	com.amazonaws.*region-code* .xray
Amazon CloudWatch 日誌	com.amazonaws.*region-code* .logs

服務	端點
AWS Security Token Service (在服務帳戶使用 IAM 角色時需要)	com.amazonaws.*region-code* .sts

考量事項

- 任何自我管理節點都必須部署至具有您所需 VPC 介面端點的子網路。如果您建立受管節點群組，VPC 介面端點安全群組必須允許子網路的 CIDR，或者您必須將建立的節點安全群組新增至 VPC 介面端點安全群組。
- 如果您Pods使用 Amazon EFS 磁碟區，則必須先變更驅動程式的庫存化 .yaml 檔案Amazon EFS CSI 驅動程式，才能將容器映像設定為使用與 Amazon EKS 叢集相同 AWS 區域 的容器映像。
- 您可以使用AWS Load Balancer Controller將 AWS 應用程式負載平衡器 (ALB) 和網路負載平衡器部署到私人叢集。部署時，您應使用命令列旗標將 `enable-shield`、`enable-waf` 和 `enable-wafv2` 設定為 false。不支援透過傳入物件的主機名稱進行憑證探索。這是因為控制器需要連線 AWS Certificate Manager，而該控制器沒有 VPC 介面端點。

  控制器支援具有 IP 目標的 Network Load Balancer，這些目標與 Fargate 一起使用。如需詳細資訊，請參閱 Amazon EKS 上的應用程式負載平衡 及 建立 Network Load Balancer。
- 支援 Cluster Autoscaler。部署 Cluster Autoscaler Pods 時，請確保命令列包含 `--aws-use-static-instance-list=true`。如需詳細資訊，請參閱 GitHub 上的使用靜態執行個體清單。工作者節點 VPC 還必須包含 VPC 端點和自動調度資源的 AWS STS VPC 端點。
- 某些容器軟體產品使用 API 呼叫來存取 AWS Marketplace Metering Service 來監視使用情況。私有叢集不允許這些呼叫，因此您無法將這些容器類型用於私有叢集。

Amazon EKS Kubernetes 版本

透過新功能，設計更新和錯誤修復，Kubernetes 得以快速進化。社群平均每四個月發行一次新的 Kubernetes 次要版本（例如 `1.30`）。Amazon EKS 遵循次要版本的上游發行和停用週期。由於 Amazon EKS 會提供新的 Kubernetes 版本，所以我們建議您主動更新您的叢集，以便使用最新的可用版本。

在發布後的前 14 個月內，次要版本在 Amazon EKS 受到標準支援。一旦版本過了標準支援日期結束時，它會自動進入未來 12 個月的延長支援。延長支援可讓您持續使用特定 Kubernetes 版本更長時間，並依叢集小時支付額外費用。如果您在延長支援期間結束之前尚未更新叢集，您的叢集會自動升級至最舊的目前支援的延伸版本。

建議您使用 Amazon EKS 支援的最新可用 Kubernetes 版本建立叢集。如果您的應用程式需要特定版本 Kubernetes，您可以選取較舊版本。您可以在標準或延長支援中提供的任何版本建立新的 Amazon EKS 叢集。

標準支援的可用版本

Amazon EKS 標準支援目前提供下列 Kubernetes 版本：

- 1.30
- 1.29
- 1.28
- 1.27
- 1.26

有關標準支援每個版本需要注意的重要變更，請參閱 標準支援版本發行說明。

延長支援的可用版本

Amazon EKS 延長支援目前提供下列 Kubernetes 版本：

- 1.25
- 1.24
- 1.23

有關延長支援每個版本需要注意的重要變更，請參閱延長支援版本發行說明。

Amazon EKS 延伸支援目前提供以下Kubernetes版本，但您無法使用這些版本建立新叢集的額外要求：

- 1.22
- 1.21

  如需這些版本的資訊，請參閱 適用於版本 1.21 和 1.22 的發行公告

Amazon EKS Kubernetes 發佈日曆

下表顯示每個 Kubernetes 版本需要考慮的重要發行和支援日期。

> **Note**
>
> 只有月份和年份的日期是近似值，並會在已知確切日期時進行更新。

Kubernetes 版本	上游發佈	Amazon EKS 發佈	標準支援結束	延長支援結束
1.30	2024年4月17日	2024年5月23 日	2025年7月23日	2026年7月23 日
1.29	2023 年 12 月 13 日	2024 年 1 月 23 日	2025年3月23日	2026年3月23 日
1.28	2023 年 8 月 15 日	2023 年 9 月 26 日	2024年11月26日	2025年11月26日
1.27	2023 年 4 月 11 日	2023 年 5 月 24 日	2024年7月24日	2025年7月24日
1.26	2022 年 12 月 9 日	2023 年 4 月 11 日	2024年6月11日	2025年6月11日
1.25	2022 年 8 月 23 日	2023 年 2 月 22 日	2024年5月1日	2025年5月1日
1.24	2022 年 5 月 3 日	2022 年 11 月 15 日	2024 年 1 月 31 日	2025 年 1 月 31 日
1.23	2021 年 12 月 7 日	2022 年 8 月 11 日	2023 年 10 月 11 日	2024 年 10 月 11 日
1.22	2021 年 8 月 4 日	2022 年 4 月 4 日	2023 年 6 月 4 日	2024年9月1日
1.21	2021 年 4 月 8 日	2021 年 7 月 19 日	2023 年 2 月 16 日	2024年7月15日

Amazon EKS 版常見問題

標準支援有多少個版本？ Kubernetes

根據 Kubernetes 社群對 Kubernetes 版本的支援，Amazon EKS 致力在任何特定時間至少為四個 Kubernetes 即可生產的版本提供標準支援。我們將至少提前 60 天宣布特定 Kubernetes 次要版本的標準支援日期結束。由於 Amazon EKS 認證和新 Kubernetes 版本的發行程序，Amazon EKS Kubernetes 版本的標準支援日期將在 Kubernetes 專案停止支援上游版本的日期或之後。

Kubernetes 接受 Amazon EKS 標準支援需要多長時間？

Kubernetes 版本在 Amazon EKS 首次提供後，已在 14 個月內獲得標準支援。即使上游 Kubernetes 不再支援 Amazon EKS 上提供的版本，也是如此。我們向後移植適用於 Amazon EKS 所支援的 Kubernetes 版本的安全性修補程式。

當 Amazon EKS Kubernetes 版本的標準支援結束時，是否會通知我？

是。如果您帳戶中有任何叢集執行的版本即將結束支援，Amazon EKS 會在 Amazon EKS 上發佈 Kubernetes版本後的 AWS Health Dashboard 大約 12 個月內寄出通知。此通知包括支援終止的日期。此日期距通知發出日期起至少 60 天。

Amazon EKS 支援哪些 Kubernetes 功能？

Amazon EKS 支援所有 Kubernetes API 一般可用 (GA) 功能。從 Kubernetes 版本 `1.24` 開始，依預設，系統不會在叢集中啟用新的 Beta API。然而，依預設，仍會繼續啟用先前的 Beta API 和新版本的現有 Beta API。Alpha 功能不受支援。

Amazon EKS 管理節點群組是否會與叢集控制平面版本一起自動更新？

不受管理節點群組會在您的帳戶建立 Amazon EC2 執行個體。當您或 Amazon EKS 更新控制平面時，這些執行個體不會自動升級。如需詳細資訊，請參閱 更新受管節點群組。建議您在控制平面和節點上維持相同的 Kubernetes 版本。

自我管理節點群組是否會與叢集控制平面版本一起自動更新？

不會。自我管理的節點群組包含您帳戶的 Amazon EC2 執行個體。當您或 Amazon EKS 代您更新控制平面版本時，這些執行個體不會自動升級。自我管理節點群組在主控台中沒有任何需要更新的指示。您可以檢視安裝在節點上的 `kubelet` 版本，方法是選取您叢集的 Overview (概觀) 標籤上的 Nodes (節點) 清單上的節點，以判斷哪些節點需要更新。您必須手動更新節點。如需詳細資訊，請參閱 自我管理的節點更新。

Kubernetes 專案針對最多三個次要版本測試控制平面與節點之間的相容性。例如，協調工作是由 `1.30` 控制平面進行時，`1.27` 節點將可繼續操作。不過，不建議執行節點持續位於控制平面後面三

個次要版本的叢集。如需詳細資訊，請參閱 Kubernetes 文件中的 Kubernetes 版本和版本偏移支援政策。建議您在控制平面和節點上維持相同的 Kubernetes 版本。

在 Fargate 執行 Pods 是否會透過自動升級叢集控制平面版本升級來自動升級？

不。我們強烈建議執行 Fargate Pods 作為複製控制器的一部分，例如 Kubernetes 部署。然後對所有 Fargate Pods 進行滾動式重新啟動。新版本的 Fargate Pod 部署了 `kubelet` 版本，其版本與您更新的叢集控制平面版本相同。如需詳細資訊，請參閱 Kubernetes 文件中的 Deployments (部署)。

> **Important**
>
> 如果您更新控制平面，您仍必須自行更新 Fargate 節點。若要更新 Fargate 節點，請刪除節點所代表的 Fargate Pod，然後重新部署 Pod。新的 Pod 部署了 `kubelet` 版本，其版本與您叢集的版本相同。

Amazon EKS 擴展支持常見問題

標準支援和延長支援術語對我來說是新的。這些術語是什麼意思？

Amazon EKS Kubernetes 版本的標準支援會在 Amazon EKS Kubernetes 發行版本時開始，並且在發布日期後 14 個月結束。對 Kubernetes 版本的延長支援將在標準支援結束後立即開始，並在接下來的 12 個月後結束。例如，Amazon EKS 版本 `1.23` 的標準支援將於 2023 年 10 月 11 日結束。版本的延伸支援從 2023 年 10 月 12 日`1.23`開始，並將於 2024 年 10 月 11 日結束。

我需要做什麼才能獲得 Amazon EKS 叢集的延長支援？

您無需採取任何動作即可獲得 Amazon EKS 叢集的延長支援。標準支援將在 Amazon EKS 發行 Kubernetes 版本後開始，並且在發布日期後 14 個月結束。對 Kubernetes 版本的延長支援將在標準支援結束後立即開始，並在接下來的 12 個月後結束。在標準支援結束後的 Kubernetes 版本執行的叢集將自動加入延長支援。

我可以針對哪些 Kubernetes 版本獲得延長支援？

延長支援適用於 Kubernetes 版本 `1.23` 及更高版本。在該版本的標準支援結束後，您可以在任何版本執行叢集，長達 12 個月。這代表每個版本將在 Amazon EKS 支援 26 個月（ 14 個月的標準支援加 12 個月的延長支援）。

如果我不想使用延長支援該怎麼辦？

如果您不想自動註冊延長支援，您可以將叢集升級到標準 Amazon EKS 支援的 Kubernetes 版本。對於未升級至標準支援的 Kubernetes 版本，其叢集將自動進入延長支援。

於延長支援 12 個月結束後會發生什麼事？

對於已完成 26 個月生命週期的 Kubernetes 版本，在其中執行叢集（14 個月的標準支援加 12 個月的延長支援）將自動升級到下一版本。

在延長支援日期結束時，您就無法再使用不支援的版本來建立新的 Amazon EKS 叢集。現有的控制平面會在終止支援日期後由 Amazon EKS 透過逐步部署程序，自動將控制平面更新為最舊的支援版本。自動更新控制平面後，請務必手動更新叢集附加元件和 Amazon EC2 節點。如需詳細資訊，請參閱 更新 Amazon EKS 叢集的 Kubernetes 版本。

於延長支援日期結束後，我的控制平面究竟何時會自動更新？

Amazon EKS 無法提供特定的時間範圍。於延長支援日期結束後，可隨時進行自動更新。更新之前，您不會收到任何通知。我們建議您主動更新控制平面，而不需依賴 Amazon EKS 自動更新程序。如需詳細資訊，請參閱 更新 Amazon EKS 叢集 Kubernetes 版本。

我可以無限期將控制平面放在 Kubernetes 版本嗎？

不。 雲安全 AWS 是最高的優先級。在過去一定時間點 (通常為 1 年) 後，Kubernetes 社群就會停止發佈常見的漏洞和風險修補程式 (CVE)，並且不鼓勵針對不支援版本提交 CVE。這表示可能甚至不會報告舊版 Kubernetes 特有的弱點。讓叢集暴露於漏洞之下，不會發出通知，也沒有修復選項。鑑於這一點，Amazon EKS 不允許控制平面保持在延長支援已結束的版本。

獲得延長支援是否需要額外費用？

是，在延伸支援中執行的 Amazon EKS 叢集需要支付額外費用。如需定價詳細資訊，請參閱 AWS 部落格上的 Amazon EKS Kubernetes 版本定價延伸支援。

延長支援包含哪些內容？

延長支援的 Amazon EKS 叢集會持續接收 Kubernetes 控制平面的安全性修補程式。此外，Amazon EKS 將針對Amazon VPC CNI 發布修補程式 `kube-proxy`，以及延長支援版本的 CoreDNS 附加元件。Amazon EKS 還將針對 AWS已發布的 Amazon EKS 針對 Amazon Linux 和視窗優化 AMI 發布修補程序Bottlerocket，以及這些版本的 Amazon EKS Fargate 節點。延伸 Support 中的所有叢集都將繼續從中存取技術支援 AWS。

> **Note**
>
> 針對由發佈的 Amazon EKS 最佳化 Windows AMI 的延伸 Support AWS 不適用於 Kubernetes版本，1.23但適用於版本1.24及更高Kubernetes版本。

延長支援的非 Kubernetes 元件修補程式是否有任何限制？

雖然延伸 Support 涵蓋了所有的Kubernetes特定元件 AWS，但它只會隨時支援 AWS已發佈的 Amazon EKS 最佳化 AMIBottlerocket，適用於 Amazon Linux 和視窗。這代表您在使用延長支援時，您可能會在 Amazon EKS 最佳化 AMI 擁有更新的元件 (例如作業系統或核心)。例如，一旦 Amazon Linux 2 在 2025 年達到生命週期結束，Amazon EKS 最佳化的 Amazon Linux AMI 將使用更新的 Amazon Linux 作業系統構建。Amazon EKS 將針對每個版本宣布並記錄重要的支援生命週期差異，例如此。 Kubernetes

我可以使用延伸支援的版本建立新叢集嗎？

是的，排除1.22和1.21。例如，您可以建立1.23叢集，但無法建立1.22叢集。

標準支援版本發行說明

本主題提供標準支援的每個 Kubernetes 版本需要注意的重要變更。於升級時，請仔細檢閱叢集舊版和新版本之間發生的變更。

> **Note**
>
> 對於 1.24 和之後的叢集，正式發佈的 Amazon EKS AMI 包括 containerd作為唯一的執行階段。低於 1.24 的 Kubernetes 版本會使用 Docker 作為預設執行階段。這些版本具有引導旗標選項，您可以使用引導旗標選項在任何支援的具有 containerd 的叢集上測試工作負載。如需詳細資訊，請參閱 Amazon EKS 已結束對 Dockershim 的支援。

Kubernetes1.30

Kubernetes 1.30 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.30 的詳細資訊，請參閱官方版本公告。

> **Important**
>
> - 從 Amazon EKS 版本1.30或更新版本開始，任何新建立的受管節點群組都會自動預設使用 Amazon Linux 2023 (AL2023) 做為節點作業系統。以前，新的節點群組會預設為 Amazon Linux 2 (AL2)。建立新節點群組時，您可以選擇 AL2 作為 AMI 類型，繼續使用 AL2。
>   - 有關 Amazon Linux 的更多信息，請參閱 Amazon Linux 用戶指南中的比較 AL2 和 AL2023。
>   - 如需為受管理節點群組指定作業系統的詳細資訊，請參閱。建立受管節點群組

- 使用 Amazon EKS 時1.30，topology.k8s.aws/zone-id標籤會新增至工作者節點。您可以使用可用區域 ID (AZ ID) 來決定一個帳戶中資源相對於另一個帳戶中資源的位置。如需詳細資訊，請參閱AWS RAM 使用者指南中的 AWS 資源的可用區域 ID。
- 從開始1.30，Amazon EKS 不再在套用至新建立叢集的gp2StorageClass資源上包含default註解。如果您按名稱引用此存儲類別，則這不會產生任何影響。如果您依賴StorageClass在叢集中具有預設值，則必須採取行動。您應該StorageClass依名稱來參照gp2。或者，您可以在安裝v1.31.0或更高版本時將defaultStorageClass.enabled參數設定為 true，部署 Amazon EBS 建議的預設儲存類別。aws-ebs-csi-driver add-on
- Amazon EKS 叢集 IAM 角色所需的最低 IAM 政策已變更。此動作ec2:DescribeAvailabilityZones為必要動作。如需詳細資訊，請參閱 Amazon EKS 叢集 IAM 角色。

如需完整的 Kubernetes 1.30 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.30.md。

Kubernetes1.29

Kubernetes 1.29 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.29 的詳細資訊，請參閱官方版本公告。

> **Important**
>
> - 的FlowSchema和已取代的 flowcontrol.apiserver.k8s.io/v1beta2 API 版本PriorityLevelConfiguration不再在中提供Kubernetesv1.29。如果您有資訊清單

或用戶端軟體使用已淘汰的 Beta API 群組，您應該在升級至v1.29之前變更這些資訊清單或用戶端軟體。

- 節點對象的.status.kubeProxyVersion字段現在已被棄用，Kubernetes項目提議在 future 的版本中刪除該字段。不推薦使用的字段不准確，歷史上已被管理 kubelet-實際上並不知道kube-proxy版本，甚至kube-proxy是否正在運行。如果您一直在用戶端軟體中使用此欄位，請停止-資訊不可靠，而且該欄位現在已被取代。
- 為了減少潛在的攻擊面，如果長時間未使用舊式自動產生的以密碼為基礎的權杖 (預設為 1 年)，LegacyServiceAccountTokenCleanUp功能會Kubernetes1.29將它們標示為無效 (預設為 1 年)，如果長時間未嘗試使用，則會自動移除它們標記為無效 (預設為額外 1 年)。要識別此類令牌，您可以運行：

```
kubectl get cm kube-apiserver-legacy-service-account-token-tracking -nkube-system
```

如需完整的 Kubernetes 1.29 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.29.md#changelog-since-v1280。

Kubernetes1.28

Kubernetes 1.28 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.28 的詳細資訊，請參閱官方版本公告。

- Kubernetes v1.28 將核心節點與控制平面元件之間的支援偏移從 n-2 到 n-3 擴充一個次要版本，以便支援最舊次要版本的節點元件 (kubelet 和 kube-proxy) 可以搭配最新支援的次要版本一起使用控制平面元件 (kube-apiserver, kube-scheduler, kube-controller-manager, cloud-controller-manager)。
- Pod GC Controller 的指標 force_delete_pods_total 和 force_delete_pod_errors_total 已經增強，負責所有強制 Pod 刪除。原因會新增至量度，以指出網繭是否因為網繭已終止、孤立、以 out-of-service 污染終止，或終止和未排定而強制刪除。
- 已修改 PersistentVolume (PV) 控制器，自動將預設 StorageClass 指定給任何未設定 PersistentVolumeClaim 的未受限制 storageClassName。此外，API 伺服器內的 PersistentVolumeClaim 接納驗證機制已經調整，以允許將值從未設定狀態變更為實際StorageClass 名稱。

如需完整的 Kubernetes 1.28 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.28.md#changelog-since-v1270。

Kubernetes1.27

Kubernetes 1.27 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.27 的詳細資訊，請參閱官方版本公告。

> **Important**
>
> - 對 Alpha seccomp 註釋 seccomp.security.alpha.kubernetes.io/pod 和 container.seccomp.security.alpha.kubernetes.io 註釋的支援已移除。Alpha seccomp 註釋已在 1.19 中棄用，在 1.27 中刪除後，seccomp 欄位將不再為包含 seccomp 註釋的 Pods 自動填入。請改為使用 Pods 的 securityContext.seccompProfile 欄位或容器來設定 seccomp 設定檔。若要檢查您是否在叢集中使用已棄用的 Alpha seccomp 註釋，請執行下列命令：
>
>   ```
>   kubectl get pods --all-namespaces -o json | grep
>    -E 'seccomp.security.alpha.kubernetes.io/pod|
>   container.seccomp.security.alpha.kubernetes.io'
>   ```
>
> - 系統已移除 kubelet 的 --container-runtime 命令行引數。containerd此後1.24，Amazon EKS 的預設容器執行階段就不需要指定容器執行階段。自 1.27 起，Amazon EKS 將忽略傳遞給任何啟動程序指令碼的 --container-runtime 引數。切勿將此引數傳遞給 --kubelet-extra-args，防止節點啟動程序期間發生錯誤。您必須移除所有節點建立工作流程和建置指令碼中的 --container-runtime 引數。

- Kubernetes 1.27 中的 kubelet 會將預設的 kubeAPIQPS 增加為 50，並將 kubeAPIBurst 增加為 100。這些增強功能允許 kubelet 處理更大量的 API 查詢，改善回應時間和效能。當 Pods 的需求增加時，由於擴展要求，修改後的預設值可確保 kubelet 能有效管理增加的工作負載。因此，Pod 啟動速度更快，且叢集作業更有效率。
- 您可以使用更精細的 Pod 拓撲來傳播政策，例如 minDomain。此參數可讓您指定 Pods 應傳播的網域數目下限。nodeAffinityPolicy 和 nodeTaintPolicy 則允許在管理 Pod 分佈中額外的細部程度。這是根據節點親和性、污點和 Pod's 規格的 topologySpreadConstraints 之中的 matchLabelKeys 欄位而定。如此可允許在滾動升級之後選取 Pods 以供分攤計算之用。
- Kubernetes1.27 已提升為測試版，此為 StatefulSets 的新政策機制，可控制其 PersistentVolumeClaims (PVCs) 的生命週期。全新的 PVC 保留政策可讓您指定，當

StatefulSet 被刪除或 StatefulSet 中的複本縮減規模時，從 StatefulSet 規格範本產生的 PVCs 是否會自動刪除或保留。

- 透過隨機關閉連線，Kubernetes API 伺服器中的 goaway-chance 選項有助於防止 HTTP/2 用戶端連線卡在單一 API 伺服器執行個體上。如果連線關閉，用戶端會嘗試重新連線，並且可能會因負載平衡而停留在不同的 API 伺服器上。Amazon EKS 版本 1.27 已啟用 goaway-chance 旗標。如果在 Amazon EKS 叢集上執行的工作負載所使用的用戶端與 HTTP GOAWAY 不相容，建議您在連線終止時透過重新連線來更新用戶端，從而處理 GOAWAY。

如需完整的 Kubernetes 1.27 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.27.md#changelog-since-v1260。

Kubernetes1.26

Kubernetes 1.26 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.26 的詳細資訊，請參閱官方版本公告。

> **Important**
>
> Kubernetes 1.26 不再支援 CRI v1alpha2。如果容器執行期不支援 CRI v1，這會導致 kubelet 不再註冊節點。這也意味著 Kubernetes 1.26 不支援 containerd 次要版本 1.5 和更早版本。如果使用的是 containerd，您需要先升級至 containerd 版本 1.6.0 或更新版本，然後再將任何節點升級至 Kubernetes 1.26。您也需要升級僅支援 v1alpha2 的任何其他容器執行期。如需詳細資訊，請參閱容器執行期廠商。依預設，Amazon Linux 和 Bottlerocket AMI 包含 containerd 版本 1.6.6。

- 在升級至 Kubernetes 1.26 之前，先將 Amazon VPC CNI plugin for Kubernetes 升級至版本 1.12 或更新版本。如果您未升級到 Amazon VPC CNI plugin for Kubernetes 版本 1.12 或更高版本，則 Amazon VPC CNI plugin for Kubernetes 會當機。如需詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。
- 透過隨機關閉連線，Kubernetes API 伺服器中的 goaway-chance 選項有助於防止 HTTP/2 用戶端連線卡在單一 API 伺服器執行個體上。如果連線關閉，用戶端會嘗試重新連線，並且可能會因負載平衡而停留在不同的 API 伺服器上。Amazon EKS 版本 1.26 已啟用 goaway-chance 旗標。如果在 Amazon EKS 叢集上執行的工作負載所使用的用戶端與 HTTP GOAWAY 不相容，建議您在連線終止時透過重新連線來更新用戶端，從而處理 GOAWAY。

如需完整的 Kubernetes 1.26 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.26.md#changelog-since-v1250。

延長支援版本發行說明

本主題提供延長支援的每個 Kubernetes 版本需要注意的重要變更。於升級時，請仔細檢閱叢集舊版和新版本之間發生的變更。

Kubernetes1.25

Kubernetes 1.25 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.25 的詳細資訊，請參閱官方版本公告。

Important

- 從 Kubernetes 版本 1.25 開始，您將無法再使用開箱即用的 Amazon EC2 P2 執行個體搭配 Amazon EKS 最佳化加速 Amazon Linux AMI 一起使用。這些適用於 Kubernetes 版本 1.25 或更高版本的 AMI 將支援 NVIDIA 525 系列或更新版本驅動程式，這些與 P2 執行個體不相容。但是，NVIDIA 525 系列或更新版本的驅動程式與 P3 P4 及 P5 執行個體相容，因此您可以在 Kubernetes 版本 1.25 或更新版本將這些執行個體搭配 AMI 一起使用。在 Amazon EKS 叢集升級至版本 1.25 之前，請將任何 P2 執行個體移轉至 P3 P4 和 P5 執行個體。您也應該主動升級您的應用程式以適用於 NVIDIA 525 系列或更新版本。我們計劃將更新的NVIDIA 525系列或更高Kubernetes版本1.23的驅動程序移植到 2024 年 1 月下旬。1.24
- PodSecurityPolicy (PSP) 已在 Kubernetes 1.25 中移除。PSPs 已取代為 Pod 安全許可 (PSA) 和 Pod 安全標準 (PSS)。 PSA 是內建的許可控制器，其實作 PSS 中概述的安全控制項。依據預設，PSA 和 PSS 在 Kubernetes1.25 中逐步進入穩定，並在 Amazon EKS 中啟用。如果您有PSPs叢集，請務必先從內建KubernetesPSS或移轉PSP至 policy-as-code 解決方案，然後再將叢集升級至版本1.25。如果您未從 PSP 遷移，則可能會遇到工作負載中斷。如需更多資訊，請參閱Pod 安全政策 (PSP) 移除常見問答集。
- Kubernetes 版本 1.25 包含可改變稱為 API 優先順序與公平性 (APF) 之現有功能的行為之變更。APF 的功能是讓 API 伺服器在處理高於往常的請求量期間不受潛在超載的影響。其作法是取代並行請求 (可在任何指定時間處理) 數量的限制。透過實施區隔明顯的優先順序層級並限制來自各種工作負載或使用者的請求，即可達成這個目標。此方法可確保關鍵應用程式或高優先順序請求能獲得優先處理，同時防止較低優先順序的請求過量，導致 API 伺服器不堪負荷。如需詳細資訊，請參閱 Kubernetes 文件中的 API 優先順序與公平性或《EKS 最佳實務指南》中的 API 優先順序與公平性。

這些更新已在 PR #10352 和 PR #118601 中推出。之前，APF 會統一處理所有類型的請求，每個請求都會耗用並行請求限制的單一單位。由於這些請求會對 API 伺服器帶來異常沉重的負擔，因此 APF 行為變更會將更高的並行單位指派給 LIST 請求。API 伺服器會預估將由 LIST 請求傳回的物件數量。它會指派一個與傳回的物件數量成比例的並行單位。

一旦升級到 Amazon EKS 版本 1.25 或更高版本，此更新過的行為可能會導致具有繁重 LIST 請求的工作負載 (先前運作不會有問題) 遭遇速率限制。此情況會透過 HTTP 429 回應碼來表示。為了避免因 LIST 請求遭到速率限制而出現的潛在工作負載中斷，我們強烈建議您調整工作負載以降低這些請求的速率。或者，您可以透過調整 APF 設定來為必要請求分配更多容量，同時減少分配給非必要請求的容量來解決此問題。如需有關這些緩解技術的詳細資訊，請參閱《EKS 最佳實務指南》中的避免遺漏的請求。

- Amazon EKS 1.25 包含叢集身分驗證的增強功能，其中內含更新的 YAML 程式庫。如果在 kube-system 命名空間中找到的 aws-auth ConfigMap 中的 YAML 值以巨集開頭，其中第一個字元是大括號，則應該在此大括號 ({ }) 前後新增引號 (" ")。必須執行此動作，以確保該 aws-iam-authenticator 版本 v0.6.3 準確地剖析 Amazon EKS 1.25 中的 aws-auth ConfigMap。
- EndpointSlice 的 beta API 版本 (discovery.k8s.io/v1beta1) 已在 Kubernetes 1.21 中被棄用，從 Kubernetes1.25 開始不再提供。此 API 已更新為 discovery.k8s.io/v1。如需詳細資訊，請參閱 Kubernetes 文件中的 EndpointSlice。AWS Load Balancer Controller v2.4.6 和更早版本使用 v1beta1 端點與 EndpointSlices 通訊。如果您將 EndpointSlices 組態用於 AWS Load Balancer Controller，則必須先升級至 AWS Load Balancer Controller v2.4.7，然後再將 Amazon EKS 叢集升級至 1.25。如果您在將 EndpointSlices 組態用於 AWS Load Balancer Controller 時升級至 1.25，控制器將損毀並導致工作負載中斷。若要升級控制器，請參閱什麼是 AWS Load Balancer Controller？。

- SeccompDefault 已在 Kubernetes 1.25 中提升為 beta 版。透過在設定 kubelet 時設定 --seccomp-default 旗標，容器執行階段會使用其 RuntimeDefault seccomp 設定檔，而不是使用無約束的 (seccomp disabled) 模式。預設設定檔提供一組強大的安全性預設值，同時保留工作負載的功能。雖然此旗標可用，但 Amazon EKS 預設不會啟用此旗標，因此 Amazon EKS 的行為實際上不會變更。如果需要，您可以在節點上開始啟用此功能。有關更多詳細資訊，請參閱 Kubernetes 文件中的使用 seccomp 限制容器的系統呼叫教學教程。
- Docker (也稱為 Dockershim) 的容器執行階段介面 (CRI) 支援已從 Kubernetes 1.24 和更新版本中移除。Amazon EKS 官方 AMIs 中適用於 Kubernetes 1.24 和更新叢集的唯一容器執行階段是

containerd。升級至 Amazon EKS 1.24 或更新版本之前，請移除任何不再受支援之引導指令碼旗標的參考。如需詳細資訊，請參閱 Amazon EKS 已結束對 Dockershim 的支援。

- 對萬用字元查詢的支援已在 CoreDNS 1.8.7 中被棄用，並在 CoreDNS 1.9 中移除。這是一種安全措施。萬用字元查詢不再起作用，並且傳回 NXDOMAIN 而非 IP 地址。
- 透過隨機關閉連線，Kubernetes API 伺服器中的 goaway-chance 選項有助於防止 HTTP/2 用戶端連線卡在單一 API 伺服器執行個體上。如果連線關閉，用戶端會嘗試重新連線，並且可能會因負載平衡而停留在不同的 API 伺服器上。Amazon EKS 版本 1.25 已啟用 goaway-chance 旗標。如果在 Amazon EKS 叢集上執行的工作負載所使用的用戶端與 HTTP GOAWAY 不相容，建議您在連線終止時透過重新連線來更新用戶端，從而處理 GOAWAY。

如需完整的 Kubernetes 1.25 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.25.md#changelog-since-v1240。

Kubernetes1.24

Kubernetes 1.24 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.24 的詳細資訊，請參閱官方版本公告。

> **Important**
>
> - 從 Kubernetes 1.24 開始，依預設，不會在叢集中啟用新的 Beta API。依預設，仍會繼續啟用現有 Beta API 和新版本的現有 Beta API。Amazon EKS 遵循與上游 Kubernetes 1.24 相同的行為。依預設，會啟用控制新的和現有 API 操作之新功能的功能閘道。這與上游 Kubernetes 對齊。如需詳細資訊，請參閱 KEP-3136：測試版 API 預設為關閉狀態為開啟 GitHub。
> - Docker (也稱為 Dockershim) 的容器執行階段介面 (CRI) 支援已從 Kubernetes 1.24 中移除。Amazon EKS 官方 AMI 將 containerd 作為唯一的執行階段。移至 Amazon EKS 1.24 或更高版本之前，您必須移除任何不再受支援之引導指令碼旗標的參考。您還必須確定已為工作節點啟用 IP 轉送。如需詳細資訊，請參閱 Amazon EKS 已結束對 Dockershim 的支援。
> - 如果您已為 Container Insights 設定了 Fluentd，則必須先將 Fluentd 遷移至 Fluent Bit，才能更新叢集。系統會將 Fluentd 剖析器設定為僅剖析 JSON 格式的日誌訊息。與 dockerd 不同的是，containerd 容器執行期具有不是 JSON 格式的日誌訊息。如果您未遷移到 Fluent Bit，則某些已設定的 Fluentd's 剖析器將在 Fluentd 容器內生成大量錯誤。如需有關移轉的詳細資訊，請參閱設定Fluent Bit為將記錄檔傳送 DaemonSet 至 CloudWatch 記錄檔。

- 在 Kubernetes `1.23` 和較早版本中，具有未驗證 IP 和 DNS 主體別名 (SAN) 的 `kubelet` 服務憑證會自動與未驗證的 SAN 一起發行。這些無法驗證的 SAN 會從佈建的憑證中省略。在版本 `1.24` 和更新版本的叢集中，如果無法驗證任何 SAN，則不會發行提供憑證的 `kubelet`。這樣可以防止 `kubectl exec` 和 `kubectl` 日誌命令運作。如需詳細資訊，請參閱 將叢集升級至 Kubernetes 1.24 之前的憑證簽署考量。
- 升級使用 Fluent Bit 的 Amazon EKS `1.23` 叢集時，您必須確保其正在執行 `k8s/1.3.12` 或更新版本。您可以從 GitHub 中重新套用最新適用的 Fluent Bit YAML 檔案來執行此操作。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南Fluent Bit中的設定。

- 當叢集工作節點跨多個可用區域部署時，您可以使用拓撲感知提示來指出將流量保留在區域中的偏好設定。在區域內路由流量有助於降低成本並改善網路效能。依預設，Amazon EKS `1.24` 中會啟用拓撲感知提示。如需詳細資訊，請參閱 Kubernetes 文件中的 Topology Aware Hints (拓撲感知提示)。
- `PodSecurityPolicy` (PSP) 已排程在 Kubernetes `1.25` 中移除。PSPs 將被取代為 Pod 安全許可 (PSA)。PSA 是內建的許可控制器，使用 Pod 安全標準 (PSS) 中概述的安全控制項。PSA 和 PSS 都是 Beta 功能，依預設會在 Amazon EKS 中啟用。若要解決在版本 `1.25` 中移除 PSP 的問題，我們建議您在 Amazon EKS 中實作 PSS。如需詳細資訊，請參閱 AWS 部落格上的 Implementing Pod Security Standards in Amazon EKS (在 Amazon EKS 中實作 Pod 安全標準)。
- 即會`client.authentication.k8s.io/v1alpha1` ExecCredential 在中移除 Kubernetes`1.24`。 ExecCredential API 在中一般可用Kubernetes`1.22`。如果您使用依賴於 `v1alpha1` API 的 Client-go 憑證外掛程式，請聯繫外掛程式的分銷商，以了解如何遷移到 `v1` API。
- 針對 Kubernetes `1.24`，我們為上游 Cluster Autoscaler 專案提供了一項功能，該專案可簡化 Amazon EKS 受管節點群組在零節點之間進行擴縮。之前，若要讓 Cluster Autoscaler 了解縮減為零節點之受管節點群組的資源、標籤和污點，您需要使用基礎 Amazon EC2 Auto Scaling 群組負責的節點詳細資訊來標記該群組。現在，當受管節點群組中沒有執行中的節點時，Cluster Autoscaler 會呼叫 Amazon EKS `DescribeNodegroup` API 操作。此 API 操作提供 Cluster Autoscaler 所需的受管節點群組資源、標籤和污點的資訊。此功能需要您將 `eks:DescribeNodegroup` 許可新增至 Cluster Autoscaler 服務帳戶 IAM 政策。當支援 Amazon EKS 受管節點群組的 Auto Scaling 群組上的 Cluster Autoscaler 標籤值與節點群組本身衝突時，Cluster Autoscaler 會偏好 Auto Scaling 群組標籤的值。如此您便能根據需要覆蓋相關的值。如需詳細資訊，請參閱 自動擴展。
- 如果您打算搭配 Amazon EKS 使用Inferentia或Trainium執行個體類型`1.24`，則必須升級至 AWS Neuron裝置外掛程式 1.9.3.0 或更新版本。如需詳細資訊，請參閱文件中的神經元 K8 版本 [1.9.3.0]。 AWS Neuron
- 依據預設，`Containerd` 已為 `IPv6` 啟用 Pods。它將節點核心設定套用至 Pod 網路命名空間。因此，Pod 中的容器會繫結至 `IPv4` (`127.0.0.1`) 和 `IPv6` (`::1`) 迴路位址。`IPv6` 是通訊的預設通訊

協定。在將叢集更新為版本 1.24 之前，建議您先測試多容器 Pods。修改應用程式，以便它們繫結至迴路界面上的所有 IP 地址。大多數程式庫都啟用 IPv6 繫結，這與 IPv4 向後相容。當無法修改應用程式碼時，您有兩個選項：

- 運行 init 容器並將 disable ipv6 設定為 true (sysctl -w net.ipv6.conf.all.disable_ipv6=1)。
- 設定變動的許可 Webhook，以便在應用程式 Pods 旁注入 init 容器。

如果您需要針對所有節點中的全部 Pods 封鎖 IPv6，則可能必須在執行個體上停用 IPv6。

- 透過隨機關閉連線，Kubernetes API 伺服器中的 goaway-chance 選項有助於防止 HTTP/2 用戶端連線卡在單一 API 伺服器執行個體上。如果連線關閉，用戶端會嘗試重新連線，並且可能會因負載平衡而停留在不同的 API 伺服器上。Amazon EKS 版本 1.24 已啟用 goaway-chance 旗標。如果在 Amazon EKS 叢集上執行的工作負載所使用的用戶端與 HTTP GOAWAY 不相容，建議您在連線終止時透過重新連線來更新用戶端，從而處理 GOAWAY。

如需完整的 Kubernetes 1.24 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.24.md#changelog-since-v1230。

Kubernetes1.23

Kubernetes 1.23 現在可在 Amazon EKS 中使用。如需有關 Kubernetes 1.23 的詳細資訊，請參閱官方版本公告。

> **Important**
>
> - Kubernetes 樹狀內至容器儲存介面 (CSI) 磁碟區遷移功能已啟用。此功能可將現有 Kubernetes Amazon EBS 的樹狀內儲存外掛程式替換為相應的 Amazon EBS CSI 驅動程式。如需詳細資訊，請參閱 Kubernetes 部落格上的 Kubernetes 1.17 功能：Kubernetes 樹狀內遷移至 CSI 磁碟區遷移至試用版。
>
>   此功能會將樹狀內 API 轉換為相等的 CSI API，並將操作委派給替代 CSI 驅動程式。透過此功能，如果您使用屬於這些工作負載的現有 StorageClass、PersistentVolume 以及 PersistentVolumeClaim 物件，可能不會有任何明顯的變化。該功能啟用 Kubernetes 以將所有儲存管理操作從樹狀內外掛程式委派給 CSI 驅動程式。如果您在現有叢集中使用 Amazon EBS 磁碟區，請先在叢集中安裝 Amazon EBS CSI 驅動程式，然後再將叢集更新為版本 1.23。如果您未在更新現有叢集之前安裝驅動程式，可能會出現中斷工作負載的情形。如果您計劃在新的 1.23 叢集中部署使用 Amazon EBS 磁碟區的工作負載，請先在叢集中安裝 Amazon EBS CSI 驅動程式，然後再在叢集中部署工作負載。如需有關如何在叢集安

裝 Amazon EBS CSI 驅動程式的說明，請參閱 Amazon EBS CSI 驅動程式。針對遷移功能的常見問題，請參閱 Amazon EBS CSI 遷移常見問答集。

- 針對由發佈的 Amazon EKS 最佳化 Windows AMI 的延伸 Support AWS 不適用於 Kubernetes版本，1.23但適用於版本1.24及更高Kubernetes版本。

- Kubernetes 在版本 1.20 中停止支援 dockershim，並在版本 1.24 中移除了 dockershim。如需詳細資訊，請參閱 Kubernetes 部落格中的 Kubernetes 正在從 Dockershim 繼續邁進：承諾和後續步驟。Amazon EKS 將從 Amazon EKS 版本 1.24 起終止支援 dockershim。從 Amazon EKS 版本 1.24 開始，Amazon EKS 官方 AMI 將包括 containerd 作為唯一執行階段。

  即使 Amazon EKS 版本 1.23 繼續支援 dockershim，我們建議您立即開始測試您的應用程式，以識別和移除任何 Docker 相依性。如此一來，您就可以將叢集更新為版本 1.24。如需 dockershim 移除的詳細資訊，請參閱 Amazon EKS 已結束對 Dockershim 的支援。
- Kubernetes 逐步全面開放推出 Pods、服務和節點的 IPv4/IPv6 雙堆疊聯網。不過，Amazon EKS 和 Amazon VPC CNI plugin for Kubernetes 不支援雙堆疊聯網。您的叢集可以指派 IPv4 或 IPv6 地址到 Pods 和服務，但無法同時指派這兩種地址類型。
- Kubernetes 讓 Pod 安全許可 (PSA) 功能逐步進入 Beta 版狀態。此功能預設為啟用。如需詳細資訊，請參閱 Kubernetes 文件中的 Pod 安全許可。PSA 取代 Pod 安全政策 (PSP) 許可控制器。不支援 PSP 許可控制器，且已排程於 Kubernetes 版本 1.25 移除。

  此 PSP 許可控制器根據設定強制實施等級的特定命名空間標籤，對命名空間中的 Pod 強制執行 Pods 安全標準。如需詳細資訊，請參閱《Amazon EKS 最佳實務指南》中的 Pod 安全標準 (PSS) 和 Pod 安全許可 (PSA)。
- 使用叢集部署的 kube-proxy 映像現在是由 Amazon EKS Distro (EKS-D) 維護的最小基礎映像。該映像包含的套件最少，而且沒有 Shell 和套件管理工具。
- Kubernetes 逐步開放暫時容器到測試版。暫時容器是作為現有 Pod 在同樣的命名空間執行的臨時容器。您可以使用暫時容器來觀察 Pods 的狀態以及用於疑難排解和除錯目的的容器。這在 kubectl exec 因容器毀損或容器映像不包含除錯公用程式而不足時，用於互動式疑難排解格外有用。包含除錯公用程式的容器範例為 distroless 映像。如需詳細資訊，請參閱 Kubernetes 文件中的使用臨時性除錯容器進行除錯。
- Kubernetes 逐步全面開放推出 HorizontalPodAutoscaler autoscaling/v2 穩定的 API 到一般可用性。HorizontalPodAutoscaler autoscaling/v2beta2 API 已棄用。在 1.26 中將無法使用。
- 透過隨機關閉連線，Kubernetes API 伺服器中的 goaway-chance 選項有助於防止 HTTP/2 用戶端連線卡在單一 API 伺服器執行個體上。如果連線關閉，用戶端會嘗試重新連線，並且可能會因負載

平衡而停留在不同的 API 伺服器上。Amazon EKS 版本 1.23 已啟用 goaway-chance 旗標。如果在 Amazon EKS 叢集上執行的工作負載所使用的用戶端與 HTTP GOAWAY 不相容，建議您在連線終止時透過重新連線來更新用戶端，從而處理 GOAWAY。

如需完整的 Kubernetes 1.23 變更日誌，請參閱 https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.23.md#changelog-since-v1220。

適用於版本 1.21 和 1.22 的發行公告

> **Important**
>
> 您無法使用這些版本建立新叢集。

本主題提供了版本1.22 和應注意的重要變更1.21。於升級時，請仔細檢閱叢集舊版和新版本之間發生的變更。

Kubernetes 版本 **1.22**

下列許可控制器已為所有 1.22 平台版本啟用：DefaultStorageClass、DefaultTolerationSeconds、LimitRanger、MutatingAdmissionW以及 DefaultIngressClass。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.22.17	eks.28	具有安全性修正和增強功能的新平台版本。	2024年5月16日
1.22.17	eks.26	具有安全性修正和增強功能的新平台版本。	2024年4月1日
1.22.17	eks.14	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 30 日
1.22.17	eks.13	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 9 日

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.22.17	eks.12	具有安全性修正和增強功能的新平台版本。	2023 年 5 月 5 日
1.22.17	eks.11	具有安全性修正和增強功能的新平台版本。	2023 年 3 月 24 日
1.22.16	eks.10	具有安全性修正和增強功能的新平台版本。	2023 年 1 月 27 日
1.22.15	eks.9	具有安全性修正和增強功能的新平台版本。	2022 年 12 月 5 日
1.22.15	eks.8	具有安全性修正和增強功能的新平台版本。	2022 年 11 月 18 日
1.22.15	eks.7	具有安全性修正和增強功能的新平台版本。	2022 年 11 月 7 日
1.22.13	eks.6	具有安全性修正和增強功能的新平台版本。	2022 年 9 月 21 日
1.22.10	eks.5	提升 etcd 彈性的新平台版本。	2022 年 8 月 15 日
1.22.10	eks.4	具有安全性修正和增強功能的新平台版本。此平台版本也推出了新的標記控制器，為所有工作節點加上 aws:eks:cluster-name 標籤，以便為這些工作節點分配成本。如需詳細資訊，請參閱 標記您的資源以便計費。	2022 年 7 月 21 日
1.22.10	eks.3	具有安全性修正和增強功能的新平台版本。	2022 年 7 月 7 日

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.22.9	eks.2	具有安全性修正和增強功能的新平台版本。	2022 年 5 月 31 日
1.22.6	eks.1	適用於 Amazon EKS 的 Kubernetes 初始版本 1.22。	2022 年 4 月 4 日

Kubernetes 版本 **1.21**

下列許可控制器已為所有 1.21 平台版本啟用：DefaultStorageClass、DefaultTolerationSeconds、LimitRanger、MutatingAdmissionW以及 DefaultIngressClass。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.21.14	eks.33	具有安全性修正和增強功能的新平台版本。	2024年5月16日
1.21.14	eks.31	具有安全性修正和增強功能的新平台版本。	2024年4月1日
1.21.14	eks.18	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 9 日
1.21.14	eks.17	具有安全性修正和增強功能的新平台版本。	2023 年 5 月 5 日
1.21.14	eks.16	具有安全性修正和增強功能的新平台版本。	2023 年 3 月 24 日
1.21.14	eks.15	具有安全性修正和增強功能的新平台版本。	2023 年 1 月 27 日
1.21.14	eks.14	具有安全性修正和增強功能的新平台版本。	2022 年 12 月 5 日

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.21.14	eks.13	具有安全性修正和增強功能的新平台版本。	2022 年 11 月 18 日
1.21.14	eks.12	具有安全性修正和增強功能的新平台版本。	2022 年 11 月 7 日
1.21.13	eks.11	提升 etcd 彈性的新平台版本。	2022 年 10 月 10 日
1.21.13	eks.10	提升 etcd 彈性的新平台版本。	2022 年 8 月 15 日
1.21.13	eks.9	具有安全性修正和增強功能的新平台版本。此平台版本也推出了新的標記控制器，為所有工作節點加上 aws:eks:cluster-name 標籤，以便為這些工作節點分配成本。如需詳細資訊，請參閱 標記您的資源以便計費。	2022 年 7 月 21 日
1.21.13	eks.8	具有安全性修正和增強功能的新平台版本。	2022 年 7 月 7 日
1.21.12	eks.7	具有安全性修正和增強功能的新平台版本。	2022 年 5 月 31 日
1.21.9	eks.6	AWS Security Token Service 端點會從先前的平台版本還原回全域端點。如果您希望在使用服務帳戶 IAM 角色時使用區域端點，則必須啟用它。如需如何啟用區域端點的說明，請參閱設定服務帳戶的 AWS Security Token Service 端點。	2022 年 4 月 8 日

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.21.5	eks.5	當您使用 服務帳戶的 IAM 角色，目前依預設使用 AWS Security Token Service 區域端點，而不是全域端點。不過，此變更已還原為 eks.6 裡的全域端點。 更新後的 Fargate 排程器在大型的部署期間以顯著提升的速度佈建節點。	2022 年 3 月 10 日
1.21.5	eks.4	Amazon VPC CNI 自我管理和 Amazon EKS 附加元件的 1.10.1-eksbuild.1 版現在是部署的預設版本。	2021 年 12 月 13 日
1.21.2	eks.3	新平台版本，支援在 Kubernetes 控制平面上執行的 VPC 資源控制器上的 Windows IPv4 地址管理。已為 Fargate Fluent Bit 記錄新增 Kubernetes 篩選條件指令。	2021 年 11 月 8 日
1.21.2	eks.2	具有安全性修正和增強功能的新平台版本。	2021 年 9 月 17 日
1.21.2	eks.1	適用於 Amazon EKS 的 Kubernetes 初始版本 1.21。	2021 年 7 月 19 日

Amazon EKS 平台版本

Amazon EKS 平台版本代表 Amazon EKS 叢集控制平面的功能，例如啟用了哪些 Kubernetes API 伺服器旗標以及目前的 Kubernetes 修補程式版本。每個 Kubernetes 次要版本皆有一或多個相關的 Amazon EKS 平台版本。適用於不同 Kubernetes 次要版本的平台版本都是彼此獨立。您可以使用

或擷取叢集目前的平台版 AWS CLI 本 AWS Management Console。如果您已開啟本機叢集 AWS Outposts，請參閱Amazon EKS 本機叢集平台版本而非本主題。

當 Amazon EKS 中有新的Kubernetes次要版本（例如 1.30）時，該次Kubernetes要版本的初始 Amazon EKS 平台版本將從開始。`eks.1`但是，Amazon EKS 會定期發佈新平台版本，以啟用新的 Kubernetes 控制平面設定，以及提供安全性問題修正。

當有新的 Amazon EKS 平台版本可供次要版本使用時：

- Amazon EKS 平台版本編號將會遞增 (`eks.`*`n+1`*)。
- Amazon EKS 會自動將所有現有的叢集升級到與其對應之 Kubernetes 次要版本的最新 Amazon EKS 平台版本。現有 Amazon EKS 平台版本的自動升級將會逐步發佈。發行程序可能需要一些時間。如果您立刻需要最新 Amazon EKS 平台版本的功能，您應該建立新的 Amazon EKS 叢集。

  如果您的叢集落後於目前平台版本超過兩個平台版本，則 Amazon EKS 可能無法自動更新叢集。如需可能導致此問題的詳細資訊，請參閱 Amazon EKS 平台版本比目前平台版本落後兩個版本以上。
- Amazon EKS 可能會發佈具有對應之修補程式版本的新節點 AMI。但是，對於指定的 Kubernetes 次要版本而言，所有修補程式版本在 EKS 控制平面和節點 AMI 之間都是相容的。

新的 Amazon EKS 平台版本不會帶來重大改變或導致服務中斷。

叢集一律會透過指定 Kubernetes 版本的最新可用 Amazon EKS 平台版本 (`eks.`*`n`*) 建立。如果您將叢集更新為新的 Kubernetes 次要版本，您的叢集會收到適用於您要更新之 Kubernetes 次要版本的目前 Amazon EKS 平台版本。

Amazon EKS 的目前及最新平台版本如下表所示。

Kubernetes 版本 **`1.30`**

下列許可控制器已為所有 `1.30` 平台版本啟用：`NodeRestriction`、`ExtendedResourceToleration`、`NamespaceLifecycle`、`LimitRanger`、

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
`1.30.1`	`eks.3`	具有安全性修正和增強功能的新平台版本。	2024年6月7日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.30.0	eks.2	適用於 EKS 的庫伯尼特版本的初始版本1.30。如需詳細資訊，請參閱Kubernetes1.30。	2024年5月23 日

Kubernetes 版本 1.29

下列許可控制器已為所有 1.29 平台版本啟用：NodeRestriction、ExtendedResourceToleration、NamespaceLifecycle、LimitRanger、

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.29.5	eks.8	具有安全性修正和增強功能的新平台版本。	2024年6月7日
1.29.4	eks.7	具有 CoreDNS 自動調度資源、安全性修正和增強功能的新平台版本。如需 CoreDNS 自動調度資源的詳細資訊，請參閱。自動調度 CoreDNS	2024年5月16日
1.29.3	eks.6	具有安全性修正和增強功能的新平台版本。	2024年4月18日
1.29.1	eks.5	具有安全性修正和增強功能的新平台版本。	2024年3月29 日
1.29.1	eks.4	具有安全性修正和增強功能的新平台版本。	2024年3月20日
1.29.1	eks.3	具有安全性修正和增強功能的新平台版本。	2024年3月12日
1.29.0	eks.1	適用於 EKS 的庫伯尼特版本的初始版本1.29。如需詳細資訊，請參閱Kubernetes1.29。	2024 年 1 月 23 日

Kubernetes 版本 **1.28**

下列許可控制器已為所有 1.28 平台版本啟
用：NodeRestriction、ExtendedResourceToleration、NamespaceLifecycle、LimitRanger、

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.28.10	eks.14	具有安全性修正和增強功能的新平台版本。	2024年6月7日
1.28.9	eks.13	具有 CoreDNS 自動調度資源、安全性修正和增強功能的新平台版本。如需 CoreDNS 自動調度資源的詳細資訊，請參閱。自動調度 CoreDNS	2024年5月16日
1.28.8	eks.12	具有安全性修正和增強功能的新平台版本。	2024年4月18日
1.28.7	eks.11	具有安全性修正和增強功能的新平台版本。	2024年3月29 日
1.28.7	eks.10	具有安全性修正和增強功能的新平台版本。	2024年3月20日
1.28.6	eks.9	具有安全性修正和增強功能的新平台版本。	2024年3月12日
1.28.5	eks.7	具有安全性修正和增強功能的新平台版本。	2024年1月17日
1.28.4	eks.6	具有存取項目功能、安全性修正和增強功能的新平台版本。	2023 年 12 月 14 日
1.28.4	eks.5	具有安全性修正和增強功能的新平台版本。	2023 年 12 月 12 日
1.28.3	eks.4	具有 EKS Pod 身分識別、安全性修正和增強功能的新平台版本。	2023 年 11 月 10 日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.28.3	eks.3	具有安全性修正和增強功能的新平台版本。	2023 年 11 月 3 日
1.28.2	eks.2	具有安全性修正和增強功能的新平台版本。	2023 年 10 月 16 日
1.28.1	eks.1	適用於 EKS 的庫伯尼特版本的初始版本1.28。如需詳細資訊，請參閱 Kubernetes1.28。	2023 年 9 月 26 日

Kubernetes 版本 **1.27**

下列許可控制器已為所有 1.27 平台版本啟用：NodeRestriction、ExtendedResourceToleration、NamespaceLifecycle、LimitRanger、

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.27.14	eks.18	具有安全性修正和增強功能的新平台版本。	2024年6月7日
1.27.13	eks.17	具有 CoreDNS 自動調度資源、安全性修正和增強功能的新平台版本。如需 CoreDNS 自動調度資源的詳細資訊，請參閱。自動調度 CoreDNS	2024年5月16日
1.27.12	eks.16	具有安全性修正和增強功能的新平台版本。	2024年4月18日
1.27.11	eks.15	具有安全性修正和增強功能的新平台版本。	2024年3月29 日
1.27.11	eks.14	具有安全性修正和增強功能的新平台版本。	2024年3月20日
1.27.10	eks.13	具有安全性修正和增強功能的新平台版本。	2024年3月12日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.27.9	eks.11	具有安全性修正和增強功能的新平台版本。	2024年1月17日
1.27.8	eks.10	具有存取項目功能、安全性修正和增強功能的新平台版本。	2023 年 12 月 14 日
1.27.8	eks.9	具有安全性修正和增強功能的新平台版本。	2023 年 12 月 12 日
1.27.7	eks.8	具有 EKS Pod 身分識別、安全性修正和增強功能的新平台版本。	2023 年 11 月 10 日
1.27.7	eks.7	具有安全性修正和增強功能的新平台版本。	2023 年 11 月 3 日
1.27.6	eks.6	具有安全性修正和增強功能的新平台版本。	2023 年 10 月 16 日
1.27.4	eks.5	具有安全性修正和增強功能的新平台版本。	2023 年 8 月 30 日
1.27.4	eks.4	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 30 日
1.27.3	eks.3	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 30 日
1.27.2	eks.2	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 9 日
1.27.1	eks.1	適用於 EKS 的庫伯尼特版本的初始版本1.27。如需詳細資訊，請參閱Kubernetes1.27。	2023 年 5 月 24 日

Kubernetes 版本 1.26

下列許可控制器已為所有 1.26 平台版本啟
用：NodeRestriction、ExtendedResourceToleration、NamespaceLifecycle、LimitRanger、

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.26.15	eks.19	具有安全性修正和增強功能的新平台版本。	2024年6月7日
1.26.15	eks.18	具有 CoreDNS 自動調度資源、安全性修正和增強功能的新平台版本。如需 CoreDNS 自動調度資源的詳細資訊，請參閱。自動調度 CoreDNS	2024年5月16日
1.26.15	eks.17	具有安全性修正和增強功能的新平台版本。	2024年4月18日
1.26.14	eks.16	具有安全性修正和增強功能的新平台版本。	2024年3月29 日
1.26.14	eks.15	具有安全性修正和增強功能的新平台版本。	2024年3月20日
1.26.13	eks.14	具有安全性修正和增強功能的新平台版本。	2024年3月12日
1.26.12	eks.12	具有安全性修正和增強功能的新平台版本。	2024年1月17日
1.26.11	eks.11	具有存取項目功能、安全性修正和增強功能的新平台版本。	2023 年 12 月 14 日
1.26.11	eks.10	具有安全性修正和增強功能的新平台版本。	2023 年 12 月 12 日
1.26.10	eks.9	具有 EKS Pod 身分識別、安全性修正和增強功能的新平台版本。	2023 年 11 月 10 日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.26.10	eks.8	具有安全性修正和增強功能的新平台版本。	2023 年 11 月 3 日
1.26.9	eks.7	具有安全性修正和增強功能的新平台版本。	2023 年 10 月 16 日
1.26.7	eks.6	具有安全性修正和增強功能的新平台版本。	2023 年 8 月 30 日
1.26.7	eks.5	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 30 日
1.26.6	eks.4	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 30 日
1.26.5	eks.3	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 9 日
1.26.4	eks.2	具有安全性修正和增強功能的新平台版本。	2023 年 5 月 5 日
1.26.2	eks.1	適用於 EKS 的庫伯尼特版本的初始版本1.26。如需詳細資訊，請參閱 Kubernetes1.26。	2023 年 4 月 11 日

Kubernetes 版本 **1.25**

下列許可控制器已為所有 1.25 平台版本啟用：NodeRestriction、ExtendedResourceToleration、NamespaceLifecycle、LimitRanger、

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.25.16	eks.20	具有安全性修正和增強功能的新平台版本。	2024年6月7日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.25.16	eks.19	具有 CoreDNS 自動調度資源、安全性修正和增強功能的新平台版本。如需 CoreDNS 自動調度資源的詳細資訊，請參閱。自動調度 CoreDNS	2024年5月16日
1.25.16	eks.18	具有安全性修正和增強功能的新平台版本。	2024年4月18日
1.25.16	eks.17	具有安全性修正和增強功能的新平台版本。	2024年3月29 日
1.25.16	eks.16	具有安全性修正和增強功能的新平台版本。	2024年3月20日
1.25.16	eks.15	具有安全性修正和增強功能的新平台版本。	2024年3月12日
1.25.16	eks.13	具有安全性修正和增強功能的新平台版本。	2024年1月17日
1.25.16	eks.12	具有存取項目功能、安全性修正和增強功能的新平台版本。	2023 年 12 月 14 日
1.25.16	eks.11	具有安全性修正和增強功能的新平台版本。	2023 年 12 月 12 日
1.25.15	eks.10	具有 EKS Pod 身分識別、安全性修正和增強功能的新平台版本。	2023 年 11 月 10 日
1.25.15	eks.9	具有安全性修正和增強功能的新平台版本。	2023 年 11 月 3 日
1.25.14	eks.8	具有安全性修正和增強功能的新平台版本。	2023 年 10 月 16 日
1.25.12	eks.7	具有安全性修正和增強功能的新平台版本。	2023 年 8 月 30 日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.25.12	eks.6	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 30 日
1.25.11	eks.5	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 30 日
1.25.10	eks.4	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 9 日
1.25.9	eks.3	具有安全性修正和增強功能的新平台版本。	2023 年 5 月 5 日
1.25.8	eks.2	具有安全性修正和增強功能的新平台版本。	2023 年 3 月 24 日
1.25.6	eks.1	適用於 EKS 的庫伯尼特版本的初始版本1.25。如需詳細資訊，請參閱 Kubernetes1.25。	2023 年 2 月 21 日

Kubernetes 版本 **1.24**

下列許可控制器已為所有 1.24 平台版本啟用：CertificateApproval、CertificateSigning、CertificateSubjectRestriction、Defaul以及 ValidatingAdmissionWebhook。

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.24.17	eks.23	具有安全性修正和增強功能的新平台版本。	2024年6月7日
1.24.17	eks.22	具有安全性修正和增強功能的新平台版本。	2024年5月16日
1.24.17	eks.21	具有安全性修正和增強功能的新平台版本。	2024年4月18日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.24.17	eks.20	具有安全性修正和增強功能的新平台版本。	2024年3月29 日
1.24.17	eks.19	具有安全性修正和增強功能的新平台版本。	2024年3月20日
1.24.17	eks.18	具有安全性修正和增強功能的新平台版本。	2024年3月12日
1.24.17	eks.16	具有安全性修正和增強功能的新平台版本。	2024年1月17日
1.24.17	eks.15	具有存取項目功能、安全性修正和增強功能的新平台版本。	2023 年 12 月 14 日
1.24.17	eks.14	具有安全性修正和增強功能的新平台版本。	2023 年 12 月 12 日
1.24.17	eks.13	具有 EKS Pod 身分識別、安全性修正和增強功能的新平台版本。	2023 年 11 月 10 日
1.24.17	eks.12	具有安全性修正和增強功能的新平台版本。	2023 年 11 月 3 日
1.24.17	eks.11	具有安全性修正和增強功能的新平台版本。	2023 年 10 月 16 日
1.24.16	eks.10	具有安全性修正和增強功能的新平台版本。	2023 年 8 月 30 日
1.24.16	eks.9	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 30 日
1.24.15	eks.8	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 30 日
1.24.14	eks.7	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 9 日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.24.13	eks.6	具有安全性修正和增強功能的新平台版本。	2023 年 5 月 5 日
1.24.12	eks.5	具有安全性修正和增強功能的新平台版本。	2023 年 3 月 24 日
1.24.8	eks.4	具有安全性修正和增強功能的新平台版本。	2023 年 1 月 27 日
1.24.7	eks.3	具有安全性修正和增強功能的新平台版本。	2022 年 12 月 5 日
1.24.7	eks.2	具有安全性修正和增強功能的新平台版本。	2022 年 11 月 18 日
1.24.7	eks.1	適用於 EKS 的庫伯尼特版本的初始版本1.24。如需詳細資訊，請參閱Kubernetes1.24。	2022 年 11 月 15 日

Kubernetes 版本 **1.23**

下列許可控制器已為所有 1.23 平台版本啟用：CertificateApproval、CertificateSigning、CertificateSubjectRestriction、Defaul以及 ValidatingAdmissionWebhook。

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.23.17	eks.25	具有安全性修正和增強功能的新平台版本。	2024年6月7日
1.23.17	eks.24	具有安全性修正和增強功能的新平台版本。	2024年5月16日
1.23.17	eks.23	具有安全性修正和增強功能的新平台版本。	2024年4月18日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.23.17	eks.22	具有安全性修正和增強功能的新平台版本。	2024年3月29 日
1.23.17	eks.21	具有安全性修正和增強功能的新平台版本。	2024年3月20日
1.23.17	eks.20	具有安全性修正和增強功能的新平台版本。	2024年3月12日
1.23.17	eks.18	具有安全性修正和增強功能的新平台版本。	2024年1月17日
1.23.17	eks.17	具有存取項目功能、安全性修正和增強功能的新平台版本。	2023 年 12 月 14 日
1.23.17	eks.16	具有安全性修正和增強功能的新平台版本。	2023 年 12 月 12 日
1.23.17	eks.15	具有安全性修正和增強功能的新平台版本。	2023 年 11 月 10 日
1.23.17	eks.14	具有安全性修正和增強功能的新平台版本。	2023 年 11 月 3 日
1.23.17	eks.13	具有安全性修正和增強功能的新平台版本。	2023 年 10 月 16 日
1.23.17	eks.12	具有安全性修正和增強功能的新平台版本。	2023 年 8 月 30 日
1.23.17	eks.11	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 30 日
1.23.17	eks.10	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 30 日
1.23.17	eks.9	具有安全性修正和增強功能的新平台版本。	2023 年 6 月 9 日

Kubernetes 版本	EKS 平台版本	版本備註	版本日期
1.23.17	eks.8	具有安全性修正和增強功能的新平台版本。	2023 年 5 月 5 日
1.23.17	eks.7	具有安全性修正和增強功能的新平台版本。	2023 年 3 月 24 日
1.23.14	eks.6	具有安全性修正和增強功能的新平台版本。	2023 年 1 月 27 日
1.23.13	eks.5	具有安全性修正和增強功能的新平台版本。	2022 年 12 月 5 日
1.23.13	eks.4	具有安全性修正和增強功能的新平台版本。	2022 年 11 月 18 日
1.23.12	eks.3	具有安全性修正和增強功能的新平台版本。	2022 年 11 月 7 日
1.23.10	eks.2	具有安全性修正和增強功能的新平台版本。	2022 年 9 月 21 日
1.23.7	eks.1	適用於 EKS 的庫伯尼特版本的初始版本1.23。如需詳細資訊，請參閱Kubernetes1.23。	2022 年 8 月 11 日

取得目前的平台版本

取得叢集的目前平台版本 (主控台)

1. 開啟 Amazon EKS 主控台。
2. 在導覽窗格中，選擇叢集。
3. 在叢集清單中，選擇要檢查平台版本的叢集名稱。
4. 選擇 Overview (概觀) 索引標籤。
5. 「平台版本」位於「詳細資料」區段中。

取得叢集 (AWS CLI) 的目前平台版本

1. 決定您要檢查平台版本之叢集的名稱。
2. 執行以下命令：

```
aws eks describe-cluster --name my-cluster --query cluster.platformVersion
```

範例輸出如下。

```
"eks.10"
```

自動擴展

自動擴展功能可自動將您的資源向內外擴展，滿足不斷變化的需求。這是 Kubernetes 的主要功能，否則需要大量人力資源才能手動執行。

Amazon EKS 支援兩種自動擴展產品：

Karpenter

Karpenter 是靈活、高效能的 Kubernetes 叢集自動擴展工具，有助於提高應用程式可用性和叢集效率。Karpenter 會啟動大小適中的運算資源 (例如 Amazon EC2 執行個體)，以便在一分鐘內回應不斷變化的應用程式負載。透過將 Kubernetes 與 AWS 整合，Karpenter 可以佈建精確符合工作負載需求的即時運算資源。Karpenter 可根據叢集工作負載的特定需求，自動佈建新的運算資源。其中包括運算、儲存、加速和排程需求。Amazon EKS 支援使用 Karpenter 的叢集，然而 Karpenter 可與所有符合標準的 Kubernetes 叢集搭配使用。如需詳細資訊，請參閱 Karpenter 文件。

Cluster Autoscaler

當 Pod 故障或重新排程到其他節點時，Kubernetes Cluster Autoscaler 會自動調整叢集中的節點數目。Cluster Autoscaler 使用 Auto Scaling 群組。如需詳細資訊，請參閱 AWS 上的 Cluster Autoscaler。

管理存取

了解如何管理 Amazon EKS 叢集的存取權限。使用 Amazon EKS 需要Kubernetes和 AWS Identity and Access Management (AWS IAM) 如何處理存取控制的知識。

本節包括：

the section called “授與庫伯內特 API 的存取權”— 瞭解如何讓應用程式或使用者對 Kubernetes API 進行驗證。您可以使用存取項目、aws-auth ConfigMap 或外部 OIDC 提供者。

the section called “使用 kubectl 存取我的叢集”— 了解如何設定 kubectl 與您的 Amazon EKS 叢集通訊。使用 AWS CLI 建立庫貝設定檔案。

the section called “授予工作負載存取權 AWS”— 了解如何將Kubernetes服務帳戶與 AWS IAM 角色建立關聯。您可以針對服務帳戶 (IRSA) 使用網繭身分識別或 IAM 角色。

一般工作：

- 授予開發人員存取 Kubernetes API 的權限。檢視中的Kubernetes資源 AWS Management Console。
  - 解決方案：使用存取項目將 Kubernetes RBAC 許可與 AWS IAM 使用者或角色建立關聯。
- 將 kubectl 設定為使用登入資料與 Amazon EKS 叢集通訊。 AWS
  - 解決方案：使用 AWS CLI 建立 Kubeconfig 檔案。
- 使用外部身分識別提供者 (例如 Ping 身分識別) 來驗證 Kubernetes API 的使用者。
  - 解決方案：連結外部 OIDC 提供者。
- 授與Kubernetes叢集上的工作負載呼叫 AWS API 的能力。
  - 解決方案：使用網繭身分將 AWS IAM 角色與Kubernetes服務帳戶建立關聯。

背景:

- 瞭解 Kubernetes 服務帳戶的運作方式。
- 檢閱以 Kubernetes 角色為基礎的存取控制 (RBAC) 模型
- 如需管理資源存取權的詳細 AWS 資訊，請參閱 AWS IAM 使用者指南。或者，接受有關使用 AWS IAM 的免費入門培訓。

授予 Kubernetes API 的存取權

您的叢集具有 Kubernetes API 端點。庫貝克特爾使用此 API。您可以使用兩種身分識別類型對此 API 進行驗證：

- AWS Identity and Access Management (IAM) 主體 (角色或使用者) — 此類型需要 IAM 進行身份驗證。使用者可以使用透過 AWS 身分識別來源提供的登入資料，以 IAM 使用者身分或使用聯合身分識別登入。如果管理員先前已設定使用 IAM 角色的聯合身分，則使用者只能夠以聯合身分登入。當使用者使 AWS 用同盟存取時，他們會間接擔任角色。當使用者使用此類身分時，您：
  - 可以為其指派 Kubernetes 許可，從而讓其能夠使用您叢集上的 Kubernetes 物件。如需進一步了解如何向 IAM 主體指派許可以便相應主體能夠存取您叢集上的 Kubernetes 物件，請參閱 管理存取項目。
  - 可以為他們指派 IAM 許可，以便他們可以使用 Amazon EKS API、、、 AWS CLI或與您的 Amazon EKS 叢集及其資源搭配使用。 AWS CloudFormation AWS Management Consoleeksctl如需詳細資訊，請參閱《服務授權參考》中的 Amazon Elastic Kubernetes Service 定義的動作一節。
  - 節點透過承擔 IAM 角色來加入叢集。AWS IAM Authenticator for Kubernetes 在 Amazon EKS 控制平面上執行，讓使用者能夠使用 IAM 主體存取您的叢集。
- 您自己的 OpenID Connect (OIDC) 提供者中的使用者：此類型需要向 OIDC 提供者進行身分驗證。如需進一步了解如何向 Amazon EKS 叢集設定您自己的 OIDC 提供者，請參閱 從OpenID Connect 身分識別提供者驗證叢集的使用者。當使用者使用此類身分時，您：
  - 可以為其指派 Kubernetes 許可，從而讓其能夠使用您叢集上的 Kubernetes 物件。
  - 無法為他們指派 IAM 許可，以便他們可以使用 Amazon EKS API、、、 AWS CLI或與您的 Amazon EKS 叢集及其資源搭配使用。 AWS CloudFormation AWS Management Consoleeksctl

您可以在叢集中使用這兩種類型的身分。無法停用 IAM 身份驗證方法。OIDC 驗證方法是選用的。

將身分識別與存取權管理權限建立關聯

AWS IAM Authenticator for Kubernetes 安裝在叢集的控制平面上。它啟用您允許存取叢集上的 Kubernetes 資源的 AWS Identity and Access Management (IAM) 主體 (角色和使用者)。您可以使用以下方法之一允許 IAM 主體存取叢集上的 Kubernetes 物件：

- 建立存取項目：如果叢集版本等於或高於叢集 Kubernetes 版本先決條件部分中列出的平台版本，則建議您使用此選項。

使用存取項目從叢集外部管理 IAM 主體的 Kubernetes 許可。您可以使用 EKS API、、 AWS Command Line Interface AWS SDK 和來新增和管理叢集的存取。 AWS CloudFormation AWS Management Console這意味著您可以使用建立叢集時使用的工具來管理使用者。

若要開始，請遵循 設定存取項目，然後 遷移現有 aws-auth ConfigMap 項目至存取項目。

- 新增項目到 **aws-auth ConfigMap**：如果叢集的平台版本低於先決條件部分中列出的版本，則必須使用此選項。如果叢集的平台版本等於或高於叢集 Kubernetes 版本先決條件部分中列出的平台版本，並且您已將項目新增至 ConfigMap，則建議您將這些項目遷移到存取項目。但是，您無法遷移 Amazon EKS 新增至 ConfigMap 的項目，例如用於受管節點群組或 Fargate 描述檔的 IAM 角色項目。如需詳細資訊，請參閱 the section called “授與庫伯內特 API 的存取權”。
  - 如果必須使用 aws-auth ConfigMap 選項，則可以使用 **eksctl create iamidentitymapping** 命令將項目新增至 ConfigMap。如需詳細資訊，請參閱 eksctl 文件中的 Manage IAM users and roles 一節。

設定叢集驗證模式

每個叢集都有自己的身分驗證模式。身分驗證模式決定您可以使用哪些方法來允許 IAM 主體存取叢集上的 Kubernetes 物件。身分驗證模式有 3 種。

> **Important**
>
> 一旦啟用存取輸入方法，就無法將其停用。
> 如果在叢集建立期間未啟用此ConfigMap方法，則稍後將無法啟用。在引入存取項目之前建立的所有叢集都會啟用此ConfigMap方法。

叢集內的 aws-auth ConfigMap

這是 Amazon EKS 叢集的原始身分驗證模式。建立叢集的 IAM 主體是可以使用 kubectl 存取叢集的初始使用者。初始使用者必須將其他使用者新增至 aws-auth ConfigMap 的清單中，並為這些使用者指派相應許可。其他使用者無法管理或移除初始使用者，因為 ConfigMap 中沒有要管理的項目。

ConfigMap 和存取項目

使用這種身分驗證模式時，您可以使用這兩種方法將 IAM 主體新增至叢集。請注意，每個方法都會儲存不同的項目；例如，如果您從中新增存取項目 AWS CLI，aws-authConfigMap則不會更新。

僅存取項目

透過此驗證模式，您可以使用 EKS API AWS Command Line Interface、 AWS SDK AWS CloudFormation，以及 AWS Management Console 管理 IAM 主體對叢集的存取。

每個存取項目都有一個類型，您可以使用存取範圍的組合將主體限制為特定命名空間，並使用存取政策來設定預先設定的可重複使用許可政策。您也可以使用 STANDARD 類型和 Kubernetes RBAC 群組來指派自訂許可。

身分驗證方式	方法
僅 ConfigMap (CONFIG_MAP)	aws-auth ConfigMap
EKS API 和 ConfigMap (API_AND_CONFIG_MAP)	存取 EKS API、 AWS Command Line Interface AWS 開發套件和和中的項目 AWS CloudFormation AWS Management Console aws-auth ConfigMap
僅 EKS API (API)	存取 EKS API、 AWS Command Line Interface AWS 開發套件和中的項目 AWS CloudFormation AWS Management Console

管理存取項目

必要條件

- 熟悉 Amazon EKS 叢集的叢集存取選項。如需詳細資訊，請參閱 授予 Kubernetes API 的存取權 。
- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。若要使用存取項目和變更叢集的身分驗證模式，叢集的平台版本必須與下表所列版本相同或更新，或是晚於表格中列出版本的 Kubernetes 版本。

Kubernetes 版本	平台版本
1.30	eks.2
1.29	eks.1

Kubernetes 版本	平台版本
1.28	eks.6
1.27	eks.10
1.26	eks.11
1.25	eks.12
1.24	eks.15
1.23	eks.17

若要檢查目前的 Kubernetes 和平台版本，您可以使用叢集名稱取代以下命令中的 *my-cluster*，然後執行修改後的命令：**aws eks describe-cluster --name *my-cluster* --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'**。

> **Important**
>
> 在 Amazon EKS 將叢集更新至表中列出的平台版本後，Amazon EKS 會為最初建立叢集的 IAM 主體，建立一個具有叢集管理員許可的存取項目。如果您不希望該 IAM 主體擁有叢集的管理員許可，請移除 Amazon EKS 建立的存取項目。
> 對於平台版本早於上表所列平台版本的叢集，叢集建立者始終為叢集管理員。您無法從建立叢集的 IAM 使用者或角色中移除叢集管理員許可。

- 對叢集具有以下許可的 IAM 主體：`CreateAccessEntry`、`ListAccessEntries`、`DescribeAccessEntry`、`DeleteAccessEntry` 和 `UpdateAccessEntry`。如需 Amazon EKS 許可的詳細資訊，請參閱《服務授權參考》中的 Amazon Elastic Kubernetes Service 定義的動作一節。
- 要為其建立存取項目的現有 IAM 主體，或要更新或刪除的現有存取項目。

設定存取項目

若要開始使用存取項目，您必須將叢集的身分驗證模式變更為 `API_AND_CONFIG_MAP` 或 `API` 模式。這會新增用於存取項目的 API。

AWS Management Console

建立存取項目

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇要在其中建立存取項目的叢集名稱。
3. 選擇存取索引標籤。
4. 身分驗證模式顯示叢集目前的身分驗證模式。如果模式顯示 EKS API，則表示您已經可以新增存取項目，您可以略過餘下步驟。
5. 選擇管理存取。
6. 對於叢集身分驗證模式，請選取帶有 EKS API 的模式。請注意，您無法將身分驗證模式變更回移除 EKS API 和存取項目的模式。
7. 選擇儲存變更。Amazon EKS 開始更新叢集，叢集的狀態變更為 Updating，並且將變更記錄在更新歷史記錄索引標籤中。
8. 等待叢集狀態變回 Active。當叢集的狀態為 Active 時，您可以依照 建立存取項目 中的步驟為 IAM 主體新增對叢集的存取權。

AWS CLI

先決條件

在您的裝置或上安裝和設定的 AWS CLI v1 的最新版本 AWS CloudShell。 AWS CLI v2 幾天不支持新功能。您可以使用 `aws --version | cut -d / -f2 | cut -d ' ' -f1` 來檢查您的目前版本。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《 AWS Command Line Interface 使用指南》`aws configure`中的〈安裝、更新 AWS CLI和解除安裝〉和〈快速設定〉。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《 AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

1.
2. 執行下列命令。使用您叢集的名稱取代 *`my-cluster`*。如果您想永久停用 `ConfigMap` 方法，則請用 `API` 取代 `API_AND_CONFIG_MAP`。

   Amazon EKS 開始更新叢集，叢集的狀態變更為 UPDATING，並且變更記錄在 aws eks list-updates 中。

```
aws eks update-cluster-config --name my-cluster --access-config
 authenticationMode=API_AND_CONFIG_MAP
```

3. 等待叢集狀態變回 Active。當叢集的狀態為 Active 時，您可以依照 建立存取項目 中的步驟為 IAM 主體新增對叢集的存取權。

建立存取項目

考量事項

在建立存取項目之前，請考慮以下事項：

- 存取項目包含一個 (且僅一個) 現有 IAM 主體的 Amazon Resource Name (ARN)。一個 IAM 主體不能包含在多個存取項目中。對於您指定的 ARN 的其他考量：
  - IAM 最佳實務建議使用具有短期憑證的 IAM 角色存取叢集，而不是使用具有長期憑證的 IAM 使用者進行存取。如需詳細資訊，請參閱 IAM 使用者指南中的要求人類使用者與身分識別提供者的聯合以 AWS 使用臨時登入資料存取。
  - 如果 ARN 用於 IAM 角色，則它可以包含路徑。`aws-auth ConfigMap` 項目中的 ARN 則不能包含路徑。例如， ARN 可以是 `arn:aws:iam::111122223333:role/development/apps/my-role` 或 `arn:aws:iam::111122223333:role/my-role`。
  - 如果存取項目的類型不是 STANDARD (請參閱下一個關於類型的考量)，ARN 必須與叢集 AWS 帳戶 所在的相同。如果類型為STANDARD，則 ARN 可以與叢集所在 AWS 帳戶 的帳戶相同或不同。
  - 建立存取項目後，您無法變更該 IAM 主體。
  - 如果您刪除具有此 ARN 的 IAM 主體，則存取項目並不會自動刪除。建議您刪除帶有您刪除之 IAM 主體的 ARN 的存取項目。如果您在不刪除存取項目的情況下重新建立 IAM 主體，則即使它具有相同的 ARN，存取項目也將無法運作。這是因為即使 ARN 對於重新建立的 IAM 主體是相同的，`roleID`或 `userID` (您可以使用`aws sts get-caller-identity` AWS CLI 命令看到) 對於重新建立的 IAM 主體而言，與原始 IAM 主體不同。即使您看不到存取項目的 IAM 主體 `roleID` 或 `userID`，Amazon EKS 也會將其與存取項目一起儲存。
- 每個存取項目都有一個類型。您可以指定 `EC2 Linux` (針對搭配 Linux 或保登機自我管理節點使用的 IAM 角色)、`EC2 Windows` (適用於搭配 Windows 自我管理節點使用的 IAM 角色)、`FARGATE_LINUX` (適用於搭配使用的 IAM 角色 AWS Fargate (Fargate)) 或做為類型。`STANDARD`如果您不指定類型，則 Amazon EKS 會自動將類型設為 STANDARD。無需為用於受管節點群組或 Fargate 描述檔的 IAM 角色建立存取項目，因為無論叢集位於哪個平台版本，Amazon EKS 都會將這些角色的項目新增至 `aws-auth ConfigMap`。

建立存取項目後，您無法變更類型。

- 如果某個存取項目的類型為 STANDARD，則可以為其指定一個使用者名稱。如果您沒有指定使用者名稱的值，Amazon EKS 會根據存取項目的類型以及您指定的 IAM 主體是 IAM 角色還是 IAM 使用者，設定下列其中一個值。除非您有特定原因需要自己指定使用者名稱，否則建議您不要指定使用者名稱，而是讓 Amazon EKS 自動為您產生該使用者名稱。如果您自己指定使用者名稱：
  - 該名稱不能以 `system:`、`eks:`、`aws:`、`amazon:` 或 `iam:` 開頭。
  - 如果該使用者名稱用於 IAM 角色，則建議您在末尾新增 `{{SessionName}}`。如果您添加`{{SessionName}}`到您的用戶名，用戶名必須在 {{SessionName}} 之前包含冒號。假設此角色時，假設角色時指定的工作階段名稱會自動傳遞至叢集，並會出現在 CloudTrail 記錄檔中。例如，使用者名稱不能為 `john{{SessionName}}`。使用者名稱必須為 `:john{{SessionName}}` 或 `jo:hn{{SessionName}}`。冒號只需位於 `{{SessionName}}` 之前。以下資料表中 Amazon EKS 產生的使用者名稱包含 ARN。由於 ARN 包含冒號，因此它符合此項要求。如果使用者名稱中不包含 `{{SessionName}}`，則不需要冒號。

IAM 主體類型	Type	Amazon EKS 自動設定的使用者名稱值
使用者	STANDARD	使用者的 ARN。範例：`arn:aws:iam::` *111122223333* `:user/`*my-user*
角色	STANDARD	角色被擔任時的 STS ARN。Amazon EKS 將 `{{SessionName}}` 附加到角色。 範例：`arn:aws:sts::` *111122223333* `:assumed-role/` *my-role*`/{{SessionName}}` 如果您指定的角色的 ARN 包含路徑，則 Amazon EKS 會將其從產生的使用者名稱中移除。

IAM 主體類型	Type	Amazon EKS 自動設定的使用者名稱值
角色	EC2 Linux 或 EC2 Windows	system:node:{{EC2PrivateDNSName}}
角色	FARGATE_LINUX	system:node:{{SessionName}}

建立存取項目後，您可以對使用者名稱進行變更。

- 如果某個存取項目的類型為 STANDARD，且您希望使用 Kubernetes RBAC 授權，則可以在該存取項目中新增一個或多個群組名稱。建立存取項目後，您可以新增和移除群組名稱。為了使 IAM 主體能夠存取叢集上的 Kubernetes 物件，您必須建立和管理 Kubernetes 角色型授權 (RBAC) 物件。在叢集上建立將群組名稱指定為 kind: Group 的 subject 的 Kubernetes RoleBinding 或 ClusterRoleBinding 物件。Kubernetes 授權 IAM 主體存取您在 Kubernetes Role 或 ClusterRole 物件中 (也在繫結的 roleRef 中) 指定的任何叢集物件。如果您指定群組名稱，則建議您熟悉 Kubernetes 角色型授權 (RBAC) 物件。如需詳細資訊，請參閱 Kubernetes 文件中的使用 RBAC 授權。

  > **Important**
  >
  > Amazon EKS 不會確認叢集上存在的任何 Kubernetes RBAC 物件是否包含您指定的任何群組名稱。

  您可以將 Amazon EKS 存取政策關聯到存取項目，而不是讓 Kubernetes 授權 IAM 主體存取叢集上的 Kubernetes 物件。Amazon EKS 授權 IAM 主體使用存取政策中的許可存取叢集上的 Kubernetes 物件。您可以將存取政策的許可範圍限定為您指定的 Kubernetes 命名空間。使用存取政策無需您管理 Kubernetes RBAC 物件。如需詳細資訊，請參閱 將存取政策與存取項目關聯和取消關聯。
- 如果您建立類型為 EC2 Linux 或 EC2 Windows 的存取項目，則建立該存取項目的 IAM 主體必須具有 iam:PassRole 許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 Granting a user permissions to pass a role to an AWS 服務 一節。
- 與標準 IAM 行為類似，存取項目的建立和更新採用最終一致模式，在初始 API 呼叫成功返回後可能需要幾秒鐘才能生效。您設計的應用程式必須能夠處理這些可能的延遲問題。建議您不要在應用程式的關鍵、高可用性程式碼路徑中包含存取項目建立或更新動作。而應在不常運作的、單獨的初始化或設定常式中進行 更改。另外，在生產工作流程套用這些變更之前，請務必確認變更已傳播完畢。

- 存取項目不支援服務連結的角色。您無法建立主參與者 ARN 是服務連結角色的存取項目。您可以依據服務連結角色的 ARN 來識別服務連結角色，格`arn:aws:iam::*:role/aws-service-role/*`式為。

您可以使用 AWS Management Console 或建立存取項目 AWS CLI。

AWS Management Console

建立存取項目

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇要在其中建立存取項目的叢集名稱。
3. 選擇存取索引標籤。
4. 選擇建立存取項目。
5. 對於 IAM 主體，選取現有 IAM 角色或使用者。IAM 最佳實務建議使用具有短期憑證的 IAM 角色存取叢集，而不是使用具有長期憑證的 IAM 使用者進行存取。如需詳細資訊，請參閱 IAM 使用者指南中的要求人類使用者與身分識別提供者的聯合以 AWS 使用臨時登入資料存取。
6. 對於類型，如果存取項目針對用於自我管理 Amazon EC2 節點的節點角色，請選取 EC2 Linux 或 EC2 Windows。否則，請接受預設值 (標準)。
7. 如果您選擇的類型是標準並且您想要指定使用者名稱，則請輸入使用者名稱。
8. 如果您選擇的類型是標準並且您想要對 IAM 主體使用 Kubernetes RBAC 授權，則請為群組指定一個或多個名稱。如果不指定任何群組名稱並希望使用 Amazon EKS 授權，則您可以在後續步驟中或在建立存取項目後關聯存取政策。
9. (選用) 您可以使用標籤為存取項目指派標籤。例如，為了更輕鬆地找到具有相同標籤的所有資源而指定標籤。
10. 選擇下一步。
11. 在新增存取政策頁面上，如果您選擇的類型是標準並且希望 Amazon EKS 授予該 IAM 主體對叢集上的 Kubernetes 物件的許可，請完成下列步驟。否則請選擇 Next (下一步)。/

    a. 對於政策名稱，請選擇存取政策。您無法檢視存取政策的許可，但其包含與 Kubernetes 面向使用者之 `ClusterRole` 物件中的許可類似的許可。如需詳細資訊，請參閱 Kubernetes 文件中的 User-facing roles 一節。

    b. 請選擇下列其中一個選項：

- 叢集：如果您希望 Amazon EKS 授予該 IAM 主體存取政策中針對叢集上所有 Kubernetes 物件的許可，請選擇此選項。
- Kubernetes 命名空間：如果您希望 Amazon EKS 授予該 IAM 主體存取政策中針對叢集上特定 Kubernetes 命名空間中所有 Kubernetes 物件的許可，則請選擇此選項。對於命名空間，請輸入叢集上 Kubernetes 命名空間的名稱。如果要新增其他命名空間，則請選擇新增命名空間並輸入命名空間名稱。

c. 如果要新增其他政策，則請選擇新增政策。您可以對每個政策設定不同的範圍，但每個政策只能新增一次。

d. 選擇下一步。

12. 檢查存取項目的組態。如果有任何內容看起來不正確，請選擇上一步以返回上一步並修正錯誤。如果組態正確，請選擇建立。

AWS CLI

先決條件

在您的裝置或上安裝和設定的 AWS CLI v1 的最新版本 AWS CloudShell。 AWS CLI v2 幾天不支持新功能。您可以使用 `aws --version | cut -d / -f2 | cut -d ' ' -f1` 來檢查您的目前版本。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《 AWS Command Line Interface 使用指南》`aws configure`中的〈安裝、更新 AWS CLI和解除安裝〉和〈快速設定〉。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《 AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

建立存取項目

您可以使用以下任何範例來建立存取項目：

- 為自我管理的 Amazon EC2 Linux 節點群組建立存取項目。將*#########*的名稱，*111122223333* 用您的 AWS 帳戶 識別碼取代我的叢集，以節點 IAM 角色的名稱取代 *EK-####-####-* NG-1。如果節點群組是 Windows 節點群組，請用**EC2_Windows** 取代 *EC2_Linux*。

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1 --type EC2_Linux
```

當指定 STANDARD 以外的類型時，則不能使用 --kubernetes-groups 選項。您無法將存取政策與此存取項目關聯，因為其類型是 STANDARD 以外的值。

- 建立這樣的一個存取項目：允許一個未用於 Amazon EC2 自我管理節點群組且您希望 Kubernetes 透過其授予對叢集的存取權的 IAM 角色。將我的*#######*的名稱，*111122223333* 用您的身分 AWS 帳戶 識別碼取代我的角色，並將我的角色取代為 IAM *##*的名稱。用您在叢集上的 Kubernetes RoleBinding 或 ClusterRoleBinding 物件中指定的群組名稱取代 *Viewers*。

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role --type STANDARD --user Viewers --
kubernetes-groups Viewers
```

- 建立一個允許 IAM 使用者向叢集進行身分驗證的存取項目。在此提供此範例只是為了說明這種可能性。IAM 最佳實務建議使用具有短期憑證的 IAM 角色存取叢集，而不是使用具有長期憑證的 IAM 使用者進行存取。如需詳細資訊，請參閱 IAM 使用者指南中的要求人類使用者與身分識別提供者的聯合以 AWS 使用臨時登入資料存取。

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:user/my-user --type STANDARD --username my-user
```

如果您希望此使用者對叢集的存取許可高於 Kubernetes API 探索角色中的許可，則您需要將存取政策關聯到該存取項目，因為不使用 --kubernetes-groups 選項。如需詳細資訊，請參閱 Kubernetes 文件中的 將存取政策與存取項目關聯和取消關聯 以及 API discovery roles 兩節。

更新存取項目

您可以使用 AWS Management Console 或更新存取項目 AWS CLI。

AWS Management Console

更新存取項目

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇要在其中建立存取項目的叢集名稱。
3. 選擇存取索引標籤。
4. 選擇要更新的存取項目。

5. 選擇編輯。
6. 對於使用者名稱，您可以變更現有值。
7. 對於群組，您可以移除現有群組名稱或新增群組名稱。如果存在以下群組名稱，請勿將其移除：system:nodes 或 system:bootstrappers。移除這些群組可能會導致叢集無法正常運作。如果您不指定任何群組名稱並希望使用 Amazon EKS 授權，請在後續步驟中關聯存取政策。
8. 您可以使用標籤為存取項目指派標籤。例如，為了更輕鬆地找到具有相同標籤的所有資源而指定標籤。您也可以移除現有的標籤。
9. 選擇儲存變更。
10. 如果您想要將存取政策關聯到項目，請參閱 將存取政策與存取項目關聯和取消關聯。

AWS CLI

先決條件

您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

更新存取項目

將我的*#######*名稱，*111122223333* 用您的 AWS 帳戶 識別碼取代我的叢集，以 IAM 角色的名稱取代 *EK-#-* 叢集-我的命名空間檢視器。

```
aws eks update-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --kubernetes-
groups Viewers
```

如果存取項目類型是 STANDARD 以外的值，則不能使用 --kubernetes-groups 選項。您也無法將存取政策與 STANDARD 以外的類型的存取項目關聯。

刪除存取項目

如果您發現某個存取項目被錯誤刪除，則您可以隨時重新建立。如果您要刪除的存取項目與任何存取政策關聯，則這些關聯將自動刪除。在刪除存取項目之前，您不必取消其與存取政策的關聯。

您可以使用 AWS Management Console 或刪除存取項目 AWS CLI。

AWS Management Console

刪除存取項目

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇要從中刪除存取項目之叢集的名稱。
3. 選擇存取索引標籤。
4. 在存取項目清單中，選擇要刪除的存取項目。
5. 選擇刪除。
6. 在確認對話方塊中，選擇 Delete (刪除)。

AWS CLI

先決條件

您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本`1.27.160`或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。`2.12.3`若要檢查您目前的版本，請使用 **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

刪除存取項目

將*####*取代為叢集的名稱，*111122223333* 用您的 AWS 帳戶 識別碼取代我的叢集，並以您不想再存取叢集的 IAM 角色名稱取代 *my-* role。

```
aws eks delete-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

將存取政策與存取項目關聯和取消關聯

您可以向類型為 STANDARD 的存取項目指派一個或多個存取政策。Amazon EKS 會自動向其他類型的存取項目授予在叢集中正常運作所需的許可。Amazon EKS 存取政策包含 Kubernetes 許可，而不是 IAM 許可。在將存取政策與存取項目關聯之前，請確認您熟悉每個存取政策中包含的 Kubernetes 許可。如需詳細資訊，請參閱 存取政策許可。如果沒有一個存取政策符合您的要求，則不要將存取政策與存取項目關聯。而是為存取項目指定一個或多個群組名稱，並建立和管理 Kubernetes 角色型存取控制物件。如需詳細資訊，請參閱 建立存取項目。

必要條件

- 現有的存取項目。若要建立服務角色，請參閱建立存取項目。
- 具有下列權限的 AWS Identity and Access Management 角色或使用者：ListAccessEntriesDescribeAccessEntryUpdateAccessEntry、ListAccessPolicies、A和DisassociateAccessPolicy。如需詳細資訊，請參閱《服務授權參考》中的 Amazon Elastic Kubernetes Service 定義的動作一節。

在將存取政策與存取項目關聯之前，請考量以下要求：

- 每個存取項目可以關聯多個存取政策，但只能將每個政策與一個存取項目關聯一次。如果您將一個存取項目與多個存取政策關聯，則該存取項目的 IAM 主體擁有所有關聯的存取政策中包含的所有許可。
- 您可以將存取政策的適用範圍限定為叢集上的所有資源，或透過指定一個或多個 Kubernetes 命名空間的名稱來限制其適用範圍。您可以將萬用字元用於命名空間名稱。例如，如果要將存取政策的適用範圍限定為以 dev- 開頭的所有命名空間，則可以指定 dev-* 作為命名空間名稱。確認叢集上存在相應命名空間，並且拼字與叢集上的實際命名空間名稱相符。Amazon EKS 不會確認叢集上命名空間的拼字或命名空間是否存在。
- 將存取政策關聯到存取項目後，您可以變更存取政策的存取範圍。如果您已將存取政策的適用範圍限定為 Kubernetes 命名空間，則可以根據需要新增和移除關聯的命名空間。
- 如果您將存取政策關聯到也指定了群組名稱的存取項目，則相應 IAM 主體擁有所有關聯存取政策中的所有許可。它還擁有在指定群組名稱的任何 Kubernetes Role 和 RoleBinding 物件中指定的任何 Kubernetes Role 或 ClusterRole 物件中的所有許可。
- 如果您執行 kubectl auth can-i --list 命令，您不會看到與您在執行該命令時所使用的 IAM 主體的存取項目關聯的存取政策指派的任何 Kubernetes 許可。只有當您已在繫結至為存取項目指定的群組名稱或使用者名稱的 Kubernetes Role 或 ClusterRole 物件中授予 Kubernetes 許可時，此命令才會顯示相應許可。

- 如果您在與叢集上的 Kubernetes 物件互動時模擬 Kubernetes 使用者或群組，例如將 `kubectl` 命令與 **`--as`** ***`username`*** 或 **`--as-group`** ***`group-name`*** 結合使用，則將強制使用 Kubernetes RBAC 授權。因此，相應 IAM 主體沒有與相應存取項目關聯的任何存取政策指派的許可。相應 IAM 主體模擬的使用者或群組擁有的 Kubernetes 許可，僅限於您在繫結到該群組名稱或使用者名稱的 Kubernetes `Role` 或 `ClusterRole` 物件中授予的 Kubernetes 許可。為了讓相應 IAM 主體擁有關聯的存取政策中的許可，請勿模擬 Kubernetes 使用者或群組。相應 IAM 主體還將擁有您在繫結至為存取項目指定的群組名稱或使用者名稱的 Kubernetes `Role` 或 `ClusterRole` 物件中授予的任何許可。如需詳細資訊，請參閱 Kubernetes 文件中的 User impersonation 一節。

您可以使用或將存取原則與存取項目 AWS Management Console 相關聯 AWS CLI。

AWS Management Console

使用 AWS Management Console將存取政策與存取項目關聯

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇要將存取政策關聯到的存取項目所在的叢集的名稱。
3. 選擇存取索引標籤。
4. 如果存取項目的類型為標準，則可以將 Amazon EKS存取政策與之關聯或取消關聯。如果存取項目的類型不為標準，則此選項不可用。
5. 選擇關聯存取政策。
6. 對於政策名稱，選取具有您希望相應 IAM 主體擁有的許可的政策。若要查看每個政策包含的許可，請參閱 存取政策許可。
7. 對於存取範圍，選擇存取範圍。如果您選擇叢集，則該存取政策中的許可將授予相應 IAM 主體對所有 Kubernetes 命名空間中的資源的存取權。如果您選擇 Kubernetes 命名空間，則可以選擇新增命名空間。在顯示的命名空間欄位中，您可以輸入叢集上的 Kubernetes 命名空間的名稱。如果您希望相應 IAM 主體擁有跨多個命名空間的許可，則可以輸入多個命名空間。
8. 選擇新增存取政策。

AWS CLI

先決條件

您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本`1.27.160`或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。`2.12.3`若要檢查您目前的版本，請使

用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

將存取政策與存取項目關聯

1. 檢視可用的存取政策。

```
aws eks list-access-policies --output table
```

範例輸出如下。

```
---------------------------------------------------------------------------------------
|                                          ListAccessPolicies                         |
                         |
+-------------------------------------------------------------------------------------
+
||                                            accessPolicies                           |
                      ||
|+-------------------------------------------------------------------
+------------------------------+|
||                                arn                                 |
     name              ||
|+-------------------------------------------------------------------
+------------------------------+|
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy        |
 AmazonEKSAdminPolicy         ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy |
 AmazonEKSClusterAdminPolicy  ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy         |
 AmazonEKSEditPolicy          ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy         |
 AmazonEKSViewPolicy          ||
|+-------------------------------------------------------------------
+------------------------------+|
```

若要查看每個政策包含的許可，請參閱 存取政策許可。

2. 檢視現有的存取項目。使用您叢集的名稱取代 *my-cluster*。

```
aws eks list-access-entries --cluster-name my-cluster
```

範例輸出如下。

```
{
    "accessEntries": [
        "arn:aws:iam::111122223333:role/my-role",
        "arn:aws:iam::111122223333:user/my-user"
    ]
}
```

3. 將存取政策與存取項目關聯。以下範例將 AmazonEKSViewPolicy 存取政策與一個存取項目關聯。每當 *my-role* IAM 角色嘗試存取叢集上的 Kubernetes 物件時，Amazon EKS 將授權該角色使用政策中的許可來存取且僅存取 *my-namespace1* 和 *my-namespace2* Kubernetes 命名空間中的 Kubernetes 物件。用您的叢集名稱取代 *my-cluster*，用您的 AWS 帳戶 ID 取代 *111122223333*，並用您想要讓 Amazon EKS 為其授予對 Kubernetes 叢集上物件之存取權的 IAM 角色取代 *my-role*。

```
aws eks associate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --access-scope type=namespace,namespaces=my-namespace1,my-namespace2 --
policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy
```

如果您希望相應 IAM 主體擁有適用於整個叢集的許可，請用 **type=cluster** 取代 **type=namespace,namespaces=*my-namespace1*,*my-namespace2***。如果要將多個存取政策關聯到該存取項目，請多次執行該命令，並每次使用不同的存取政策。每個關聯的存取政策都有自身的適用範圍。

> **Note**
>
> 如果您稍後想要變更關聯的存取政策的適用範圍，請再次執行先前的命令並指定新的適用範圍。例如，如果您想要移除 *my-namespace2*，則只需使用 **type=namespace,namespaces=*my-namespace1*** 再次執行命令。如果您想將適用範圍從 **namespace** 變更為 **cluster**，您需要使用 **type=cluster** 再次執行該命令，而不使用 **type=namespace,namespaces=*my-namespace1*,*my-namespace2***。

將存取政策與存取項目取消關聯

1. 確定哪些存取政策與某個存取項目關聯。

```
aws eks list-associated-access-policies --cluster-name my-cluster --principal-
arn arn:aws:iam::111122223333:role/my-role
```

範例輸出如下。

```
{
    "clusterName": "my-cluster",
    "principalArn": "arn:aws:iam::111122223333",
    "associatedAccessPolicies": [
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSViewPolicy",
            "accessScope": {
                "type": "cluster",
                "namespaces": []
            },
            "associatedAt": "2023-04-17T15:25:21.675000-04:00",
            "modifiedAt": "2023-04-17T15:25:21.675000-04:00"
        },
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSAdminPolicy",
            "accessScope": {
                "type": "namespace",
                "namespaces": [
                    "my-namespace1",
                    "my-namespace2"
                ]
            },
            "associatedAt": "2023-04-17T15:02:06.511000-04:00",
            "modifiedAt": "2023-04-17T15:02:06.511000-04:00"
        }
    ]
}
```

在前面的範例中，相應存取項目的 IAM 主體擁有相應叢集上所有命名空間的檢視許可，以及兩個 Kubernetes 命名空間的管理員許可。

2. 將存取政策與存取項目取消關聯。在此範例中，AmazonEKSAdminPolicy 政策與存取項目取消關聯。但是，相應 IAM 主體保留 AmazonEKSViewPolicy 存取政策中對 *my-namespace1* 和 *my-namespace2* 命名空間中物件的許可，因為該存取政策並未與該存取項目取消關聯。

```
aws eks disassociate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy
```

存取政策許可

存取政策包含 `rules`，其中包含 Kubernetes `verbs` (許可) 和 `resources`。存取政策不包含 IAM 許可或資源。與 Kubernetes `Role` 和 `ClusterRole` 物件類似，存取政策僅包含 `allow` `rules`。您無法修改存取政策的內容。您無法建立自己的存取政策。如果存取政策中的許可無法滿足您的需求，您可以建立 Kubernetes RBAC 物件並為存取項目指定群組名稱。如需詳細資訊，請參閱 建立存取項目。存取政策中包含的許可類似於 Kubernetes 面向使用者之叢集角色中的許可。如需詳細資訊，請參閱 Kubernetes 文件中的 User-facing roles 一節。

選擇任意存取政策以查看其內容。每個存取政策中每個表的每一行都是一項單獨規則。

AmazonEKS AdminPolicy

此存取政策包含向相應 IAM 主體授予對資源的大部分許可的許可。當將之與某個存取項目關聯時，其存取範圍通常是一個或多個 Kubernetes 命名空間。如果您希望相應 IAM 主體對叢集上的所有資源具有管理員存取權，請將 AmazonEKS ClusterAdminPolicy 存取政策與相應存取項目關聯。

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy`

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
apps	daemonsets , deployments , deployments/rollback , deployments/scale , replicasets , replicasets/scale, statefulsets , statefulsets/scale	create, delete, deletecollection , patch, update
apps	controllerrevisions , daemonsets , daemonset	get, list, watch

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
	s/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	
authorization.k8s.io	localsubjectaccessreviews	create
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
rbac.authorization.k8s.io	rolebindings , roles	create, delete, deletecollection , get, list, patch, update, watch
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get,list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update
	pods, pods/attach , pods/exec , pods/portforward , pods/proxy	create, delete, deletecollection , patch, update
	serviceaccounts	impersonate

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , resourcequotas/status	get, list, watch
	namespaces	get,list, watch

AmazonEKS ClusterAdminPolicy

此存取政策包含授予相應 IAM 主體對叢集的管理員存取權的許可。當將之與某個存取項目關聯時，其存取範圍通常是叢集，而不是某個 Kubernetes 命名空間。如果您希望限制相應 IAM 主體的管理範圍，請考慮將 AmazonEKS AdminPolicy 存取政策與相應存取項目關聯。

ARN – arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy

Kubernetes API 群組	Kubernetes nonResourceURLs	Kubernetes 資源	Kubernetes 動詞 (許可)
*		*	*
	*		*

AmazonEKS AdminViewPolicy

此存取政策包括授與 IAM 主體存取權限以列出/檢視叢集中所有資源的許可。請注意，這包括Kubernetes秘密。

ARN – arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminViewPolicy

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
*	*	get, list, watch

AmazonEKS EditPolicy

此存取政策包含允許相應 IAM 主體編輯大多數 Kubernetes 資源的許可。

ARN – arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
apps	daemonsets , deployments , deployments/rollback , deployments/scale , replicasets , replicasets/scale, statefulsets , statefulsets/scale	create, delete, deletecollection , patch, update
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
networking.k8s.io	ingresses , ingresses /status , networkpo licies	get, list, watch
policy	poddisruptionbudgets	create, delete, deletecol lection , patch, update
policy	poddisruptionbudgets , poddisruptionbudge ts/status	get, list, watch
	namespaces	get, list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	serviceaccounts	impersonate
	pods, pods/attach , pods/exec , pods/port forward , pods/proxy	create, delete, deletecol lection , patch, update
	configmaps , events, persistentvolumecl aims , replicati oncontrollers , replicationcontrol lers/scale , secrets, serviceaccounts , services, services/ proxy	create, delete, deletecol lection , patch, update

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get, list, watch
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , resourcequotas/status	get, list, watch

AmazonEKS ViewPolicy

此存取政策包含允許相應 IAM 主體檢視大多數 Kubernetes 資源的許可。

ARN – arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicase	get, list, watch

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
	ts/status , statefuls ets , statefulsets/ scale , statefulsets/ status	
autoscaling	horizontalpodautos calers , horizonta lpodautoscalers/st atus	get, list, watch
batch	cronjobs, cronjobs/ status , jobs, jobs/stat us	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets ,daemonset s/status ,deploymen ts ,deployments/ scale,deployments/ status ,ingresses ,ingresses/ status,networkpo licies ,replicase ts , replicasets/ scale,replicasets/ status , replicati oncontrollers/scale	get, list, watch
networking.k8s.io	ingresses , ingresses /status , networkpo licies	get, list, watch
policy	poddisruptionbudgets , poddisruptionbudge ts/status	get, list, watch

Kubernetes API 群組	Kubernetes 資源	Kubernetes 動詞 (許可)
	configmaps ，endpoints ，persistentvolumeclaims ，persistentvolumeclaims/status ，pods, replicationcontrollers ，replicationcontrollers/scale ，serviceaccounts ，services, services/status	get, list, watch
	bindings,events,limitranges ,namespaces/status,pods/log, rpods/status ,eplicationcontrollers/status ，rresourcequotas ，resourcequotas/status	get, list, watch
	namespaces	get, list, watch

存取政策更新

檢視有關存取政策更新 (自引進以來) 的詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 Amazon EKS 文件歷程記錄頁面上的 RSS 摘要。

變更	描述	日期
加 AmazonEKSAdminViewPolicy	新增擴充檢視存取權限的新原則，包括 Secret 等資源。	2024年4月23 日
引進存取政策。	Amazon EKS 引進了存取政策。	2023 年 5 月 29 日

遷移現有 aws-auth ConfigMap 項目至存取項目

如果您已將項目新增至叢集上的 aws-auth ConfigMap，則建議您為 aws-auth ConfigMap 中的現有項目建立存取項目。建立存取項目後，您可以從 ConfigMap 中移除對應項目。您無法將存取政策與 aws-auth ConfigMap 中的項目相關聯。如果您想要將存取政策與 IAM 主體相關聯，則請建立存取項目。

> **Important**
>
> 請勿移除現有的您將受管節點群組或 Fargate 描述檔新增至叢集時 Amazon EKS 建立的 aws-auth ConfigMap 項目。如果您移除 Amazon EKS 在 ConfigMap 中建立的項目，叢集將無法正常運作。但是，在為自我管理節點群組建立存取項目後，您可以移除它們的任何項目。

必要條件

- 熟悉存取項目和存取政策。如需詳細資訊，請參閱 管理存取項目 及 將存取政策與存取項目關聯和取消關聯。
- 現有叢集的平台版本等於或高於允許 IAM 角色或使用者存取 Amazon EKS 叢集上的 Kubernetes 物件主題之「先決條件」部分中列出的版本。
- 已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。
- 修改 kube-system 命名空間中的 aws-auth ConfigMap 的 Kubernetes 許可。
- 具有下列權限的 AWS Identity and Access Management 角色或使用者：CreateAccessEntry和ListAccessEntries。如需詳細資訊，請參閱《服務授權參考》中的 Amazon Elastic Kubernetes Service 定義的動作一節。

將項目從 aws-auth ConfigMap 遷移至存取項目

1. 檢視 aws-auth ConfigMap 中的現有項目。使用您叢集的名稱取代 *my-cluster*。

   ```
   eksctl get iamidentitymapping --cluster my-cluster
   ```

   範例輸出如下。

```
ARN
              USERNAME                                      GROUPS
                            ACCOUNT
arn:aws:iam::111122223333:role/EKS-my-cluster-Admins
              Admins                                        system:masters
arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers
              my-namespace-Viewers                          Viewers
arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1
                              system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:user/my-user
              my-user
arn:aws:iam::111122223333:role/EKS-my-cluster-fargateprofile1
                              system:node:{{SessionName}}
 system:bootstrappers,system:nodes,system:node-proxier
arn:aws:iam::111122223333:role/EKS-my-cluster-managed-ng
                              system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. 為在上一個輸出中傳回的您建立的任何 ConfigMap 項目建立存取項目。建立存取項目時，請確保為 ARN、USERNAME、GROUPS 和 ACCOUNT 指定與輸出中傳回的值相同的值。在範例輸出中，您將為最後兩個項目之外的所有項目建立存取項目，因為這些項目是由 Amazon EKS 為 Fargate 描述檔和受管節點群組建立的。
3. 從 ConfigMap 中刪除您建立的任何存取項目對應的項目。如果您沒有從 ConfigMap 中刪除對應項目，則對應 IAM 主體 ARN 的存取項目的設定將覆寫 ConfigMap 項目。將 *1111223333* 替換為您的 AWS 帳戶 ID 和 *EKS-#-#-##-######-####-###*，替換為. ConfigMap 如果您要刪除的項目用於 IAM 使用者而不是 IAM 角色，請用 **user** 取代 **role**，並用使用者名稱取代 *EKS-my-cluster-my-namespace-Viewers*。

```
eksctl delete iamidentitymapping --arn arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --cluster my-cluster
```

啟用 IAM 主體的叢集存取權

> **Important**
>
> 已棄用。aws-auth ConfigMap管理 Kubernetes API 存取權的建議方法是存取項目。

Kubernetes 專用AWS IAM 驗證器在 Amazon EKS 控制平面上執行，可以使用 IAM 主體啟用對叢集的存取權。驗證器會從 aws-auth ConfigMap 獲得其組態資訊。如需所有 aws-auth ConfigMap 設定的詳細資訊，請參閱 GitHub 上的完整組態格式。

將 IAM 主體新增至 Amazon EKS 叢集

當您建立 Amazon EKS 叢集時，建立該叢集的 IAM 主體會由 Amazon EKS 控制平面中叢集的角色型存取控制 (RBAC) 組態來自動授予 system:masters 許可。此主體不會出現在任何可見的組態中，因此請務必記下最初建立叢集的主體。若要賦予其他 IAM 主體與叢集互動的能力，您必須編輯 Kubernetes 中的 aws-auth ConfigMap 並用 group 的名稱建立一個 Kubernetes rolebinding 或 clusterrolebinding，其名稱是您在 aws-auth ConfigMap 中指定的名稱。

> **Note**
>
> 如需有關 Kubernetes 角色型存取控制 (RBAC) 組態的詳細資訊，請參閱 Kubernetes 文件中的使用 RBAC 授權。

將 IAM 主體新增至 Amazon EKS 叢集

1. 判斷 kubectl 正使用哪些憑證來存取叢集。您可以在電腦上透過一下命令查看 kubectl 正在使用的憑證。如果不使用預設路徑，則使用 kubeconfig 檔案的路徑取代 ***~/.kube/config***。

   ```
   cat ~/.kube/config
   ```

   範例輸出如下。

   ```
   [...]
   contexts:
   - context:
       cluster: my-cluster.region-code.eksctl.io
       user: admin@my-cluster.region-code.eksctl.io
     name: admin@my-cluster.region-code.eksctl.io
   current-context: admin@my-cluster.region-code.eksctl.io
   [...]
   ```

   在前述範例輸出中，已針對名為 *my-cluster* 的叢集設定名為 *admin* 的使用者的憑證。若該叢集是由這名使用者建立，則其已擁有您叢集的存取權。若叢集不是由此使用者建立，則您需要完成

剩餘的步驟，才能為其他 IAM 主體啟用叢集存取權。IAM 最佳實務建議您將許可授予角色而非使用者。您可以使用以下命令查看目前哪些其他主體可以存取您的叢集：

```
kubectl describe -n kube-system configmap/aws-auth
```

範例輸出如下。

```
Name:         aws-auth
Namespace:    kube-system
Labels:       <none>
Annotations:  <none>

Data
====
mapRoles:
----
- groups:
  - system:bootstrappers
  - system:nodes
  rolearn: arn:aws:iam::111122223333:role/my-node-role
  username: system:node:{{EC2PrivateDNSName}}


BinaryData
====

Events:  <none>
```

上一個範例是預設的 aws-auth ConfigMap。只有節點執行個體角色有權存取叢集。

2. 請確定您現有的 Kubernetes roles 和 rolebindings 或 clusterroles 和 clusterrolebindings (您可以將 IAM 主體映射至此)。如需有關這些資源的詳細資訊，請參閱 Kubernetes 文件中的使用 RBAC 授權。

   1. 檢視您現有的 Kubernetes roles 或 clusterroles。Roles 的作用域為 namespace，但 clusterroles 的作用域限定為叢集。

      ```
      kubectl get roles -A
      ```

      ```
      kubectl get clusterroles
      ```

2. 檢視先前輸出中傳回的任何 role 或 clusterrole 的詳細資訊，並確認其具有您希望 IAM 主體在叢集中擁有的許可 (rules)。

   將 *role-name* 取代為上一個命令的輸出中傳回的 role 名稱。將 *kube-system* 取代為 role 的命名空間。

   ```
   kubectl describe role role-name -n kube-system
   ```

   將 *cluster-role-name* 取代為上一個命令的輸出中傳回的 clusterrole 名稱。

   ```
   kubectl describe clusterrole cluster-role-name
   ```

3. 檢視您現有的 Kubernetes rolebindings 或 clusterrolebindings。Rolebindings 的作用域為 namespace，但 clusterrolebindings 的作用域限定為叢集。

   ```
   kubectl get rolebindings -A
   ```

   ```
   kubectl get clusterrolebindings
   ```

4. 檢視任何 rolebinding 或 clusterrolebinding 的詳細資訊，並確認其具有上一步中列為 roleRef 的 role 或 clusterrole，以及為 subjects 列出的群組名稱。

   將 *role-binding-name* 取代為上一個命令的輸出中傳回的 rolebinding 名稱。將 *kube-system* 取代為 rolebinding 的 namespace。

   ```
   kubectl describe rolebinding role-binding-name -n kube-system
   ```

   範例輸出如下。

   ```
   apiVersion: rbac.authorization.k8s.io/v1
   kind: RoleBinding
   metadata:
     name: eks-console-dashboard-restricted-access-role-binding
     namespace: default
   subjects:
   - kind: Group
     name: eks-console-dashboard-restricted-access-group
     apiGroup: rbac.authorization.k8s.io
   roleRef:
   ```

```
  kind: Role
  name: eks-console-dashboard-restricted-access-role
  apiGroup: rbac.authorization.k8s.io
```

將 *cluster-role-binding-name* 取代為上一個命令的輸出中傳回的 clusterrolebinding 名稱。

```
kubectl describe clusterrolebinding cluster-role-binding-name
```

範例輸出如下。

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks-console-dashboard-full-access-binding
subjects:
- kind: Group
  name: eks-console-dashboard-full-access-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: eks-console-dashboard-full-access-clusterrole
  apiGroup: rbac.authorization.k8s.io
```

3. 編輯 aws-auth ConfigMap。您可以使用例如 eksctl 的工具來更新 ConfigMap，或者您可以透過編輯來手動更新它。

> **Important**
>
> 我們建議您使用 eksctl 或其他工具來編輯 ConfigMap。如需有關您可使用的其他工具的資訊，請參閱《Amazon EKS 最佳實務指南》中的使用工具對 aws-authConfigMap 進行變更。格式錯誤的 aws-auth ConfigMap可能會導致您失去叢集存取權。

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

1. 在 ConfigMap 檢視目前的映射項目。使用您叢集的名稱取代 *my-cluster*。*region-code*以叢集所 AWS 區域 在的位置取代。

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

範例輸出如下。

```
ARN
                    USERNAME                                GROUPS
              ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA     system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. 為角色新增映射項目。將 *my-role* 取代為您的角色名稱。將 *eks-console-dashboard-full-access-group* 取代為 Kubernetes RoleBinding 或 ClusterRoleBinding 物件中指定的群組名稱。使用您的帳戶 ID 取代 *111122223333*。您可以將 *admin* 取代為您選擇的任何名稱。

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws:iam::111122223333:role/my-role --username admin --group eks-
console-dashboard-full-access-group \
    --no-duplicate-arns
```

> **Important**
>
> 角色 ARN 不能包含例如 role/my-team/developers/my-role 的路徑。ARN 的格式必須是 arn:aws:iam::*111122223333*:role/*my-role*。在此範例中，需移除 my-team/developers/。

範例輸出如下。

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-
role" to auth ConfigMap
```

3. 為使用者新增映射項目。IAM 最佳實務建議您將許可授予角色而非使用者。將 *my-user* 取代為您的使用者名稱。將 *eks-console-dashboard-restricted-access-group* 取代為 Kubernetes RoleBinding 或 ClusterRoleBinding 物件中指定的群組名稱。使用您的帳戶 ID 取代 *111122223333*。您可以將 *my-user* 取代為您選擇的任何名稱。

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user --username my-user --
group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

範例輸出如下。

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-
user" to auth ConfigMap
```

4. 再次檢視 ConfigMap 中的映射項目。

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

範例輸出如下。

```
ARN
                    USERNAME                                GROUPS
                      ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:role/admin
                        my-role                                 eks-console-
dashboard-full-access-group
arn:aws:iam::111122223333:user/my-user
                        my-user                                 eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

1. 開啟 ConfigMap 進行編輯。

```
kubectl edit -n kube-system configmap/aws-auth
```

> **Note**
>
> 若您收到錯誤訊息，指出 "Error from server (NotFound): configmaps "aws-auth" not found"，請使用 將 aws-authConfigMap 套用至您的叢集 中的程序來套用儲存的 ConfigMap。

2. 將您的 IAM 主體新增至 ConfigMap。IAM 群組不是 IAM 主體，因此無法新增至 ConfigMap。
   - 新增 IAM 角色 (例如：為聯合身分使用者)：在 data 下位於 ConfigMap 中的 mapRoles 區段新增角色詳細資訊。若此區段在檔案不存在，則將其新增。每個項目支援以下參數：
     - rolearn：要新增的 IAM 角色之 ARN。此值不能包含路徑。例如，您無法指定 ARN (如 arn:aws:iam::*111122223333*:role/my-team/developers/*role-name*)。ARN 需要改為 arn:aws:iam::*111122223333*:role/*role-name*。
     - username：Kubernetes 內映射至 IAM 角色的使用者名稱。
     - 群組：要將角色映射到的 Kubernetes 群組的群組或清單。群組可以是預設群組，也可以是 clusterrolebinding 或 rolebinding 中指定的群組。如需詳細資訊，請參閱 Kubernetes 文件中的預設角色和角色連結。
   - 若要新增 IAM 使用者：IAM 最佳實務建議您將許可授予角色而非使用者。將使用者詳細資訊新增至 ConfigMap 的 mapUsers 區段 (在 data 下方)。若此區段在檔案不存在，則將其新增。每個項目支援以下參數：
     - userarn：要新增之 IAM 使用者的 ARN。
     - username：Kubernetes 內映射至 IAM 使用者的使用者名稱。
     - 群組：要將使用者映射到的 Kubernetes 群組的群組或清單。群組可以是預設群組，也可以是 clusterrolebinding 或 rolebinding 中指定的群組。如需詳細資訊，請參閱 Kubernetes 文件中的預設角色和角色連結。

   例如，下列的 YAML 區塊包含：
   - mapRoles 區段，將 IAM 節點執行個體映射至 Kubernetes 群組，讓節點可以將自己註冊到叢集，以及 my-console-viewer-role IAM 角色，映射至 Kubernetes 群組，可以檢視所有叢集的所有 Kubernetes 資源。如需 my-console-viewer-role IAM 角色所需的 IAM 和 Kubernetes 群組許可的清單，請參閱 所需的許可。

- 將 admin IAM 使用者從預設 AWS 帳戶對應至system:mastersKubernetes群組，以及對應至可檢視特定命名空間Kubernetes資源之Kubernetes群組的其他 AWS 帳戶的my-user使用者的mapUsers區段。如需 my-user IAM 使用者所需的 IAM 和 Kubernetes 群組許可的清單，請參閱 所需的許可。

根據需要新增或移除行，並將所有 *example values* 取代為您自己的值。

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this
 file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::111122223333:role/my-role
      username: system:node:{{EC2PrivateDNSName}}
    - groups:
      - eks-console-dashboard-full-access-group
      rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
      username: my-console-viewer-role
  mapUsers: |
    - groups:
      - system:masters
      userarn: arn:aws:iam::111122223333:user/admin
      username: admin
    - groups:
      - eks-console-dashboard-restricted-access-group
      userarn: arn:aws:iam::444455556666:user/my-user
      username: my-user
```

3. 儲存檔案並結束您的文字編輯器。

將 aws-authConfigMap 套用至您的叢集

在建立受管節點群組或以 eksctl 建立節點群組時，便會自動建立並套用 aws-auth ConfigMap 至您的叢集。最初建立的目的是要允許將節點加入您的叢集，但您也可以使用此 ConfigMap 來新增角色

型存取控制 (RBAC) 以存取 IAM 主體。若尚未啟動自我管理節點且尚未套用 aws-auth ConfigMap 至叢集，則您可以使用以下程序來完成此操作。

將 aws-authConfigMap 套用至您的叢集

1. 檢查您是否已套用 aws-auth ConfigMap。

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

   若您收到錯誤訊息，指出 "Error from server (NotFound): configmaps "aws-auth" not found"，請使用以下步驟的程序來套用儲存的 ConfigMap。

2. 下載、編輯和套用 AWS 驗證器設定對應。

   a. 下載組態對應。

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

   b. 在 aws-auth-cm.yaml 檔案中，將 rolearn 設定至與您節點關聯的 IAM 角色之 Amazon Resource Name (ARN)。您可以使用文字編輯器來完成此操作，或者透過替換 *my-node-instance-role* 並執行以下命令：

   ```
   sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-
   instance-role|' aws-auth-cm.yaml
   ```

   請勿修改此檔案中的任何其他行。

   > **Important**
   >
   > 角色 ARN 不能包含例如 role/my-team/developers/my-role 的路徑。ARN 的格式必須是 arn:aws:iam::*111122223333*:role/*my-role*。在此範例中，需移除 my-team/developers/。

   您可以檢查節點群組的 AWS CloudFormation 堆疊輸出，並尋找下列值：

   - InstanceRoleARN — 針對使用建立的節點群組 eksctl
   - NodeInstance角色 — 針對使用 Amazon EKS 提供 AWS CloudFormation 範本所建立的節點群組 AWS Management Console

c. 套用組態。此命令可能需要幾分鐘的時間來完成。

```
kubectl apply -f aws-auth-cm.yaml
```

> **Note**
>
> 如果您收到任何授權或資源類型錯誤，請參閱故障診斷主題中的未經授權或存取遭拒 (kubectl)。

3. 查看節點的狀態，並等待他們到達 Ready 狀態。

```
kubectl get nodes --watch
```

輸入 Ctrl+C 傳回 Shell 提示。

從OpenID Connect身分識別提供者驗證叢集的使用者

Amazon EKS 支援使用 OpenID Connect (OIDC) 身分識別供應商做為叢集使用者驗證的方法。 OIDC 身分識別提供者可與 (IAM) 搭配使用，或作為 AWS Identity and Access Management (IAM) 的替代方案。如需使用 IAM 的詳細資訊，請參閱 the section called “授與庫伯內特 API 的存取權”。設定叢集身分驗證後，您可以建立 Kubernetes roles 和 clusterroles 指派許可給角色，然後使用 Kubernetes rolebindings 和 clusterrolebindings 將角色繫結至身分。如需詳細資訊，請參閱 Kubernetes 文件中的使用 RBAC 授權。

考量事項

- 您可以將一個 OIDC 身分提供者與叢集建立關聯。
- Kubernetes 不提供 OIDC 身分提供者。您可以使用現有的公有 OIDC 身分提供者，也可以執行您自己的身分提供者。如需認證提供商的清單，請參閱 OpenID 網站上的 OpenID 認證。
- OIDC 身分提供者的發行者 URL 必須可公開存取，以便 Amazon EKS 可以探索簽署金鑰。Amazon EKS 不支援具有自我簽署憑證的 OIDC 身分提供者。
- 您無法停用叢集上的 IAM 身分驗證器，因為將節點連接到叢集仍然需要此身分驗證器。
- Amazon EKS 叢集仍必須由 AWS IAM 主體建立，而不是OIDC身分識別提供者使用者。這是因為叢集建立者會與 Amazon EKS API 進行互動，而不是與 Kubernetes API 進行互動。
- OIDC如果開啟控制平面的記錄，則會在叢集的稽核 CloudWatch 記錄檔中列出身分識別提供者驗證的使用者。如需詳細資訊，請參閱 啟用和停用控制平面日誌。

- 您無法使用供應OIDC商 AWS Management Console 的帳戶登入。您只能透過 AWS Management Console 使用 AWS Identity and Access Management 帳戶登入來檢視主控台中的Kubernetes資源。

關聯 OIDC 身分提供者

在可以將 OIDC 身分提供者與叢集建立關聯之前，您需要提供商的下列資訊：

發行者 URL

OIDC 身分提供者的 URL，可讓 API 伺服器探索用於驗證權杖的公有簽署金鑰。URL 必須以 `https://` 開頭並且應該對應至提供商之 OIDC ID 字符中的 `iss` 宣告。根據 OIDC 標準，允許路徑元件，但不允許查詢參數。通常，URL 只包含一個主機名稱，如 `https://server.example.org` 或 `https://example.com`。此 URL 應指向 `.well-known/openid-configuration` 以下的層級，必須可透過網際網路公開存取。

用戶端 ID (也稱為對象)

向 OIDC 身分提供者提出驗證請求的用戶端應用程式 ID。

您可以使用 `eksctl` 或 AWS Management Console關聯身分提供者。

eksctl

使用 **`eksctl`** 將 OIDC 身分提供者與您的叢集關聯

1. 使用下列內容建立名為 *`associate-identity-provider.yaml`* 的檔案。使用自己的取代 *`example values`*。`identityProviders` 區段中的值是從您的 OIDC 身分提供者取得的。值僅對 `identityProviders` 下的 `name`、`type`、`issuerUrl` 和 `clientId` 設定為必要項目。

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: your-region-code

identityProviders:
  - name: my-provider
    type: oidc
```

```
    issuerUrl: https://example.com
    clientId: kubernetes
    usernameClaim: email
    usernamePrefix: my-username-prefix
    groupsClaim: my-claim
    groupsPrefix: my-groups-prefix
    requiredClaims:
      string: string
    tags:
      env: dev
```

> **Important**
>
> 請勿指定 `system:`，或該字串的任何部分，對於 `groupsPrefix` 或 `usernamePrefix`。

2. 建立供應商。

```
eksctl associate identityprovider -f associate-identity-provider.yaml
```

3. 若要讓 `kubectl` 與叢集和 OIDC 身分提供者搭配使用，請參閱 Kubernetes 文件中的 Using kubectl 一節。

AWS Management Console

使用將OIDC身分識別提供者與叢集相關聯 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選取您的叢集，然後選取 [存取] 索引標籤。
3. 在身分OIDC識別提供者區段中，選取關聯身分識別提供者
4. 在關聯 OIDC 身分提供者頁面上，輸入或選取下列選項，然後選取關聯。
   - 對於 Name (名稱)，輸入提供商的唯一名稱。
   - 對於 Issuer URL (發行者 URL)，輸入您的提供商 URL。此 URL 必須可透過網際網路存取。
   - 對於用戶端 ID，輸入 OIDC 身分提供者的用戶端 ID (也稱為受眾)。
   - 對於 Username claim (使用者名稱宣告)，輸入要用作使用者名稱的宣告。
   - 對於 Groups claim (群組宣告)，輸入要作為使用者群組使用的宣告。

- (選用) 選取 Advanced options (進階選項)，輸入或選取下列資訊。
  - Username prefix (使用者名稱字首)：輸入要在使用者名稱宣告前面加上的字首。字首會加入使用者名稱宣告，以防止與現有名稱發生衝突。如果未提供值，且使用者名稱是非 `email` 的值，字首預設為 Issuer URL (發行者 URL) 的值。您可以使用值 `-` 停用所有字首。請勿指定 `system:` 或該字串的任何部分。
  - Groups prefix (群組字首)：輸入要在群組宣告前面加上的字首。字首會加入群組宣告，以防止與現有名稱發生衝突 (例如 `system: groups`)。例如，值 `oidc:` 會建立群組名稱，例如 `oidc:engineering` 和 `oidc:infra`。請勿指定 `system:` 或該字串的任何部分。
  - Required claims (必要宣告)：選取 Add claim (新增宣告)，然後在用戶端 ID 字符中輸入一或多個描述必要宣告的鍵值對。鍵值對在 ID 字符中描述了必要宣告。如果設定，則會驗證每個宣告是否存在於具有相符值的 ID 字符中。
5. 若要讓 `kubectl` 與叢集和 OIDC 身分提供者搭配使用，請參閱 Kubernetes 文件中的 Using `kubectl` 一節。

取消 OIDC 身分提供者與叢集的關聯

如果您取消 OIDC 身分提供者與叢集的關聯，則提供商中包含的使用者將無法再存取叢集。不過，您仍然可以透過 IAM 主體存取叢集。

使用 AWS Management Console取消 OIDC 身分提供者與叢集的關聯

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在 OIDC 身分提供者區段中，選取取消關聯，輸入身分提供者名稱，然後選取 `Disassociate`。

IAM 政策範例

如果您想要防止 OIDC 身分提供者與叢集產生關聯，請建立下列 IAM 政策並將其與 Amazon EKS 管理員的 IAM 帳戶關聯。如需詳細資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策和新增 IAM 身分許可，以及《服務授權參考》中的 Amazon Elastic Kubernetes Service 的動作、資源和條件索引鍵。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "denyOIDC",
            "Effect": "Deny",
```

```
            "Action": [
                "eks:AssociateIdentityProviderConfig"
            ],
            "Resource": "arn:aws:eks:us-west-2.amazonaws.com:111122223333:cluster/*"

        },
        {
            "Sid": "eksAdmin",
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        }
    ]
}
```

下列範例政策允許 OIDC 身分提供者關聯，如果 `clientID` 為 `kubernetes` 以及 `issuerUrl` 為 `https://cognito-idp.us-west-2amazonaws.com/*`。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCognitoOnly",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotLikeIfExists": {
                    "eks:issuerUrl": "https://cognito-idp.us-west-2.amazonaws.com/*"
                }
            }
        },
        {
            "Sid": "DenyOtherClients",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotEquals": {
                    "eks:clientId": "kubernetes"
                }
```

```
            }
        },
        {
            "Sid": "AllowOthers",
            "Effect": "Allow",
            "Action": "eks:*",
            "Resource": "*"
        }
    ]
}
```

合作夥伴驗證的 OIDC 身分提供者

Amazon EKS 會持續與提供相容 OIDC 身分提供者支援的合作夥伴網路建立關係。有關如何將身分提供者與 Amazon EKS 整合的詳細資訊，請參閱下列合作夥伴的文件。

合作夥伴	產品	文件
PingIdentity	PingOne 適用於企業	安裝說明

Amazon EKS 旨在為您提供各種選擇來涵蓋所有使用案例。如果開發此處未列出的商業支援 OIDC 相容身分提供者，請聯絡我們的合作夥伴團隊：aws-container-partners@amazon.com 以取得詳細資訊。

建立或更新 Amazon EKS 叢集的 `kubeconfig` 檔案

在本主題中，您將為您的叢集建立 `kubeconfig` 檔案 (或更新現有的檔案)。

`kubectl` 命令列工具會使用 `kubeconfig` 檔案中的組態資訊，以與叢集的 API 伺服器進行通訊。如需詳細資訊，請參閱 Kubernetes 文件中的 Organizing Cluster Access Using kubeconfig Files (使用 kubeconfig 檔案組織叢集存取)。

Amazon EKS 使用 `aws eks get-token` 命令搭配 `kubectl` 進行叢集身分驗證。依預設，會 AWS CLI 使用與下列命令傳回的相同認證：

```
aws sts get-caller-identity
```

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。

- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 具有對您指定的叢集使用 eks:DescribeCluster API 動作許多的 IAM 使用者或角色。如需詳細資訊，請參閱 Amazon EKS 身分型政策範例。如果您使用自己的 OpenID Connect 提供者提供的身分來存取叢集，則請參閱 Kubernetes 文件中的 Using kubectl 一節以了解如何建立或更新 kube config 檔案。

自動建立 kubeconfig 檔案

必要條件

- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 對您指定的叢集使用 eks:DescribeCluster API 動作的許可。如需詳細資訊，請參閱 Amazon EKS 身分型政策範例。

若要使用建立您的**kubeconfig**檔案 AWS CLI

1. 為您的叢集建立或更新 kubeconfig 檔案。##### ##

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

根據預設，產生的組態檔案會在主目錄預設的 kubeconfig 路徑下 (.kube)，或與位在該處的現有 config 檔案合併。您可以使用 **--kubeconfig** 選項指定其他路徑。

您可以使用 **--role-arn** 選項指定 IAM 角色 ARN，用於在您發出 kubectl 命令時進行身分驗證。否則，會使用預設 AWS CLI 或 SDK 憑證鏈中的 IAM 主體。您可以通過運行aws sts get-caller-identity命令來查看默認 AWS CLI 或 SDK 身份。

如需所有可用的選項，請執行 aws eks update-kubeconfig help 命令，或參閱 AWS CLI Command Reference 中的 update-kubeconfig 一節。

2. 測試組態。

```
kubectl get svc
```

範例輸出如下。

```
NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
```

如果您收到任何授權或資源類型錯誤，請參閱故障診斷主題中的未經授權或存取遭拒 (kubectl)。

使用服務帳戶授予 Kubernetes 工作負載存取 AWS 權 Kubernetes

Kubernetes 服務帳戶為在 Pod 中執行的程序提供身分。如需詳細資訊，請參閱 Kubernetes 文件中的管理服務帳戶。如果您Pod需要存取 AWS 服務，您可以將服務帳戶對應至 AWS Identity and Access Management 身分以授予該存取權。如需詳細資訊，請參閱 服務帳戶的 IAM 角色。

服務帳戶字符

BoundServiceAccountTokenVolume 功能依 Kubernetes 版本的預設啟用。此功能透過允許在 Kubernetes 上執行的工作負載請求 JSON Web 字符，其為對象、時間和金鑰綁定，藉此提升安全性。服務帳戶字符的過期時間為一小時。在舊版 Kubernetes 版本中，字符沒有到期期間。這表示仰賴這些字符的客戶端必須在一小時內重新整理字符。以下 Kubernetes 用戶端開發套件會在要求的時間範圍內重新整理字符：

- Go 版本 `0.15.7` 和更新版本
- Python 版本 `12.0.0` 和更新版本
- Java 版本 `9.0.0` 和更新版本
- JavaScript 版本`0.10.3`及更新版本
- Ruby `master` 分支
- Haskell 版本 `0.3.0.0`
- C# 版本 `7.0.5` 和更新版本

如果工作負載使用較早的客戶端版本，則必須進行更新。為使客戶端順暢遷移到更新的限時服務帳戶字符，Kubernetes 對服務帳戶字符新增了超過預設的一小時的延長到期期間。針對 Amazon EKS 叢集，延長的到期期間為 90 天。您的 Amazon EKS 叢集的 Kubernetes API 伺服器拒絕超過 90 天的字符的請求。我們建議您檢查應用程式及其相依性，以確保 Kubernetes 用戶端開發套件與先前列出的版本相同或為更新版本。

當 API 服務器收到使用超過一小時的字符的請求時，它會使用 `annotations.authentication.k8s.io/stale-token` 註釋 API 稽核日誌事件。註釋的值如下範例所示：

```
subject: system:serviceaccount:common:fluent-bit, seconds after warning threshold:
 4185802.
```

如果您的叢集啟用了控制平面記錄，則註釋位於稽核日誌中。您可以使用下列CloudWatch 日誌見解查詢來識別 Amazon EKS 叢集Pods中使用過時權杖的所有項目：

```
fields @timestamp
| filter @logStream like /kube-apiserver-audit/
| filter @message like /seconds after warning threshold/
| parse @message "subject: *, seconds after warning threshold:*\"" as subject,
 elapsedtime
```

`subject` 指的是該 Pod 使用的服務帳戶。`elapsedtime` 表示讀取最新字符後的經過時間（以秒為單位）。當 `elapsedtime` 超過 90 天 (7,776,000 秒)，對 API 伺服器的請求將遭到拒絕。您應主動更新應用程式的 Kubernetes 用戶端開發套件，以使用先前列出的自動重新整理字符的版本之一。如果您使用的服務帳戶字符接近 90 天，並且沒有足夠的時間在字符過期前更新用戶端軟體開發套件版本，則可以終止現有 Pods 並建立新的 Pod。這會導致重新擷取服務帳戶字符，進而給您額外的 90 天來更新的客戶端版本 SDK。

如果 Pod 為部署的一部分，則在維持高可用性的同時終止 Pods 的建議方法是使用以下命令執行推出。使用您的部署名稱替換 *my-deployment*。

```
kubectl rollout restart deployment/my-deployment
```

叢集附加元件

下列叢集附加元件已更新以使用自動重新擷取服務帳戶字符的 Kubernetes 客戶端 SDK。建議您確保已在叢集上安裝列出的版本或更新版本。

- Amazon VPC CNI plugin for Kubernetes 和指標輔助外掛程式版本 `1.8.0` 和更新版本。若要檢查您目前的版本或進行更新，請參閱使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件和cni-metrics-helper。
- CoreDNS 版 `1.8.4` 和更新版本 若要檢查目前版本或更新版本，請參閱 使用 CoreDNS Amazon EKS 附加元件。
- AWS Load Balancer Controller 版 `2.0.0` 和更新版本 若要檢查目前版本或更新版本，請參閱 什麼是 AWS Load Balancer Controller？。
- 目前的 `kube-proxy` 版本。若要檢查目前版本或更新版本，請參閱 使用庫伯尼特附kube-proxy加元件。
- AWS 適用於流利位元版本`2.25.0`或更高版本。若要更新您的目前版本，請參閱 GitHub 上的 Releases (版本)。
- Fluentd 映像 1.14.6-1.2 版或更新版本，以及適用於 Kubernetes 中繼資料的 Fluentd 篩選器外掛程式 2.11.1 版或更新版本。

授予 AWS Identity and Access Management 權限給 Amazon Elastic Kubernetes Service 叢集上的工作負載

Amazon EKS 提供兩種授與 AWS Identity and Access Management 許可給在 Amazon EKS 叢集中執行的工作負載的方式：適用於服務帳戶的 IAM 角色和 EKS 網繭身分。

服務帳戶的 IAM 角色

> 服務帳戶的 IAM 角色 (IRSA) 可設定執行在其上執行的 Kubernetes 應用程式，使用精細 AWS 的 IAM 許可存取各種其他 AWS 資源，例如 Amazon S3 儲存貯體、Amazon DynamoDB 資料表等。您可以在同一個 Amazon EKS 叢集中同時執行多個應用程式，並確保每個應用程式只有其所需的最低許可集。IRSA 是為了支援各種Kubernetes部署選項而建置， AWS 例如 Amazon

EKS、Amazon EKS Anywhere 不在，以及 Amazon EC2 執行個體上的自我管理Kubernetes叢集。Red Hat OpenShift Service on AWS因此，IRSA 是使用基礎 AWS 服務 (如 IAM) 建置的，而且沒有對 Amazon EKS 服務和 EKS API 採取任何直接相依性。如需詳細資訊，請參閱 服務帳戶的 IAM 角色。

EKS Pod 身分識別

EKS 網繭身分識別為叢集管理員提供簡化的工作流程，以驗證應用程式以存取各種其他 AWS 資源，例如 Amazon S3 儲存貯體、Amazon DynamoDB 表等。EKS Pod 身分識別僅適用於 EKS，因此可簡化叢集管理員設定 Kubernetes 應用程式以取得 IAM 許可的方式。這些權限現在可以透過 AWS Management Console EKS API 直接以較少的步驟輕鬆設定 AWS CLI，而且在任何Kubernetes物件中都不需要在叢集內採取任何動作。叢集管理員不需要在 EKS 和 IAM 服務之間切換，也不需要使用特殊權限 IAM 操作來設定應用程式所需的許可。IAM 角色現在可跨多個叢集使用，無需在建立新叢集時更新角色信任政策。EKS Pod 身分識別提供的 IAM 憑證包含角色工作階段標籤，並具有叢集名稱、命名空間、服務帳戶名稱等屬性。角色工作階段標記可讓系統管理員撰寫可跨服務帳戶使用的單一角色，方法是允許根據相符標籤存取 AWS 資源。如需詳細資訊，請參閱 EKS Pod 身分識別。

比較 EKS Pod 身分識別和 IRSA

整體上來說，EKS Pod 身分識別和 IRSA 都可讓您將 IAM 許可授予在 Kubernetes 叢集上執行的應用程式。但是，它們在設定方式、支援的限制和啟用的功能完全不同。下面，我們比較兩種解決方案的一些關鍵面向。

	EKS Pod 身分識別	IRSA
角色擴充性	您必須設定每個角色一次，才能與新推出的 Amazon EKS 服務主體 `pods.eks.amazonaws.com` 建立信任。完成此一次性步驟之後，您不需要每次在新叢集中使用角色的信任政策時更新該政策。	每次要在新叢集中使用 IAM 角色時，您都必須使用新的 EKS 叢集 OIDC 提供者端點更新該角色的信任政策。
叢集可擴展性	EKS Pod 身分識別不需要使用者設定 IAM OIDC 提供者，因此不適用此限制。	每個叢集具有與其相關聯的 OpenID Connect (OIDC) 發行者 URL。若要使用 IRSA，

	EKS Pod 身分識別	IRSA
		則必須為 IAM 中的每個 EKS 叢集建立唯一的 OpenID Connect 提供者。IAM 的預設全域限制為每個 AWS 帳戶 100 個 OIDC 提供者。如果您計劃每個具有 IRSA 的 EKS 叢集都 AWS 帳戶 有超過 100 個，則您將達到 IAM OIDC 提供者限制。
角色可擴展性	EKS Pod 身分識別不需要使用者在信任政策中定義 IAM 角色和服務帳戶之間的信任關係，因此不適用此限制。	在 IRSA 中，您可以在角色的信任政策中定義 IAM 角色和服務帳戶之間的信任關係。根據預設，信任政策大小的長度為 2048。這表示您通常可以在單一信任政策中定義 4 個信任關係。雖然您可以提高信任政策長度限制，但在單一信任政策中，您通常只能使用最多 8 個信任關係。
角色可重複使用性	AWS STS EKS Pod 身分提供的臨時認證包括角色工作階段標記，例如叢集名稱、命名空間、服務帳戶名稱。角色工作階段標籤可讓管理員編寫可搭配多個服務帳戶使用的單一 IAM 角色，並具有不同的有效許可，方法是允許根據連接至它們的標籤存取 AWS 資源。這也稱為屬性型存取控制 (ABAC)。如需詳細資訊，請參閱 定義 EKS Pod 身分識別的許可，以根據標籤擔任角色。	AWS STS 不支援階段作業標籤。您可以在叢集之間重複使用角色，但是每個 Pod 都會收到該角色的所有許可。

	EKS Pod 身分識別	IRSA
支援的環境	EKS Pod 身分識別僅適用於 Amazon EKS。	IRSA 可用於 Amazon EKS、Amazon EKS Anywhere，以及亞馬 Amazon EC2 執行個體上的自我管理 Kubernetes叢集。 Red Hat OpenShift Service on AWS
支援 EKS 版本	EKS Kubernetes 版本 `1.24` 或更新版本。如需特定平台版本，請參閱 EKS Pod 身分識別叢集版本。	所有支援的 EKS 叢集版本。

EKS Pod 身分識別

容器中Pod的應用程式可以使用 AWS SDK 或 AWS CLI 向 AWS 服務 使用 AWS Identity and Access Management (IAM) 許可發出 API 請求。應用程式必須使用 AWS 認證簽署其 AWS API 要求。

EKS Pod 身分識別提供管理應用程式憑證的功能，類似 Amazon EC2 執行個體設定檔將憑證提供給 Amazon EC2 執行個體的方式。您可以將 IAM 角色與服務帳戶建立關聯，並將其設定為使用 Kubernetes服務帳戶，而不是建立和分發 AWS 登入資料Pods到容器或使用 Amazon EC2 執行個體的角色。

每個 EKS Pod 身分識別關聯會將角色對應至指定叢集內命名空間中的服務帳戶。如果您在多個叢集中具有相同的應用程式，則可以在每個叢集中建立相同的關聯，而不必修改角色的信任政策。

如果 Pod 使用具有關聯的服務帳戶，則 Amazon EKS 會在 Pod 的容器中設定環境變數。環境變數會將 AWS SDK (包括) 設定為使用 EKS 網繭身分認證。 AWS CLI

EKS Pod 身分識別的優勢

EKS Pod 身分識別提供下列優勢：

- 最低權限 – 您可以將 IAM 許可範圍限定為服務帳戶，只有使用該服務帳戶的 Pods 才能存取這些許可。有此功能也就不需要第三方解決方案，例如 `kiam` 或 `kube2iam`。
- 憑證隔離：Pod's 的容器只能擷取與容器所使用服務帳戶相關聯之 IAM 角色的憑證。容器一律無法訪問其他 Pods 中其他容器使用的憑證。當使用 Pod 身分識別時，Pod's 容器也會向 Amazon EKS

節點 IAM 角色指派許可，除非您封鎖 Amazon EC2 執行個體中繼資料服務 (IMDS) 對 Pod 的存取權限。如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

- 可稽核性 — 可透過存取和事件記錄 AWS CloudTrail 來協助追溯性稽核。

相較於 服務帳戶的 IAM 角色，EKS Pod 身分識別是比較簡單的方法，因為此方法不使用 OIDC 身分識別提供者。EKS Pod 身分識別具有下列增強功能：

- 獨立操作 – 在許多組織中，建立 OIDC 身分識別提供者和管理 Kubernetes 叢集是不同團隊的責任。EKS Pod 身分識別具有完整的職責分離，其中 EKS Pod 身分識別關聯的所有組態都在 Amazon EKS 中完成，而 IAM 許可的所有組態則都在 IAM 中完成。
- 可重複使用性 – EKS Pod 身分識別會針對服務帳戶的 IAM 角色使用的每個叢集使用單一 IAM 主體，而不是使用不同的主體。您的 IAM 管理員會將下列主體新增至任何角色的信任政策，以便供 EKS Pod 身分識別使用。

```
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            }
```

- 延展性 — 每組臨時登入資料由 EKS Pod 身分中的EKS Auth服務承擔，而不是您在每個網繭中執行的每個 AWS SDK。然後，在每個節點上執行的 Amazon EKS Pod 身分識別代理程式會向 SDK 發出憑證。因此，每個節點的負載會減少為一次，而且不會在每個 Pod 中重複。如需該程序的詳細資訊，請參閱 EKS Pod 身分識別的運作方式。

如需比較兩種替代方案的詳細資訊，請參閱 使用服務帳戶授予 Kubernetes 工作負載存取 AWS 權 Kubernetes。

設定 EKS Pod 身分識別的概觀

完成下列程序以開啟 EKS Pod 身分識別：

1. 設定 Amazon EKS 網繭身分代理程式 – 對每個叢集只完成此程序一次。
2. 設定Kubernetes服務帳戶以採用具有 EKS 網繭身分識別的 IAM 角色 – 對您希望應用程式擁有的每種非重複可組合完成此程序。
3. 設定Pods為使用服Kubernetes務帳戶— 針對每個需要存取Pod的項目完成此程序 AWS 服務。
4. 使用支援的 AWS SDK— 確認工作負載使用受支援版本的 AWS SDK，並確認工作負載使用預設認證鏈結。

EKS Pod 身分識別考量

- 您可以在每個叢集中將一個 IAM 角色與每個 Kubernetes 服務帳戶建立關聯。您可以透過編輯 EKS Pod 身分識別關聯來變更對應至服務帳戶的角色。
- 您只能關聯與叢集位於相 AWS 帳戶 同的角色。您可以將存取權從另一個帳戶委派給此帳戶 (您為要使用的 EKS Pod 身分識別所設定) 中的角色。如需委派存取權的教學課程`AssumeRole`，請參閱《IAM 使用者指南》中的〈使用 IAM 角色在各個 AWS 帳戶之間委派存取權限〉。
- EKS Pod 身分識別代理程式是必要項目。它會在節點上以 Kubernetes `DaemonSet` 的形式執行，並且僅提供憑證給其執行所在節點上的 Pod。如需 EKS Pod 身分識別代理程式相容性的詳細資訊，請參閱下節 EKS Pod 身分識別限制。
- EKS Pod 身分識別代理程式會使用節點的 `hostNetwork`，並使用節點上 link-local 地址上的連接埠 `80` 和連接埠 `2703`。此地址對於 IPv4 為 `169.254.170.23`，對於 IPv6 叢集則為 `[fd00:ec2::23]`。

  如果您停用位`IPv6`址或以其他方式阻止本機主機 `IPv6` IP 位址，則代理程式無法啟動。若要在無法使用的節點上啟動代理程式`IPv6`，請遵循中`IPv6`在 EKS 網繭身分識別代理程式中停用的步驟停用`IPv6`組態。

EKS Pod 身分識別叢集版本

若要使用 EKS Pod 身分識別，叢集的平台版本必須與下表所列版本相同或更新，或是晚於表格中列出版本的 Kubernetes 版本。

Kubernetes 版本	平台版本
`1.30`	`eks.2`
`1.29`	`eks.1`
`1.28`	`eks.4`
`1.27`	`eks.8`
`1.26`	`eks.9`
`1.25`	`eks.10`
`1.24`	`eks.13`

與 EKS 網繭身分相容的附加版本

> **Important**
>
> 若要將 EKS 網繭身分與 EKS 附加元件搭配使用，您必須手動建立 EKS 網繭身分關聯。請勿在中的附加元件設定中選擇 IAM 角色 AWS Management Console，該角色只能與 IRSA 搭配使用。

需要 IAM 登入資料的 Amazon EKS 附加元件和自我管理附加元件可以使用 EKS 網繭身分識別、IRSA 或執行個體角色。使用支援 EKS Pod 身分的 IAM 登入資料的附加元件清單如下：

- Amazon VPC CNI plugin for Kubernetes`1.15.5-eksbuild.1`或更新版本
- AWS Load Balancer Controller`2.7.0`或更高版本。請注意，AWS Load Balancer Controller無法以 EKS 附加元件的形式提供，但可以作為自我管理的附加元件使用。

EKS Pod 身分識別限制

EKS Pod 身分識別適用於下列情況：

- 上一個主題 EKS Pod 身分識別叢集版本 中列出的 Amazon EKS 叢集版本。
- 叢集中為 Linux Amazon EC2 執行個體的工作節點。

EKS Pod 身分識別不適用於下列情況：

- 中國區域。
- AWS GovCloud (US).
- AWS Outposts.
- Amazon EKS Anywhere。
- 您在 Amazon EC2 上建立和執行的 Kubernetes 叢集。EKS Pod 身分識別元件僅適用於 Amazon EKS。

您無法將 EKS Pod 身分識別與下列項目搭配使用：

- 在 Linux Amazon EC2 執行個體以外的任何位置執行的 Pod。 AWS Fargate (Fargate) 不支援在上執行的 Linux 和視窗網繭。不支援在 Windows Amazon EC2 執行個體上執行的 Pod。

- 需要 IAM 憑證的 Amazon EKS 附加元件。EKS 附加元件只能改用服務帳戶的 IAM 角色。使用 IAM 憑證的 EKS 附加元件清單包含：
  - CSI存儲驅動程序：EBS CSI，EFS CSI，Amazon FSx for Lustre CSI 驅動程序，Amazon FSX 用於 NetApp ONTAP CSI 驅動程序，Amazon FSX 用於 OpenZFS CSI 驅動程序，Amazon 文件緩存 CSI 驅動程序， AWS 秘密和配置提供商（ASCP）的秘密存儲 CSI 驅動程序 Kubernetes

> **Note**
>
> 如果這些控制器、驅動程式和外掛程式安裝為自我管理的附加元件而非 EKS 附加元件，只要這些控制器、驅動程式和外掛程式已更新為使用最新 SDK，它們就會支援 EKS Pod 身分。 AWS

EKS Pod 身分識別的運作方式

Amazon EKS Pod 身分識別關聯提供管理應用程式憑證的功能，類似 Amazon EC2 執行個體設定檔將憑證提供給 Amazon EC2 執行個體的方式。

Amazon EKS Pod 身分識別透過其他 EKS 驗證 API 和在每個節點上執行的代理程式 Pod，為您的工作負載提供憑證。

在您的附加元件中，例如 Amazon EKS 附加元件和自我管理控制器、操作員和其他附加元件，作者需要更新其軟體以使用最 AWS 新的 SDK。如需 EKS Pod 身分識別與 Amazon EKS 產生的附加元件之間的相容性清單，請參閱上節 EKS Pod 身分識別限制。

在程式碼中使用 EKS Pod 身分識別

在您的程式碼中，您可以使用 AWS SDK 存取 AWS 服務。您撰寫程式碼以使用 SDK 建立 AWS 服務的用戶端，依預設，SDK 會在一連串位置中搜尋要使用的 AWS Identity and Access Management 認證。找到有效的憑證後，系統就會停止搜尋。有關使用的預設位置的詳細資訊，請參閱 AWS SDK 和工具參考指南中的認證提供者鏈。

EKS Pod 身分識別已新增至容器憑證提供者，系統會在預設憑證鏈中的某個步驟搜尋此提供者。如果您的工作負載目前使用憑證鏈中較早的憑證，那麼即使您為相同工作負載設定 EKS Pod 身分識別關聯，系統仍會繼續使用這些憑證。如此一來，您就可以先建立關聯，然後再移除舊憑證，以安全地從其他類型的憑證遷移。

容器憑證提供者會從每個節點上執行的代理程式提供臨時憑證。在 Amazon EKS 中，代理程式為 Amazon EKS Pod 身分識別代理程式，而在 Amazon Elastic Container Service 上，代理程式則為 `amazon-ecs-agent`。SDK 使用環境變數來尋找要連線的代理程式。

相反地，服務帳戶的 IAM 角色會提供 AWS SDK 必須使用的 Web 身分權杖交換`AssumeRoleWithWebIdentity`。 AWS Security Token Service

EKS Pod 身分識別代理程式如何搭配 Pod 使用

1. 當 Amazon EKS 啟動使用具有 EKS Pod 身分識別關聯之服務帳戶的全新 Pod 時，叢集會將下列內容新增至 Pod 清單檔案：

```
    env:
    - name: AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE
      value: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/eks-pod-
identity-token"
    - name: AWS_CONTAINER_CREDENTIALS_FULL_URI
      value: "http://169.254.170.23/v1/credentials"
    volumeMounts:
    - mountPath: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/"
      name: eks-pod-identity-token
  volumes:
  - name: eks-pod-identity-token
    projected:
      defaultMode: 420
      sources:
      - serviceAccountToken:
          audience: pods.eks.amazonaws.com
          expirationSeconds: 86400 # 24 hours
          path: eks-pod-identity-token
```

2. Kubernetes 會選取要在哪個節點上執行 Pod。然後，節點上的 Amazon EKS 網繭身分代理程式會使用該AssumeRoleForPodIdentity動作從 EKS 驗證 API 擷取臨時登入資料。
3. EKS Pod 身分識別代理程式可讓您在容器內執行的 AWS SDK 使用這些認證。
4. 您可以在應用程式中使用 SDK，無需指定憑證提供者來使用預設憑證鏈。或者，您可以指定容器憑證提供者。有關使用的預設位置的詳細資訊，請參閱 AWS SDK 和工具參考指南中的認證提供者鏈。
5. SDK 會使用環境變數來連線至 EKS Pod 身分識別代理程式，並擷取憑證。

> **Note**
>
> 如果您的工作負載目前使用憑證鏈中較早的憑證，那麼即使您為相同工作負載設定 EKS Pod 身分識別關聯，系統仍會繼續使用這些憑證。

設定 Amazon EKS 網繭身分代理程式

Amazon EKS Pod 身分識別關聯提供管理應用程式憑證的功能，類似 Amazon EC2 執行個體設定檔將憑證提供給 Amazon EC2 執行個體的方式。

Amazon EKS Pod 身分識別透過其他 EKS 驗證 API 和在每個節點上執行的代理程式 Pod，為您的工作負載提供憑證。

考量事項

- **IPv6**

  依預設，EKS 網繭身分識別代理程式會接聽網繭的IPv4和IPv6位址以要求認證。代理程式使用的回送 (本機主機) IP 位址，IPv4並使169.254.170.23用的本機主機 IP 位址。[fd00:ec2::23] IPv6

  如果您停用位IPv6址或以其他方式阻止本機主機 IPv6 IP 位址，則代理程式無法啟動。若要在無法使用的節點上啟動代理程式IPv6，請遵循中IPv6在 EKS 網繭身分識別代理程式中停用的步驟停用IPv6組態。

建立 Amazon EKS Pod 身分識別代理程式

代理程式先決條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。叢集版本和平台版本必須與 EKS Pod 身分識別叢集版本 中列出的版本相同或為更新版本。
- 節點角色具有代理程式在 EKS 驗證 API 中執行 AssumeRoleForPodIdentity 動作的許可。您可以使用 AWS 管理策略：AmazonEKS WorkerNodePolicy 或新增與下列類似的自訂政策：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks-auth:AssumeRoleForPodIdentity"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

此動作可能會受到標籤的限制，以限制使用代理程式的 Pod 可以擔任哪些角色。

- 這些節點可以從 Amazon ECR 連上和下載映像。附加元件的容器映像位於 Amazon 容器映像登錄檔中列出的登錄檔內。

  請注意 AWS Management Console，您可以變更映像位置，並在中的選擇性組態設定中提供 `imagePullSecrets` EKS 附加元件。`--configuration-values` AWS CLI
- 這些節點可以連上 Amazon EKS 驗證 API。對於私人叢集，中的`eks-auth`端點 AWS PrivateLink 是必要的。

AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取叢集，然後選取您要為其設定 EKS Pod 身分識別代理程式附加元件之叢集的名稱。
3. 選擇附加元件索引標籤。
4. 選擇取得更多附加元件。
5. 選取 EKS Pod 身分識別代理程式之附加元件方塊右上方的方塊，然後選擇下一步。
6. 在設定選取的附加元件設定頁面上，選取版本下拉式清單中的任何版本。
7. (選用) 展開選用組態設定以輸入其他組態。例如，您可以提供替代容器映像位置和 `ImagePullSecrets`。具有已接受金鑰的 JSON Schema 會顯示在附加元件組態結構描述中。

   在組態值中輸入組態金鑰和值。
8. 選擇下一步。
9. 確認 EKS Pod 身分識別代理程式 Pod 正在您的叢集上執行。

   ```
   kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
   ```

   範例輸出如下。

   ```
   eks-pod-identity-agent-gmqp7                                          1/1
    Running   1 (24h ago)   24h
   ```

```
eks-pod-identity-agent-prnsh                                    1/1
 Running   1 (24h ago)   24h
```

您現在可以在叢集中使用 EKS Pod 身分識別關聯。如需詳細資訊，請參閱 設定Kubernetes服務帳戶以採用具有 EKS 網繭身分識別的 IAM 角色。

AWS CLI

1. 執行下列 AWS CLI 命令。使用您叢集的名稱取代 `my-cluster`。

```
aws eks create-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --addon-version v1.0.0-eksbuild.1
```

> **Note**
>
> EKS Pod 身分識別代理程式不會針對服務帳戶的 IAM 角色使用 `service-account-role-arn`。您必須為 EKS Pod 身分識別代理程式提供節點角色的許可。

2. 確認 EKS Pod 身分識別代理程式 Pod 正在您的叢集上執行。

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

範例輸出如下。

```
eks-pod-identity-agent-gmqp7                                    1/1
 Running   1 (24h ago)   24h
eks-pod-identity-agent-prnsh                                    1/1
 Running   1 (24h ago)   24h
```

您現在可以在叢集中使用 EKS Pod 身分識別關聯。如需詳細資訊，請參閱 設定Kubernetes服務帳戶以採用具有 EKS 網繭身分識別的 IAM 角色。

更新 Amazon EKS Pod 身分識別代理程式

更新 Amazon EKS 類型的附加元件。如果您尚未將 Amazon EKS 類型的附加元件新增至叢集，請參閱 建立 Amazon EKS Pod 身分識別代理程式。

AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取叢集，然後選取您要為其設定 EKS Pod 身分識別代理程式附加元件之叢集的名稱。
3. 選擇附加元件索引標籤。
4. 如果有新版本的附加元件可用，則 EKS Pod 身分識別代理程式會出現更新版本按鈕。選取更新版本。
5. 在設定 Amazon EKS Pod 身分識別代理程式頁面上，從版本下拉式清單中選取新版本。
6. 選取儲存變更。

   更新動作可能需要幾秒鐘的時間才能完成。然後，檢查狀態來確保附加元件版本已更新。

AWS CLI

1. 查看叢集上目前安裝了哪些附加元件版本。使用您的叢集名稱取代 *my-cluster*。

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --query "addon.addonVersion" --output text
   ```

   範例輸出如下。

   ```
   v1.0.0-eksbuild.1
   ```

   您必須先建立附加元件，才能使用此程序進行更新。
2. 使用 AWS CLI更新您的附加元件。如果您想要使用 AWS Management Console 或 `eksctl` 更新附加元件，請參閱 更新附加元件。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令。
   - 使用您叢集的名稱取代 *my-cluster*。
   - 使用您想要的版本取代 *v1.0.0-eksbuild.1*。
   - 使用您的帳戶 ID 取代 *111122223333*。
   - 執行以下命令：

```
aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-
agent --addon-version v1.0.0-eksbuild.1'
```

更新動作可能需要幾秒鐘的時間才能完成。

3. 確認附加元件版本已更新。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-
agent
```

更新動作可能需要幾秒鐘的時間才能完成。

範例輸出如下。

```
{
    "addon": {
        "addonName": "eks-pod-identity-agent",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.0.0-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/eks-pod-
identity-agent/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "tags": {}
    }
}
```

EKS 網繭識別代理程式組態

IPv6在 EKS 網繭身分識別代理程式中停用

AWS Management Console

IPv6在中停用 AWS Management Console

1. 若要IPv6在 EKS 網繭身分識別代理程式中停用，請將下列組態新增至 EKS 附加元件的選用組態設定。

   a. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

   b. 在左側導覽窗格中，選取 Clusters (叢集)，然後選取您要為其設定附加元件之叢集的名稱。

   c. 選擇附加元件索引標籤。

   d. 選取 EKS 網繭識別代理程式附加元件方塊右上角的核取方塊，然後選擇 [編輯]。

   e. 在 [設定 EKS 網繭身分識別代理程式] 頁面上：

      i. 選取您要使用的 Version (版本)。我們建議您保留與上一步相同的版本，並以單獨的動作更新版本和配置。

      ii. 展開選用組態設定。

      iii. 輸入巢狀 JSON 物件的 JSON 金鑰`"agent":`和值，並`"additionalArgs":`在組態值中輸入索引鍵。產生的文字必須是有效的 JSON 物件。如果此金鑰和值是文字方塊中唯一的資料，請以大括號 `{}` 括住該金鑰和值。下列範例顯示網路原則已啟用：

      ```
      {
          "agent": {
              "additionalArgs": {
                  "-b": "169.254.170.23"
              }
          }
      }
      ```

      此組態會將位IPv4址設定為代理程式使用的唯一位址。

   f. 若要透過取代 EKS 網繭身分識別代理程式網繭來套用新組態，請選擇儲存變更。

   Amazon EKS 使用KubernetesDaemonSet針對 EKS 網繭身分識別代理程式的部署，將變更套用至 EKS 附加元件。您可以在 AWS Management Console 和中的附加元件的更

新歷史記錄中追蹤推出的狀態。`kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`

`kubectl rollout`具有以下命令：

```
$ kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

如果部署時間太長，Amazon EKS 將撤銷推出，並在附加元件的更新歷史記錄中新增附加元件更新類型和失敗狀態的訊息。若要調查任何問題，請從部署的歷史記錄開始，然後在 EKS 網繭身分識別代理程式網繭`kubectl logs`上執行，以查看 EKS 網繭識別代理程式的記錄。

2. 如果更新歷程記錄中的新項目狀態為「成功」，則表示推出已完成，且附加元件正在使用所有 EKS 網繭身分識別代理程式網繭中的新組態。

AWS CLI

IPv6在中停用 AWS CLI

- 若要IPv6在 EKS 網繭身分識別代理程式中停用，請將下列組態新增至 EKS 附加元件的組態值。

  執行下列 AWS CLI 命令。將 `my-cluster` 取代為您的叢集名稱，並將 IAM 角色 ARN 取代為您正在使用的角色。

  ```
  aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent \
      --resolve-conflicts PRESERVE --configuration-values '{"agent": {"additionalArgs": { "-b": "169.254.170.23"}}}'
  ```

  此組態會將位IPv4址設定為代理程式使用的唯一位址。

Amazon EKS 使用Kubernetes DaemonSet 針對 EKS 網繭身分識別代理程式的部署，將變更套用至 EKS 附加元件。您可以在 AWS Management Console 和中的附加元件的更新歷史記錄中追蹤推出的狀態。`kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`

`kubectl rollout`具有以下命令：

```
kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

如果部署時間太長，Amazon EKS 將撤銷推出，並在附加元件的更新歷史記錄中新增附加元件更新類型和失敗狀態的訊息。若要調查任何問題，請從部署的歷史記錄開始，然後在 EKS 網繭身分識別代理程式網繭`kubectl logs`上執行，以查看 EKS 網繭識別代理程式的記錄。

設定Kubernetes服務帳戶以採用具有 EKS 網繭身分識別的 IAM 角色

本主題說明如何將Kubernetes服務帳戶設定為具有 EKS Pod 身分識別的 AWS Identity and Access Management (IAM) 角色。接Pods著，任何設定為使用服務帳戶的使用者都可以存取 AWS 服務 該角色具有存取權限的任何項目。

若要建立 EKS 網繭身分識別關聯，只需一個步驟；您可以透過 AWS Management Console、、 AWS CLI AWS SDK AWS CloudFormation 和其他工具在 EKS 中建立關聯。在任何 Kubernetes 物件中，叢集內部的關聯沒有任何資料或中繼資料，也不會將任何註解新增至服務帳戶。

必要條件

- 現有的叢集。如果您沒有，則可以按照其中一個 Amazon EKS 入門指南來建立。
- 建立關聯的 IAM 主體必須具有 `iam:PassRole`。
- 在您的裝置或上 AWS CLI 安裝和設定的最新版本 AWS CloudShell。您可以使用 `aws --version | cut -d / -f2 | cut -d ' ' -f1` 來檢查您的目前版本。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最

新版本，請參閱《 AWS Command Line Interface 使用指南》aws configure中的〈安裝、更新 AWS CLI和解除安裝〉和〈快速設定〉。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《 AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。
- 包含叢集組態的現有 kubectl config 檔案。若要建立 kubectl config 檔案，請參閱建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

建立 EKS Pod 身分識別關聯

AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取叢集，然後選取您要為其設定 EKS Pod 身分識別代理程式附加元件之叢集的名稱。
3. 選擇存取索引標籤。
4. 在 Pod 身分識別關聯中，選擇建立。
5. 對於 IAM 角色，選取具有您希望工作負載擁有許可的 IAM 角色。

> **Note**
>
> 此清單僅包含具有下列信任政策的角色，允許 EKS Pod 身分識別使用這些角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
```

```
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole`

EKS Pod 身分識別會先使用 `AssumeRole` 來擔任 IAM 角色，然後將臨時憑證傳至您的 Pod。

`sts:TagSession`

EKS Pod 身分識別會用 `TagSession` 在向 AWS STS的請求中包含工作階段標籤。

您可以在信任政策的 condition keys 中使用這些標籤，用來限制哪些服務帳戶、命名空間和叢集可以使用此角色。

如需 Amazon EKS 條件金鑰的清單，請參閱服務授權參考中的 Amazon Elastic Kubernetes Service 定義的條件。若要了解您可以搭配哪些動作和資源使用條件索引鍵，請參閱 Amazon Elastic Kubernetes Service 定義的動作。

6. 對於 Kubernetes 命名空間，選取包含服務帳戶和工作負載的 Kubernetes 命名空間。或者，您可以使用名稱 (不存在於叢集中) 指定命名空間。
7. 對於 Kubernetes 服務帳戶，選取要使用的 Kubernetes 服務帳戶。Kubernetes 工作負載的清單檔案必須指定此服務帳戶。或者，您可以使用名稱 (不存在於叢集中) 指定服務帳戶。
8. (選用) 對於標籤，請選擇新增標籤，以在鍵值對中新增中繼資料。這些標籤會套用至關聯，可用於 IAM 政策。

   您可以多次重複此步驟以新增多個標籤。
9. 選擇建立。

AWS CLI

1. 如果您要將現有 IAM 政策與 IAM 角色建立關聯，請跳到下一步驟。

建立 IAM 政策。您可以建立自己的政策，或複製已授予您所需的某些許可的 AWS 受管政策，並根據您的特定要求加以自訂。如需詳細資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策。

a. 根據您要讓 Pods 存取的 AWS 服務 ，建立內含相關許可的檔案。如需所有動作的清單 AWS 服務，請參閱服務授權參考。

   您可以執行以下命令來建立允許唯讀存取 Amazon S3 儲存貯體的範例政策檔案。您可以選擇性地將組態資訊或引導指令碼存放在此儲存貯體中，而 Pod 中的容器可從儲存貯體讀取檔案，並載入至您的應用程式。如果您要建立此範例政策，請將以下內容複製到您的裝置。將 *my-pod-secrets-bucket* 取代為您的儲存貯體名稱並執行命令。

```
cat >my-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
EOF
```

b. 建立 IAM 政策。

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. 建立 IAM 角色並與 Kubernetes 服務帳戶建立關聯。

1. 如果您有要擔任 IAM 角色的現有 Kubernetes 服務帳戶，則可略過此步驟。

   建立 Kubernetes 服務帳戶。將以下內容複製到您的裝置。視需要將 *my-service-account* 取代為您要使用的名稱，將 *default* 取代為其他命名空間。如果您變更 *default*，命名空間必須已經存在。

```
cat >my-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
```

```
metadata:
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

執行下列命令。

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

2. 執行下列命令以建立 IAM 角色的信任政策檔案。

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
EOF
```

3. 建立角色。將 *my-role* 取代為您的 IAM 角色名稱，並將 *my-role-description* 取代為您角色的描述。

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

4. 將 IAM 政策連接至您的角色。將 *my-role* 取代為您的 IAM角色名稱，將 *my-policy* 取代為您建立的現有政策名稱。

```
aws iam attach-role-policy --role-name my-role --policy-
arn=arn:aws:iam::111122223333:policy/my-policy
```

> **Note**
>
> 與服務帳戶的 IAM 角色不同，EKS Pod 身分識別不會在服務帳戶上使用註釋。

5. 執行下列命令以建立關聯。視需要使用叢集名稱取代 `my-cluster`，使用您要使用的名稱取代 *`my-service-account`*，使用其他命名空間取代 *`default`*。

```
aws eks create-pod-identity-association --cluster-name my-cluster --role-
arn arn:aws:iam::111122223333:role/my-role --namespace default --service-
account my-service-account
```

範例輸出如下。

```
{
    "association": {
        "clusterName": "my-cluster",
        "namespace": "default",
        "serviceAccount": "my-service-account",
        "roleArn": "arn:aws:iam::111122223333:role/my-role",
        "associationArn": "arn:aws::111122223333:podidentityassociation/my-
cluster/a-abcdefghijklmnop1",
        "associationId": "a-abcdefghijklmnop1",
        "tags": {},
        "createdAt": 1700862734.922,
        "modifiedAt": 1700862734.922
    }
}
```

> **Note**
>
> 您可以使用名稱 (不存在於叢集中) 指定命名空間和服務帳戶。您必須建立命名空間、服務帳戶以及使用服務帳戶進行 EKS Pod 身分識別關聯才能運作的工作負載。

3. 確認角色和服務帳戶設定正確。

   a. 確認 IAM 角色的信任政策設定正確。

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

範例輸出如下。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow EKS Auth service to assume this role for Pod
 Identities",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

b. 確認您在上一步連接至角色的政策已連接至該角色。

```
aws iam list-attached-role-policies --role-name my-role --query
 AttachedPolicies[].PolicyArn --output text
```

範例輸出如下。

```
arn:aws:iam::111122223333:policy/my-policy
```

c. 設定變數以存放您要使用之政策的 Amazon Resource Name (ARN)。將 *my-policy* 取代為您要確認許可的政策名稱。

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. 檢視預設政策版本。

```
aws iam get-policy --policy-arn $policy_arn
```

範例輸出如下。

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

e. 檢視政策內容，以確定政策包含您的 Pod 需要的所有許可。視需要將以下命令中的 *1* 取代為上一個輸出中傳回的版本。

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

範例輸出如下。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

如果您在上一步建立了範例政策，則輸出結果相同。如果您建立了不同的政策，則 *example* 內容有所不同。

下一步驟

[設定Pods為使用服Kubernetes務帳戶]

設定Pods為使用服Kubernetes務帳戶

如果Pod需要存取 AWS 服務，則必須將其設定為使用Kubernetes服務帳戶。服務帳戶必須與具有存取權限的 AWS Identity and Access Management (IAM) 角色相關聯 AWS 服務。

必要條件

- 現有的叢集。如果您沒有，則可以使用其中一個 Amazon EKS 入門 指南來建立一個。
- 將服務帳戶與 IAM 角色建立關聯的現有 Kubernetes 服務帳戶和 EKS Pod 身分識別關聯。該角色必須具 有相關聯的 IAM 政策，其中包含您希望 Pods 必須具備以便使用 AWS 服務的許可。如需有關服務帳戶和角色之建立和設定方式的詳細資訊，請參閱設定Kubernetes服務帳戶以採用具有 EKS 網繭身分識別的 IAM 角色。
- 在您的裝置或上 AWS CLI 安裝和設定的最新版本 AWS CloudShell。您可以使用 `aws --version | cut -d / -f2 | cut -d ' ' -f1` 來檢查您的目前版本。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《 AWS Command Line Interface 使用指南》`aws configure`中的〈安裝、更新 AWS CLI和解除安裝〉和〈快速設定〉。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《 AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 已在裝置或 AWS CloudShell上安裝 `kubectl` 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 `1.29`，則可以搭配使用 `kubectl` `1.28`、`1.29` 或 `1.30` 版。若要安裝或升級 `kubectl`，請參閱 安裝或更新 kubectl。
- 包含叢集組態的現有 `kubectl config` 檔案。若要建立 `kubectl config` 檔案，請參閱建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

設定 Pod 以使用服務帳戶

1. 使用以下命令，以建立部署清單檔案，讓您可部署 Pod 以確認組態。以您自己的值取代 *example values*。

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
```

```
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. 將清單檔案部署到叢集。

```
kubectl apply -f my-deployment.yaml
```

3. 確認您的 Pod 有必要的環境變數。

   a. 檢視使用上一步中的部署作業進行部署的 Pods。

   ```
   kubectl get pods | grep my-app
   ```

   範例輸出如下。

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. 確認 Pod 具有服務帳戶字符檔案掛載。

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
    AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:
   ```

   範例輸出如下。

   ```
   AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:  /var/run/secrets/
   pods.eks.amazonaws.com/serviceaccount/eks-pod-identity-token
   ```

4. 確認您Pods可以 AWS 服務 使用您在附加到角色的 IAM 政策中指派的許可與互動。

> **Note**
>
> 當Pod使用與服務帳戶相關聯的 IAM 角色的 AWS 登入資料時，容器中的 AWS CLI 或其他 SDK Pod 使用該角色提供的認證。如果您沒有對提供給 Amazon EKS 節點 IAM 角色的認證限制存取權，則 Pod 仍然可以存取這些認證。如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

如果您的 Pods 無法如預期與服務互動，請完成以下步驟以確認所有項目都已正確設定。

a. 確認您Pods使用支援透過 EKS 網繭身分識別關聯假設 IAM 角色的 AWS SDK 版本。如需詳細資訊，請參閱 使用支援的 AWS SDK。

b. 確認部署使用服務帳戶。

```
kubectl describe deployment my-app | grep "Service Account"
```

範例輸出如下。

```
Service Account:  my-service-account
```

定義 EKS Pod 身分識別的許可，以根據標籤擔任角色

EKS Pod 身分識別會將標籤連接至具有叢集名稱、命名空間、服務帳戶名稱等屬性的每個 Pod 的臨時憑證。這些角色工作階段標記可讓系統管理員撰寫可跨服務帳戶使用的單一角色，方法是允許根據相符標籤存取 AWS 資源。透過新增對角色工作階段標籤的支援，客戶可以在叢集之間和叢集內的工作負載之間強制執行更嚴格的安全界限，同時重複使用相同的 IAM 角色和 IAM 政策。

例如，如果物件標記為 EKS 叢集名稱，下列政策即可執行 `s3:GetObject` 動作。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectTagging"
            ],
            "Resource": "*",
            "Condition": {
```

```
                "StringEquals": {
                    "s3:ExistingObjectTag/eks-cluster-name": "${aws:PrincipalTag/eks-
cluster-name}"
                }
            }
        }
    ]
}
```

由 EKS Pod 身分識別新增的工作階段標籤清單

下列清單包含新增至 Amazon EKS 所提出 AssumeRole 請求之標籤的所有金鑰。以 ${aws:PrincipalTag/kubernetes-namespace} 為例，若要在政策中使用這些標籤，請在 ${aws:PrincipalTag/ 後面使用金鑰。

- eks-cluster-arn
- eks-cluster-name
- kubernetes-namespace
- kubernetes-service-account
- kubernetes-pod-name
- kubernetes-pod-uid

跨帳戶標籤

EKS Pod 身分識別新增的所有工作階段標籤都可轉移；標籤金鑰和值會傳遞至工作負載用於將角色切換至其他帳戶的任何 AssumeRole 動作。您可以在其他帳戶的政策中使用這些標籤，以限制跨帳戶情況中的存取。如需詳細資訊，請參閱《IAM 使用者指南》中的使用工作階段標籤鏈結角色。

自訂標籤

EKS Pod 身分識別無法將其他自訂標籤新增至其執行的 AssumeRole 動作。不過，您套用至 IAM 角色的標籤永遠都可以使用，但格式相同：${aws:PrincipalTag/ 後面是金鑰，例如 ${aws:PrincipalTag/MyCustomTag}。

> **Note**
>
> 在發生衝突的情況下，透過 sts:AssumeRole 請求新增至工作階段的標籤具有優先順序。例如，假設當 EKS 擔任客戶角色時，Amazon EKS 將金鑰 eks-cluster-name 和值 my-cluster 新增至工作階段。您也將 eks-cluster-name 標籤新增至具有值 my-own-

cluster 的 IAM 角色。在這種情況下，前者優先，eks-cluster-name 標籤的值會是 my-cluster。

使用支援的 AWS SDK

Important

較早版本的文件不正確。適用於 Java V1 的 AWS 開發套件不支援 EKS 網繭身分識別。

使用時EKS Pod 身分識別，您的容器Pods必須使用 AWS SDK 版本，該版本支援從 EKS Pod 身分識別代理程式假設 IAM 角色。請確定您的 AWS SDK 使用下列版本或更新版本：

- Java (版本 2) – 2.21.30
- Go v1 – v1.47.11
- Go v2 – release-2023-11-14
- Python（肉毒桿菌 3）— 1.34.41
- Python（肉毒核）— 1.34.41
- AWS CLI — 1.30.0

  AWS CLI — 2.15.0
- JavaScript V 2 —
- JavaScript V3
- 科特林 — V1.0.1
- Ruby – 3.188.0
- 銹-釋放
- C ++
- 無線網路 — 3.7.734.0
- PowerShell — 4.1.502
- PHP – 3.287.1

為了確保您使用支援的開發套件，在建立您的容器時，請遵循在 AWS上建立的工具中您偏好開發套件的安裝指示。

如需支援 EKS Pod 身分識別的附加元件清單，請參閱與 EKS 網繭身分相容的附加版本。

使用 EKS Pod 身分識別憑證

若要使用來自 EKS Pod 身分識別關聯的認證，您的程式碼可以使用任何 AWS SDK 為具有 SDK 的 AWS 服務建立用戶端，依預設，SDK 會在一連串位置中搜尋要使用的認 AWS Identity and Access Management 證。如果您在建立用戶端或以其他方式初始化 SDK 時未指定憑證提供者，則系統會使用 EKS Pod 身分識別憑證。

這樣是有效的，因為 EKS Pod 身分識別已新增至容器憑證提供者，系統會在預設憑證鏈中的某個步驟搜尋此提供者。如果您的工作負載目前使用憑證鏈中較早的憑證，那麼即使您為相同工作負載設定 EKS Pod 身分識別關聯，系統仍會繼續使用這些憑證。

如需 EKS Pod 身分識別運作方式的詳細資訊，請參閱 EKS Pod 身分識別的運作方式。

EKS Pod 身分識別角色

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole`

EKS Pod 身分識別會先使用 `AssumeRole` 來擔任 IAM 角色，然後將臨時憑證傳至您的 Pod。

`sts:TagSession`

EKS Pod 身分識別會用 `TagSession` 在向 AWS STS的請求中包含工作階段標籤。

您可以在信任政策的 condition keys 中使用這些標籤，用來限制哪些服務帳戶、命名空間和叢集可以使用此角色。

如需 Amazon EKS 條件金鑰的清單，請參閱服務授權參考中的 Amazon Elastic Kubernetes Service 定義的條件。若要了解您可以搭配哪些動作和資源使用條件索引鍵，請參閱 Amazon Elastic Kubernetes Service 定義的動作。

服務帳戶的 IAM 角色

容器中Pod的應用程式可以使用 AWS SDK 或 AWS CLI 向 AWS 服務 使用 AWS Identity and Access Management (IAM) 許可發出 API 請求。應用程式必須使用 AWS 認證簽署其 AWS API 要求。服務帳戶的 IAM 角色提供管理應用程式憑證的功能，類似 Amazon EC2 執行個體設定檔將憑證提供給 Amazon EC2 執行個體的方式。您可以將 IAM 角色與服務帳戶建立關聯，並將其設定為使用Kubernetes服務帳戶，而不是建立和分發 AWS 登入資料Pods到容器或使用 Amazon EC2 執行個體的角色。您無法將 IAM 角色用於在 AWS Outposts上具有 Amazon EKS 本機叢集的服務帳戶。

服務帳戶的 IAM 角色提供下列優點：

- 最低權限 – 您可以將 IAM 許可範圍限定為服務帳戶，只有使用該服務帳戶的 Pods 才能存取這些許可。有此功能也就不需要第三方解決方案，例如 `kiam` 或 `kube2iam`。
- 憑證隔離：Pod's 的容器只能擷取與容器所使用服務帳戶相關聯之 IAM 角色的憑證。容器一律無法訪問其他 Pods 中其他容器使用的憑證。當使用服務帳戶的 IAM 角色時，Pod's 容器也會向 Amazon EKS 節點 IAM 角色指派權限，除非您封鎖 Amazon EC2 執行個體中繼資料服務 (IMDS) 對 Pod 的存取權限。如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。
- 可稽核性 — 可透過存取和事件記錄， AWS CloudTrail 以協助確保回溯性稽核。

完成下列程序，為服務帳戶啟用 IAM 角色：

1. 為您的叢集建立 IAM OIDC 提供者 – 對每個叢集只完成此程序一次。

> **Note**
>
> 如果您啟用 EKS VPC 端點，則無法從該 VPC 內部存取 EKS OIDC 服務端點。因此，您的作業 (例如在 VPC 利用 `eksctl` 建立 OIDC 提供者) 將無法運作，並且當嘗試請求 `https://oidc.eks.region.amazonaws.com` 時，會導致逾時。以下是範例錯誤訊息：

```
** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
```

若要完成此步驟，您可以在 VPC 之外執行命令，例如在連接至網際網路的電腦上 AWS CloudShell 或在連線到網際網路的電腦上執行該命令。

2. 設定Kubernetes服務帳戶以擔任 IAM 角色 – 對您希望應用程式擁有的每種非重複可組合完成此程序。
3. 設定Pods為使用服Kubernetes務帳戶— 針對每個需要存取Pod的項目完成此程序 AWS 服務。
4. 使用支援的 AWS 開發套件— 確認工作負載使用受支援版本的 AWS SDK，並確認工作負載使用預設認證鏈結。

IAM、Kubernetes 和 OpenID Connect (OIDC) 背景資訊

在 2014 年，使用 OpenID Connect (OIDC) AWS Identity and Access Management 新增了對聯合身分的支援。此功能可讓您透過支援的身分識別提供者驗證 AWS API 呼叫，並接收有效的 OIDC JSON Web Token (JWT)。您可以將此權杖傳遞至 AWS STS `AssumeRoleWithWebIdentity` API 作業，並接收 IAM 臨時角色登入資料。您可以使用這些憑證與任何 AWS 服務互動，例如 Amazon S3 和 DynamoDB。

每個 JWT 權杖均由一個簽署金鑰對簽署。這些金鑰是由 Amazon EKS 管理的 OIDC 供應商提供服務，且私有金鑰每 7 天就會輪換一次。Amazon EKS 會保留公有金鑰，直到它們過期為止。如果您連接外部 OIDC 用戶端，請注意，您需要在公開金鑰到期之前重新整理簽署金鑰。了解如何 the section called “擷取簽署金鑰”。

Kubernetes 長期以來一直使用服務帳戶作為本身的內部身分系統。Pods 可以使用只有 Kubernetes API 伺服器才能驗證的自動掛載 Token (非 OIDC JWT) 向 Kubernetes API 伺服器進行驗證。這些舊的服務帳戶 Token 不會過期，而輪換簽署金鑰是一項困難的程序。Kubernetes 版本 `1.12` 進一步支援新的 `ProjectedServiceAccountToken` 功能。此功能是 OIDC JSON Web Token，同時包含服務帳戶身分，並支援可設定的對象。

Amazon EKS 為每個叢集託管公有 OIDC 探索端點，其中包含 `ProjectedServiceAccountToken` JSON Web Token 的簽署金鑰，讓外部系統 (例如 IAM) 可以驗證和接受 Kubernetes 發出的 OIDC Token。

為您的叢集建立 IAM OIDC 提供者

您的叢集具有與其相關聯的 OpenID Connect (OIDC) 發行者 URL。若要將 AWS Identity and Access Management (IAM) 角色用於服務帳戶，叢集的OIDC發行OIDC者 URL 必須存在 IAM 提供者。

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到您的主目錄〉。
- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。
- 包含叢集組態的現有 kubectl config 檔案。若要建立 kubectl config 檔案，請參閱建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

您可以使用 eksctl 或 AWS Management Console為叢集建立 IAM OIDC 提供者。

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

使用 **eksctl** 為您的叢集建立 IAM OIDC 身分提供者

1. 確定叢集的 OIDC 發行者 ID。

   擷取叢集的 OIDC 發行者 ID 並將其存放在變數中。使用您自己的值取代 *my-cluster*。

```
cluster_name=my-cluster
```

```
oidc_id=$(aws eks describe-cluster --name $cluster_name --query
 "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
```

```
echo $oidc_id
```

2. 判斷您的帳戶中是否已經有擁有叢集發行者 ID 的 IAM OIDC 供應商。

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

如果有輸出傳回，表示您已經有叢集 IAM OIDC 提供者，可以略過下一步驟。如果未傳回任何輸出，則您必須為叢集建立 IAM OIDC 供應商。

3. 使用下列命令為您的叢集建立 IAM OIDC 身分提供者。

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name --approve
```

> **Note**
>
> 如果您啟用 EKS VPC 端點，則無法從該 VPC 內部存取 EKS OIDC 服務端點。因此，您的作業 (例如在 VPC 利用 `eksctl` 建立 OIDC 提供者) 將無法運作，並且當嘗試請求 `https://oidc.eks.region.amazonaws.com` 時，會導致逾時。以下是範例錯誤訊息：
>
> ```
> ** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
> ```
>
> 若要完成此步驟，您可以在 VPC 之外執行命令，例如在連接至網際網路的電腦上 AWS CloudShell 或在連線到網際網路的電腦上執行該命令。

AWS Management Console

若要建立您的叢集的 IAM OIDC 身分識別提供者，請使用 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

2. 在左側窗格中，選取 Clusters (叢集)，然後在 Clusters (叢集) 頁面上選取您的叢集名稱。
3. 在 Overview (概觀) 標籤的 Details (詳細資訊) 區段中，記下 OpenID Connect provider URL (OpenID Connect 供應商 URL) 的值。
4. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
5. 在左側導覽窗格中，選擇 Access management (存取管理) 下的 Identity Providers (身分提供者)。如果列出的提供商與您叢集的 URL 相符，則表示您已經擁有叢集提供商。如果沒有列出符合叢集 URL 的提供商，則您必須建立一個。
6. 若要建立供應商，請選擇 Add provider (新增供應商)。
7. 對於 Provider type (供應商類型)，選取 OpenID Connect。
8. 對於 Provider URL (供應商 URL)，輸入叢集的 OIDC 供應商 URL，然後選擇 Get thumbprint (取得指紋)。
9. 對於 Audience (對象)，輸入 **`sts.amazonaws.com`**，然後選擇 Add provider (新增提供商)。

下一步驟

設定Kubernetes服務帳戶以擔任 IAM 角色

設定Kubernetes服務帳戶以擔任 IAM 角色

本主題介紹如何將Kubernetes服務帳戶設定為擔任 AWS Identity and Access Management (IAM) 角色。任何設定為使用該服務帳戶的 Pods 都可以存取該角色有權存取的任何 AWS 服務 。

必要條件

- 現有的叢集。如果您沒有，則可以按照其中一個 Amazon EKS 入門指南來建立。
- 叢集的現有 IAM OpenID Connect (OIDC) 供應商。若要了解是否已經擁有，或是了解如何建立，請參閱為您的叢集建立 IAM OIDC 提供者。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本`1.27.160`或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。`2.12.3`若要檢查您目前的版本，請使用 **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到您的主目錄〉。

- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。
- 包含叢集組態的現有 kubectl config 檔案。若要建立 kubectl config 檔案，請參閱建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

將 IAM 角色與 Kubernetes 服務帳戶建立關聯

1. 如果您要將現有 IAM 政策與 IAM 角色建立關聯，請跳到下一步驟。

   建立 IAM 政策。您可以建立自己的原則，或複製已授與您需要的某些權限的 AWS 受管理原則，並根據您的特定需求自訂該原則。如需詳細資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策。

   a. 建立一個檔案，其中包 AWS 服務 含您要存取Pods的權限。如需所有動作的清單 AWS 服務，請參閱服務授權參考。

      您可以執行以下命令來建立允許唯讀存取 Amazon S3 儲存貯體的範例政策檔案。您可以選擇性地將組態資訊或引導指令碼存放在此儲存貯體中，而 Pod 中的容器可從儲存貯體讀取檔案，並載入至您的應用程式。如果您要建立此範例政策，請將以下內容複製到您的裝置。將 *my-pod-secrets-bucket* 取代為您的儲存貯體名稱並執行命令。

      ```
      cat >my-policy.json <<EOF
      {
          "Version": "2012-10-17",
          "Statement": [
              {
                  "Effect": "Allow",
                  "Action": "s3:GetObject",
                  "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
              }
          ]
      }
      EOF
      ```

   b. 建立 IAM 政策。

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. 建立 IAM 角色並與 Kubernetes 服務帳戶建立關聯。您可以使用 eksctl 或 AWS CLI。

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

將 *my-service-account* 取代為您希望 eksctl 建立並與 IAM 角色建立關聯的 Kubernetes 服務帳戶名稱。將 *default* 取代為您要 eksctl 在其中建立服務的命名空間。使用您叢集的名稱取代 *my-cluster*。將 *my-role* 取代為您希望與服務帳戶建立關聯的角色名稱。如果不存在，eksctl 會為您建立。使用您的帳戶 ID 取代 *111122223333*，再以現有政策名稱取代 *my-policy*。

```
eksctl create iamserviceaccount --name my-service-account --namespace default --cluster my-cluster --role-name my-role \
    --attach-policy-arn arn:aws:iam::111122223333:policy/my-policy --approve
```

> **Important**
>
> 如果角色或服務帳戶已經存在，上一個命令可能會失敗。您可以為 eksctl 提供在這些情況下的不同選項。如需詳細資訊，請執行 **eksctl create iamserviceaccount --help**。

AWS CLI

1. 如果您有要擔任 IAM 角色的現有 Kubernetes 服務帳戶，則可略過此步驟。

   建立 Kubernetes 服務帳戶。將以下內容複製到您的裝置。視需要將 *my-service-account* 取代為您要使用的名稱，將 *default* 取代為其他命名空間。如果您變更 *default*，命名空間必須已經存在。

```
cat >my-service-account.yaml <<EOF
apiVersion: v1
```

```
kind: ServiceAccount
metadata:
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

2. 使用以下命令將 AWS 帳戶 ID 設置為環境變量。

```
account_id=$(aws sts get-caller-identity --query "Account" --output text)
```

3. 使用下列命令將叢集的 OIDC 身分提供者設定為環境變數。使用您叢集的名稱取代 *my-cluster*。

```
oidc_provider=$(aws eks describe-cluster --name my-cluster --region
 $AWS_REGION --query "cluster.identity.oidc.issuer" --output text | sed -e "s/
^https:\/\///")
```

4. 設定命名空間和服務帳戶名稱的變數。將 *my-service-account* 取代為要擔任角色的 Kubernetes 服務帳戶。將 *default* 取代為服務帳戶的命名空間。

```
export namespace=default
export service_account=my-service-account
```

5. 執行下列命令以建立 IAM 角色的信任政策檔案。如果您想要允許命名空間中的所有服務帳戶使用該角色，請將以下內容複製到您的裝置。替換*StringEquals*為**StringLike**並替換*$ ####*。*您可以在 StringEquals 和 StringLike 條件中新增多個項目，以允許多個服務帳戶或命名空間擔任角色。若要允許來自其他 AWS 帳戶 (叢集所在帳戶以外的帳戶) 的角色擔任該角色，請參閱跨帳戶 IAM 許可以了解詳情。

```
cat >trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::$account_id:oidc-provider/$oidc_provider"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
```

```
          "$oidc_provider:aud": "sts.amazonaws.com",
          "$oidc_provider:sub": "system:serviceaccount:
$namespace:$service_account"
        }
      }
    }
  ]
}
EOF
```

6. 建立角色。將 *my-role* 取代為您的 IAM 角色名稱，並將 *my-role-description* 取代為您角色的描述。

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

7. 將 IAM 政策連接至您的角色。將 *my-role* 取代為您的 IAM角色名稱，將 *my-policy* 取代為您建立的現有政策名稱。

```
aws iam attach-role-policy --role-name my-role --policy-arn=arn:aws:iam::
$account_id:policy/my-policy
```

8. 使用您希望服務帳戶擔任之 IAM 角色的 Amazon Resource Name (ARN) 標註您的服務帳戶。使用現有 IAM 角色的名稱取代 *my-role*。假設您允許與叢集所在帳戶 AWS 帳戶 不同的角色擔任上一個步驟中的角色。然後，確保從另一個帳戶指定 AWS 帳戶 和角色。如需詳細資訊，請參閱 跨帳戶 IAM 許可。

```
kubectl annotate serviceaccount -n $namespace $service_account
 eks.amazonaws.com/role-arn=arn:aws:iam::$account_id:role/my-role
```

3. 確認角色和服務帳戶設定正確。

   a. 確認 IAM 角色的信任政策設定正確。

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

   範例輸出如下。

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:default:my-
service-account",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
```

b. 確認您在上一步連接至角色的政策已連接至該角色。

```
aws iam list-attached-role-policies --role-name my-role --query
 AttachedPolicies[].PolicyArn --output text
```

範例輸出如下。

```
arn:aws:iam::111122223333:policy/my-policy
```

c. 設定變數以存放您要使用之政策的 Amazon Resource Name (ARN)。將 *my-policy* 取代為您要確認許可的政策名稱。

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. 檢視預設政策版本。

```
aws iam get-policy --policy-arn $policy_arn
```

範例輸出如下。

```
{
```

```
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

e. 檢視政策內容，以確定政策包含您的 Pod 需要的所有許可。視需要將以下命令中的 *1* 取代為上一個輸出中傳回的版本。

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

範例輸出如下。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

如果您在上一步建立了範例政策，則輸出結果相同。如果您建立了不同的政策，則 *example* 內容有所不同。

f. 確認 Kubernetes 服務帳戶標註了角色。

```
kubectl describe serviceaccount my-service-account -n default
```

範例輸出如下。

```
Name:                my-service-account
Namespace:           default
Annotations:         eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/my-role
```

```
Image pull secrets:  <none>
Mountable secrets:   my-service-account-token-qqjfl
Tokens:              my-service-account-token-qqjfl
[...]
```

4. (選擇性) 設定服務帳戶的 AWS Security Token Service 端點。 AWS 建議使用區域 AWS STS 端點而非全域端點。這樣可以減少延遲、提供內建備援，並增加工作階段字符的有效性。

下一步驟

設定Pods為使用服Kubernetes務帳戶

設定Pods為使用服Kubernetes務帳戶

如果Pod需要存取 AWS 服務，則必須將其設定為使用Kubernetes服務帳戶。服務帳戶必須與具有存取權限的 AWS Identity and Access Management (IAM) 角色相關聯 AWS 服務。

必要條件

- 現有的叢集。如果您沒有，則可以使用其中一個 Amazon EKS 入門 指南來建立一個。
- 叢集的現有 IAM OpenID Connect (OIDC) 供應商。若要了解是否已經擁有，或是了解如何建立，請參閱為您的叢集建立 IAM OIDC 提供者。
- 與 IAM 角色相關聯的現有 Kubernetes 服務帳戶。服務帳戶必須標註 IAM 角色的 Amazon Resource Name (ARN)。該角色必須具 有相關聯的 IAM 政策，其中包含您希望 Pods 必須具備以便使用 AWS 服務的許可。如需有關服務帳戶和角色之建立和設定方式的詳細資訊，請參閱設定Kubernetes服務帳戶以擔任 IAM 角色。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。

- 包含叢集組態的現有 kubectl config 檔案。若要建立 kubectl config 檔案，請參閱建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

設定 Pod 以使用服務帳戶

1. 使用以下命令，以建立部署清單檔案，讓您可部署 Pod 以確認組態。以您自己的值取代 *example values*。

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. 將清單檔案部署到叢集。

```
kubectl apply -f my-deployment.yaml
```

3. 確認您的 Pod 有必要的環境變數。

   a. 檢視使用上一步中的部署作業進行部署的 Pods。

   ```
   kubectl get pods | grep my-app
   ```

   範例輸出如下。

```
my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
```

b. 檢視 Pod 所使用 IAM 角色的 ARN。

```
kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_ROLE_ARN:
```

範例輸出如下。

```
AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/my-role
```

角色 ARN 必須與您用來標註現有服務帳戶的角色 ARN 相符。如需有關標註服務帳戶的詳細資訊，請參閱設定Kubernetes服務帳戶以擔任 IAM 角色。

c. 確認 Pod 有 Web 身分字符檔案掛載。

```
kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
 AWS_WEB_IDENTITY_TOKEN_FILE:
```

範例輸出如下。

```
AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

kubelet 請求並代表 Pod 存放字符。根據預設，如果字符超過總存留時間的 80% 或 24 小時，kubelet 會重新整理字符。您可以使用 Pod 規格中的設定來修改任何帳戶 (預設服務帳戶除外) 的過期持續時間。如需詳細資訊，請參閱 Kubernetes 文件中的服務帳戶字符磁碟區投影。

叢集上的 Amazon EKS Pod Identity Webhook 會監視使用具有以下註釋之服務帳戶的 Pods：

```
eks.amazonaws.com/role-arn: arn:aws:iam::111122223333:role/my-role
```

Webhook 會將先前的環境變數套用至這些 Pods。您的叢集不需要使用 Webhook 來設定環境變數和字符檔案掛載。您可以手動將 Pods 設定為具有這些環境變數。支援的 AWS SDK 版本會先在憑證鏈提供者中尋找這些環境變數。符合此條件的 Pods 會使用角色憑證。

4. 確認您Pods可以 AWS 服務 使用您在附加到角色的 IAM 政策中指派的許可與互動。

> **Note**
>
> 當Pod使用與服務帳戶相關聯的 IAM 角色的 AWS 登入資料時，容器中的 AWS CLI 或其他 SDK Pod 使用該角色提供的認證。如果您沒有對提供給 Amazon EKS 節點 IAM 角色的認證限制存取權，則 Pod 仍然可以存取這些認證。如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

如果您的 Pods 無法如預期與服務互動，請完成以下步驟以確認所有項目都已正確設定。

a. 確認您Pods使用的 AWS SDK 版本支持通過 OpenID Connect Web 身份令牌文件假設 IAM 角色。如需詳細資訊，請參閱 使用支援的 AWS 開發套件。

b. 確認部署使用服務帳戶。

```
kubectl describe deployment my-app | grep "Service Account"
```

範例輸出如下。

```
Service Account:  my-service-account
```

c. 如果您的 Pods 仍然無法存取服務，請查看設定Kubernetes服務帳戶以擔任 IAM 角色中描述的步驟，以確認您的角色和服務帳戶設定正確。

設定服務帳戶的 AWS Security Token Service 端點

如果您在搭配使用Kubernetes服務帳戶服務帳戶的 IAM 角色，則可以設定服務帳戶所使用的 AWS Security Token Service 端點類型 (如果叢集和平台版本與下表所列版本相同或更新)。如果您的 Kubernetes 或平台版本早於表中列出的版本，則服務帳戶只能使用全域端點。

Kubernetes 版本	平台版本	預設端點類型
1.30	eks.2	區域性
1.29	eks.1	區域性
1.28	eks.1	區域性

Kubernetes 版本	平台版本	預設端點類型
1.27	eks.1	區域性
1.26	eks.1	區域性
1.25	eks.1	區域性
1.24	eks.2	區域性
1.23	eks.1	區域性

AWS 建議使用區域 AWS STS 端點而非全域端點。這樣可以減少延遲、提供內建備援，並增加工作階段字符的有效性。 AWS Security Token Service 必須在正在執行的 AWS 區域 位置處處於作Pod用中狀態。此外，您的應用程式必須具 AWS 區域 有不同的內建備援，以便在 AWS 區域. 如需詳細資訊，請參閱 AWS STS《IAM 使用者指南》 AWS 區域中的「管理」。

必要條件

- 現有的叢集。如果您沒有，則可以使用其中一個 Amazon EKS 入門 指南來建立一個。
- 叢集的現有 IAM OIDC 提供商。如需詳細資訊，請參閱 為您的叢集建立 IAM OIDC 提供者。
- 現有的 Kubernetes 服務帳戶設定為與服務帳戶的 Amazon EKS IAM 功能搭配使用。

設定 Kubernetes 服務帳戶使用的端點類型

以下範例均使用由 Amazon VPC CNI 外掛程式所使用的 aws-node Kubernetes 服務帳戶。您可以將 *example values* 取代為您自己的服務帳戶、Pods、命名空間和其他資源。

1. 根據您要變更端點的服務帳戶，選取使用該服務帳戶的 Pod。確定Pod運 AWS 區域 行在哪一個。將 *aws-node-6mfgv* 取代為 Pod 名稱，並將 *kube-system* 取代為 Pod 的命名空間。

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep Node:
   ```

   範例輸出如下。

   ```
   ip-192-168-79-166.us-west-2/192.168.79.166
   ```

   AWS 區域

2. 判斷 Pod's 服務帳戶所使用的端點類型。

```
kubectl describe pod aws-node-6mfgv -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
```

範例輸出如下。

```
AWS_STS_REGIONAL_ENDPOINTS: regional
```

如果目前端點是全域範圍，則輸出中會傳回 `global`。如果未傳回任何輸出，則預設端點類型正在使用中且未被覆寫。

3. 如果您的叢集或平台版本與表中列出的版本相同或更高，則可以使用以下命令之一，將服務帳戶使用的端點類型從預設類型變更為其他類型。將 *aws-node* 取代為您服務帳戶的名稱，以及將 *kube-system* 取代為服務帳戶的命名空間。

   - 如果您的預設或目前端點類型是全域範圍，且您想將其範圍變更為區域：

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=true
     ```

     如果您是使用 服務帳戶的 IAM 角色 來產生在 Pods 容器中執行的應用程式中的預簽署 S3 URL，則區域端點的 URL 格式會與以下範例類似：

     ```
     https://bucket.s3.us-west-2.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
     ```

   - 如果您的預設或目前端點類型是區域範圍，且您想將其範圍變更為全域：

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=false
     ```

     如果您的應用程式明確向 AWS STS 全球端點發出請求，且您未覆寫在 Amazon EKS 叢集中使用區域端點的預設行為，則請求將失敗並顯示錯誤。如需詳細資訊，請參閱 Pod 容器會接收到下列錯誤：An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region。

     如果您使用 服務帳戶的 IAM 角色 來產生在 Pods 容器中執行的應用程式中的預簽署 S3 URL，則全域端點的 URL 格式會與以下範例類似：

```
https://bucket.s3.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
```

如果您的自動化需要以特定格式使用預先簽署的 URL，或者您的應用程式或使用預先簽署 URL 的下游相依性 AWS 區域 對目標有期望，請進行必要的變更以使用適當 AWS STS 的端點。

4. 刪除並重新建立任何與服務帳戶相關聯的現有 Pods，以套用登入資料環境變數。變動 Webhook 不會將這些變數套用到已在執行中的 Pods。您可以使用所設定標註的 Pods 資訊，來取代 *Pods*、*kube-system* 和 *-l k8s-app=aws-node*。

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

5. 確認 Pods 全部重新啟動。

```
kubectl get Pods -n kube-system -l k8s-app=aws-node
```

6. 檢視其中一個 Pods 的環境變數。確認 AWS_STS_REGIONAL_ENDPOINTS 值是您在上一個步驟中所設定的值。

```
kubectl describe pod aws-node-kzbtr -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
```

範例輸出如下。

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

跨帳戶 IAM 許可

您可以從另一個帳戶的叢集建立身分提供者，或使用鏈結的 AssumeRole 操作，設定跨帳戶 IAM 許可。在以下範例中，帳戶 A 擁有的 Amazon EKS 叢集支援服務帳戶的 IAM 角色。在該叢集上執行的 Pods 必須從帳戶 B 取得 IAM 許可。

Example 從另一個帳戶的叢集建立身分提供者

Example

在此範例中，帳戶 A 將為帳戶 B 提供其叢集的 OpenID Connect (OIDC) 發行者 URL。帳戶 B 遵循 為您的叢集建立 IAM OIDC 提供者 和 設定Kubernetes服務帳戶以擔任 IAM 角色 中的指示，使用來自帳

戶 A 叢集的 OIDC 發行者 URL。然後，叢集管理員會註釋帳戶 A 叢集中的服務帳戶，以使用帳戶 B (*444455556666*) 中的角色。

```
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::444455556666:role/account-b-role
```

Example 使用鏈結的 **`AssumeRole`** 操作

Example

在此範例中，帳戶 B 會建立 IAM 政策，並授予許可給帳戶 A 叢集中的 Pods。帳戶 B (*444455556666*) 將該政策連接至 IAM 角色，該角色有信任關係將 AssumeRole 許可授予帳戶 A (*111122223333*)。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

帳戶 A 使用信任政策建立角色，該政策會從使用叢集 OIDC 發行者位址建立的身分提供者取得憑證。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.region-
code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
```

```
      "Action": "sts:AssumeRoleWithWebIdentity"
    }
  ]
}
```

帳戶 A 將政策連接到該角色，並給予以下許可來擔任帳戶 B 建立的角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::444455556666:role/account-b-role"
        }
    ]
}
```

Pods 擔任帳戶 B 角色的應用程式碼使用兩個設定檔：account_b_role 和 account_a_role。account_b_role 描述檔使用 account_a_role 描述檔作為其來源。對於 AWS CLI，~/.aws/config檔案類似下列內容。

```
[profile account_b_role]
source_profile = account_a_role
role_arn=arn:aws:iam::444455556666:role/account-b-role

[profile account_a_role]
web_identity_token_file = /var/run/secrets/eks.amazonaws.com/serviceaccount/token
role_arn=arn:aws:iam::111122223333:role/account-a-role
```

若要為其他 AWS SDK 指定鏈結的設定檔，請參閱您正在使用之 SDK 的說明文件。如需詳細資訊，請參閱建置在其上的工具 AWS。

使用支援的 AWS 開發套件

使用時服務帳戶的 IAM 角色，您的容器Pods必須使用支援透過 OpenID Connect Web 身分權杖檔案假設 IAM 角色的 AWS SDK 版本。請確定您的 AWS SDK 使用下列版本或更新版本：

- Java (第 2 版) – 2.10.11
- Java – 1.11.704
- Go – 1.23.13

- Python (Boto3) – 1.9.220
- Python (botocore) – 1.12.200
- AWS CLI — 1.16.232
- 節點：2.525.0 和 3.27.0
- Ruby – 3.58.0
- C++ – 1.7.174
- .NET：3.3.659.1 – 您也必須包含 `AWSSDK.SecurityToken`。
- PHP – 3.110.7

許多熱門的 Kubernetes 附加元件，例如 Cluster Autoscaler、什麼是 AWS Load Balancer Controller？和 Amazon VPC CNI plugin for Kubernetes，皆支援服務帳戶的 IAM 角色。

為了確保您使用支援的開發套件，在建立您的容器時，請遵循在 AWS上建立的工具中您偏好開發套件的安裝指示。

使用憑證

若要將 IAM 角色的登入資料用於服務帳戶，您的程式碼可以使用任何 AWS SDK 為具有 SDK 的 AWS 服務建立用戶端，依預設，SDK 會在一連串位置中搜尋要使用的認 AWS Identity and Access Management 證。如果您在建立用戶端或以其他方式初始化 SDK 時未指定憑證提供者，則系統會使用服務帳戶憑證的 IAM 角色。

這樣是有效的，因為服務帳戶的 IAM 角色已新增為預設憑證鏈中的一個步驟。如果您的工作負載目前使用憑證鏈中較早的憑證，那麼即使您為相同工作負載設定服務帳戶的 IAM 角色，系統仍會繼續使用這些憑證。

SDK 會 AWS Security Token Service 透過使用動`AssumeRoleWithWebIdentity`作，自動將服務帳戶OIDC權杖交換為暫時認證。Amazon EKS 和此 SDK 動作會繼續輪換臨時憑證，方法是在臨時憑證到期前進行更新。

擷取簽署金鑰

Kubernetes每個發`ProjectedServiceAccountToken`出一個KubernetesService Account。這個令牌是一個OIDC令牌，它進一步是一種類型JSON web token (JWT)。Amazon EKS 會為每個叢集託管一個公有OIDC端點，其中包含權杖的簽署金鑰，以便外部系統可以對其進行驗證。

要驗證`ProjectedServiceAccountToken`，您需要獲取OIDC公共簽名密鑰，也稱為JSON Web Key Set (JWKS). 在應用程式中使用這些金鑰來驗證權杖。例如，您可以使用 PyJWT Python 庫來驗證

使用這些密鑰的令牌。如需有關的詳細資訊ProjectedServiceAccountToken，請參閱the section called “IAM、Kubernetes 和 OpenID Connect (OIDC) 背景資訊”。

必要條件

- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。
- AWS CLI— 用於處理包括 Amazon EKS 在內的 AWS 服務的命令行工具。如需詳細資訊，請參閱《 AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI。安裝之後 AWS CLI，我們建議您也對其進行配置。如需詳細資訊，請參閱《 AWS Command Line Interface 使用者指南》中的使用 aws configure 進行快速組態設定。

獲取OIDC公共簽名密鑰（ AWS CLI ）

1. 使用擷取 Amazon EKS 叢集的 OIDC URL。 AWS CLI

```
$ aws eks describe-cluster --name my-cluster --query 'cluster.identity.oidc.issuer'
"https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE"
```

2. 使curl用或類似工具檢索公共簽名密鑰。結果是一個 JSON Web Key Set (JWKS).

> **Important**
>
> Amazon EKS 節流到端點的呼叫。OIDC您應該緩存公共簽名密鑰。尊重響應中包含的cache-control標題。

> **Important**
>
> Amazon EKS 每七天輪換一次OIDC簽名密鑰。

```
$ curl https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE/keys
{"keys":
[{"kty":"RSA","kid":"2284XXXX4a40","use":"sig","alg":"RS256","n":"wklbXXXXMVfQ","e":"AQAB"}
```

Amazon EKS 節點

Kubernetes 節點是執行容器化應用程式的機器。每個節點皆具有下列元件：

- 容器執行階段：負責執行容器的軟體。
- **kubelet**：確保容器健康狀態良好並在其關聯的 Pod 中執行。
- **kube-proxy**：維護允許與 Pods 通訊的網路規則。

如需詳細資訊，請參閱 Kubernetes 文件中的節點。

您的 Amazon EKS 叢集可以在自我管理的節點、Amazon EKS 受管節點群組和 AWS Fargate 的任何組合上排程 Pods。若要進一步了解在叢集中部署的節點，請參閱 檢視 Kubernetes 資源。

Important

AWS Fargate Amazon EKS 在 AWS GovCloud （美國東部）和 AWS GovCloud （美國西部）中不可用。

Note

節點必須與您在建立叢集時選取的子網位於同一 VPC 中。不過，節點不需要位於相同的子網中。

下表提供了幾個準則，以便在決定哪些選項最符合您的要求時進行評估。此表不包括在 Amazon EKS 之外建立的連接節點，這些節點只能檢視。

Note

Bottlerocket 與此表格中的一般資訊有一些特定的差異。如需詳細資訊，請參閱 GitHub 上的 Bottlerocket 文件。

條件	EKS 受管節點群組	自我管理的節點	AWS Fargate
可以部署到 AWS Outposts	否	是	否
可以部署到 AWS Local Zone	否	是：如需詳細資訊，請參閱 Amazon EKS 和 AWS 本機區域。	否
可以執行需要 Windows 的容器	是	是：您的叢集仍然需要至少一個 (建議使用兩個) Linux 節點。	否
可以執行需要 Linux 的容器	是	是	是
可以執行需要 Inferentia 晶片的工作負載	是：僅限 Amazon Linux 節點	是：僅限 Amazon Linux	否
可以執行需要 GPU 的工作負載	是：僅限 Amazon Linux 節點	是：僅限 Amazon Linux	否
可以執行需要 Arm 處理器的工作負載	是	是	否
可以運行 AWS 瓶子	是	是	否
Pod 與其他 Pods 共享核心執行階段階段環境	是：每個節點上的所有 Pods	是：每個節點上的所有 Pods	否：每個 Pod 皆有專用核心
Pod 會與其他 Pods 共享 CPU、記憶體、儲存體和網路資源。	是：可能會導致每個節點上未使用的資源	是：可能會導致每個節點上未使用的資源	否：每個 Pod 皆有專用資源，可以獨立調整大小，以最大化資源使用率。

條件	EKS 受管節點群組	自我管理的節點	AWS Fargate
Pod 可以使用的硬體和記憶體比 Pod 規格要求的更多	是：若 Pod 需要的資源超過請求，且節點上有資源可用，則 Pod 可以使用其他資源。	是：若 Pod 需要的資源超過請求，且節點上有資源可用，則 Pod 可以使用其他資源。	否：Pod 可以使用較大的 vCPU 和記憶體組態重新部署。
必須部署和管理 Amazon EC2 執行個體	是：如果您部署了 Amazon EKS 優化 AMI，則可透過 Amazon EKS 自動化。如果部署了自訂 AMI，則必須手動更新執行個體。	是：提供手動設定或使用 Amazon EKS 提供的 AWS CloudFormation 範本部署 Linux (x86)、Linux (Arm) 或 Windows 節點。	否
必須保護、維護和修補 Amazon EC2 執行個體的作業系統	是	是	否
可以在部署節點時提供啟動程序引數，例如額外的 `kubelet` 引數。	是 – 以自訂 AMI 使用 `eksctl` 或啟動範本	是：如需詳細資訊，請檢視 GitHub 上的引導指令碼使用資訊。	否
可以將 IP 地址指派給來自不同於將 IP 地址指派給節點之 CIDR 區塊的 Pods。	是：以自訂 AMI 使用啟動範本。如需詳細資訊，請參閱 使用啟動範本自訂受管節點。	是：如需詳細資訊，請參閱 Pod 的自訂聯網。	否

條件	EKS 受管節點群組	自我管理的節點	AWS Fargate
可以對節點執行 SSH	是	是	否：沒有節點主機作業系統可供 SSH 使用。
可以將自己的自訂 AMI 部署至節點	是：使用啟動範本	是	否
可以將您自己的自訂 CNI 部署至節點	是：以自訂 AMI 使用啟動範本	是	否
必須自行更新節點 AMI	是：如果已部署 Amazon EKS 最佳化 AMI，Amazon EKS 主控台會在有可用更新時通知您。您只要在主控台中按一下即可執行更新。如果已部署自訂 AMI，Amazon EKS 主控台不會在有更新可用時通知您。您必須自行執行更新。	是：使用 Amazon EKS 主控台以外的工具。這是因為無法使用 Amazon EKS 主控台來管理自我管理節點。	否

條件	EKS 受管節點群組	自我管理的節點	AWS Fargate
必須自行更新節點 Kubernetes 版本	是：如果已部署 Amazon EKS 最佳化 AMI，Amazon EKS 主控台會在有可用更新時通知您。您只要在主控台中按一下即可執行更新。如果已部署自訂 AMI，Amazon EKS 主控台不會在有更新可用時通知您。您必須自行執行更新。	是：使用 Amazon EKS 主控台以外的工具。這是因為無法使用 Amazon EKS 主控台來管理自我管理節點。	否：您未管理節點。
可以透過 Pods 使用 Amazon EBS 儲存體	是	是	否
可以透過 Pods 使用 Amazon EFS 儲存體	是	是	是
可以透過 Pods 使用 Amazon FSx for Lustre 儲存體	是	是	否
可以針對服務使用 Network Load Balancer	是	是	是，當使用 建立 Network Load Balancer
Pod 可以在公有子網路中執行	是	是	否
可以將不同的 VPC 安全群組指派給個別 Pods	是：僅 Linux 節點	是：僅 Linux 節點	是
可以執行 Kubernetes DaemonSets	是	是	否

條件	EKS 受管節點群組	自我管理的節點	AWS Fargate
支援 Pod 清單檔案中的 `HostPort` 和 `HostNetwork`	是	是	否
AWS 區域 可用性	所有 Amazon EKS 支援的區域	所有 Amazon EKS 支援的區域	部分 Amazon EKS 支援的區域
可以在 Amazon EC2 專用主機上執行容器	是	是	否
定價	執行多個 Pods 的 Amazon EC2 執行個體成本。如需詳細資訊，請參閱 Amazon EC2 定價。	執行多個 Pods 的 Amazon EC2 執行個體成本。如需詳細資訊，請參閱 Amazon EC2 定價。	獨立 Fargate 記憶體和 CPU 組態的成本。每個 Pod 皆有自己的成本。如需詳細資訊，請參閱 AWS Fargate 定價。

受管節點群組

Amazon EKS 受管節點群組會自動化 Amazon EKS Kubernetes 叢集節點 (Amazon EC2 執行個體) 的佈建和生命週期管理。

使用 Amazon EKS 受管節點群組時，不需要個別佈建或註冊提供運算容量的 Amazon EC2 執行個體，來執行 Kubernetes 應用程式。您可以透過單一操作來建立、自動更新或終止叢集的節點。節點更新和終止會自動耗盡節點，以確保您的應用程式保持可用。

每個受管節點均會佈建為 Amazon EKS 為您管理的 Amazon EC2 Auto Scaling 群組的一部分。包括執行個體和 Auto Scaling 群組的每個資源都會在您的 AWS 帳戶中執行。每個節點群組都可以跨您定義的多個可用區域執行。

您可以使用 Amazon EKS 主控台、 AWS CLI; AWS API 或基礎設施做為程式碼工具`eksctl`，將受管節點群組新增至新的或現有 AWS CloudFormation叢集。作為受管節點群組的一部份而啟動的節點，會自動標記為由 KubernetesCluster Autoscaler 進行自動探索。您可以使用節點群組將 Kubernetes 標籤套用至節點，並隨時加以更新。

使用 Amazon EKS 受管節點群組不需要額外費用，您只需為佈建的 AWS 資源付費。其中包括 Amazon EC2 執行個體、Amazon EBS 磁碟區、Amazon EKS 叢集時數和任何其他 AWS 基礎設施。沒有最低費用，也沒有前期承諾。

若要開始使用新的 Amazon EKS 叢集和受管節點群組，請參閱 開始使用 Amazon EKS — AWS Management Console 和 AWS CLI。

若要將受管理的節點群組新增至現有叢集，請參閱 建立受管節點群組。

受管節點群組概念

- Amazon EKS 受管節點群組會為您建立和管理 Amazon EC2 執行個體。
- 每個受管節點均會佈建為 Amazon EKS 為您管理的 Amazon EC2 Auto Scaling 群組的一部分。此外，包括 Amazon EC2 執行個體和 Auto Scaling 群組在內的所有資源都在您的 AWS 帳戶中執行。
- 受管節點群組的 Auto Scaling 群組會跨越您在建立群組時指定的每個子網路。
- Amazon EKS 會標記受管節點群組資源，以便將其設定為使用 Kubernetes Cluster Autoscaler。

> **Important**
>
> 如果您跨越多個由 Amazon EBS 磁碟區提供的可用區域執行狀態應用程式，並且使用 Kubernetes 自動擴展，您應該設定多個節點群組，每個群組的範圍皆為單一可用區域。另外，您應該啟用 `--balance-similar-node-groups` 功能。

- 部署受管節點時，您可以使用自訂啟動範本來獲得更高層級的靈活性和自訂能力。例如，可指定額外的 `kubelet` 引數並使用自訂 AMI。如需詳細資訊，請參閱 使用啟動範本自訂受管節點。如果在第一次建立受管節點群組時未使用自訂啟動範本，會有一個自動產生的啟動範本。請勿手動修改此自動產生的範本，否則會發生錯誤。
- Amazon EKS 會遵循受管節點群組上 CVE 和安全修補程式的共同責任模型。受管節點執行 Amazon EKS 最佳化 AMI 時，Amazon EKS 負責在報告錯誤或問題時建置 AMI 的修補版本。我們可以發佈修正程式。不過，您必須負責將這些修補 AMI 版本部署至受管節點群組。當受管節點執行自訂 AMI 時，您必須負責在報告錯誤或問題時建置 AMI 的修補版本，然後部署 AMI。如需詳細資訊，請參閱 更新受管節點群組。
- Amazon EKS 受管節點群組可以在公有和私有子網路中啟動。如果在 2020 年 4 月 22 日或之後啟動公有子網路中的受管節點群組，則子網路必須將 `MapPublicIpOnLaunch` 設定為「true」，執行個體才能成功加入叢集。如果公有子網路是在 2020 年 3 月 26 日`eksctl`或之後使用或 Amazon EKS 供貨 AWS CloudFormation 範本建立的，則此設定已設定為 true。如果公有子網路是在 2020 年 3

月 26 日之前所建立，則必須手動變更設定。如需詳細資訊，請參閱修改子網路的公有 IPv4 定址屬性。

- 在私有子網路中部署受管節點群組時，您必須確保該群組可存取 Amazon ECR 以提取容器映像。您可以透過將 NAT 閘道連接至子網路的路由表，或新增以下 AWS PrivateLink VPC 端點來執行此操作：
  - Amazon ECR API 端點界面 – `com.amazonaws.region-code.ecr.api`
  - Amazon ECR Docker 登錄檔 API 端點界面 – `com.amazonaws.region-code.ecr.dkr`
  - Amazon S3 閘道端點 – `com.amazonaws.region-code.s3`

  如需了解其他常用的服務和端點，請參閱 私有叢集要求。
- 受管節點群組無法部署在或本機區域中，AWS Outposts或部署在 AWS Wavelength L AWS ocal Zones 中。
- 您可以在單一叢集內建立多個受管節點群組。例如，您可以使用標準 Amazon EKS 針對某些工作負載最佳化的 Amazon Linux AMI 建立一個節點群組，另一個使用 GPU 變體來建立節點群組，用於需要 GPU 支援的工作負載。
- 如果受管節點群組遇到 Amazon EC2 執行個體狀態檢查故障問題，Amazon EKS 會傳回錯誤代碼，以協助您診斷問題。如需詳細資訊，請參閱 受管節點群組錯誤代碼。
- Amazon EKS 會將 Kubernetes 標籤新增至受管節點群組執行個體。這些 Amazon EKS 提供的標籤前面會加上 `eks.amazonaws.com`。
- 在終止或更新期間，Amazon EKS 會使用 Kubernetes API 自動耗盡節點。
- 使用 `AZRebalance` 終止節點或減少所需的節點計數時，不會遵守 Pod 中斷預算。這些動作會嘗試在該節點上移出 Pods。但如果需要超過 15 分鐘，則無論節點上的所有 Pods 是否終止，都會終止節點。若要延長期間直到節點終止，請將 lifecycle hook 新增至 Auto Scaling 群組。如需詳細資訊，請參閱 Amazon EC2 Auto Scaling 使用者指南中的新增 lifecycle hook。
- 若要在收到 Spot 中斷通知或容量重新平衡通知後正確執行耗盡程序，必須將 `CapacityRebalance` 設定為 `true`。
- 更新受管理節點群組會遵守您為 Pods 設定的 Pod 中斷預算。如需詳細資訊，請參閱 受管節點更新行為。
- 使用 Amazon EKS 受管節點群組不會產生額外成本。您只需為佈建的 AWS 資源付費。
- 如果要為節點加密 Amazon EBS 磁碟區，則可以使用啟動範本部署節點。若要在不使用啟動範本的情況下部署具有加密 Amazon EBS 磁碟區的受管節點，請加密帳戶中建立的所有新 Amazon EBS 磁碟區。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的預設加密。

受管節點群組容量類型

建立受管節點群組時，您可以選擇隨需或 Spot 容量類型。Amazon EKS 部署具有 Amazon EC2 Auto Scaling 群組的受管節點群組，該群組僅包含隨需執行個體或僅包含 Amazon EC2 Spot 執行個體。您可以將具有容錯能力之應用程式的 Pods 排程至 Spot 受管節點群組，以及將其排程至單一 Kubernetes 叢集中的隨需節點群組。依預設，受管節點群組會部署隨需 Amazon EC2 執行個體。

On-Demand

透過隨需執行個體，您只需要按秒數支付運算容量開銷，無需簽訂長期合約。

運作方式

依預設，如果未指定 Capacity Type (容量類型)，則會使用隨需執行個體佈建受管節點群組。受管節點群組會代表您設定 Amazon EC2 Auto Scaling 群組，並套用下列設定：

- 佈建隨需容量的配置策略設定為 `prioritized`。在滿足隨需容量時，受管節點群組會 API 中傳遞執行個體類型的順序，決定先使用哪一種執行個體類型。例如，您可以按照下列順序指定三種執行個體類型：`c5.large`、`c4.large`，以及 `c3.large`。當您的隨需執行個體啟動時，受管節點群組會從 `c5.large` 開始，然後是 `c4.large`，再來是 `c3.large`，以滿足隨需容量。如需詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的 Amazon EC2 Auto Scaling 群組。
- Amazon EKS 會將下列 Kubernetes 標籤新增至指定容量類型之受管節點群組中的所有節點：`eks.amazonaws.com/capacityType: ON_DEMAND`。您可以使用此標籤，在隨需節點上排程可設定狀態或可容錯的應用程式。

Spot

Amazon EC2 Spot 執行個體是備用的 Amazon EC2 容量，可提供隨需價格的極低折扣。EC2 需要取回容量時，就可能會以兩分鐘中斷通知的方式中斷 Amazon EC2 Spot 執行個體。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的競價型執行個體。您可以使用 Amazon EC2 Spot 執行個體設定受管節點群組，以最佳化 Amazon EKS 叢集中執行之運算節點的成本。

運作方式

若要在受管節點群組內使用 Spot 執行個體，請將容量類型設定為`spot`，以建立受管節點群組。受管節點群組代表您設定 Amazon EC2 Auto Scaling 群組，並套用下列 Spot 最佳實務：

- 為確保您的 Spot 節點已在最佳 Spot 容量集區中佈建，配置策略會設定為下列之一：

  - `price-capacity-optimized` (PCO) – 在具有 Kubernetes 版本 `1.28` 或以上版本的叢集中建立新節點群組時，配置策略會設定為 `price-capacity-optimized`。但是，在 Amazon EKS 受管節點群組開始支援 PCO 之前，系統不會變更已使用 `capacity-optimized` 建立的節點群組的配置策略。
  - `capacity-optimized` (CO) – 在具有 Kubernetes 版本 `1.27` 或以下版本的叢集中建立新節點群組時，配置策略會設定為 `capacity-optimized`。

  若要增加可用於配置容量的 Spot 容量集區數量，請將受管節點群組設定為使用多個執行個體類型。
- 已啟用 Amazon EC2 Spot 容量重新平衡功能，以便 Amazon EKS 可以正常地消耗和重新平衡 Spot 節點，在 Spot 節點中斷風險提高時，將應用程式中斷情況降至最低。如需詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的 Amazon EC2 Auto Scaling 容量重新平衡。
  - Spot 節點收到重新平衡建議時，Amazon EKS 會自動嘗試啟動新的替代 Spot 節點。
  - 如果 Spot 兩分鐘中斷通知在取代 Spot 節點處於 `Ready` 狀態前到達，則 Amazon EKS 會開始消耗收到重新平衡建議的 Spot 節點。Amazon EKS 將竭盡全力耗盡節點。因此，無法保證 Amazon EKS 會在耗盡現有節點之前等待替代節點加入叢集。
  - 引導取代 Spot 節點，並且在 Kubernetes 處於 `Ready` 狀態時，Amazon EKS 會包圍隔離並消耗收到重新平衡建議的 Spot 節點。包圍隔離 Spot 節點可確保服務控制器不會將任何新的請求傳送到此 Spot 節點。它也會從狀況良好、作用中的 Spot 節點清單中移除。消耗 Spot 節點可確保正在執行的 Pods 可以正常地移出。
- Amazon EKS 會將下列 Kubernetes 標籤新增至指定容量類型之受管節點群組中的所有節點：`eks.amazonaws.com/capacityType: SPOT`。您可以使用此標籤在 Spot 節點上排程可容錯的應用程式。

選取容量類型的考慮因素

決定是否要部署具有隨需或 Spot 容量的節點群組時，應考慮下列條件：

- Spot 執行個體適用於無狀態、容錯、靈活的應用程式。其中包括批次和機器學習訓練工作負載、大數據 ETL (例如 Apache Spark)、佇列處理應用程式，以及無狀態 API 端點。由於 Spot 是可能隨時間變更的備用 Amazon EC2 容量，因此建議您將 Spot 容量用於可接受中斷的工作負載。更具體地說，Spot 容量適用於容錯期間無法使用所需容量的工作負載。
- 我們建議您將隨需模式用於不容錯的應用程式。這包括叢集管理工具 (例如監控和操作工具)、需要 `StatefulSets` 的部署，以及狀態應用程式 (例如資料庫)。
- 為了在使用 Spot 執行個體時最大化應用程式的可用性，建議您將 Spot 受管節點群組設定為使用多個執行個體類型。建議您在使用多個執行個體類型時，套用下列規則：

- 在受管節點群組中，如果您使用 Cluster Autoscaler，則建議使用具有相同數量 vCPU 和記憶體資源的彈性執行個體類型集。這是為了確保叢集中的節點如預期般擴展。例如，如果您需要四個 vCPU 和八個 GiB 記憶體，請使用 `c3.xlarge`、`c4.xlarge`、`c5.xlarge`、`c5d.xlarge`、`c5a.xlarge`、`c5n.xlarge` 或其他類似的執行個體類型。
- 若要增強應用程式可用性，建議部署多個 Spot 受管節點群組。為此，每個群組應該使用具有相同 vCPU 和記憶體資源的彈性執行個體類型集。例如，如果您需要 4 個 vCPU 和 8 個 GiB 記憶體，建議您使用 `c3.xlarge`、`c4.xlarge`、`c5.xlarge`、`c5d.xlarge`、`c5a.xlarge`、`c5n.xlarge` 或其他類似的執行個體類型建立一個受管節點群組，以及使用 `m3.xlarge`、`m4.xlarge`、`m5.xlarge`、`m5d.xlarge`、`m5a.xlarge`、`m5n.xlarge` 或其他類似的執行個體類型建立另一個受管節點。
- 透過使用自訂啟動範本的 Spot 容量類型部署節點群組時，請使用 API 傳遞多個執行個體類型。請勿透過啟動範本傳遞單一執行個體類型。如需使用啟動範本部署節點群組的詳細資訊，請參閱 使用啟動範本自訂受管節點。

建立受管節點群組

本主題會描述如何啟動已向 Amazon EKS 叢集註冊的節點的 Amazon EKS 受管節點群組。節點加入叢集後，您就可以將 Kubernetes 應用程式部署至其中。

如果這是您第一次啟動 Amazon EKS 受管節點群組，建議您改為按照其中一個 Amazon EKS 入門 指南的步驟進行。本指南提供建立具有節點的 Amazon EKS 叢集的演練。

> **Important**
>
> - Amazon EKS 節點是標準的 Amazon EC2 執行個體。我們會根據正常 Amazon EC2 價格向您收費。如需詳細資訊，請參閱 Amazon EC2 定價。
> - 您無法在啟用 AWS Outposts、 AWS Wavelength或 L AWS ocal Zones 的 AWS 區域 位置建立受管節點。您可以在已啟用 AWS Wavelength L AWS ocal Zones 的 AWS 區域 位置建立自我管理的節點。 AWS Outposts如需詳細資訊，請參閱 啟動自我管理的 Amazon Linux 節點、正在啟動自我管理的 Windows 節點 和 正在啟動自我管理的 Bottlerocket 節點。您也可以在 Outpost 上建立自我管理的 Amazon Linux 節點群組。如需詳細資訊，請參閱 啟動 Outpost 上自我管理的 Amazon Linux 節點。
> - 如果您沒有為 Amazon EKS 最佳化 Linux 或 Bottlerocket 隨附的 `bootstrap.sh` 檔案指定 AMI ID，則受管節點群組會對 `maxPods` 的值強制執行數量上限。對於少於 30 個 vCPU

的執行個體，數量上限為 110。對於具有 30 個以上 vCPU 的執行個體，數量上限會跳至 250。這些是根據 Kubernetes 可擴展性閾值和內部 Amazon EKS 可擴展性團隊測試所建議設定得到的數量。如需詳細資訊，請參閱 Amazon VPC CNI 外掛程式增加每節點的 Pod 限制部落格文章。

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 建立 Amazon EKS 叢集。
- 供節點使用的現有 IAM 角色。若要建立服務角色，請參閱Amazon EKS 節點 IAM 角色。如果此角色沒有任何 VPC CNI 的政策，則 VPC CNI Pod 需要以下個別角色。
- (選用，但建議) Amazon VPC CNI plugin for Kubernetes 附加元件設定有自己的 IAM 角色，該角色已連接必要的 IAM 政策。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。
- 熟悉 選擇 Amazon EC2 執行個體類型 中列出的考量。取決於您選擇的執行個體類型，您的叢集和 VPC 可能還有其他先決條件。
- 若要新增 Windows 受管節點群組，您必須先啟用叢集的 Windows 支援。如需詳細資訊，請參閱 為您的 Amazon EKS 叢集啟用 Windows 支援。

您可以使用 `eksctl` 或 AWS Management Console建立受管節點群組。

eksctl

使用 **`eksctl`** 建立受管節點群組

此程序需要 `eksctl` 版本 `0.183.0` 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的 Installation 一節。

1. (選用) 如果 AmazonEKS_CNI_Policy 受管 IAM 政策已連接至您的 Amazon EKS 節點 IAM 角色，建議您改為將其指派給您與 Kubernetes `aws-node` 服務帳戶相關聯的 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

2. 使用或不使用自訂啟動範本來建立受管節點群組。手動指定啟動範本可讓您更靈活地自訂節點群組。例如，此方式可以允許部署自訂 AMI，或對 Amazon EKS 最佳化 AMI 中的 boostrap.sh 指令碼提供引數。如需每個可用選項和預設值的完整清單，請輸入以下命令。

```
eksctl create nodegroup --help
```

在以下命令中，將 *my-cluster* 取代為您的叢集名稱，並將 *my-mng* 取代為您的節點群組名稱。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。

> **Important**
>
> 如果在第一次建立受管的節點群組時未使用自訂啟動範本，請勿在之後針對節點群組使用自訂啟動範本。如果沒有指定自訂啟動範本，系統會自動產生啟動範本 (不建議手動修改該範本)。手動修改此自動產生的啟動範本可能會導致錯誤。

沒有啟動範本

eksctl 會在您的帳戶中建立預設 Amazon EC2 啟動範本，並使用根據您指定之選項建立的啟動範本來部署節點群組。指定 --node-type 的值之前，請參閱 選擇 Amazon EC2 執行個體類型。

使用允許的關鍵字取代 *ami-family*。如需詳細資訊，請參閱 eksctl 文件中的 Setting the node AMI Family (設定節點 AMI 系列)。使用 Amazon EC2 金鑰對或公有金鑰的名稱取代 *my-key*。此金鑰會在節點啟動後用於將 SSH 套用至節點。

> **Note**
>
> 對於 Windows，此命令不會啟用 SSH。而是將 Amazon EC2 金鑰對與執行個體建立關聯，讓您可以 RDP 至該執行個體中。

如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳Linux細資訊，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 金鑰配對和Linux執行個體。如需詳Windows細資訊，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 金鑰配對和Windows執行個體。

如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：

- 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
- 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

若要封鎖 Pod 對 IMDS 的存取，請將 **--disable-pod-imds** 選項新增至下列命令。

```
eksctl create nodegroup \
  --cluster my-cluster \
  --region region-code \
  --name my-mng \
  --node-ami-family ami-family \
  --node-type m5.large \
  --nodes 3 \
  --nodes-min 2 \
  --nodes-max 4 \
  --ssh-access \
  --ssh-public-key my-key
```

您的執行個體可以選擇性地為 Pods 指派更多的 IP 地址、將 IP 地址指派給不同於執行個體之 CIDR 區塊的 Pods，以及在沒有網際網路存取的情況下部署至叢集。如需詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量、Pod 的自訂聯網 和 私有叢集要求 以取得要新增至之前命令的其他選項。

受管節點群組會根據執行個體類型，計算並套用可在節點群組之每個節點上執行的 Pods 數量上限單一數值。若您建立具有不同執行個體類型的節點群組，則在所有執行個體類型中計算的最小值，會套用為可在節點群組中每種執行個體類型上執行的 Pods 數量上限。受管節點群組會使用 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限 中參考的指令碼來計算數值。

搭配啟動範本

啟動範本必須已經存在，且必須符合 啟動範本組態基礎知識 中指定的要求。

如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：

- 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
- 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

若您要封鎖 Pod 對 IMDS 的存取，請在啟動範本中指定必要的設定。

a. 將以下內容複製到您的裝置。取代 *example values*，然後執行修改後的命令來建立 `eks-nodegroup.yaml` 檔案。在不使用啟動範本的情況下部署時指定的數個設定會移至啟動範本。如果您未指定 `version`，即使用範本的預設版本。

```
cat >eks-nodegroup.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
managedNodeGroups:
- name: my-mng
  launchTemplate:
    id: lt-id
    version: "1"
EOF
```

如需 `eksctl` 組態檔案設定的完整清單，請參閱 `eksctl` 文件中的組態檔案結構描述。您的執行個體可以選擇性地為 Pods 指派更多的 IP 地址、將 IP 地址指派給不同於執行個體之 CIDR 區塊的 Pods、使用 `containerd` 執行階段，以及在沒有傳出網際網路存取的情況下部署至叢集。如需詳細資訊，請參閱適用於新增至組態檔案之其他選項的增加 Amazon EC2 節點的可用 IP 地址數量、Pod 的自訂聯網、測試從移轉Docker到 `containerd` 和 私有叢集要求。

若您未在啟動範本中指定 AMI ID，受管節點群組會根據執行個體類型，計算並套用可在節點群組之每個節點上執行的 Pods 數量上限單一數值。若您建立具有不同執行個體類型的節點群組，則在所有執行個體類型中計算的最小值，會套用為可在節點群組中每種執行個體類型上執行的 Pods 數量上限。受管節點群組會使用 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限 中參考的指令碼來計算數值。

若您已在啟動範本中指定 AMI ID，且正使用自訂聯網或想增加指派至執行個體的 IP 地址數量，請指定可在節點群組之每個節點上執行的 Pods 數量上限。如需詳細資訊，請參閱 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限。

b. 使用下列命令部署節點群組。

```
eksctl create nodegroup --config-file eks-nodegroup.yaml
```

AWS Management Console

若要使用建立受管理節點群組 AWS Management Console

1. 等待您的叢集狀態顯示為 ACTIVE。您無法針對尚未進入 ACTIVE 狀態的叢集建立受管節點群組。
2. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
3. 選擇要在其中建立受管節點群組的叢集名稱。
4. 選取 Compute (運算) 標籤。
5. 選擇 Add node group (新增節點群組)。
6. 在 Configure node group (配置節點群組) 頁面上，相應地填寫參數，然後選擇 Next (下一步)。

- Name (名稱) – 輸入受管節點群組的唯一名稱。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。
- Node IAM role (節點 IAM 角色)：選擇要與節點群組搭配使用的節點執行個體角色。如需詳細資訊，請參閱 Amazon EKS 節點 IAM 角色。

> **Important**
>
> - 您無法使用用於建立任何叢集的相同角色。
> - 建議使用尚未被任何自我管理節點群組使用的角色。否則，您要計劃與新的自我管理節點組一起使用。如需詳細資訊，請參閱 刪除受管節點群組。

- Use launch template (使用啟動範本)：(選用) 選擇是否要使用現有的啟動範本。選取啟動範本名稱。然後，選取 Launch template version (啟動範本版本)。如果您沒有選取版本，則 Amazon EKS 會使用範本的預設版本。啟動範本允許對節點群組進行更多自訂，譬如允許您部署自訂 AMI、為 Pods 指派更多的 IP 地址、將 IP 地址指派給不同於執行個體之 Pods 區

塊的 Pod、啟用執行個體的 containerd 執行階段，以及將節點部署至沒有傳出網際網路存取的叢集。如需詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量、Pod 的自訂聯網、測試從移轉Docker到 containerd 和 私有叢集要求。

啟動範本必須符合 使用啟動範本自訂受管節點 要求。如果您不使用自己的啟動範本，Amazon EKS API 會在您的帳戶中建立預設的 Amazon EC2 啟動範本，並使用預設啟動範本部署節點群組。

若您實作服務帳戶的 IAM 角色，請直接將必要的許可指派至需要存取 AWS 服務的每個 Pod，且叢集中沒有 Pods 基於其他原因 (例如擷取目前的 AWS 區域) 需要存取 IMDS，而您亦可針對未在啟動範本中使用主機聯網的 Pods 停用對 IMDS 的存取。如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

- Kubernetes 標籤 – (選用) 您可以選擇將 Kubernetes 標籤套用至受管節點群組中的節點。
- Kubernetes 污點 – (選用) 您可以選擇將 Kubernetes 污點套用至受管節點群組中的節點。Effect (效果) 功能表中的可用選項是 **NoSchedule**、**NoExecute** 及 **PreferNoSchedule**。如需詳細資訊，請參閱 受管節點群組上的節點污點。
- Tags (標籤) – (選用) 您可以選擇標記 Amazon EKS 受管節點群組。這些標籤不會傳播到節點群組中的其他資源，例如 Auto Scaling 群組或執行個體。如需詳細資訊，請參閱 為您的 Amazon EKS 資源加上標籤。

7. 在 Set compute and scaling configuration (設定運算和擴展組態) 頁面上，相應地填寫參數，然後選擇 Next (下一步)。

   - AMI type (AMI 類型)：選取 AMI 類型。如果您在部署 Arm 執行個體，請務必在部署前檢閱 Amazon EKS 最佳化的 Arm Amazon Linux AMI 中的考量事項。

     如果您在上一頁中已指定啟動範本，並在啟動範本中指定了 AMI，則無法選取值。系統會顯示範本中的值。範本中指定的 AMI 必須符合 指定 AMI 中的要求。
   - Capacity type (容量類型) – 選取容量類型。如需選擇容量類型的詳細資訊，請參閱 受管節點群組容量類型。您無法在同一節點群組中混合使用不同的容量類型。如果要同時使用這兩種容量類型，請建立個別的節點群組，每個群組都有自己的容量和執行個體類型。
   - Instance types (執行個體類型)：依預設會指定一或多個執行個體類型。若要移除預設執行個體類型，請選取執行個體類型的右側的 X。選擇要在受管節點群組中使用的執行個體類型。如需詳細資訊，請參閱 選擇 Amazon EC2 執行個體類型。

     主控台會顯示一組常用的執行個體類型。如果需要使用未顯示的執行個體類型建立受管節點群組，則請使用 eksctl、 AWS CLI、 AWS CloudFormation或開發套件以建立節點群組。如果您在上一頁指定了啟動範本，則無法選取值，因為必須在啟動範本中指定執行個體類

型。系統會顯示啟動範本中的值。如果為 Capacity type (容量類型) 選取了 Spot，則建議您指定多個執行個體類型以增強可用性。

- Disk size (磁碟大小) – 輸入要用於節點根磁碟區的磁碟大小 (以 GiB 為單位)。

  如果您在上一頁指定了啟動範本，則無法選取值，因為必須在啟動範本中指定該值。

- Desired size (所需大小) – 指定受管節點群組目前在啟動時應該維護的節點數量。

> **Note**
>
> Amazon EKS 不會自動擴展或縮減您的節點群組。不過，您可以設定 Kubernetes Cluster Autoscaler 為您執行這項操作。

- Minimum size (大小下限) – 指定受管節點群組可縮減至的節點數量下限。
- Maximum size (大小上限)：指定受管節點群組可擴增至的節點數量上限。
- Node group update configuration (節點群組更新組態) – (選用) 您可以選取要平行更新的節點數量或百分比。這些節點在更新期間將無法使用。對於 Maximum unavailable (無法使用的上限)，選取下列其中一個選項，然後指定 Value (值)：
  - Number (數量)：選取並指定可平行更新節點群組的節點數量。
  - Percentage (百分比)：選取並指定可平行更新節點群組的節點百分比。如果您的節點群組中有大量節點，這會很有用。

8. 在 Specify networking (指定聯網) 頁面上，據此填寫參數，然後選擇 Next (下一步)。

- Subnets (子網路)：選擇要將受管節點啟動至其中的子網路。

> **Important**
>
> 如果您跨越多個由 Amazon EBS 磁碟區提供的可用區域執行狀態應用程式，並且使用 Kubernetes 自動擴展，您應該設定多個節點群組，每個群組的範圍皆為單一可用區域。另外，您應該啟用 `--balance-similar-node-groups` 功能。

> **Important**
>
> - 如果選擇公有子網路，而且您的叢集只啟用了公有 API 伺服器端點，則子網路的必須將 `MapPublicIPOnLaunch` 設定為 `true`，讓執行個體成功加入叢集。如果公有子網路是使用 `eksctl` 或 2020 年 3 月 26 日或之後 Amazon EKS 發佈的 AWS CloudFormation 範本，則此設定已設定為 `true`。如果子網路是使

用eksctl或在 2020 年 3 月 26 日之前建立的 AWS CloudFormation 範本，則您需要手動變更設定。如需詳細資訊，請參閱修改子網路的公有 IPv4 定址屬性。

- 如果使用啟動範本並指定多個網路介面，Amazon EC2 便不會自動指派公有 IPv4 地址，即使 MapPublicIpOnLaunch 已設定為 true。對於在此案例中加入叢集的節點，您必須啟用叢集的私有 API 伺服器端點，或使用透過其他方法 (例如 NAT Gateway) 提供的出站網際網存取，啟動私有子網路中的節點。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 執行個體 IP 定址。

- 設定 SSH 存取節點 (選用)。啟用 SSH 可讓您連接到執行個體，並在發生問題時收集診斷資訊。強烈建議您在建立節點群組時啟用遠端存取。您無法在建立節點群組之後啟用遠端存取。

  如果您選擇使用啟動範本，則不會顯示此選項。若要啟用節點的遠端存取，請在啟動範本中指定金鑰對，並確定您在啟動範本中指定之安全群組中的節點已開啟適當的連接埠。如需詳細資訊，請參閱 使用自訂安全群組。

  > **Note**
  >
  > 對於 Windows，此命令不會啟用 SSH。而是將 Amazon EC2 金鑰對與執行個體建立關聯，讓您可以 RDP 至該執行個體中。

- 對於 SSH key pair (SSH 金鑰對) (選用)，選擇要使用的 Amazon EC2 SSH 金鑰。如需詳Linux細資訊，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 金鑰配對和Linux執行個體。如需詳Windows細資訊，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 金鑰配對和Windows執行個體。如果您選擇使用啟動範本，則無法選取範本。當使用 Bottlerocket AMI 為節點群組提供 Amazon EC2 SSH 金鑰時，也會啟用管理容器。如需詳細資訊，請參閱 GitHub 上的管理容器。
- 對於 Allow SSH remote access from (允許 SSH 遠端存取來自)，如果要限制對特定執行個體的存取，請選取與這些執行個體相關聯的安全群組。如果沒有選取特定的安全群組，則允許從網際網路 (0.0.0.0/0) 上的任何地方存取 SSH。

9. 在 Review and create (檢閱並建立) 頁面上，檢閱您的受管節點群組組態，然後選擇 Create (建立)。

   如果節點無法加入叢集，請參閱故障診斷指南中的 節點無法加入叢集。

10. 查看節點的狀態，並等待他們到達 Ready 狀態。

```
kubectl get nodes --watch
```

11. (僅限 GPU 節點) 若選擇了 GPU 執行個體類型以及 Amazon EKS 最佳化之加速 AMI，則必須套用 Kubernetes 專用 NVIDIA 裝置外掛程式作為叢集上的 DaemonSet。請先以您想要的 NVIDIA/k8s-device-plugin 版本來取代 *vX.X.X*，再執行下列命令。

    ```
    kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
    ```

現在您已具備含節點之可正常運作的 Amazon EKS 叢集，您即可開始安裝 Kubernetes 附加元件並將應用程式部署到叢集。以下文件主題可協助您擴展叢集的功能。

- 建立叢集的 IAM 主體是唯一可以使用 `kubectl` 或 AWS Management Console對 Kubernetes API 伺服器進行呼叫的主體。如果要其他 IAM 主體能夠存取叢集，則需新增這些主體。如需詳細資訊，請參閱 授予 Kubernetes API 的存取權 及 所需的許可。
- 如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：
  - 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
  - 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

  如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。
- 自動擴展：設定 Kubernetes Cluster Autoscaler，以自動調整節點群組中的節點數量。
- 將範例應用程式部署至叢集。
- 叢集管理：了解如何使用重要工具來管理叢集。

更新受管節點群組

啟動受管節點群組更新後，Amazon EKS 會自動為您更新節點，完成 受管節點更新行為 中列出的步驟。如果使用的是 Amazon EKS 最佳化 AMI，則 Amazon EKS 會自動將最新的安全修補程式和作業系統更新套用至您的節點，作為最新 AMI 發行版本的一部分。

有數個案例可用於更新 Amazon EKS 受管節點群組的版本或組態：

- 您已更新 Amazon EKS 叢集的 Kubernetes 版本，並且想要更新您的節點，以使用相同的 Kubernetes 版本。
- 新的 AMI 發行版本可供受管節點群組使用。如需 AMI 版本的詳細資訊，請參閱下列各節：
  - Amazon EKS 最佳化 Amazon Linux AMI 版本

  - Amazon EKS 最佳化 Bottlerocket AMI
  - Amazon EKS 最佳化 Windows AMI 版本
- 您想要調整受管節點群組中執行個體的下限、上限或所需計數。
- 您想要從受管節點群組中的執行個體新增或移除 Kubernetes 標籤。
- 您想要在受管節點群組中新增或移除 AWS 標籤。
- 您需要部署具有組態變更的新版啟動範本，例如更新的自訂 AMI。
- 您已部署 Amazon VPC CNI 附加元件的版本`1.9.0`或更新版本、為前綴委派啟用附加元件，並希望節點群組中的新 AWS Nitro System執行個體支援大幅增加的數量。Pods如需詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。
- 您已為 Windows 節點啟用 IP 前置詞委派，並希望節點群組中的新 AWS Nitro System 執行個體支援大幅增加的Pods數目。如需詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。

如果有較新的受管節點群組之 Kubernetes 版本的 AMI 發行版本，則可以更新節點群組的版本，以使用較新的 AMI 版本。同樣地，如果叢集執行的 Kubernetes 版本比節點群組還新，則可以更新節點群組，以使用符合叢集 Kubernetes 版本的最新 AMI 發行版本。

當受管節點群組中的節點由於擴展操作或更新而終止時，該節點中的 Pods 會先被耗盡。如需詳細資訊，請參閱 受管節點更新行為。

更新節點群組版本

您可以使用 `eksctl` 或 AWS Management Console更新節點群組版本。您更新的版本不能比控制平面的版本還新。

eksctl

使用 **`eksctl`** 更新節點群組版本

- 使用下列命令，將受管節點群組更新至目前節點上所部署相同 Kubernetes 版本的最新 AMI 版本。使用您自己的值取代每一個 *`example value`*。

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code
```

> **Note**
>
> 如果要將使用啟動範本部署的節點群組升級至新的啟動範本版本，則請將 `--launch-template-version` *`version-number`* 新增至上述命令。啟動範本必須符合 使用啟動範本自訂受管節點 中描述的要求。如果啟動範本包含自訂 AMI，則 AMI 必須符合 指定 AMI 中的要求。當您將節點群組升級至較新版本的啟動範本時，每個節點都會回收，以符合指定之啟動範本版本的新組態。
> 您無法直接將未部署啟動範本的節點群組升級至新的啟動範本版本。相反地，您必須使用啟動範本部署新的節點群組，才能將節點群組更新為新的啟動範本版本。

您可以將節點群組升級至與控制平面的 Kubernetes 版本相同的版本。例如，當您有一個正在執行 Kubernetes `1.29` 的叢集時，您可以使用下列命令，將目前執行 Kubernetes `1.28` 的節點升級到版本 `1.29`。

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code \
  --kubernetes-version=1.29
```

AWS Management Console

使用更新節點群組版本 AWS Management Console

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 選擇包含要更新之節點群組的叢集。
3. 如果至少有一個節點群組有可用的更新，則頁面頂部會出現一個方塊，通知您可用的更新。如果選取 Compute (運算) 索引標籤，您會看到針對有可用更新之節點群組的 Node groups (節點群組) 資料表中 AMI release version (AMI 發行版本) 欄位中的 Update now (立即更新)。若要更新節點群組，請選擇 Update now (立即更新)。

   您不會看到使用自訂 AMI 部署之節點群組的通知。如果節點是使用自訂 AMI 進行部署，則請完成以下步驟來部署新的已更新自訂 AMI。

   a. 建立 AMI 的新版本。

   b. 使用新的 AMI ID 建立新的啟動範本版本。

c. 將節點升級至啟動範本的新版本。

4. 在 Update node group version (更新節點群組版本) 對話方塊中，啟用或停用以下選項：

   - Update node group version (更新節點群組版本)：如果您部署了自訂 AMI，或經 Amazon EKS 優化的 AMI 目前是叢集的最新版本，則無法使用此選項。
   - Change launch template version (變更啟動範本版本)：如果節點群組是在沒有自訂啟動範本的情況下進行部署，則無法使用此選項。您只能更新已使用自訂啟動範本部署之節點群組的啟動範本版本。選取節點群組要更新至的 Launch template version (啟動範本版本)。如果節點群組設定了自訂 AMI，則您選取的版本也必須指定 AMI。當您升級至較新版本的啟動範本時，每個節點都會回收，以符合指定之啟動範本版本的新組態。

5. 對於 Update strategy (更新策略)，選取下列其中一個選項：

   - Rolling update (滾動更新) – 此選項會遵循叢集的 Pod 中斷預算。若發生 Pod 中斷預算問題，導致 Amazon EKS 無法正常地耗盡正在此節點群組上執行的 Pods，則更新會失敗。
   - Force update (強制更新) – 此選項不會遵循 Pod 中斷預算。無論 Pod 中斷預算問題是否出現，都會強制節點重新啟動以進行更新。

6. 選擇 Update (更新)。

編輯節點群組組態

您可以修改受管節點群組中的部分組態。

編輯節點群組組態

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 選擇包含要編輯之節點群組的叢集。
3. 選取 Compute (運算) 標籤。
4. 選取要編輯的節點群組，然後選擇 Edit (編輯)。
5. (選用) 在 Edit node group (編輯節點群組) 頁面上，執行以下作業：

   a. 編輯 Node group scaling configuration (節點群組擴展組態)。

      - Desired size (所需大小) – 指定受管節點群組目前應該維護的工作者節點數量。
      - Minimum size (大小下限) – 指定受管節點群組可縮減至的節點數量下限。
      - Maximum size (大小上限)：指定受管節點群組可擴增至的節點數量上限。如需節點群組中支援的節點數量上限，請參閱 Amazon EKS 服務配額。

b. (選用) 將 Kubernetes 標籤新增至節點群組中的節點或從中移除這些標籤。這裡顯示的標籤只是您使用 Amazon EKS 所套用的標籤。其他標籤可能存在於這裡未顯示的節點上。

c. (選用) 將 Kubernetes 污點新增至節點群組中的節點或從中移除這些污點。已新增的污點可能會有 **`NoSchedule`**、**`NoExecute`** 或 **`PreferNoSchedule`** 效果。如需詳細資訊，請參閱 受管節點群組上的節點污點。

d. (選用) 將 Tags (標籤) 新增至節點群組或從中移除標籤。這些標籤僅適用於 Amazon EKS 節點群組。標籤不會傳播到其他資源，例如節點群組中的子網路或 Amazon EC2 執行個體。

e. (選用) 編輯 Node Group update configuration (節點群組更新組態)。選取任何一個 Number (數字) 或 Percentage (百分比)。

- Number (數量)：選取並指定可平行更新節點群組的節點數量。這些節點在更新期間將無法使用。
- Percentage (百分比)：選取並指定可平行更新節點群組的節點百分比。這些節點在更新期間將無法使用。如果您的節點群組中有許多節點，這會很有用。

f. 完成編輯後，請選擇 Save changes (儲存變更)。

受管節點更新行為

Amazon EKS 受管工作節點升級策略有四個不同的階段，如下節所述。

設定階段

設定階段具有下列步驟：

1. 它為與節點群組相關聯的 Auto Scaling 群組建立新的 Amazon EC2 啟動範本版本。新的啟動範本版本使用目標 AMI 或自訂啟動範本版本進行更新。
2. 它會更新 Auto Scaling 群組，以使用最新的啟動範本。
3. 它使用節點群組的 `updateConfig` 屬性決定要平行升級的節點數量上限。不可用節點上限配額為 100。預設值為一個節點。如需詳細資訊，請參閱《Amazon EKS API 參考》中的 `updateConfig` 屬性。

擴充規模階段

升級受管節點群組中的節點時，已升級的節點會在與正在升級的節點相同的可用區域中啟動。為了保證這種配置，我們使用 Amazon EC2 的可用區域重新平衡。如需詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的可用區域容量重新平衡。為了滿足這項需求，我們可在受管節點群組中每個可用區域啟動最多兩個執行個體。

擴充規模階段具有下列步驟：

1. 它按下面的任一大小 (以較大者為准) 增加 Auto Scaling 群組的最大大小和所需大小：
   - Auto Scaling 群組部署的可用區域數量的兩倍。
   - 升級無法使用的上限。

     例如，如果節點群組有五個可用區域，而 maxUnavailable 為一個，則升級程序最多能啟動 10 個節點。但maxUnavailable是，當 20（或高於 10 的任何內容）時，該過程將啟動 20 個新節點。
2. 擴展 Auto Scaling 群組後，會檢查使用最新配置的節點是否存在於節點群組中。只有在符合下列條件時，此步驟才會成功：
   - 在節點所在的每個可用區域中，至少會啟動一個新節點。
   - 每個新節點都應該處於 Ready 狀態。
   - 新節點應該有 Amazon EKS 套用標籤。

     以下是一般節點群組中工作節點上的 Amazon EKS 套用標籤：
     - eks.amazonaws.com/nodegroup-image=*$amiName*
     - eks.amazonaws.com/nodegroup=*$nodeGroupName*

     以下是自訂啟動範本或 AMI 節點群組中工作節點上的 Amazon EKS 套用標籤：
     - eks.amazonaws.com/nodegroup-image=*$amiName*
     - eks.amazonaws.com/nodegroup=*$nodeGroupName*
     - eks.amazonaws.com/sourceLaunchTemplateId=*$launchTemplateId*
     - eks.amazonaws.com/sourceLaunchTemplateVersion=*$launchTemplateVersion*
3. 它將節點標記為不可排程，以避免排程新的 Pods。它還使用 node.kubernetes.io/exclude-from-external-load-balancers=true 標記節點，以便在終止節點之前從負載平衡器中移除節點。

以下是在這個階段中會導致 NodeCreationFailure 錯誤的已知原因：

可用區域容量不足

可用區域可能沒有所請求執行個體類型的容量。建議在建立受管節點群組時設定多個執行個體類型。

您帳戶中的 EC2 執行個體限制

您可能需要增加帳戶可以使用 Service Quotas 同時執行的 Amazon EC2 執行個體的數量。如需詳細資訊，請參閱《Amazon Elastic Compute Cloud Linux 執行個體使用者指南》中的 EC2 Service Quotas。

自訂使用者資料

自訂使用者資料有時會中斷引導程序。這種情況可能會導致 `kubelet` 未在節點上啟動或節點上未取得預期的 Amazon EKS 標籤。如需詳細資訊，請參閱 指定 AMI。

任何使節點運作狀態不良或未準備就緒的變更

節點磁碟壓力、記憶體壓力和類似情況，可能導致節點無法進入 `Ready` 狀態。

升級階段

升級階段具有下列步驟：

1. 它隨機選取需要升級的節點，最大不超過為節點群組設定的無法使用上限。
2. 它耗盡節點中的 Pods。若 Pods 未在 15 分鐘內離開節點且未強制標記，則升級階段將失敗，並出現 `PodEvictionFailure` 錯誤。在這種情況下，您可以應用強制標記，同時使用 `update-nodegroup-version` 請求刪除 Pods。
3. 在移出每個 Pod 之後，它包圍隔離節點並等待 60 秒。這樣做是為了讓服務控制器不會將任何新的請求傳送到此節點，並將此節點從其作用中的節點清單中移除。
4. 它將終止請求傳送之包圍隔離節點的 Auto Scaling 群組。
5. 它重複步驟先前的升級步驟，直到節點群組中沒有使用舊版啟動範本部署的節點為止。

以下是在這個階段中會導致 `PodEvictionFailure` 錯誤的已知原因：

積極的 PDB

在 Pod 上定義積極的 PDB，或有多個 PDB 指向同一個 Pod。

容忍所有污點的部署

一旦移出每個 Pod，節點預期為空，因為節點在先前的步驟中會受到污染。但是，如果部署容忍每一個污點，則節點更有可能是非空白，導致 Pod 移出失敗。

縮減規模階段

縮減規模階段會將 Auto Scaling 群組的大小上限和所需大小遞減一，以便回到更新開始之前的值。

如果升級工作流程判斷 Cluster Autoscaler 在工作流程的縮減規模階段期間擴充節點群組，它會立即結束，而不會將節點群組恢復至原始大小。

受管節點群組上的節點污點

Amazon EKS 支援透過受管節點群組設定 Kubernetes 污點。污點和容差共同運作，以確保 Pods 不會排程至不適當的節點上。一個或多個污點可以套用至一個節點。這表示節點不應該接受無法容許污點的任何 Pods。容差會套用至 Pods，並允許 (但不需要) Pods 排程到具有相符污點的節點上。如需詳細資訊，請參閱 Kubernetes 文件中的污點和容差。

Kubernetes 節點污點可使用 AWS Management Console 或透過 Amazon EKS API 套用至新的和現有的受管理節點群組。

- 如需使用 AWS Management Console 建立具有污點之節點群組的資訊，請參閱 建立受管節點群組。
- 以下是使用 AWS CLI 建立具有污點之節點群組的範例：

```
aws eks create-nodegroup \
 --cli-input-json '
{
  "clusterName": "my-cluster",
  "nodegroupName": "node-taints-example",
  "subnets": [
     "subnet-1234567890abcdef0",
     "subnet-abcdef01234567890",
     "subnet-021345abcdef67890"
   ],
  "nodeRole": "arn:aws:iam::111122223333:role/AmazonEKSNodeRole",
  "taints": [
     {
         "key": "dedicated",
         "value": "gpuGroup",
         "effect": "NO_SCHEDULE"
     }
   ]
}'
```

如需詳細資訊和用量範例，請參閱 Kubernetes 參考文件中的污點。

> **Note**
>
> - 污點可以在使用 UpdateNodegroupConfig API 建立節點群組後更新。
> - 污點金鑰必須以字母或數字開頭。可包含字母、數字、連字號 (-)、句點 (.) 及底線 (_)。長度上限為 63 個字元。
> - 或者，污點金鑰可以使用 DNS 子網域字首和單一 / 開頭。如果其以 DNS 子網域字首開頭，則長度可為 253 個字元。
> - 值是選用的，必須以字母或數字開頭。可包含字母、數字、連字號 (-)、句點 (.) 及底線 (_)。長度上限為 63 個字元。
> - 當直接使用 Kubernetes 或 AWS Management Console 時，污點效果必須是 **NoSchedule**、**PreferNoSchedule** 或 **NoExecute**。不過，使用 AWS CLI 或 API 時，污點效果必須是 **NO_SCHEDULE**、**PREFER_NO_SCHEDULE** 或 **NO_EXECUTE**。
> - 每個節點群組最多允許 50 個污點。
> - 如果從節點手動移除使用受管節點群組建立的污點，則 Amazon EKS 不會將污點重新加入節點。即使在受管節點群組組態中指定了污點，也是如此。

您可以使用 `aws eks update-nodegroup-config` AWS CLI 命令來新增、移除或取代受管節點群組的污點。

使用啟動範本自訂受管節點

若要取得最高層級的自訂，您可以使用自己的啟動範本來部署受管節點。使用啟動範本可提供下列功能：

- 在部署節點時提供啟動程序引數，例如額外的 `kubelet` 引數。
- 將 IP 地址指派給來自不同於將 IP 地址指派給節點之 CIDR 區塊的 Pods。
- 將自己的自訂 AMI 部署至節點。
- 將您自己的自訂 CNI 部署至節點。

如果您在第一次建立受管節點群組期間提供自己的啟動範本，稍後也會得到更出色的靈活性。只要使用自己的啟動範本部署受管節點群組，您就可以使用不同版本的相同啟動範本來反覆更新。將節點群組更新為不同版本的啟動範本時，群組中的所有節點都會回收，以符合指定啟動範本版本的新組態。

一律使用 Amazon EC2 Auto Scaling 群組所使用的啟動範本部署受管節點群組。如果未提供啟動範本，Amazon EKS API 會使用帳戶中的預設值自動建立此啟動範本。然而，我們不建議您修改自動產生的啟動範本。此外，未使用自訂啟動範本的現有節點群組無法直接更新。相反，您必須建立具有自訂啟動範本的新節點群組，才能執行此動作。

啟動範本組態基礎知識

您可以使用 AWS Management Console、 AWS CLI或開發 AWS 套件建立 Amazon EC2 自動擴展啟動範本。如需詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的建立 Auto Scaling 群組的啟動範本。啟動範本中的某些設定與用於受管理節點組態的設定類似。使用啟動範本部署或更新節點群組時，必須在節點群組組態或啟動範本其中一個位置中指定某些設定。請勿同時在兩個位置指定同一設定。如果設定存在於不應該的地方，則建立或更新節點群組之類的動作會失敗。

下表會列出啟動範本中禁止的設定。該表也會列出受管節點群組組態中需要的類似設定 (如果有)。列出的設定是顯示在主控台中的設定。它們在 AWS CLI 和 SDK 中可能具有相似但不同的名稱。

啟動範本：禁止	Amazon EKS 節點群組組態
Network interfaces (網路介面) (Add network interface (新增網路介面)) 下的 Subnet (子網)	Specify networking (指定聯網) 頁面上 Node Group network configuration (節點群組網路組態) 下的 Subnets (子網)
Advanced details (進階詳細資訊) 下的 IAM instance profile (IAM 執行個體描述檔)	Configure Node Group (設定節點群組) 頁面上 Node Group configuration (節點群組組態) 下的 Node IAM Role (節點 IAM 角色)
Advanced details (進階詳細資訊) 下的 Shutdown behavior (關機行為) 和 Stop - Hibernate behavior (停用 – 休眠行為)。為兩種設定在啟動範本中保留預設 Don't include in launch template setting (請勿包含在啟動範本設定中)。	無同等。Amazon EKS 必須控制執行個體生命週期，而非 Auto Scaling 群組。

下表會列出受管節點群組組態中禁止的設定。該表也會列出類似的設定 (如果有)，這些設定在啟動範本中是必要設定。列出的設定是顯示在主控台中的設定。它們可能在 AWS CLI 和 SDK 中具有類似的名稱。

Amazon EKS 節點群組組態設定：禁止	啟動範本

(僅當您在啟動模板中指定了自訂 AMI 時) 在 Set compute and scaling configuration (設定運算和擴展組態) 頁面上 Node Group compute configuration (節點群組運算組態) 下的 AMI type (AMI 類型) – 主控台顯示 Specified in launch template (在啟動範本中指定) 以及指定的 AMI ID。
 	
如果未在啟動範本中指定 Application and OS Images (Amazon Machine Image) (應用程式和作業系統映像 (Amazon Machine Image))，您可在節點群組組態中選取 AMI。
	Launch template contents (啟動範本內容) 下的 Application and OS Images (Amazon Machine Image) (應用程式和作業系統映像 (Amazon Machine Image))：如果您有下列任一要求，則必須指定 ID：
	
	• 使用自訂 AMI。如果您指定的 AMI 不符合 指定 AMI 中所列要求，則節點群組部署將會失敗。
	• 想要提供使用者資料將引數提供給 `bootstrap.sh` 檔案，其中包含在 Amazon EKS 最佳化 AMI。您可以讓執行個體指派大量的 IP 位址給Pods、Pods從不同的 CIDR 區塊指派 IP 位址給執行個體，或是部署沒有輸出網際網路存取權的私人叢集。如需詳細資訊，請參閱下列主題：
	  • 增加 Amazon EC2 節點的可用 IP 地址數量
	  • Pod 的自訂聯網
	  • 私有叢集要求
	  • 指定 AMI

Set compute and scaling configuration (設定運算和擴展組態) 頁面上 Node Group compute configuration (節點群組運算組態) 下的 Disk size (磁碟大小)：主控台顯示 Specified in launch template (在啟動範本中指定)。	Storage (Volumes) (儲存 (磁碟區)) (Add new volume (新增磁碟區)) 下的 Size (大小)。您必須在啟動範本中指定此選項。
Specify Networking (指定聯網) 頁面上 Node Group configuration (節點群組組態) 下的 SSH key pair (SSH 金鑰對) – 主控台會顯示在啟動範本中指定的金鑰，或顯示 Not specified in launch template (未在啟動範本中指定)。	Key pair (login) (金鑰對 (登入)) 下的 Key pair name (金鑰對名稱)。

Amazon EKS 節點群組組態設定：禁止	啟動範本
使用啟動範本時，您無法指定允許遠端存取的來源安全群組。	針對執行個體的 Network settings (網路設定) 下的 Security groups (安全群組) 或 Network interfaces (網路介面) 下的 Security groups (安全群組) (Add network interface (新增網路介面))，但不能兩者同時選擇。如需詳細資訊，請參閱使用自訂安全群組。

Note

- 如果使用啟動範本部署節點群組，請在啟動範本的 Launch template contents (啟動範本內容) 中指定零或一個 Instance type (執行個體類型)。您也可以在主控台上的 Set compute and scaling configuration (設定運算和擴展組態) 頁面中，為 Instance types (執行個體類型) 指定 0 到 20 個執行個體類型。您還可以使用其他使用 Amazon EKS API 的工具來執行這項操作。如果在啟動範本中指定執行個體類型，並使用該啟動範本來部署節點群組，則無法在主控台中指定任何執行個體類型，或使用其他使用 Amazon EKS API 的工具。如果未在啟動範本、主控台中或使用其他使用 Amazon EKS API 的工具指定執行個體類型，請使用 `t3.medium` 執行個體類型。如果您的節點群組正在使用 Spot 容量類型，則建議使用主控台指定多個執行個體類型。如需詳細資訊，請參閱 受管節點群組容量類型。
- 如果您部署到節點群組的任何容器使用執行個體中繼資料服務版本 2，請確定在啟動範本中將 Metadata response hop limit (中繼資料回應躍點限制) 設定為 2。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的執行個體中繼資料與使用者資料。如果在未使用自訂啟動範本的情況下部署受管節點群組，則會自動為預設啟動範本中的節點群組設定此值。

標記 Amazon EC2 執行個體

您可以使用啟動範本的 `TagSpecification` 參數，以指定要將哪些標籤套用至節點群組中的 Amazon EC2 執行個體。IAM 實體呼叫 `CreateNodegroup` 或 `UpdateNodegroupVersion` API 必須擁有 `ec2:RunInstances` 和 `ec2:CreateTags` 許可，而且標籤必須新增至啟動範本。

使用自訂安全群組

您可以使用啟動範本來指定自訂 Amazon EC2 安全群組以套用至節點群組中的執行個體。這可以是在執行個體層級安全群組參數中，也可以是網路介面組態參數的一部分。不過，您無法建立同時指定執行

個體層級和網路介面安全群組的啟動範本。請考慮下列適用於搭配受管節點群組使用之自訂安全群組的條件：

- Amazon EKS 僅允許使用單一網路介面規格的啟動範本。
- 預設情況下，Amazon EKS 將叢集安全群組套用至節點群組中的執行個體，以促進節點與控制平面之間的通訊。如果使用先前提到的任一選項在啟動範本中指定自訂安全群組，則 Amazon EKS 不會新增叢集安全群組。因此，您必須確保安全群組的傳入和傳出規則可啟用與叢集端點的通訊。如果安全群組規則不正確，工作節點便無法加入叢集。如需安全群組規則的詳細資訊，請參閱 Amazon EKS 安全群組與考量。
- 如需 SSH 存取節點群組中的執行個體，請包含允許該存取的安全群組。

Amazon EC2 使用者資料

啟動範本包含自訂使用者資料的區段。您可以在此區段中指定節點群組的組態設定，無需手動建立個別自訂 AMI。如需有關 Bottlerocket 可用設定的詳細資訊，請參閱 GitHub 上的 Using user data (使用使用者資料)。

您可以在啟動執行個體時使用 `cloud-init` 提供啟動範本中的 Amazon EC2 使用者資料。如需詳細資訊，請參閱 cloud-init 文件。您的使用者資料可用來執行一般組態操作。這包含下列操作：

- 包括使用者或群組
- 安裝套件

與受管節點群組搭配使用的啟動範本中的 Amazon EC2 使用者資料，必須為 Amazon Linux AMI 的 MIME 分段封存格式和 Bottlerocket AMI 的 TOML 格式。這是因為您的使用者資料與節點加入叢集所需的 Amazon EKS 使用者資料合併。請勿在啟動或修改 `kubelet` 的使用者資料中指定任何命令。這會作為 Amazon EKS 合併使用者資料的一部分執行。某些 `kubelet` 參數 (例如在節點上設定標籤) 可以透過受管節點群組 API 直接設定。

> **Note**
>
> 如果有關進階 `kubelet` 自訂 (包括手動啟動或傳入自訂組態參數) 的詳細資訊，請參閱 指定 AMI。如果啟動範本中指定自訂 AMI ID 時，Amazon EKS 不會合併使用者資料。

下列詳細資訊提供有關使用者資料區段的詳細資訊。

Amazon Linux 2 user data

您可以將多個使用者資料區塊組合在一起成為單一 MIME 分段檔案。例如，您可以將設定 Docker 常駐程式的雲端 Boothook 與安裝自訂套件的使用者資料 Shell 指令碼合併。MIME 分段檔案包含下列元件：

- 內容類型和部分邊界宣告：`Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="`
- MIME 版本宣告：`MIME-Version: 1.0`
- 包含以下元件的一或多個使用者資料區塊：
  - 開啟邊界，表示使用者資料區塊的開始 – `--==MYBOUNDARY==`
  - 區塊的內容類型宣告：`Content-Type: text/cloud-config; charset="us-ascii"`。如需內容類型的詳細資訊，請參閱 cloud-init 文件。
  - 使用者資料的內容 (例如，Shell 命令或 `cloud-init` 指令的清單)。
  - 結束邊界，表示 MIME 分段檔案的結束：`--==MYBOUNDARY==--`

以下是您可以用來建立自己的 MIME 分段檔案的範例。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo "Running custom user data script"

--==MYBOUNDARY==--
```

Amazon Linux 2023 user data

Amazon 2023 (AL2023) 引入了使用 YAML 組態結構描述`nodeadm`的新節點初始化程序。如果您使用自我管理的節點群組或 AMI 搭配啟動範本，您現在需要在建立新節點群組時明確提供額外的叢集中繼資料。最小必要參數的範例如下，其中`apiServerEndpoint`、`certificateAuthority`、和服務現`cidr`在需要：

```
---
apiVersion: node.eks.aws/v1alpha1
```

```
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

您通常會在用戶數據中設置此配置，無論是按原樣還是嵌入在 MIME 多部分文檔中：

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="BOUNDARY"

--BOUNDARY
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig spec: [...]

--BOUNDARY--
```

在 AL2 中，這些參數的中繼資料是從 Amazon EKS `DescribeCluster` API 呼叫中探索到的。使用 AL2023 時，自大型節點擴展期間額外 API 呼叫風險限制以來，此行為已發生變更。如果您使用的是沒有啟動範本的受管節點群組，或您正在使用，則此變更不會影響您Karpenter。如需有關`certificateAuthority`和服務的詳細資訊`cidr`，請參閱 Amazon EKS API 參考*`DescribeCluster`*中的。

Bottlerocket user data

Bottlerocket 會以 TOML 格式建構使用者資料。您可以提供要與 Amazon EKS 提供的使用者資料合併的使用者資料。例如，您可以提供額外的 `kubelet` 設定。

```
[settings.kubernetes.system-reserved]
cpu = "10m"
memory = "100Mi"
ephemeral-storage= "1Gi"
```

如需有關受支援設定的詳細資訊，請參閱 Bottlerocket 文件。您可以在使用者資料中設定節點標籤和污點。不過，建議您改為在節點群組內設定。在執行這項操作時，Amazon EKS 會套用這些組態。

合併使用者資料時，不會保留格式設定，但內容保持不變。您在使用者資料中提供的組態會覆寫 Amazon EKS 所設定的任何設定。因此，如果您設置`settings.kubernetes.max-pods`或`settings.kubernetes.cluster-dns-ip`，則用戶數據中的這些值將應用於節點。

Amazon EKS 不支援所有有效的 TOML。以下是已知不支援格式的清單：

- 引號鍵內的引號：`'quoted "value"' = "value"`
- 值中的溢出引號：`str = "I'm a string. \"You can quote me\""`
- 混合浮點數和正整數：`numbers = [ 0.1, 0.2, 0.5, 1, 2, 5 ]`
- 陣列中的混合類型：`contributors = ["foo@example.com", { name = "Baz", email = "baz@example.com" }]`
- 帶引號鍵的括號標題：`[foo."bar.baz"]`

Windows user data

Windows 使用者資料會使用 PowerShell 命令。建立受管節點群組時，您的自訂使用者資料會與 Amazon EKS 受管使用者資料結合。您的 PowerShell 命令為優先，然後是受管的使用者資料命令，所有這些命令都在一個 **<powershell></powershell>** 標籤中。

> **Note**
>
> 如果啟動範本中未指定任何 AMI ID，請勿在使用者資料中使用 Windows Amazon EKS 引導指令碼來設定 Amazon EKS。

範例使用者資料如下。

```
<powershell>
Write-Host "Running custom user data script"
</powershell>
```

指定 AMI

如果有下列其中一項要求，請在啟動範本 `ImageId` 欄位中指定 AMI ID。選取您對其他資訊的要求。

提供使用者資料以將引數傳遞給 **bootstrap.sh** 檔案，其中包含經 Amazon EKS 優化的 Linux/Bottlerocket AMI

引導是一個術語，用於描述新增在執行個體啟動時可以執行的命令。例如，引導允許使用額外的 kubelet 引數。您可以使用 eksctl 將引數傳遞給 bootstrap.sh 而不指定啟動範本。您也可以在啟動範本的使用者資料區段中指定資訊來執行此動作。

eksctl without specifying a launch template

使用下列內容建立名為 *my-nodegroup.yaml* 的檔案。使用您自己的值取代每一個 *example value*。--apiserver-endpoint、--b64-cluster-ca，和 --dns-cluster-ip 引數為選用。但是，定義引述會允許 bootstrap.sh 指令碼避免進行 describeCluster 呼叫。這在經常縮減和擴增節點的私有叢集設定或叢集中非常有用。如需 bootstrap.sh 指令碼的詳細資訊，請參閱 GitHub 上的 bootstrap.sh 檔案。

- 唯一需要的引數是叢集名稱 (*my-cluster*)。
- 若要擷取 ami-*1234567890abcdef0* 的最佳畫 AMI ID，您可以使用以下各節中的表格：
  - 擷取 Amazon EKS 最佳化 Amazon Linux AMI ID
  - 擷取 Amazon EKS 最佳化 Bottlerocket AMI ID
  - 擷取 Amazon EKS 最佳化 Windows AMI ID
- 若要擷取您的叢集的 *certificate-authority*，請執行以下命令。

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- 若要擷取您的叢集的 *api-server-endpoint*，請執行以下命令。

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- --dns-cluster-ip 的值是結尾處為 .10 的服務 CIDR。若要擷取您的叢集的 *service-cidr*，請執行以下命令。例如，如果傳回的值是 ipv4 10.100.0.0/16，則您的值是 *10.100.0.10*。

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- 此範例使用其中包含 Amazon EKS 最佳化 AMI 的 bootstrap.sh 指令碼提供 kubelet 引數來設定自訂 max-pods 值。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘

字元也可以包含連字符和底線。如需選取 *my-max-pods-value* 的協助，請參閱 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限。

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code

managedNodeGroups:
  - name: my-nodegroup
    ami: ami-1234567890abcdef0
    instanceType: m5.large
    privateNetworking: true
    disableIMDSv1: true
    labels: { x86-al2-specified-mng }
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster \
        --b64-cluster-ca certificate-authority \
        --apiserver-endpoint api-server-endpoint \
        --dns-cluster-ip service-cidr.10 \
        --kubelet-extra-args '--max-pods=my-max-pods-value' \
        --use-max-pods false
```

針對每一個可用的 `eksctl config` 檔案選項，請參閱 `eksctl` 文件中的 Config file schema (組態檔案結構描述)。此 `eksctl` 公用程式仍會為您建立啟動範本，並使用在 `config` 檔案提供的資料來填入其使用者資料。

使用下列命令來建立節點群組。

```
eksctl create nodegroup --config-file=my-nodegroup.yaml
```

User data in a launch template

在啟動範本的使用者資料區段中指定下列資訊。使用您自己的值取代每一個 *example value*。`--apiserver-endpoint`、`--b64-cluster-ca`，和 `--dns-cluster-ip` 引數為選用。但是，定義引述會允許 `bootstrap.sh` 指令碼避免進行 `describeCluster` 呼叫。這在經常縮減和擴增

節點的私有叢集設定或叢集中非常有用。如需 bootstrap.sh 指令碼的詳細資訊，請參閱 GitHub 上的 bootstrap.sh 檔案。

- 唯一需要的引數是叢集名稱 (*my-cluster*)。
- 若要擷取您的叢集的 *certificate-authority*，請執行以下命令。

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- 若要擷取您的叢集的 *api-server-endpoint*，請執行以下命令。

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- --dns-cluster-ip 的值是結尾處為 .10 的服務 CIDR。若要擷取您的叢集的 *service-cidr*，請執行以下命令。例如，如果傳回的值是 ipv4 10.100.0.0/16，則您的值是 *10.100.0.10*。

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- 此範例使用其中包含 Amazon EKS 最佳化 AMI 的 bootstrap.sh 指令碼提供 kubelet 引數來設定自訂 max-pods 值。如需選取 *my-max-pods-value* 的協助，請參閱 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
set -ex
/etc/eks/bootstrap.sh my-cluster \
  --b64-cluster-ca certificate-authority \
  --apiserver-endpoint api-server-endpoint \
  --dns-cluster-ip service-cidr.10 \
  --kubelet-extra-args '--max-pods=my-max-pods-value' \
  --use-max-pods false
```

```
--==MYBOUNDARY==--
```

提供使用者資料以將引數傳遞給 **Start-EKSBootstrap.ps1** 檔案，其中包含經 Amazon EKS 優化的 Windows AMI

引導是一個術語，用於描述新增在執行個體啟動時可以執行的命令。您可以使用 eksctl 將引數傳遞給 Start-EKSBootstrap.ps1 而不指定啟動範本。您也可以在啟動範本的使用者資料區段中指定資訊來執行此動作。

如果您想指定自訂 Windows AMI ID，請注意下列考量事項：

- 您必須使用啟動範本，並在使用者資料區段中提供所需引導命令。若要擷取所需 Windows ID，您可以使用 Amazon EKS 最佳化 Windows AMI 中的資料表。
- 您需滿足幾個限制和條件。例如，您必須新增eks:kube-proxy-windows至 AWS IAM 驗證器設定對應。如需詳細資訊，請參閱 指定 AMI ID 時的限制和條件。

在啟動範本的使用者資料區段中指定下列資訊。使用您自己的值取代每一個 *example value*。-APIServerEndpoint、-Base64ClusterCA，和 -DNSClusterIP 引數為選用。但是，定義引述會允許 Start-EKSBootstrap.ps1 指令碼避免進行 describeCluster 呼叫。

- 唯一需要的引數是叢集名稱 (*my-cluster*)。
- 若要擷取您的叢集的 *certificate-authority*，請執行以下命令。

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text --
name my-cluster --region region-code
```

- 若要擷取您的叢集的 *api-server-endpoint*，請執行以下命令。

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-cluster
 --region region-code
```

- --dns-cluster-ip 的值是結尾處為 .10 的服務 CIDR。若要擷取您的叢集的 *service-cidr*，請執行以下命令。例如，如果傳回的值是 ipv4 10.100.0.0/16，則您的值是 *10.100.0.10*。

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr" --
output text --name my-cluster --region region-code
```

- 如需其他引數，請參閱 引導指令碼組態參數。

> **Note**
>
> 如果您使用的是自訂服務 CIDR，則需要使用 `-ServiceCIDR` 參數來對其進行指定。否則，叢集中 Pods 的 DNS 解析將會失敗。

```
<powershell>
[string]$EKSBootstrapScriptFile = "$env:ProgramFiles\Amazon\EKS\Start-EKSBootstrap.ps1"
& $EKSBootstrapScriptFile -EKSClusterName my-cluster `
  -Base64ClusterCA certificate-authority `
  -APIServerEndpoint api-server-endpoint `
  -DNSClusterIP service-cidr.10
</powershell>
```

由於特定安全、合規或內部政策要求，執行自訂 AMI

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon Machine Images (AMI)。Amazon EKS AMI 構建規格包含用於基於 Amazon Linux 構建自定義 Amazon EKS AMI 的資源和配置腳本。如需詳細資訊，請參閱 GitHub 上的 Amazon EKS AMI Build Specification (Amazon EKS AMI 建置規格)。若要建置使用其他作業系統安裝的自訂 AMI，請參閱 GitHub 上的 Amazon EKS Sample Custom AMIs (Amazon EKS 範例自訂 AMI)。

> **Important**
>
> 指定 AMI 時，Amazon EKS 不會合併任何使用者資料。相反，您必須負責提供所需的 `bootstrap` 命令，讓節點加入叢集。如果節點無法加入叢集，則 Amazon EKS `CreateNodegroup` 和 `UpdateNodegroupVersion` 動作也會失敗。

指定 AMI ID 時的限制和條件

以下是使用受管節點群組指定 AMI ID 所涉及的限制和條件：

- 您必須建立新的節點群組，才能在指定啟動範本的 AMI ID 與不指定 AMI ID 之間切換。
- 有較新的 AMI 版本可用時，您不會在主控台中收到通知。若要將節點群組更新為較新的 AMI 版本，您需要使用更新的 AMI ID 來建立新版本的啟動範本。然後，您需要使用新的啟動範本版本更新節點群組。
- 如果指定 AMI ID，則無法在 API 中設定下列欄位：

- amiType
- releaseVersion
- version

- 如果您指定 AMI ID，則 API 的任何 taints 集合都會以非同步方式套用。若要在節點加入叢集之前套用污點，您必須使用 --register-with-taints 命令列旗標將污點傳遞給使用者資料的 kubelet。如需詳細資訊，請參閱 Kubernetes 文件中的 kubelet。
- 為Windows受管節點群組指定自訂 AMI ID 時，請新增eks:kube-proxy-windows至 AWS IAM 驗證器組態對應。必須要有這項，DNS 才能正常運作。

1. 開啟 AWS IAM 驗證器設定對應以進行編輯。

```
kubectl edit -n kube-system cm aws-auth
```

2. 將此項目新增至與 Windows 節點關聯之每個 rolearn 下的 groups 清單中。您的組態映射看起來應類似於 aws-auth-cm-windows.yaml。

```
- eks:kube-proxy-windows
```

3. 儲存檔案並結束您的文字編輯器。

刪除受管節點群組

本主題會描述可以如何刪除 Amazon EKS 受管節點群組。刪除受管節點群組時，Amazon EKS 會先將 Auto Scaling 群組的最小、最大和所需大小設定為零。這就會讓節點群組縮減規模。

在每個執行個體終止之前，Amazon EKS 會傳送訊號，從該節點消耗 Pods。如果幾分鐘後沒有消耗 Pods，Amazon EKS 會讓 Auto Scaling 繼續終止執行個體。在終止每個執行個體之後，便會刪除 Auto Scaling 群組。

> **Important**
>
> 如果刪除使用節點 IAM 角色的受管節點群組，且角色未被叢集中任何其他受管節點群組使用，則此角色會從 aws-auth ConfigMap 移除。如果叢集中有任何自我管理節點群組使用相同的節點 IAM 角色，則自我管理節點會轉為 NotReady 狀態。此外，叢集操作也會中斷。若要僅針對自我管理節點群組新增您正在使用之角色的映射，請參閱 建立存取項目，前提是您的叢集的平台版本不低於 管理存取項目「先決條件」部分中列出的最低版本。如果您的平台版本早於

存取項目所需的最低版本，您可以將項目新增回 aws-auth ConfigMap。如需詳細資訊，請在您的終端機中輸入 eksctl create iamidentitymapping --help。

您可以使用 eksctl 或 AWS Management Console 刪除受管節點群組。

eksctl

使用 **eksctl** 刪除受管節點群組

輸入以下命令。使用您自己的值取代每一個 *example value*。

```
eksctl delete nodegroup \
  --cluster my-cluster \
  --name my-mng \
  --region region-code
```

如需更多選項，請參閱 eksctl 文件中的刪除和耗盡節點群組。

AWS Management Console

若要刪除您的受管節點群組 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在叢集頁面上，選擇包含要刪除之節點群組的叢集。
3. 在所選叢集頁面上，選擇運算索引標籤。
4. 在 Node Groups (節點群組) 區段中，選擇要刪除的節點群組。然後選擇 Delete (刪除)。
5. 在刪除節點群組確認對話方塊中，輸入節點群組的名稱。然後選擇 Delete (刪除)。

AWS CLI

若要刪除您的受管節點群組 AWS CLI

1. 輸入以下命令。使用您自己的值取代每一個 *example value*。

```
aws eks delete-nodegroup \
  --cluster-name my-cluster \
  --nodegroup-name my-mng \
  --region region-code
```

2. 使用鍵盤上的方向鍵來捲動回應輸出。完成後按下 **q** 鍵。

如需詳細資訊，請參閱《AWS CLI 命令參考》中的 delete-nodegroup 命令。

自我管理的節點

叢集包含排程 Pods 的一或多個 Amazon EC2 節點。Amazon EKS 節點在您的 AWS 帳戶中執行，並透過叢集 API 伺服器端點連接到叢集的控制平面。會根據 Amazon EC2 價格收費。如需詳細資訊，請參閱 Amazon EC2 定價。

一個叢集可以包含數個節點群組。每個節點群組包含一個或多個部署在 Amazon EC2 Auto Scaling 群組的節點。群組節點的執行個體類型可能會有所不同，例如使用以屬性為基礎的 Karpenter 執行個體類型選項時 。節點群組的所有執行個體都必須使用 Amazon EKS 節點 IAM 角色。

Amazon EKS 提供專門的Amazon 機器映像 (AMI)，稱為Amazon EKS 最佳化 AMI。AMI 已設定為搭配 Amazon EKS 一起使用。它們的組件包括 containerdkubelet，和 AWS IAM 身份驗證器。AMI 還包含特殊化啟動程序指令碼，使其可自動探索並連接至您叢集的控制平面。

如果使用 CIDR 區塊限制對叢集公有端點的存取，則建議您也啟用私有端點存取。這樣可以讓節點與叢集通訊。若不啟用私有端點，您指定用於公有存取的 CIDR 區塊必須包含來自 VPC 的輸出來源。如需詳細資訊，請參閱 Amazon EKS 叢集端點存取控制。

若要將自我管理節點新增至 Amazon EKS 叢集，請參閱以下主題。如果手動啟動自我管理節點，請將以下標籤新增至每個節點。如需詳細資訊，請參閱新增和刪除個別資源的標籤。如果遵循以下指南中的步驟，系統會為您自動新增必要的標籤至節點。

金鑰	值
kubernetes.io/cluster/ *my-cluster*	owned

如需 Kubernetes 對節點普遍觀點的詳細資訊，請參閱 Kubernetes 文件中的 Nodes (節點)。

主題

- 啟動自我管理的 Amazon Linux 節點
- 正在啟動自我管理的 Bottlerocket 節點
- 正在啟動自我管理的 Windows 節點
- 正在啟動自我管理的 Ubuntu 節點

- 自我管理的節點更新

啟動自我管理的 Amazon Linux 節點

本主題會說明如何啟動已向 Amazon EKS 叢集註冊的 Linux 節點的 Auto Scaling 群組。節點加入叢集後，您就可以將 Kubernetes 應用程式部署至其中。您也可以使用eksctl或啟動自我管理的 AWS Management Console Amazon Linux 節點。如果您需要在上啟動節點 AWS Outposts，請參閱啟動 Outpost 上自我管理的 Amazon Linux 節點。

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 建立 Amazon EKS 叢集。如果您已啟用 AWS Outposts、 AWS Wavelength或 L AWS ocal Zones 的子網路，則在您建立叢集時不得傳入這些子網路。 AWS 區域
- 供節點使用的現有 IAM 角色。若要建立服務角色，請參閱Amazon EKS 節點 IAM 角色。如果此角色沒有任何 VPC CNI 的政策，則 VPC CNI Pod 需要以下個別角色。
- (選用，但建議) Amazon VPC CNI plugin for Kubernetes 附加元件設定有自己的 IAM 角色，該角色已連接必要的 IAM 政策。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。
- 熟悉 選擇 Amazon EC2 執行個體類型 中列出的考量。取決於您選擇的執行個體類型，您的叢集和 VPC 可能還有其他先決條件。

eksctl

> **Note**
>
> eksctl目前不支持 Amazon Linux 2023。

先決條件

已在裝置或 AWS CloudShell上安裝版本 `0.183.0` 或更新版本的 `eksctl` 命令列工具。如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的安裝一節。

使用 **`eksctl`** 啟動自我管理的 Linux 節點

1. (選用) 如果 AmazonEKS_CNI_Policy 受管 IAM 政策已連接至您的 Amazon EKS 節點 IAM 角色，建議您改為將其指派給您與 Kubernetes `aws-node` 服務帳戶相關聯的 IAM 角色。如需

詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

2. 以下命令會在現有的叢集建立節點群組。將 *al-nodes* 取代為您的節點群組名稱。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。使用您叢集的名稱取代 *my-cluster*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。以您自己的值取代其餘的 *example value*。依預設，會使用與控制平面相同的 Kubernetes 版本來建立節點。

   在為 `--node-type` 選擇值之前，請先參閱 選擇 Amazon EC2 執行個體類型。

   使用 Amazon EC2 金鑰對或公有金鑰的名稱取代 *my-key*。此金鑰會在節點啟動後用於將 SSH 套用至節點。如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 金鑰對。

   使用下列命令來建立您的節點群組。

   > **Important**
   >
   > 如果您想要將節點群組部署到 AWS Outposts、Wavelength 或本機區域子網路，還有其他考量：
   >
   > - 在建立叢集時，不得傳入子網路。
   > - 您必須使用指定子網路和 `volumeType: gp2` 的組態檔案來建立節點群組。如需詳細資訊，請參閱 `eksctl` 文件中的從組態檔案建立節點群組和組態檔案結構描述。

```
eksctl create nodegroup \
  --cluster my-cluster \
  --name al-nodes \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --ssh-access \
  --managed=false \
  --ssh-public-key my-key
```

   若要部署節點群組：

- 其可以將比預設組態更大量的 IP 地址指派給 Pods，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。
- 其可以從與執行個體不同的 CIDR 區塊將 IPv4 地址指派給 Pods，請參閱 Pod 的自訂聯網。
- 其可以指派 IPv6 地址至 Pods 和服務，請參閱 IPv6叢集的位址Pods、和 services。
- 您必須使用 config 檔案部署節點群組來使用 containerd 執行時間。如需詳細資訊，請參閱 測試從移轉Docker到 containerd。
- 沒有對外網際網路存取，請參閱 私有叢集要求。

如需所有可用選項和預設值的完整清單，請輸入以下命令。

```
eksctl create nodegroup --help
```

如果節點無法加入叢集，請參閱故障診斷指南中的 節點無法加入叢集。

範例輸出如下。建立節點時，會有數行輸出。輸出的最後幾行之一類似於以下的範例行。

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (選用) 部署範例應用程式以測試您的叢集和 Linux 節點。
4. 如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：
   - 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
   - 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

   如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

AWS Management Console

步驟 1：使用 AWS Management Console啟動自我管理的 Linux 節點

1. 下載最新版本的 AWS CloudFormation 模板。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/
amazon-eks-nodegroup.yaml
```

2. 等待您的叢集狀態顯示為 ACTIVE。如果在叢集處於作用中狀態之前啟動節點，則節點向叢集註冊將會失敗，導致必須再次啟動。
3. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation
4. 選擇 Create stack (建立堆疊)，然後選取 With new resources (standard) (使用新資源 (標準))。
5. 針對 Specify template (指定範本)，選取 Upload a template file (上傳範本檔案)，然後選取 Choose file (選擇檔案)。
6. 選取您下載的 `amazon-eks-nodegroup.yaml` 檔案。
7. 選取下一步。
8. 在 Specify stack details (指定堆疊詳細資訊) 頁面上，據此填寫下列參數，然後選擇 Next (下一步)：

   - Stack name：為您的 AWS CloudFormation 堆疊選擇堆疊名稱。例如，您可以稱它為 ***my-cluster-nodes***。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - ClusterName：輸入您在建立 Amazon EKS 叢集時使用的名稱。此名稱必須與叢集名稱相同，否則節點無法加入叢集。
   - ClusterControlPlaneSecurity群組：從建立 VPC 時產生的 AWS CloudFormation 輸出中選擇SecurityGroups值。

     下列步驟顯示擷取適用群組的一種操作。

     1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
     2. 選擇叢集的名稱。
     3. 選擇 Networking (網路) 索引標籤。
     4. 從「群組」下拉式清單中選取時，請使用其他安全性群組值作為參考。ClusterControl PlaneSecurity

- NodeGroup名稱：輸入節點群組的名稱。此名稱稍後可用以識別為您節點建立的 Auto Scaling 節點群組。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。
- NodeAutoScalingGroupMinSize：輸入節點「自動調整」群組可擴充至的最小節點數。
- NodeAutoScalingGroupDesiredCapacity：輸入堆疊建立時要擴展到的所需節點數。
- NodeAutoScalingGroupMaxSize：輸入節點「自動調整」群組可向外擴充至的節點數目上限。
- NodeInstance類型：選擇節點的執行個體類型。如需詳細資訊，請參閱 選擇 Amazon EC2 執行個體類型。
- NodeImageIDSSmParam：針對可變版本的最新 Amazon EKS 優化 AMI 預先填充了 Amazon EC2 Systems Manager 參數。Kubernetes若要使用 Amazon EKS 支援不同的 Kubernetes 次要版本，請將 *1.XX* 取代為不同的受支援版本。建議指定與叢集相同的 Kubernetes 版本。

  您也可以用不同*amazon-linux-2*的 AMI 類型替換。如需詳細資訊，請參閱 擷取 Amazon EKS 最佳化 Amazon Linux AMI ID。

  > **Note**
  >
  > Amazon EKS 節點 AMI 基於 Amazon Linux。您可以透過 Amazon Linux 安全中心或是訂閱關聯的 RSS 摘要，追蹤 Amazon Linux 2 的安全或隱私權事件。安全與隱私權事件包含問題的概觀、哪些套件受到影響，以及如何更新您的執行個體以修正問題。

- NodeImageID：(選用) 如果您使用自己的自訂 AMI (而不是 Amazon EKS 最佳化 AMI)，請輸入您 AWS 區域的節點 AMI ID。如果您在此處指定一個值，它會取代 NodeImageIDSSmParam 欄位中的任何值。
- NodeVolume大小：指定節點的根磁碟區大小 (以 GiB 為單位)。
- NodeVolume類型：指定節點的根磁碟區類型。
- KeyName：輸入 Amazon EC2 安全殼層 key pair 的名稱，您可以在啟動後使用 SSH 連線到節點。如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 金鑰對。

> **Note**
>
> 如果您沒有在這裡提供 key pair， AWS CloudFormation 堆疊建立會失敗。

- BootstrapArguments：指定要傳遞給節點啟動程序檔的任何選用引數，例如額外kubelet引數。如需詳細資訊，請檢視 GitHub 上的 bootstrap script usage information (啟動程序指令碼使用資訊)。

  若要部署節點群組：

  - 其可以將比預設組態更大量的 IP 地址指派給 Pods，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。
  - 其可以從與執行個體不同的 CIDR 區塊將 IPv4 地址指派給 Pods，請參閱 Pod 的自訂聯網。
  - 其可以指派 IPv6 地址至 Pods 和服務，請參閱 IPv6叢集的位址Pods、和 services。
  - 您必須使用 config 檔案部署節點群組來使用 containerd 執行時間。如需詳細資訊，請參閱 測試從移轉Docker到 containerd。
  - 沒有對外網際網路存取，請參閱 私有叢集要求。
- DisableIMDSv1：預設情況下，每個節點都支援執行個體中繼資料服務版本 1 (IMDSv1) 和 IMDSv2。您可以停用 IMDSv1。若要防止節點群組中的未來節點和 Pods 使用 MDSv1，請將 DisableIMDSv1 設定為 true。如需 IMDS 的詳細資訊，請參閱設定執行個體中繼資料服務。如需在節點上限制存取的詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。
- VpcId：輸入您建立之 VPC 的識別碼。
- 子網路：選擇您為 VPC 建立的子網路。如果您依照 為 Amazon EKS 叢集建立 VPC 中所述的步驟建立 VPC，則只要指定要啟動節點的 VPC 中的私有子網。您可以看到哪些子網是私有子網，方法是從叢集的 Networking (聯網) 標籤打開每一個子網連結。

> **Important**
>
> - 如有任何子網路是公有子網路，則必須啟用自動公有 IP 地址指派設定。如果未針對公用子網路啟用此設定，則您部署到該公用子網路的任何節點都不會被指派公用 IP 位址，也無法與叢集或其他 AWS 服務通訊。如果子網路是在 2020 年 3 月 26 日之前使用任一 Amazon EKS AWS CloudFormation VPC 範本或使用eksctl部署，則公有子網路的自動公用 IP 位址指派將停用。如需如何啟用子網的公有 IP 地

址指派的相關資訊，請參閱修改您子網的公有 IPv4 定址屬性。如果將節點部署到私有子網路，則可以透過 NAT 閘道與叢集和其他 AWS 服務進行通訊。

- 如果子網路無法存取網際網路，請確定您已知悉 私有叢集要求 中的考量和額外步驟。
- 如果您選取「 AWS Outposts Wavelength」或「本機區域」子網路，則在建立叢集時不得傳入子網路。

9. 請在 Configure stack options (設定堆疊選項) 頁面上選取所需的選項，然後選擇 Next (下一頁)。
10. 選取我了解 AWS CloudFormation 會建立 IAM 資源。左側的核取方塊，然後選擇 Create stack (建立堆疊)。
11. 當當堆疊已完成建立時，從主控台將其選取，然後選擇 Outputs (輸出)。
12. 記錄已建立之節點群組的「NodeInstance角色」。當您設定 Amazon EKS 節點時會需要此值。

步驟 2：讓節點加入叢集

> **Note**
>
> 如果您在沒有傳出網際網路存取權的私有 VPC 內啟動節點，則必須啟用節點才能從 VPC 內加入叢集。

1. 檢查以瞭解是否有 `aws-auth` `ConfigMap`。

```
kubectl describe configmap -n kube-system aws-auth
```

2. 如果您看到 `aws-auth` `ConfigMap`，請視需要更新之。

   a. 開啟 `ConfigMap` 進行編輯。

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. 視需要新增 `mapRoles` 個項目。將`rolearn`值設定為您在上一個程序中記錄的「NodeInstance角色」值。

```
[...]
```

```
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

c. 儲存檔案並結束您的文字編輯器。

3. 如果您收到一個錯誤，說明 "Error from server (NotFound): configmaps "aws-auth" not found，那麼請套用股票 ConfigMap。

a. 下載組態對應。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm.yaml
```

b. 在aws-auth-cm.yaml檔案中，將rolearn值設定為您在上一個程序中記錄的「NodeInstance角色」值。您可以使用文字編輯器來完成此操作，或者透過替換 *my-node-instance-role* 並執行以下命令：

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-
instance-role|' aws-auth-cm.yaml
```

c. 套用組態。此命令可能需要幾分鐘的時間來完成。

```
kubectl apply -f aws-auth-cm.yaml
```

4. 查看節點的狀態，並等待他們到達 Ready 狀態。

```
kubectl get nodes --watch
```

輸入 Ctrl+C 傳回 Shell 提示。

> **Note**
>
> 如果您收到任何授權或資源類型錯誤，請參閱故障診斷主題中的未經授權或存取遭拒 (kubectl)。

如果節點無法加入叢集，請參閱故障診斷指南中的 節點無法加入叢集。

5. (僅限 GPU 節點) 若選擇了 GPU 執行個體類型以及 Amazon EKS 最佳化之加速 AMI，您就必須套用 Kubernetes 專用 NVIDIA 裝置外掛程式作為叢集上的 DaemonSet。請先以您想要的 NVIDIA/k8s-device-plugin 版本來取代 *vX.X.X*，再執行下列命令。

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
```

步驟 3：其他動作

1. (選用) 部署範例應用程式以測試您的叢集和 Linux 節點。
2. (選用) 若將 AmazonEKS_CNI_Policy 受管 IAM 政策 (如果您有 IPv4 叢集) 或 *AmazonEKS_CNI_IPv6_Policy* (您具有 IPv6 叢集時自我建立的政策) 連接至 the section called “節點 IAM 角色”，建議改為將其指派給您與 Kubernetes aws-node 服務帳戶關聯的 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。
3. 如果下列條件為真，我們建議封鎖 Pod 對 IMDS 的存取：
   - 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
   - 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

   如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

ML 的容量區塊

> **Important**
>
> - 容量區塊僅適用於特定 Amazon EC2 執行個體類型和 AWS 區域。如需相容性資訊，請參閱 Amazon EC2 Linux 執行個體使用者指南中的使用容量區塊先決條件。
> - 容量區塊目前無法與 Amazon EKS 受管節點群組或Karpenter一起使用。

機器學習 (ML) 的容量區塊可讓您指定未來日期保留 GPU 執行個體，以支援短期 ML 工作負載。在容量區塊內執行的執行個體會自動放置在 Amazon EC2 內 UltraClusters，因此不需要使用叢集置放群組。如需詳細資訊，請參閱 Amazon EC2 Linux 執行個體使用者指南中的 ML 適用容量區塊。

您可以將容量區塊與 Amazon EKS 搭配使用，藉此佈建和擴展您的自我管理節點。下列步驟提供一般範例概觀。

1. 在 AWS Management Console建立啟動範本 如需詳細資訊，請參閱 Amazon EC2 Auto Scaling 使用者指南中的機器學習工作負載使用容量區塊。

   請務必包含執行個體類型和 Amazon Machine Image (AMI) 的組態。
2. 使用容量保留 ID 將容量區塊連結至啟動範本。

   以下範例 AWS CloudFormation 範本可用來建立以容量區塊為目標的啟動範本：

   ```
   NodeLaunchTemplate:
       Type: "AWS::EC2::LaunchTemplate"
       Properties:
         LaunchTemplateData:
           InstanceMarketOptions:
             MarketType: "capacity-block"
           CapacityReservationSpecification:
             CapacityReservationTarget:
               CapacityReservationId: "cr-02168da1478b509e0"
           IamInstanceProfile:
             Arn: iam-instance-profile-arn
           ImageId: image-id
           InstanceType: p5.48xlarge
           KeyName: key-name
           SecurityGroupIds:
           - sg-05b1d815d1EXAMPLE
           UserData: user-data
   ```

   由於容量區塊為區域性，因此您必須在進行保留的可用區域中傳遞子網路。
3. 如果您要先建立自我管理節點群組，再讓容量保留變成作用中，請將所需的容量設定為 0。建立節點群組時，請確定您只為保留容量的可用區域指定個別子網路。

   以下是可以使用的 CloudFormation 範例範本。我們利用上一個範例中所示的 AWS::Amazon EC2::LaunchTemplate 資源，針對其中的 LaunchTemplateId 與 Version 來示範取得的值，這個範例也會取得在相同範本中其他位置宣告的 DesiredCapacity、MaxSize、MinSize 和 VPCZoneIdentifier 值。

```
  NodeGroup:
    Type: "AWS::AutoScaling::AutoScalingGroup"
    Properties:
      DesiredCapacity: !Ref NodeAutoScalingGroupDesiredCapacity
      LaunchTemplate:
        LaunchTemplateId: !Ref NodeLaunchTemplate
        Version: !GetAtt NodeLaunchTemplate.LatestVersionNumber
      MaxSize: !Ref NodeAutoScalingGroupMaxSize
      MinSize: !Ref NodeAutoScalingGroupMinSize
      VPCZoneIdentifier: !Ref Subnets
      Tags:
        - Key: Name
          PropagateAtLaunch: true
          Value: !Sub ${ClusterName}-${NodeGroupName}-Node
        - Key: !Sub kubernetes.io/cluster/${ClusterName}
          PropagateAtLaunch: true
          Value: owned
```

4. 成功建立節點群組之後，請務必記錄所建立節點群組的 NodeInstanceRole。這麼做才能確保節點群組擴展時，新節點會加入叢集，且 Kubernetes 能夠識別節點。如需詳細資訊，請參閱啟動自我管理的 Amazon Linux 節點中的 AWS Management Console 指示。
5. 建議您根據容量區塊的保留時間，為 Auto Scaling 群組建立排程擴展政策。如需詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的為 Amazon EC2 Auto Scaling 排程擴展。

   在容量區塊結束時間的 30 分鐘前，您保留的所有執行個體都可以使用。屆時仍在執行的執行個體將會開始終止。為了能有足夠的時間正常耗盡節點，建議您在容量區塊保留結束前預留 30 分鐘以上，排程擴展至零。

   如果您想要在容量保留變成 Active 時改為手動縱向擴展，則需要在容量區塊保留的開始時間更新 Auto Scaling 群組的所需容量。然後，您還需要在容量區塊保留結束前預留 30 分鐘以上手動縮減規模。
6. 節點群組現在已準備就緒，可供工作負載和 Pods 進行排程。
7. 為了Pods使您的正常排空，我們建議您設置 AWS 節點終止處理程序。此處理常式將能夠使用 Amazon EC2 Auto Scaling 觀看來自 Amazon EC2 Auto Scaling 的「ASG 擴展」生命週期事件，EventBridge 並允許Kubernetes控制平面在執行個體無法使用之前採取必要的動作。否則，您的 Pods 和 Kubernetes 物件都會卡在待處理狀態。如需詳細資訊，請參閱上的AWS 節點終止處理常式 GitHub。

如果您未設定節點終止處理常式，建議您在接近 30 分鐘前開始手動耗盡 Pods，以便它們有足夠的時間正常耗盡。

正在啟動自我管理的 Bottlerocket 節點

> **Note**
>
> 受管節點群組可能會為您的使用案例提供一些優勢。如需詳細資訊，請參閱 受管節點群組。

本主題說明如何啟動向 Amazon EKS 叢集註冊的自動擴展節點群組。 Bottlerocket是以開放原始碼為Linux基礎 AWS 的作業系統，可用來在虛擬機器或裸機主機上執行容器。節點加入叢集後，您就可以將 Kubernetes 應用程式部署至其中。如需 Bottlerocket 的詳細資訊，請參閱 GitHub 上的搭配 Amazon EKS 使用 Bottlerocket AMI，以及 `eksctl` 文件中的自訂 AMI 支援。

如需就地升級的相關資訊，請參閱 GitHub 上的Bottlerocket 更新運算子。

> **Important**
>
> - Amazon EKS 節點為標準 Amazon EC2 執行個體，會根據一般 Amazon EC2 執行個體價格向您收取這些節點的費用。如需詳細資訊，請參閱 Amazon EC2 定價。
> - 您可以在 Outposts 上的 Amazon EKS 延伸叢集中啟動保安特勒章節點，但無法在 AWS Outposts 的本機叢集中啟動它們。 AWS 如需詳細資訊，請參閱 AWS Outposts 上的 Amazon EKS。
> - 您可以使用 x86 或 Arm 處理器部署至 Amazon EC2 執行個體。不過，您無法部署至具有 Inferentia 晶片的執行個體。
> - Bottlerocket兼容 AWS CloudFormation. 但是，沒有可以複製的官方 CloudFormation 範本來部署 Amazon EKS 的Bottlerocket節點。
> - Bottlerocket 映像不會隨附 SSH 伺服器或 Shell。您可以使用 out-of-band 訪問方法來允許SSH啟用管理容器，並通過一些引導配置步驟與用戶數據。如需詳細資訊，請參閱 GitHub 上 bottlerocket README.md 中的這些章節：
>   - 探勘
>   - 管理員容器
>   - Kubernetes 設定

使用 **eksctl** 啟動 Bottlerocket 節點

此程序需要 eksctl 版本 0.183.0 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的 Installation 一節。

> **Note**
>
> 此程序只適用於使用 eksctl 所建立的叢集。

1. 將以下內容複製到您的裝置。使用您叢集的名稱取代 *my-cluster*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。將 *ng-bottlerocket* 取代為您的節點群組名稱。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。若要在 Arm 執行個體上部署，請以 Arm 執行個體類型取代 *m5.large*。使用 Amazon EC2 SSH 金鑰對名稱取代 *my-ec2-keypair-name*，您可以在節點啟動後使用該金鑰對來透過 SSH 連接至節點。如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 金鑰對。以您自己的值取代其餘所有 *example values*。完成取代後，請執行修改後的命令以建立 bottlerocket.yaml 檔案。

   如果指定 Arm Amazon EC2 執行個體類型，則請在部署前檢閱 Amazon EKS 最佳化的 Arm Amazon Linux AMI 的考量事項。如需如何使用自訂 AMI 部署的說明，請參閱 GitHub 上的建置 Bottlerocket，以及 eksctl 文件中的自訂 AMI 支援。若要部署受管節點群組，請使用啟動範本部署自訂 AMI。如需詳細資訊，請參閱 使用啟動範本自訂受管節點。

   > **Important**
   >
   > 若要將節點群組部署到 AWS Outposts AWS Wavelength、或 AWS 本機區域子網路，請勿在建立叢集時通過 AWS Outposts或 AWS 本機區域子網路。 AWS Wavelength您必須在下列範例中指定子網路。如需詳細資訊，請參閱 eksctl 文件中的從組態檔案建立節點群組和組態檔案結構描述。*region-code*以叢集所 AWS 區域 在的位置取代。

   ```
   cat >bottlerocket.yaml <<EOF
   ```

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.30'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-bottlerocket
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Bottlerocket
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

2. 使用下列命令部署節點。

```
eksctl create nodegroup --config-file=bottlerocket.yaml
```

範例輸出如下。

建立節點時，會有數行輸出。輸出的最後幾行之一類似於以下的範例行。

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (選用) 使用 Amazon EBS CSI 外掛程式在 Bottlerocket 節點上建立 Kubernetes 持續性磁碟區。預設 Amazon EBS 驅動程式依賴於未包含 Bottlerocket 的檔案系統工具。如需使用驅動程式建立儲存類別的詳細資訊，請參閱 Amazon EBS CSI 驅動程式。

4. (選用) 依預設 kube-proxy 會將 nf_conntrack_max 核心參數設定為預設值，此值可能與開機時 Bottlerocket 原先設定的不同。若要保留 Bottlerocket 的預設設定，請使用以下命令編輯 kube-proxy 組態。

```
kubectl edit -n kube-system daemonset kube-proxy
```

新增 --conntrack-max-per-core 和 --conntrack-min 到 kube-proxy 引數，這些引數位於以下範例中。0 設定意味著沒有改變。

```
      containers:
      - command:
        - kube-proxy
        - --v=2
        - --config=/var/lib/kube-proxy-config/config
        - --conntrack-max-per-core=0
        - --conntrack-min=0
```

5. (選用) 部署範例應用程式來測試您的 Bottlerocket 節點。
6. 如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：
   - 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
   - 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

   如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

正在啟動自我管理的 Windows 節點

本主題會說明如何啟動向 Amazon EKS 叢集註冊之 Windows 節點的 Auto Scaling 群組。節點加入叢集後，您就可以將 Kubernetes 應用程式部署至其中。

> **Important**
>
> - Amazon EKS 節點為標準 Amazon EC2 執行個體，會根據一般 Amazon EC2 執行個體價格向您收取這些節點的費用。如需詳細資訊，請參閱 Amazon EC2 定價。

- 您可以在 Outposts 上的 Amazon EKS 延伸叢集中啟動 Windows 節點，但無法在 AWS Outposts 的本機叢集中啟動它們。 AWS 如需詳細資訊，請參閱 AWS Outposts 上的 Amazon EKS。

為您的叢集啟用 Windows 支援。我們建議您在啟動 Windows 節點群組之前，先檢閱重要的考量。如需詳細資訊，請參閱 啟用 Windows 支援。

您可以使用 `eksctl` 或 AWS Management Console啟動自我管理的 Windows 節點。

eksctl

使用 **`eksctl`** 啟動自我管理的 Windows 節點

此程序需要您已安裝 `eksctl`，且您的 `eksctl` 版本至少是 `0.183.0`。您可使用以下命令檢查您的版本。

```
eksctl version
```

如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的 Installation 一節。

> **Note**
>
> 此程序只適用於使用 `eksctl` 所建立的叢集。

1. (選用) 若將 AmazonEKS_CNI_Policy 受管 IAM 政策 (如果您有 `IPv4` 叢集) 或 *`AmazonEKS_CNI_IPv6_Policy`* (您具有 `IPv6` 叢集時自我建立的政策) 連接至 the section called “節點 IAM 角色”，建議改為將其指派給您與 Kubernetes `aws-node` 服務帳戶關聯的 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。
2. 此程序假設您有現有的叢集。如果您還沒有要新增節點群組的 Amazon EKS 叢集和 Amazon Linux Windows 節點群組，建議您遵循指Amazon EKS 入門：`eksctl`南。指南提供了建立具有 Amazon Linux 節點的 Amazon EKS 叢集的完整演練。

   使用下列命令來建立您的節點群組。*`region-code`*以叢集所 AWS 區域 在的位置取代。使用您的叢集名稱取代 *`my-cluster`*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。將 *`ng-windows`* 取代為您的節點群組名稱。節點群組名

稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。對於 Kubernetes 1.24 版或更新版本，您可以將 *2019* 取代為 2022 以使用 Windows Server 2022。以您自己的值取代其餘的 *example values*。

> **Important**
>
> 若要將節點群組部署到 AWS Outposts AWS Wavelength、或 AWS 本機區域子網路，請勿在建立叢集時傳遞 AWS Outposts、Wavelength 或本機區域子網路。使用設定檔建立節點群組，並指定 AWS Outposts、Wavelength 或本機區域子網路。如需詳細資訊，請參閱 eksctl 文件中的從組態檔案建立節點群組和組態檔案結構描述。

```
eksctl create nodegroup \
    --region region-code \
    --cluster my-cluster \
    --name ng-windows \
    --node-type t2.large \
    --nodes 3 \
    --nodes-min 1 \
    --nodes-max 4 \
    --managed=false \
    --node-ami-family WindowsServer2019FullContainer
```

> **Note**
>
> - 如果節點無法加入叢集，請參閱故障診斷指南中的 節點無法加入叢集。
> - 若要查看 eksctl 命令可用的選項，請輸入下列命令。
>
>   ```
>   eksctl command -help
>   ```

範例輸出如下。建立節點時，會有數行輸出。輸出的最後幾行之一類似於以下的範例行。

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (選用) 部署範例應用程式以測試您的叢集和 Windows 節點。
4. 如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：

- 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
- 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

AWS Management Console

必要條件

- 現有 Amazon EKS 叢集和 Linux 節點群組。如果您沒有這些資源，我們建議您遵循我們的 Amazon EKS 入門 指南來建立這些資源。本指南描述如何建立具有 Linux 節點的 Amazon EKS 叢集。
- 符合 Amazon EKS 叢集要求的現有 VPC 和安全群組。如需詳細資訊，請參閱 Amazon EKS VPC 與子網要求和注意事項 及 Amazon EKS 安全群組與考量。Amazon EKS 入門 指南會建立符合要求的 VPC。您也可以遵循 為 Amazon EKS 叢集建立 VPC 手動建立 VPC。
- 現有的 Amazon EKS 叢集，其使用符合 Amazon EKS 叢集要求的 VPC 和安全群組。如需詳細資訊，請參閱 建立 Amazon EKS 叢集。如果您已啟用 AWS Outposts、 AWS Wavelength或 L AWS ocal Zones 的子網路，則在建立叢集時不得傳入這些子網路。 AWS 區域

步驟 1：若要啟動自我管理Windows節點 AWS Management Console

1. 等待您的叢集狀態顯示為 ACTIVE。如果在叢集處於作用中狀態之前啟動節點，則節點向叢集註冊將會失敗，導致您必須再次啟動。
2. 請在以下位置開啟 AWS CloudFormation 主控台 https://console.aws.amazon.com/cloudformation
3. 選擇建立堆疊。
4. 對於 Specify template (指定範本)，選取 Amazon S3 URL。
5. 複製以下 URL 並將其貼到 Amazon S3 URL 中。

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2023-02-09/amazon-eks-windows-nodegroup.yaml
```

6. 選取 Next (下一步) 兩次。

7. 在 Quick create stack (快速建立堆疊) 頁面上，視需要輸入下列參數：

   - Stack name：為您的 AWS CloudFormation 堆疊選擇堆疊名稱。例如，您可以稱它為 ***my-cluster*-nodes**。
   - ClusterName：輸入您在建立 Amazon EKS 叢集時使用的名稱。

     > **Important**
     >
     > 此名稱必須完全符合您在 步驟 1：建立您的 Amazon EKS 叢集 中使用的名稱。否則，您的節點無法加入叢集。

   - ClusterControlPlaneSecurity群組：從建立 VPC 時產生的 AWS CloudFormation 輸出中選擇安全群組。

     下列步驟顯示擷取適用群組的一種方法。

     1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
     2. 選擇叢集的名稱。
     3. 選擇 Networking (網路) 索引標籤。
     4. 從「群組」下拉式清單中選取時，請使用其他安全性群組值作為參考。ClusterControlPlaneSecurity

   - NodeGroup名稱：輸入節點群組的名稱。此名稱稍後可用以識別為您節點建立的 Auto Scaling 節點群組。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。
   - NodeAutoScalingGroupMinSize：輸入節點「自動調整」群組可擴充至的最小節點數。
   - NodeAutoScalingGroupDesiredCapacity：輸入堆疊建立時要擴展到的所需節點數。
   - NodeAutoScalingGroupMaxSize：輸入節點「自動調整」群組可向外擴充至的節點數目上限。
   - NodeInstance類型：選擇節點的執行個體類型。如需詳細資訊，請參閱 選擇 Amazon EC2 執行個體類型。

     > **Note**
     >
     > 最新版本的 Amazon VPC CNI plugin for Kubernetes 支援執行個體類型會在 GitHub 上的 vpc_ip_resource_limit.go 中列出。您可能需要更新 CNI 版本，才能使用最新

支援的執行個體類型。如需詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。

- NodeImageIDSSmParam：預先填入目前推薦的 Amazon EKS 優化核心 AMI ID 的 Amazon EC2 Systems Manager 參數。Windows若要使用完整版本的 Windows，請將 *Core* 取代為 Full。
- NodeImage識別碼：(選用) 如果您使用自己的自訂 AMI (而不是 Amazon EKS 最佳化 AMI)，請輸入您 AWS 區域的節點 AMI ID。如果您為此欄位指定一個值，它會取代 NodeImageIDSSm Param 欄位中的任何值。
- NodeVolume大小：指定節點的根磁碟區大小 (以 GiB 為單位)。
- KeyName：輸入 Amazon EC2 安全殼層 key pair 的名稱，您可以在啟動後使用 SSH 連線到節點。如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 金鑰對。

> **Note**
>
> 如果您沒有在這裡提供 key pair，則無法建立 AWS CloudFormation 堆疊。

- BootstrapArguments：指定要傳遞給節點啟動程序檔的任何選用引kubelet數，例如使用-KubeletExtraArgs.
- DisableIMDSv1：預設情況下，每個節點都支援執行個體中繼資料服務版本 1 (IMDSv1) 和 IMDSv2。您可以停用 IMDSv1。若要防止節點群組中的未來節點和 Pods 使用 MDSv1，請將 DisableIMDSv1 設定為 true。如需 IMDS 的詳細資訊，請參閱設定執行個體中繼資料服務。
- VpcId：選取您所建立之 VPC 的識別碼。
- NodeSecurity群組：選取建立 VPC 時為Linux節點群組建立的安全性群組。如果 Linux 節點連接了一個以上的安全群組，請指定所有的群組。例如，如果 Linux 節點群組是使用 eksctl 建立。
- Subnets (子網路)：選擇您建立的子網路。如果您依照 為 Amazon EKS 叢集建立 VPC 中的步驟建立 VPC，則只要指定要啟動節點的 VPC 中的私有子網。

> **Important**
>
> - 如有任何子網路是公有子網路，則必須啟用自動公有 IP 地址指派設定。如果未針對公用子網路啟用此設定，則您部署到該公用子網路的任何節點都不會被指派公用 IP 位址，也無法與叢集或其他 AWS 服務通訊。如果子網路是在 2020 年 3 月 26

日之前使用任一 Amazon EKS AWS CloudFormation VPC 範本或使用eksctl部署，則公有子網路的自動公用 IP 位址指派將停用。如需如何啟用子網的公有 IP 地址指派的相關資訊，請參閱修改您子網的公有 IPv4 定址屬性。如果將節點部署到私有子網路，則可以透過 NAT 閘道與叢集和其他 AWS 服務進行通訊。

- 如果子網路無法存取網際網路，請確定您已知悉 私有叢集要求 中的考量和額外步驟。
- 如果您選取「 AWS Outposts Wavelength」或「本機區域」子網路，則在建立叢集時不得傳入子網路。

8. 確認堆疊可建立 IAM 資源，然後選擇 Create stack (建立堆疊)。
9. 當堆疊已完成建立時，從主控台將其選取，然後選擇 Outputs (輸出)。
10. 記錄已建立之節點群組的「NodeInstance角色」。當您設定 Amazon EKS Windows 節點時會需要此值。

步驟 2：讓節點加入叢集

1. 檢查以瞭解是否有 aws-auth ConfigMap。

```
kubectl describe configmap -n kube-system aws-auth
```

2. 如果您看到 aws-auth ConfigMap，請視需要更新之。

   a. 開啟 ConfigMap 進行編輯。

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. 視需要新增 mapRoles 個項目。將這些rolearn值設定為您在先前程序中記錄的「NodeInstance角色」值。

```
[...]
data:
  mapRoles: |
- rolearn: <ARN of linux instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: <ARN of windows instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
```

```
      groups:
        - system:bootstrappers
        - system:nodes
        - eks:kube-proxy-windows
[...]
```

c. 儲存檔案並結束您的文字編輯器。

3. 如果您收到一個錯誤，說明 "Error from server (NotFound): configmaps "aws-auth" not found，那麼請套用股票 ConfigMap。

a. 下載組態對應。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm-windows.yaml
```

b. 在aws-auth-cm-windows.yaml檔案中，將這些rolearn值設定為您在先前程序中記錄的適用「NodeInstance角色」值。您可以使用文字編輯器來完成此操作，或者透過替換 *example values* 並執行以下命令：

```
sed -i.bak -e 's|<ARN of linux instance role (not instance profile)>|my-
node-linux-instance-role|' \
    -e 's|<ARN of windows instance role (not instance profile)>|my-node-
windows-instance-role|' aws-auth-cm-windows.yaml
```

> **Important**
>
> - 請勿修改此檔案中的任何其他行。
> - 請勿為 Windows 節點和 Linux 節點使用相同的 IAM 角色。

c. 套用組態。此命令可能需要幾分鐘的時間來完成。

```
kubectl apply -f aws-auth-cm-windows.yaml
```

4. 查看節點的狀態，並等待他們到達 Ready 狀態。

```
kubectl get nodes --watch
```

輸入 Ctrl+C 傳回 Shell 提示。

> **Note**
>
> 如果您收到任何授權或資源類型錯誤，請參閱故障診斷主題中的未經授權或存取遭拒 (kubectl)。

如果節點無法加入叢集，請參閱故障診斷指南中的 節點無法加入叢集。

步驟 3：其他動作

1. (選用) 部署範例應用程式以測試您的叢集和 Windows 節點。
2. (選用) 若將 AmazonEKS_CNI_Policy 受管 IAM 政策 (如果您有 IPv4 叢集) 或 *AmazonEKS_CNI_IPv6_Policy* (您具有 IPv6 叢集時自我建立的政策) 連接至 the section called “節點 IAM 角色”，建議改為將其指派給您與 Kubernetes aws-node 服務帳戶關聯的 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。
3. 如果下列條件為真，我們建議封鎖 Pod 對 IMDS 的存取：
   - 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
   - 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

   如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

正在啟動自我管理的 Ubuntu 節點

> **Note**
>
> 受管節點群組可能會為您的使用案例提供一些優勢。如需詳細資訊，請參閱 受管節點群組。

本主題說明如何在 Amazon E lastic Kubernetes Service (EKS) 或Ubuntu在向您的 Amazon EKS 叢集註冊的亞馬遜彈性 Kubernetes 服務 (EKS) 節點Ubuntu Pro上啟動 Auto Scaling 群組。 Ubuntu而 Ubuntu Pro對於 EKS 是基於官方的Ubuntu最小 LTS，包括與共同開發的自定義內 AWS 核 AWS，並

且專門為 EKS 構建。 Ubuntu Pro通過支持 EKS 擴展支持期，內核livepatch，FIPS 合規性以及運行無限Pro容器的功能來增加額外的安全性涵蓋範圍。

節點加入叢集之後，您可以將容器化應用程式部署到這些節點。如需詳細資訊，請參閱文件中的 Ubuntuon AWS 和 Custom AMI 支援eksctl文件。

> **Important**
>
> - Amazon EKS 節點為標準 Amazon EC2 執行個體，會根據一般 Amazon EC2 執行個體價格向您收取這些節點的費用。如需詳細資訊，請參閱 Amazon EC2 定價。
> - 您可以在 Outposts 上啟動 Amazon EKS 延伸叢集中的Ubuntu節點，但無法在 AWS Outposts 的本機叢集中啟動節點 AWS 。如需詳細資訊，請參閱 AWS Outposts 上的 Amazon EKS。
> - 您可以使用 x86 或 Arm 處理器部署至 Amazon EC2 執行個體。但是，具有Inferentia晶片的執行個體可能需要先安裝 NeuronSDK。

若要Ubuntu針對 EKS 或 EKS 節點啟動，Ubuntu Pro請使用 **`eksctl`**

此程序需要 `eksctl` 版本 `0.183.0` 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的 Installation 一節。

> **Note**
>
> 此程序只適用於使用 `eksctl` 所建立的叢集。

1. 將以下內容複製到您的裝置。使用您叢集的名稱取代 `my-cluster`。此名稱僅能使用英數字元 (區分大小寫) 和連字號。必須以字母字元開頭，且長度不可超過 100 個字元。將 `ng-ubuntu` 取代為您的節點群組名稱。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。若要在執行個Arm體上部署，請`m5.large`以執行個Arm體類型取代。使用 Amazon EC2 SSH 金鑰對名稱取代 `my-ec2-keypair-name`，您可以在節點啟動後使用該金鑰對來透過 SSH 連接至節點。如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 金鑰配

對。以您自己的值取代其餘所有 *example values*。完成取代後，請執行修改後的命令以建立 ubuntu.yaml 檔案。

> **Important**
>
> 若要將節點群組部署到 AWS Outposts AWS Wavelength、或 AWS 本機區域子網路，請勿在建立叢集時通過 AWS Outposts或 AWS 本機區域子網路。 AWS Wavelength您必須在下列範例中指定子網路。如需詳細資訊，請參閱 eksctl 文件中的從組態檔案建立節點群組和組態檔案結構描述。*region-code*以叢集所 AWS 區域 在的位置取代。

```
cat >ubuntu.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.30'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-ubuntu
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Ubuntu22.04
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

若要建立Ubuntu Pro節點群組，只要將`amiFamily`值變更為即可`UbuntuPro2204`。

2. 使用下列命令部署節點。

```
eksctl create nodegroup --config-file=ubuntu.yaml
```

範例輸出如下。

建立節點時，會有數行輸出。輸出的最後幾行之一類似於以下的範例行。

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (選用) 部署範例應用程式來測試您的 Ubuntu 節點。
4. 如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：

   - 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
   - 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

   如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

自我管理的節點更新

全新 Amazon EKS 最佳化 AMI 發行時，請考量以新 AMI 取代自我管理節點群組中的節點。同樣地，如果更新 Amazon EKS 叢集的 Kubernetes 版本，請更新節點以使用具相同 Kubernetes 版本的節點。

> **Important**
>
> 本主題涵蓋自我管理節點的節點更新。如果使用 受管節點群組，請參閱 更新受管節點群組。

有兩種基本方法可以將叢集中的自我管理節點群組更新成使用新的 AMI：

遷移至新的節點群組

建立新的節點群組，並將 Pods 遷移至該群組。遷移至新節點群組比在現有 AWS CloudFormation 堆疊中更新 AMI ID 更順利。這是因為遷移過程中會將舊節點群組污染為 `NoSchedule`，並在新堆疊準備好接受現有的 Pod 工作負載之後耗盡節點。

[更新現有的自我管理節點群組]

更新現有節點群組的 AWS CloudFormation 堆疊，以使用新的 AMI。不支援對使用 `eksctl` 建立的節點群組使用這個方法。

遷移至新的節點群組

此主題會說明如何建立新的節點群組，將現有的應用程式從容遷移至新群組，接著從叢集移除舊的節點群組。您可以使用 `eksctl` 或 AWS Management Console遷移至新的節點群組。

eksctl

使用 **`eksctl`** 將應用程式遷移至新的節點群組

如需有關使用 eksctl 進行移轉的詳細資訊，請參閱文件中的非受管節點群組。`eksctl`

此程序需要 `eksctl` 版本 `0.183.0` 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的 Installation 一節。

> **Note**
>
> 此程序只適用於使用 `eksctl` 所建立的叢集和節點群組。

1. 擷取現有節點群組的名稱，使用叢集名稱取代 *`my-cluster`*。

   ```
   eksctl get nodegroups --cluster=my-cluster
   ```

   範例輸出如下。

   ```
   CLUSTER      NODEGROUP          CREATED               MIN SIZE      MAX SIZE
    DESIRED CAPACITY     INSTANCE TYPE     IMAGE ID
   default      standard-nodes   2019-05-01T22:26:58Z  1             4             3
                       t3.medium         ami-05a71d034119ffc12
   ```

2. 使用以下命令啟動具備 `eksctl` 的新節點群組。在命令中，使用您自己的值取代每一個 *`example value`*。版本編號不能比控制平面的 Kubernetes 版本還要舊。也不能為比控制平面的 Kubernetes 版本舊兩個版本以上的次要版本。建議使用與控制平面相同的版本。

   如果下列條件為真，則建議封鎖 Pod 對 IMDS 的存取：

   - 您計劃將 IAM 角色指派給您的所有 Kubernetes 服務帳戶，以便 Pods 僅具有所需的最低權限。
   - 叢集Pods中的否需要存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的其他原因，例如擷取目前 AWS 區域的執行個體。

   如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

   若要封鎖 Pod 對 IMDS 的存取權，請將 `--disable-pod-imds` 選項新增至下列命令。

   > **Note**
   >
   > 如需更多可用旗標及其描述的資訊，請參閱 https://eksctl.io/。

   ```
   eksctl create nodegroup \
     --cluster my-cluster \
     --version 1.30 \
     --name standard-nodes-new \
     --node-type t3.medium \
     --nodes 3 \
     --nodes-min 1 \
     --nodes-max 4 \
     --managed=false
   ```

3. 之前的命令完成時，使用下列命令來確認所有節點已達到 `Ready` 狀態：

   ```
   kubectl get nodes
   ```

4. 使用下列命令來刪除原始的節點群組。在命令中，將每個 *`example value`* 取代為您的叢集和節點群組名稱：

   ```
   eksctl delete nodegroup --cluster my-cluster --name standard-nodes-old
   ```

AWS Management Console and AWS CLI

使用 AWS Management Console 和將您的應用程式移轉至新節點群組 AWS CLI

1. 依照 啟動自我管理的 Amazon Linux 節點 中概述的步驟啟動新的節點群組。
2. 當堆疊已完成建立時，從主控台將其選取，然後選擇 Outputs (輸出)。
3. 記錄已建立之節點群組的「NodeInstance角色」。您需要這樣才能新增 Amazon EKS 節點至叢集。

> **Note**
>
> 如果已將任何額外的 IAM 政策連接至舊節點群組 IAM 角色，則請將相同政策連接至新節點群組 IAM 角色，以便在新群組維持該功能。例如，如果為 Kubernetes Cluster Autoscaler 新增許可，則適用此操作。

4. 更新兩個節點群組的安全群組，使其能夠互相通訊。如需詳細資訊，請參閱 Amazon EKS 安全群組與考量。

   a. 記錄兩個節點群組的安全群組 ID。這在 AWS CloudFormation 堆棧輸出中顯示為「NodeSecurity組」值。

   您可以使用下列 AWS CLI 命令從堆疊名稱取得安全群組 ID。在這些命令中 AWS CloudFormation ，`oldNodes`是舊節點堆疊的堆疊名稱，`newNodes`也是您要移轉到的堆疊名稱。使用您自己的值取代每一個 *`example value`*。

   ```
   oldNodes="old_node_CFN_stack_name"
   newNodes="new_node_CFN_stack_name"

   oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
    $oldNodes \
   --query 'StackResources[?
   ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
   --output text)
   newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
    $newNodes \
   --query 'StackResources[?
   ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
   --output text)
   ```

   b. 新增輸入規則至每個節點安全群組，使其接受彼此的流量。

下列 AWS CLI 命令會將輸入規則新增至每個安全性群組，以允許來自其他安全性群組的所有通訊協定上的所有流量。此組態可讓每個節點群組中的 Pods 在您將工作負載遷移至新群組時互相通訊。

```
aws ec2 authorize-security-group-ingress --group-id $oldSecGroup \
--source-group $newSecGroup --protocol -1
aws ec2 authorize-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

5. 編輯 aws-auth configmap 以在 RBAC 中對應新的節點執行個體角色。

```
kubectl edit configmap -n kube-system aws-auth
```

為新節點群組新增 mapRoles 項目。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov:

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: ARN of instance role (not instance profile)
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

將*ARN of instance role (not instance profile)*程式碼片段取代為您在上一個步驟中記錄的「NodeInstance角色」值。接著，儲存並關閉檔案以套用更新的 configmap。

6. 注意節點狀態並等待新的節點加入叢集和達到 Ready 狀態。

```
kubectl get nodes --watch
```

7. (選用) 如果您使用的是 Kubernetes Cluster Autoscaler，請將部署縮減為零 (0) 個複本以避免衝突的擴展動作。

```
kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
```

8. 使用以下命令污染每個您想要用 NoSchedule 移除的節點。這樣就不會在您要取代的節點上排程或重新排程新的 Pods。如需詳細資訊，請參閱 Kubernetes 文件中的污點和容差。

```
kubectl taint nodes node_name key=value:NoSchedule
```

如果要將節點升級至新的 Kubernetes 版本，您可以使用以下程式碼片段識別並標示特定 Kubernetes 版本 (在此情況下為 1.28) 的所有節點。版本編號不能比控制平面的 Kubernetes 版本還要舊。也不能為比控制平面的 Kubernetes 版本舊兩個以上的次要版本。建議使用與控制平面相同的版本。

```
K8S_VERSION=1.28
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Tainting $node"
    kubectl taint nodes $node key=value:NoSchedule
done
```

9. 判斷叢集的 DNS 供應商。

```
kubectl get deployments -l k8s-app=kube-dns -n kube-system
```

範例輸出如下。此叢集針對 DNS 解析度使用 CoreDNS，但您的叢集可能會傳回 kube-dns。

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
coredns   1         1         1            1           31m
```

10. 如果您目前的部署執行少於 2 個複本，請將部署擴增為 2 個複本。如果您先前的命令輸出傳回該項目，請以 **kubedns** 取代 *coredns*。

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

11. 使用以下命令清空您想要從叢集移除的每個節點：

```
kubectl drain node_name --ignore-daemonsets --delete-local-data
```

如果要將節點升級至新的 Kubernetes 版本，請使用以下程式碼片段識別並耗盡特定 Kubernetes 版本 (在此情況下為 *1.28*) 的所有節點。

```
K8S_VERSION=1.28
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Draining $node"
    kubectl drain $node --ignore-daemonsets --delete-local-data
done
```

12. 在舊的節點完成消耗後，請撤銷您先前授權的安全群組傳入規則。然後，刪除 AWS CloudFormation 堆疊以終止執行個體。

> **Note**
>
> 如果您將任何其他 IAM 政策附加到舊節點群組 IAM 角色，例如為Kubernetes叢集自動配置器新增許可)，請先將這些其他政策從角色中分離，然後才能刪除堆疊。 AWS CloudFormation

a. 撤銷您先前為節點安全群組建立的傳入規則。在這些命令中 AWS CloudFormation ，`oldNodes`是舊節點堆疊的堆疊名稱，`newNodes`也是您要移轉到的堆疊名稱。

```
oldNodes="old_node_CFN_stack_name"
newNodes="new_node_CFN_stack_name"

oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $oldNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $newNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
aws ec2 revoke-security-group-ingress --group-id $oldSecGroup \
```

```
--source-group $newSecGroup --protocol -1
aws ec2 revoke-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

b. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation

c. 選取舊的節點堆疊。

d. 選擇刪除。

e. 在 Delete stack (刪除堆疊) 確認對話方塊中，選擇 Delete stack (刪除堆疊)。

13. 編輯 aws-auth configmap 以從 RBAC 移除舊的節點執行個體角色。

```
kubectl edit configmap -n kube-system aws-auth
```

刪除舊節點群組的 mapRoles 項目。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov:

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-W70725MZQFF8
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-15-NodeInstanceRole-U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
```

儲存並關閉檔案以套用更新的 configmap。

14. (選用) 如果您使用的是 Kubernetes Cluster Autoscaler，請將部署縮減回 1 個複本。

> **Note**
>
> 您也必須適當標記新的 Auto Scaling 群組 (例如 k8s.io/cluster-autoscaler/enabled,k8s.io/cluster-autoscaler/*my-cluster*)，並更新 Cluster

Autoscaler 部署的命令，以指向新標記的 Auto Scaling 群組。如需詳細資訊，請參閱上的叢集自動配置器。 AWS

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

15. (選用) 確認您使用的是最新版本的 Kubernetes 專用 Amazon VPC CNI 外掛程式。您可能需要更新 CNI 版本，才能使用最新支援的執行個體類型。如需詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。
16. 如果您的叢集針對 DNS 解析度使用 kube-dns (請參閱先前步驟)，請將 kube-dns 部署縮減至 1 個複本。

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

更新現有的自我管理節點群組

本主題說明如何使用新 AMI 更新現有的 AWS CloudFormation 自我管理節點堆疊。您可以使用此程序在叢集更新後將節點更新至新版本的 Kubernetes。否則，您可以針對現有 Kubernetes 版本更新至最新的 Amazon EKS 最佳化 AMI。

Important

本主題涵蓋自我管理節點的節點更新。如需使用 受管節點群組 的詳細資訊，請參閱 更新受管節點群組。

最新的預設 Amazon EKS 節點 AWS CloudFormation 範本已設定為在您的叢集中啟動具有新 AMI 的執行個體，然後再移除舊的執行個體 (一次一個)。此組態可確保您在滾動更新期間，都能掌握叢集中 Auto Scaling 群組所需的作用中執行個體計數。

Note

不支援對使用 eksctl 建立的節點群組使用這個方法。如果透過 eksctl 建立叢集或節點群組，則請參閱 遷移至新的節點群組。

更新現有的節點群組

1. 判斷叢集的 DNS 供應商。

   ```
   kubectl get deployments -l k8s-app=kube-dns -n kube-system
   ```

   範例輸出如下。此叢集針對 DNS 解析度使用 CoreDNS，但您的叢集可能會傳回 kube-dns。根據您使用的 kubectl 版本，輸出可能會有所不同。

   ```
   NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
   coredns   1         1         1            1           31m
   ```

2. 如果您目前的部署執行少於 2 個複本，請將部署擴增為 2 個複本。如果您先前的命令輸出傳回該項目，請以 **kube-dns** 取代 *coredns*。

   ```
   kubectl scale deployments/coredns --replicas=2 -n kube-system
   ```

3. (選用) 如果您使用的是 Kubernetes Cluster Autoscaler，請將部署縮減為零 (0) 個複本以避免衝突的擴展動作。

   ```
   kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
   ```

4. 決從目前節點群組的執行個體類型和所需執行個體計數。稍後當您更新群組的 AWS CloudFormation 範本時，請輸入這些值。

   a. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。

   b. 在左側導覽窗格中，選擇 Launch Configurations (啟動組態)，並記下現有節點啟動組態的執行個體類型。

   c. 在左側導覽窗格中，選擇 Auto Scaling Groups (Auto Scaling 群組)，並注意現有節點 Auto Scaling 群組的 Desired (所需) 執行個體計數。

5. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation

6. 選取您的節點群組堆疊，然後選擇 Update (更新)。

7. 選取 Replace current template (取代目前的範本)，然後選取 Amazon S3 URL。

8. 對於 Amazon S3 URL，請將以下 URL 貼到文字區域，以確保您使用的是最新版本的節點 AWS CloudFormation 範本。然後選擇 Next (下一步)：

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
```

9. 在 Specify stack details (指定堆疊詳細資訊) 頁面上，填寫下列參數，然後選擇 Next (下一步)：

   - NodeAutoScalingGroupDesiredCapacity— 輸入您在上一個步驟中記錄的所需執行個體計數。或者，在堆疊更新時，輸入欲擴展的所需節點數量。
   - NodeAutoScalingGroupMaxSize— 輸入節點「自動調整」群組可向外延展的節點數目上限。此值必須至少超過所需容量一個節點。這樣才能夠在更新期間執行節點滾動更新，而不必減少節點計數。
   - NodeInstance類型 — 選擇您在上一步中記錄的執行個體類型。或者，為節點選擇不同的執行個體類型。選擇不同的執行個體類型之前，請先檢閱 選擇 Amazon EC2 執行個體類型。每個 Amazon EC2 執行個體類型都支援最大數量的彈性網路介面 (網路介面)，且每個網路介面支援最大數量的 IP 地址。由於為每個工作節點和 Pod 指派了自己的 IP 地址，因此請務必選擇支援要在每個 Amazon EC2 節點上執行之最大 Pods 數量的執行個體類型。如需執行個體類型支援的網路介面和 IP 地址數量清單，請參閱每種執行個體類型每個網路介面的 IP 地址。例如，`m5.large` 執行個體類型最多支援工作節點和 Pods 的 30 個 IP 地址。

     > **Note**
     >
     > 最新版本的 Amazon VPC CNI plugin for Kubernetes 支援執行個體類型會在 GitHub 上的 vpc_ip_resource_limit.go 中顯示。您可能需要更新 Amazon VPC CNI plugin for Kubernetes 版本，才能使用最新支援的執行個體類型。如需詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。

     > **Important**
     >
     > 某些執行個體類型可能並非全部可用 AWS 區域。

   - NodeImageIDSSmParam — 您要更新的目標之 AMI 識別碼的 Amazon EC2 Systems Manager 參數。以下值對 Kubernetes 版本 `1.30` 使用最新的 Amazon EKS 最佳化 AMI。

     ```
     /aws/service/eks/optimized-ami/1.30/amazon-linux-2/recommended/image_id
     ```

     可以將 *`1.30`* 取代為版本號碼相同的受支援的 Kubernetes 版本。或者應為比控制平面上執行的 Kubernetes 版本舊一個版本號碼的版本。建議您將節點保持在與控制平面相同的版本。您也可

以用不同*amazon-linux-2*的 AMI 類型替換。如需詳細資訊，請參閱 擷取 Amazon EKS 最佳化 Amazon Linux AMI ID。

> **Note**
>
> 使用 Amazon EC2 Systems Manager 參數可讓您在未來更新節點，而無需查詢和指定 AMI ID。如果 AWS CloudFormation 堆疊使用此值，任何堆疊更新一律會為您指定的 Kubernetes 版本啟動最新建議的 Amazon EKS 最佳化 AMI。即使沒有變更範本中的任何值，情況也會如此。

- NodeImageID — 若要使用您自己的自訂 AMI，請輸入要使用的 AMI 的 ID。

> **Important**
>
> 此值會取代為 NodeImageIDSS mParam 指定的任何值。如果您要使用 NodeImageIDSSmParam 值，請確定 Id 的值是空白的。NodeImage

- DisableIMDSv1：在預設情況下，每個節點都支援執行個體中繼資料服務版本 1 (IMDSv1) 和 IMDSv2。不過，您可以停用 IMDSv1。若不希望任何節點或任何節點群組中排定的 Pods 使用 IMDSv1，則請選取 true (是)。如需 IMDS 的詳細資訊，請參閱設定執行個體中繼資料服務。如果您已為服務帳戶實作 IAM 角色，請將必要的許可直接指派給所有需Pods要存取 AWS 服務的使用者。如此一來，叢集Pods中沒有任何需要存取 IMDS 的其他原因，例如擷取目前 AWS 區域的。然後，您也可以針對不使用主機聯網的 Pods 停用其對 IMDSv2 的存取。如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

10. (選用) 在 Options (選項) 頁面上，為堆疊資源加上標籤。選擇下一步。
11. 在 Review (檢閱) 頁面上檢閱您的資訊，確認該堆疊可建立 IAM 資源，然後選擇 Update stack (更新堆疊)。

> **Note**
>
> 叢集中每個節點的更新，都需要幾分鐘的時間。請等待所有節點更新完成再執行後續步驟。

12. 如果您叢集的 DNS 供應商為 `kube-dns`，請將 `kube-dns` 部署縮減至 1 個複本。

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

13. (選用) 如果您使用的是 Kubernetes Cluster Autoscaler，請將部署縮減回您想要的複本數量。

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

14. (選用) 確認您使用的是最新版本的 Amazon VPC CNI plugin for Kubernetes。您可能需要更新 Amazon VPC CNI plugin for Kubernetes 版本，才能使用最新支援的執行個體類型。如需更多詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。

AWS Fargate

> **Important**
>
> AWS Fargate Amazon EKS 在 AWS GovCloud （美國東部）和 AWS GovCloud （美國西部）中不可用。

本主題討論如何使用 Amazon EKS 在 AWS Fargate 上執行 Kubernetes Pods。Fargate 是一種為容器提供隨需、適當大小運算容量的科技。有了 Fargate，就不再需要自行佈建、設定或擴展虛擬機器的群組來執行容器。您也無需選擇伺服器類型、決定何時擴展節點群組，或最佳化叢集壓縮。

您可以控制哪些 Pods 要在 Fargate 上啟動，以及其如何搭配 Fargate 設定檔執行。Fargate 描述檔會定義為 Amazon EKS 叢集的一部分。Amazon EKS 透過使用由提供的上游可擴充模型建置的控制器，藉 AWS 此Kubernetes與 Fargate 整合。Kubernetes這些控制器會作為 Amazon EKS 受管 Kubernetes 控制平面的一部分執行，且負責將原生 Kubernetes Pods 排程至 Fargate。Fargate 控制器包括新的排程器，除了數個變換與驗證許可控制器之外，也會隨著預設 Kubernetes 排程器執行。當您啟動符合在 Fargate 上執行之條件的 Pod 時，在叢集中執行的 Fargate 控制器便會辨識、更新及將 Pod 排程至 Fargate。

本主題說明在 Fargate 上執行的不同 Pods 元件，並注意搭配使用 Fargate 與 Amazon EKS 的特殊考慮因素。

AWS Fargate 考量

以下是在 Amazon EKS 上使用 Fargate 的考慮因素。

- 在 Fargate 上執行的每個 Pod 都有自己的隔離界限。這些 Pod 不會與其他 Pod 共用基礎核心、CPU 資源、記憶體資源或彈性網路介面。

- Network Load Balancer 和 Application Load Balancer (ALB) 可以僅與具有 IP 目標的 Fargate 搭配使用。如需詳細資訊，請參閱 建立 Network Load Balancer 和 Amazon EKS 上的應用程式負載平衡。
- Fargate 的公開服務僅能在目標類型 IP 模式下執行，不能在節點 IP 模式下執行。若要檢查在受管節點上和在 Fargate 上執行的服務的連線狀態，建議透過服務名稱來連線。
- Pod 必須符合其排程時的 Fargate 描述檔，以在 Fargate 上執行。與 Fargate 描述檔不符的 Pod 可能會卡在 `Pending` 狀態。若具有符合的 Fargate 設定檔，您可刪除已建立的待定 Pods，以將其重新排程至 Fargate。
- Fargate 不支援 Daemonset。若您的應用程式需要常駐程式，請重新設定該常駐程式，以作為您 Pods 中的附屬容器來執行。
- Fargate 不支援具有特殊權限的容器。
- 在 Fargate 上執行的 Pod 不能指定 Pod 清單檔案中的 `HostPort` 或 `HostNetwork`。
- 預設 `nofile` 和 `nproc` 軟性限制為 1024，Fargate Pods 的硬性限制為 65535。
- GPU 目前無法在 Fargate 上使用。
- 在 Fargate 上執行的網繭僅支援私有子網路 (具有 AWS 服務的 NAT 閘道存取權，而不是直接路由至網 Internet Gateway)，因此叢集的 VPC 必須具有可用的私人子網路。如需沒有傳出網際網路存取權的叢集，請參閱 私有叢集要求。
- 您可以使用 Vertical Pod Autoscaler 為 Fargate Pods 設定 CPU 和記憶體的初始正確大小，然後使用 Horizontal Pod Autoscaler 來擴展這些 Pods。若要讓 Vertical Pod Autoscaler 自動將 Pods 重新部署至具有較大 CPU 和記憶體組合的 Fargate，請將 Vertical Pod Autoscaler 的模式設定為 `Auto` 或 `Recreate` 以確保正確的功能。如需詳細資訊，請參閱 GitHub 上的 Vertical Pod Autoscaler 文件。
- 您的 VPC 必須啟用 DNS 解析和 DNS 主機名稱。如需詳細資訊，請參閱檢視與更新 VPC 的 DNS 支援。
- Amazon EKS Fargate 透過隔離虛擬機器 (VM) 內的每個網繭， defense-in-depth 為Kubernetes應用程式新增。此 VM 界限可防止在容器逸出時存取其他 Pod 使用的主機型資源，這是攻擊容器化應用程式並存取容器外資源的常用方法。

  使用 Amazon EKS 並不會變更您在共同責任模型下所需承擔的責任。您應該審慎考慮叢集安全性和控管控制項的組態。隔離應用程式最安全的方法是永遠在個別的叢集中執行。
- Fargate 描述檔支援從 VPC 次要 CIDR 區塊指定子網路。您可能想要指定次要 CIDR 區塊。這是因為子網路僅提供數量有限的 IP 地址。因此，叢集中可建立的 Pods 數量也有限。若為 Pods 使用不同的子網路，您可增加可用 IP 地址的數量。如需詳細資訊，請參閱為 VPC 新增 IPv4 CIDR 區塊。

- 部署至 Fargate 節點的 Pods 不提供 Amazon EC2 執行個體中繼資料服務 (IMDS)。若有部署至 Fargate 且需要 IAM 憑證的 Pods，請使用 服務帳戶的 IAM 角色 將其指派至 Pods。若您的 Pods 需要存取 IMDS 提供的其他資訊，則必須將此資訊硬式編碼至 Pod 規格中。這包括部署到的 AWS 區域 或可用區域。Pod
- 您無法將 Fargate 部署Pods到 AWS Outposts AWS Wavelength、或 L AWS ocal Zones。
- Amazon EKS 必須定期修補 Fargate Pods 來確保其安全。我們嘗試以減少影響的方式進行更新，但有時，如果沒有成功移出 Pods，則必須將其刪除。您可採取一些措施以盡可能地減少中斷。如需詳細資訊，請參閱 Fargate OS 修補。
- 適用於 Amazon EKS 的 Amazon VPC CNI 外掛程式會安裝在 Fargate 叢集上。您無法搭配使用 替代相容的 CNI 外掛程式 與 Fargate 節點。
- 在 Fargate 上執行的 Pod 會自動掛載 Amazon EFS 檔案系統。您不能將動態持續性磁碟區佈建與 Fargate 節點搭配使用，但可以使用靜態佈建。
- 您無法將 Amazon EBS 磁碟區掛載到 Fargate Pods。
- 您可以在 Fargate 節點上執行 Amazon EBS CSI 控制器，但是 Amazon EBS CSI 節點 DaemonSet 僅能在 Amazon EC2 執行個體上執行。
- 在 Kubernetes Job 標記了 `Completed` 或 `Failed` 之後，Job 建立的 Pods 通常會繼續存在。此行為允許您檢視日誌和結果，但使用 Fargate，如果您之後不清理 Job，將產生費用。

  若要在Job完成或失敗Pods後自動刪除相關內容，您可以使用 time-to-live (TTL) 控制器指定時段。下列範例顯示在您的 Job 清單檔案中指定 `.spec.ttlSecondsAfterFinished`。

```
apiVersion: batch/v1
kind: Job
metadata:
  name: busybox
spec:
  template:
    spec:
      containers:
      - name: busybox
        image: busybox
        command: ["/bin/sh", "-c", "sleep 10"]
      restartPolicy: Never
  ttlSecondsAfterFinished: 60 # <-- TTL controller
```

開始 AWS Fargate 使用 Amazon EKS

> **Important**
>
> AWS Fargate Amazon EKS 在 AWS GovCloud （美國東部）和 AWS GovCloud （美國西部）中不可用。

本主題說明如何開始 AWS Fargate 使用 Amazon EKS 叢集執行Pods。

如果使用 CIDR 區塊限制對叢集公有端點的存取，則建議您也啟用私有端點存取。這樣一來，Fargate Pods 即可與叢集通訊。若不啟用私有端點，您指定用於公有存取的 CIDR 區塊必須包含來自 VPC 的傳出來源。如需詳細資訊，請參閱 Amazon EKS 叢集端點存取控制。

先決條件

現有的叢集。如果還沒有 Amazon EKS 叢集，請參閱 Amazon EKS 入門。

確認現有節點可以與 Fargate Pods 進行通訊

如果正在使用無節點的新叢集，或叢集只有受管節點群組，您可以跳至 建立 Fargate Pod 執行角色。

假設您正在使用的現有叢集已經具有與其相關聯的節點。確保這些節點上的 Pods 可以與 Fargate 上執行的 Pods 自由通訊。在 Fargate 上執行的 Pods 會自動設定為使用相關聯叢集的叢集安全群組。請務必確認叢集中的任何現有節點可以傳送與接收叢集安全群組的流量。受管節點群組 也會自動設為使用叢集安全群組，因此您不需要為此相容性修改或檢查。

對於使用`eksctl`或 Amazon EKS 受管 AWS CloudFormation 範本建立的現有節點群組，您可以手動將叢集安全群組新增至節點。您還可以修改節點群組的 Auto Scaling 群組啟動範本，從而將叢集安全群組連接至執行個體。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的變更執行個體的安全群組。

您可以在叢集的 [網路] 區段 AWS Management Console 下方檢查叢集的安全性群組。或者，您可以使用以下 AWS CLI 命令執行此操作。使用此命令時，請以您叢集的名稱取代 *my-cluster*。

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

建立 Fargate Pod 執行角色

當您的叢集建立時 AWS Fargate，Pods在 Fargate 基礎結構上執行的元件必須代表您呼叫 AWS API。Amazon EKS Pod 執行角色提供進行此類工作的 IAM 許可。若要建立 AWS Fargate Pod執行角色，請參閱Amazon EKS Pod 執行 IAM 角色。

> **Note**
>
> 若您使用 `--fargate` 選項以 `eksctl` 建立叢集，則您的叢集已擁有 Pod 執行角色，您可以在 IAM 主控台中使用模式 `eksctl-`*`my-cluster`*`-FargatePodExecutionRole-`*`ABCDEFGHIJKL`* 找到該角色。同樣地，若您使用 `eksctl` 建立 Fargate 設定檔，`eksctl` 會建立 Pod 執行角色 (如果該角色尚未建立)。

為您的叢集建立 Fargate 描述檔

您必須定義 Fargate 描述檔，指定哪些 Pods 在啟動時使用 Fargate，然後才能排程在叢集 Fargate 上執行的 Pods。如需詳細資訊，請參閱 AWS Fargate 設定檔。

> **Note**
>
> 若您使用 `--fargate` 選項以 `eksctl` 建立叢集，則您叢集的 Fargate 描述檔已建立完成，並包含在 `kube-system` 和 `default` 命名空間中所有 Pods 的選擇器。使用下列程序來為您想要搭配 Fargate 使用的任何其他命名空間建立 Fargate 描述檔。

您可以使用 `eksctl` 或 AWS Management Console建立 Fargate 描述檔。

eksctl

此程序需要 `eksctl` 版本 `0.183.0` 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的 Installation 一節。

使用 **`eksctl`** 建立 Fargate 設定檔

使用以下 `eksctl` 命令建立您的 Fargate 設定檔，並以您自己的值取代每一個 *`example value`*。必須指定一個命名空間。不過，`--labels` 選項並非必要項目。

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

對於 *my-kubernetes-namespace* 和 *key=value* 標籤可以使用某些萬用字元。如需詳細資訊，請參閱 Fargate 設定檔萬用字元。

AWS Management Console

Fargate 使用 AWS Management Console

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 選擇要為其建立 Fargate 設定檔的叢集。
3. 選擇 Compute (運算) 索引標籤。
4. 在 Fargate profiles (Fargate 設定檔) 下，選擇 Add Fargate profile (新增 Fargate 設定檔)。
5. 在 Configure Fargate profile (設定 Fargate 設定檔) 頁面上，執行以下操作：
   a. 對於 Name (名稱)，輸入您 Fargate 描述檔的名稱。名稱必須是唯一的。
   b. 針對 Pod execution role (Pod 執行角色)，請選擇要搭配 Fargate 設定檔使用的 Pod 執行角色。只會顯示具有 `eks-fargate-pods.amazonaws.com` 服務主體的 IAM 角色。如果未列出任何角色，您必須建立一個角色。如需詳細資訊，請參閱 Amazon EKS Pod 執行 IAM 角色。
   c. 視需要修改選取的子網路。

      > **Note**
      > 只有私有子網路支援在 Fargate 上執行的 Pods。

   d. 對於 Tags (標籤)，您可以選擇標記 Fargate 設定檔。這些標籤不會傳播至與描述檔相關聯的其他資源，例如 Pods。
   e. 選擇下一步。
6. 在 Configure Pod selection (設定 選擇) 頁面上，執行以下操作：
   a. 針對 Namespace (命名空間)，輸入符合 Pods 的命名空間。
      - 您可以使用特定的命名空間進行比對，例如 **kube-system** 或 **default**。

- 您可以使用某些萬用字元 (例如 **prod-***) 來比對多個命名空間 (例如 prod-deployment 和 prod-test)。如需詳細資訊，請參閱 Fargate 設定檔萬用字元。

b. (選用) 將 Kubernetes 標籤新增至選取器。將其特別新增至需與特定命名空間中的 Pods 相符的選取器。

- 您可以將標籤 **infrastructure: fargate** 新增至選取器，因此只有在也有 infrastructure: fargate Kubernetes 標籤之指定命名空間中的 Pods 符合選取器。
- 您可以使用某些萬用字元 (例如 **key?: value?**) 來比對多個命名空間 (例如 keya: valuea 和 keyb: valueb)。如需詳細資訊，請參閱 Fargate 設定檔萬用字元。

c. 選擇下一步。

7. 在 Review and create (檢閱和建立) 頁面，檢閱您 Fargate 設定檔的資訊並選擇 Create (建立)。

更新 CoreDNS

CoreDNS 預設為在 Amazon EKS 叢集上的 Amazon EC2 基礎設施上執行。若您希望僅在叢集的 Fargate 上執行 Pods，請完成以下步驟。

> **Note**
>
> 如果透過 --fargate 選項使用 eksctl 建立叢集，您可以直接跳至 後續步驟。

1. 使用下列命令來刪除 CoreDNS 的 Fargate 設定檔。使用您的叢集名稱取代 *my-cluster*、使用您的帳戶 ID 取代 *111122223333*、使用您的 Pod 執行角色名稱取代 *AmazonEKSFargatePodExecutionRole*，並以私有子網路的 ID 取代 *0000000000000001*、*0000000000000002* 和 *0000000000000003*。若沒有 Pod 執行角色，則必須先建立一個。

> **Important**
>
> 角色 ARN 不能包含 / 之外的路徑。例如，如果您的角色名稱是 development/apps/my-role，則您必須在指定角色的 ARN 時將其變更為 my-role。角色 ARN 的格式必須是 arn:aws:iam::*111122223333*:role/*role-name*。

```
aws eks create-fargate-profile \
    --fargate-profile-name coredns \
    --cluster-name my-cluster \
    --pod-execution-role-arn
 arn:aws:iam::111122223333:role/AmazonEKSFargatePodExecutionRole \
    --selectors namespace=kube-system,labels={k8s-app=kube-dns} \
    --subnets subnet-0000000000000001 subnet-0000000000000002
 subnet-0000000000000003
```

2. 執行以下命令，從 CoreDNS Pods 移除 `eks.amazonaws.com/compute-type : ec2` 註釋。

```
kubectl patch deployment coredns \
    -n kube-system \
    --type json \
    -p='[{"op": "remove", "path": "/spec/template/metadata/annotations/
eks.amazonaws.com~1compute-type"}]'
```

後續步驟

- 您可以透過以下工作流程開始遷移現有的應用程式，以在 Fargate 上執行。
  1. 建立符合您應用程式的 Kubernetes 命名空間和 Kubernetes 標籤的 Fargate 描述檔。
  2. 刪除並重新建立任何現有 Pods，以便在 Fargate 上對其進行排程。例如，以下命令會觸發 `coredns` 部署的推展。您可以修改命名空間和部署類型，以更新您的特定 Pods。

     ```
     kubectl rollout restart -n kube-system deployment coredns
     ```

- 部署 Amazon EKS 上的應用程式負載平衡 以允許您在 Fargate 執行的 Pods 輸入物件。
- 您可以使用 Vertical Pod Autoscaler 為 Fargate Pods 設定 CPU 和記憶體的初始正確大小，然後使用 Horizontal Pod Autoscaler 來擴展這些 Pods。若要讓 Vertical Pod Autoscaler 自動將 Pods 重新部署到具有較高 CPU 和記憶體組合的 Fargate，請將 Vertical Pod Autoscaler 的模式設定為 `Auto` 或 `Recreate`。這是為了確認功能可以運作正確。如需詳細資訊，請參閱 GitHub 上的 Vertical Pod Autoscaler 文件。
- 請遵循以下說明設定 AWS Distro for OpenTelemetry (ADOT) 收集器，以便監控應用程式。

AWS Fargate 設定檔

> **Important**
>
> AWS Fargate Amazon EKS 在 AWS GovCloud （美國東部）和 AWS GovCloud （美國西部）中不可用。

您必須定義至少一個 Fargate 設定檔，指定哪些 Pods 在啟動時使用 Fargate，然後才能排程在叢集 Fargate 上的 Pods。

管理員可以使用 Fargate 設定檔宣告哪些 Pods 要在 Fargate 上執行。您可以透過設定檔的選取器來執行此操作。您最多可以將五個選取器新增至每個設定檔。每個選取器必須包含一個命名空間。選取器還可以包括標籤。標籤欄位由多個選用鍵/值對組成。與選取器相符的 Pod 會排程在 Fargate 上執行。比對 Pod 時，會使用命名空間和在選取器中指定的標籤。如果命名空間選取器未以標籤定義，Amazon EKS 會嘗試使用設定檔，將以該命名空間執行的所有 Pods 排程至 Fargate。如果與 Fargate to-be-scheduled Pod 配置文件中的任何選擇器匹配，則該Pod選擇器在 Fargate 上進行計劃。

如果 Pod 符合多個 Fargate 設定檔，您可以將以下 Pod 標籤新增至 Kubernetes 規格，以指定 Pod 使用哪個設定檔：`eks.amazonaws.com/fargate-profile: my-fargate-profile`。不過，Pod 必須符合該設定檔中的選取器，才能排程至 Fargate。Kubernetes 親和性/反親和性規則並不適用，對 Amazon EKS Fargate Pods 而言也非必要。

建立 Fargate 設定檔時，您必須指定 Pod 執行角色。此執行角色適用於使用設定檔在 Fargate 基礎設施上執行的 Amazon EKS 元件。此角色會新增至叢集的 Kubernetes 角色型存取控制 (RBAC)，以進行授權。這樣一來，在 Fargate 基礎設施上執行的 `kubelet` 就可以註冊到您的 Amazon EKS 叢集，並作為節點出現在您的叢集中。Pod 執行角色也為 Fargate 基礎設施提供 IAM 許可，以允許對 Amazon ECR 映像儲存庫的讀取存取權。如需詳細資訊，請參閱 Amazon EKS Pod 執行 IAM 角色。

Fargate 設定檔無法變更。不過，您可以建立新的已更新設定檔來取代現有設定檔，然後刪除原始的設定檔。

> **Note**
>
> 刪除設定檔時，使用 Fargate 設定檔執行的任何 Pods 將停止並進入待定狀態。

如果在叢集中的任何 Fargate 設定檔處於 `DELETING` 狀態，您必須等候該 Fargate 設定檔刪除後，才能在該叢集建立任何其他設定檔。

Amazon EKS 和 Fargate 會嘗試在 Fargate 設定檔所定義的每個子網路中散佈 Pods。不過，最終可能會產生不均勻的散佈結果。如果必須產生均勻的散佈結果，請使用兩個 Fargate 設定檔。在您想要部署兩個副本且不希望任何停機時間的情況下，均勻的散佈結果相當重要。我們建議每個設定檔只定義一個子網路。

Fargate 描述檔元件

下列元件包含在 Fargate 描述檔中。

Pod 執行角色

當您的叢集建立時 AWS Fargate`kubelet`，Pods在 Fargate 基礎結構上執行的叢集必須代表您呼叫 AWS API。例如，它必須透過呼叫從 Amazon ECR 提取容器映像。Amazon EKS Pod 執行角色提供進行此類工作的 IAM 許可。

建立 Fargate 設定檔時，您必須指定要搭配 Pods 使用的 Pod 執行角色。此角色會新增至叢集的 Kubernetes 角色型存取控制 (RBAC)，以進行授權。這是為了讓在 Fargate 基礎設施上執行的 `kubelet` 可以註冊到您的 Amazon EKS 叢集並作為節點出現在您的叢集中。如需詳細資訊，請參閱 Amazon EKS Pod 執行 IAM 角色。

子網

要在其中啟動 Pods 並使用該設定檔的子網路 ID。目前不會對在 Fargate 上執行的 Pods 指派公用 IP 地址。因此，此參數僅接受私有子網路 (不具網際網路閘道的直接路由)。

選取器

要與 Pods 比對以使用此 Fargate 設定檔的選擇器。您最多可在一個 Fargate 設定檔中指定五個選取器。選取器具有下列元件：

- 命名空間：您必須指定選取器的命名空間。選取器只會比對在此命名空間中建立的 Pods。不過，您可以建立多個選取器來定義多個命名空間。
- 標籤：您可以選擇指定符合選取器的 Kubernetes 標籤。選取器僅符合擁有在選取器中指定之所有標籤的 Pods。

Fargate 設定檔萬用字元

除了 Kubernetes 允許的字元之外，您還可以在命名空間、標籤索引鍵和標籤值的選取器條件中使用 ***** 和 **?**：

- ***** 代表無、一個或多個字元。例如，**`prod*`** 可以代表 `prod` 和 `prod-metrics`。

- **?** 代表單一字元 (例如，**value?** 可以代表 valuea)。不過，它不能代表 value 和 value-a，因為 ? 只能代表確切一個字元。

這些萬用字元適用於任何位置和組合 (例如，**prod***、***dev**，以及 **frontend*?**)。其他萬用字元和模式比對形式 (例如規則運算式) 則不支援。

如果 Pod 規格中的命名空間和標籤有多個相符的設定檔，Fargate 會根據設定檔名稱的英數字元排序選擇設定檔。例如，如果設定檔 A (名稱為 beta-workload) 和設定檔 B (名稱為 prod-workload) 都具有要啟動的 Pods 的相符選取器，則 Fargate 將為 Pods 選擇設定檔 A (beta-workload)。Pods 在 Pods 上具有含設定檔 A 的標籤 (例如 eks.amazonaws.com/fargate-profile=beta-workload)。

如果您想要將現有的 Fargate Pods 遷移至使用萬用字元的新設定檔，有兩種方法：

- 使用相符的選取器建立新的設定檔，然後刪除舊設定檔。標有舊設定檔的 Pod 會重新排程為新的相符設定檔。
- 如果您想要遷移工作負載，但不確定每個 Fargate Pod 上有哪些 Fargate 標籤，則可使用以下方法。建立新的設定檔，其名稱在同一叢集上的設定檔中按英數字元順序排列在第一位。然後，回收需要遷移到新設定檔的 Fargate Pods。

建立 Fargate 設定檔

本主題說明如何建立 Fargate 設定檔。您還必須建立用於 Fargate 設定檔的 Pod 執行角色。如需詳細資訊，請參閱Amazon EKS Pod 執行 IAM 角色。 Pods只有在具有 NAT 閘道存取權的私有子網路上才支援在 Fargate 上執行 AWS 服務，但不支援直接路由到網 Internet Gateway。因此，您叢集的 VPC 必須具有可用的私有子網路。您可以使用 eksctl 或 AWS Management Console建立描述檔。

此程序需要 eksctl 版本 0.183.0 或更新版本。您可使用以下命令檢查您的版本：

```
eksctl version
```

如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的 Installation 一節。

eksctl

使用 **eksctl** 建立 Fargate 設定檔

使用以下 eksctl 命令建立您的 Fargate 設定檔，並以您自己的值取代每一個 *example value*。必須指定一個命名空間。不過，--labels 選項並非必要項目。

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

對於 *my-kubernetes-namespace* 和 *key=value* 標籤可以使用某些萬用字元。如需詳細資訊，請參閱 Fargate 設定檔萬用字元。

AWS Management Console

Fargate 使用 AWS Management Console

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 選擇要為其建立 Fargate 設定檔的叢集。
3. 選擇 Compute (運算) 索引標籤。
4. 在 Fargate profiles (Fargate 設定檔) 下，選擇 Add Fargate profile (新增 Fargate 設定檔)。
5. 在 Configure Fargate profile (設定 Fargate 設定檔) 頁面上，執行以下作業：
   a. 對於 Name (名稱)，輸入 Fargate 設定檔的唯一名稱，例如 ***my-profile***。
   b. 針對 Pod execution role (Pod 執行角色)，請選擇要搭配 Fargate 設定檔使用的 Pod 執行角色。只會顯示具有 `eks-fargate-pods.amazonaws.com` 服務主體的 IAM 角色。如果未列出任何角色，您必須建立一個角色。如需詳細資訊，請參閱 Amazon EKS Pod 執行 IAM 角色。
   c. 視需要修改選取的子網路。

      > **Note**
      > 只有私有子網路支援在 Fargate 上執行的 Pods。

   d. 對於 Tags (標籤)，您可以選擇標記 Fargate 設定檔。這些標籤不會傳播至與設定檔相關聯的其他資源，例如 Pods。
   e. 選擇下一步。
6. 在 Configure Pod selection (設定 選擇) 頁面上，執行以下操作：
   a. 針對 Namespace (命名空間)，輸入符合 Pods 的命名空間。
      - 您可以使用特定的命名空間進行比對，例如 **kube-system** 或 **default**。

- 您可以使用某些萬用字元 (例如 **`prod-*`**) 來比對多個命名空間 (例如 `prod-deployment` 和 `prod-test`)。如需詳細資訊，請參閱 Fargate 設定檔萬用字元。

b. (選用) 將 Kubernetes 標籤新增至選取器。將其特別新增至需與特定命名空間中的 Pods 相符的選取器。

- 您可以將標籤 **`infrastructure: fargate`** 新增至選取器，因此只有在也有 `infrastructure: fargate` Kubernetes 標籤之指定命名空間中的 Pods 符合選取器。
- 您可以使用某些萬用字元 (例如 **`key?: value?`**) 來比對多個命名空間 (例如 `keya: valuea` 和 `keyb: valueb`)。如需詳細資訊，請參閱 Fargate 設定檔萬用字元。

c. 選擇下一步。

7. 在 Review and create (檢閱和建立) 頁面，檢閱您 Fargate 設定檔的資訊並選擇 Create (建立)。

刪除 Fargate 描述檔

本主題說明如何刪除 Fargate 設定檔。

當您刪除 Fargate 設定檔時，使用設定檔排程至 Fargate 的任何 Pods 皆會遭到刪除。若這些 Pods 符合其他 Fargate 設定檔，其便會使用該設定檔在 Fargate 上排程。如果其不再符合任何 Fargate 設定檔，則不會排程至 Fargate，且可能仍為待定狀態。

一個叢集中一次只能有一個 Fargate 描述檔可以處於 `DELETING` 狀態。請等候 Fargate 描述檔完成刪除動作，才可以刪除該叢集中的任何其他描述檔。

您可以使用`eksctl`、或刪除 AWS Management Console設定檔 AWS CLI。選取含有要用來刪除描述檔之工具名稱的索引標籤。

eksctl

使用 **`eksctl`** 刪除 Fargate 設定檔

使用下列命令從叢集刪除設定檔。使用您自己的值取代每一個 *`example value`*。

```
eksctl delete fargateprofile  --name my-profile --cluster my-cluster
```

AWS Management Console

若要從叢集中刪除 Fargate 設定檔，請使用 AWS Management Console

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。在叢集清單中，選擇您要從中刪除 Fargate 設定檔的叢集。
3. 選擇 Compute (運算) 索引標籤。
4. 選擇要刪除的 Fargate 設定檔，然後選擇 Delete (刪除)。
5. 在 Delete Fargate profile (刪除 Fargate 設定檔) 頁面上，輸入設定檔名稱，然後選擇 Delete (刪除)。

AWS CLI

使用 AWS CLI刪除 Fargate 設定檔

使用下列命令從叢集刪除設定檔。使用您自己的值取代每一個 *example value*。

```
aws eks delete-fargate-profile --fargate-profile-name my-profile --cluster-name my-cluster
```

Fargate Pod 組態

> **Important**
>
> AWS FargateAmazon EKS 在 AWS GovCloud （美國東部）和 AWS GovCloud （美國西部）中不可用。

本節會描述部分適用於在 AWS Fargate 上執行 Kubernetes Pods 的專用 Pod 組態詳細資訊。

Pod CPU 和記憶體

運用 Kubernetes，您可以定義請求、最小 vCPU 數量，以及分配給 Pod 中每個容器的記憶體資源。Kubernetes 會排程 Pods，藉此確保為每個 Pod 所請求的資源均可在計算資源上使用。如需詳細資訊，請參閱 Kubernetes 文件中的管理容器的運算資源。

> **Note**
>
> 由於 Amazon EKS Fargate 的每個節點僅會執行一個 Pod，因此不會發生在資源較少的情況下將 Pods 移出的情況。所有 Amazon EKS Fargate Pods 都是以保證優先順序的方式執行，因此請求的 CPU 和記憶體必須等同於所有容器的限制。如需詳細資訊，請參閱 Kubernetes 文件中的設定 Pods 的服務品質。

當 Pods 在 Fargate 上排程時，在 Pod 規格內的 vCPU 與記憶體保留決定了多少 CPU 和記憶體會佈建給 Pod。

- 任何 Init 容器的最大請求用於判斷 Init 請求 vCPU 和記憶體需求。
- 所有長期執行容器的請求會加總，以判斷長期執行請求 vCPU 和記憶體需求。
- 先前兩個值中較大的值會選擇作為用於您 Pod 的 vCPU 和記憶體請求。
- Fargate 會將 256 MB 新增至所需 Kubernetes 元件之每個 Pod 的記憶體保留 (`kubelet`、`kube-proxy` 和 `containerd`)。

Fargate 會四捨五入至以下最接近 vCPU 和記憶體請求總和的運算組態，以確保 Pods 隨時擁有其執行所需的資源。

若您未指定 vCPU 和記憶體的組合，則會使用最小可用的組合 (.25 vCPU 和 0.5 GB 記憶體)。

下表顯示可用於在 Fargate 上執行之 Pods 的 vCPU 和記憶體組合。

vCPU 數值	記憶體數值
.25 vCPU	0.5 GB、1 GB、2 GB
.5 vCPU	1 GB、2 GB、3 GB、4 GB
1 vCPU	2 GB、3 GB、4 GB、5 GB、6 GB、7 GB、8 GB
2 vCPU	介於 4 GB 與 16 GB 之間，以 1 GB 為單位遞增
4 vCPU	介於 8 GB 與 30 GB 之間，以 1 GB 為單位遞增

vCPU 數值	記憶體數值
8 vCPU	介於 16 GB 與 60 GB 之間，以 4 GB 為單位遞增
16 vCPU	介於 32 GB 與 120 GB 之間，以 8 GB 為單位遞增

為 Kubernetes 元件預訂的額外記憶體，可能會導致 Fargate 任務佈建了超出原先請求數量的 vCPU。例如，對 1 個 vCPU 和 8 GB 記憶體的請求將會新增 256 MB 至其記憶體請求，並且會佈建具有 2 個 vCPU 和 9 GB 記憶體的 Fargate 任務，因為沒有任何具有 1 個 vCPU 和 9 GB 記憶體的可用任務。

在 Fargate 上執行的 Pod 大小，與 Kubernetes 使用 `kubectl get nodes` 報告的節點大小之間沒有相關性。報告的節點大小通常大於 Pod 的容量。您可以使用下列命令確認 Pod 容量。以您 Pod 的命名空間來取代 *`default`*，並以您 Pod 的名稱來取代 *`pod-name`*。

```
kubectl describe pod --namespace default pod-name
```

範例輸出如下。

```
[...]
annotations:
    CapacityProvisioned: 0.25vCPU 0.5GB
[...]
```

`CapacityProvisioned` 註釋代表強制執行的 Pod 容量，並決定在 Fargate 上執行 Pod 的成本。如需運算組態的定價資訊，請參閱 AWS Fargate 定價。

Fargate 儲存

在 Fargate 上執行的 Pod 會自動掛載 Amazon EFS 檔案系統。您不能將動態持續性磁碟區佈建與 Fargate 節點搭配使用，但可以使用靜態佈建。如需詳細資訊，請參閱上的 Amazon EFS CSI 驅動程式 GitHub。

佈建時，在 Fargate 上執行的每個 Pod 均會收到預設 20 GiB 的暫時性儲存。此類型的儲存會在 Pod 停止後刪除。在 Fargate 上啟動的新 Pods，預設皆會啟用暫時性儲存磁碟區的加密。暫時性 Pod 儲存會以使用 AWS Fargate 受管金鑰的 AES-256 加密演算法來加密。

> **Note**
>
> 在 Fargate 上執行的 Amazon EKS Pods 預設可用儲存空間少於 20 GiB。這是因為部分空間由 Pod 內部載入的 `kubelet` 和其他 Kubernetes 模組使用。

您可以增加暫時性儲存的總量，最多可達 175 GiB。若要使用 Kubernetes 設定大小，請將 `ephemeral-storage` 資源的請求指定到 Pod 中的每個容器。當 Kubernetes 排程了 Pods，它可以確保每個 Pod 的資源請求總計少於 Fargate 任務的容量。如需詳細資訊，請參閱 Kubernetes 文件中的適用於 Pods 和容器的資源管理。

Amazon EKS Fargate 佈建的暫時性儲存空間比系統使用目的之要求更多。例如，100 GiB 的請求將佈建具有 115 GiB 暫時性儲存的 Fargate 任務。

Fargate OS 修補

> **Important**
>
> AWS FargateAmazon EKS 在 AWS GovCloud （美國東部）和 AWS GovCloud （美國西部）中不可用。

Amazon EKS 會定期修補 AWS Fargate 節點的 OS 來確保節點的安全。作為修補程序的一部分，我們會回收節點以安裝作業系統修補程式。以對您的服務產生最小影響的方式嘗試更新。但是，如果未成功移出 Pods，有時必須將其刪除。您可以採取以下措施，以最大限度地減少潛在的中斷：

- 設定適當的 Pod 中斷預算 (PDB)，以控制同時關閉的 Pods 數量。
- 建立 Amazon EventBridge 規則以在刪除失敗的驅逐之前處Pods理。
- 在 AWS 使用者通知中建立通知組態。

Amazon EKS 與 Kubernetes 社群密切合作，以盡快提供錯誤修復和安全修補程式。所有 Fargate Pods 均從最新的 Kubernetes 修補程式版本開始，該修補程式版本可從 Amazon EKS 取得，適用於您叢集的 Kubernetes 版本。若您的 Pod 具有較舊的修補程式版本，Amazon EKS 可能會予以回收來將其更新到最新版本。這可確保您的 Pods 已配備最新的安全更新。這樣，如果有一個關鍵常見漏洞和風險 (CVE) 問題，您可隨時了解最新資訊，以降低安全風險。

若要限制在回收 Pods 時同時關閉的 Pods 數量，可以設定 Pod 中斷預算 (PDB)。您可以使用 PDB 根據每個應用程式的要求定義最低可用性，同時仍允許進行更新。如需詳細資訊，請參閱 Kubernetes 文件中的 Specifying a Disruption Budget for your Application (為應用程式指定中斷預算)。

Amazon EKS 使用移出 API 以安全地消耗 Pod，同時顧及您為應用程式設定的 PDB。可用區域將 Pod 移出，以最大限度地減少影響。如果移出成功，新 Pod 將會取得最新的修補程序，且無須採取進一步的動作。

當 Pod 移出失敗時，Amazon EKS 會向您的帳戶傳送一個事件，其中包含有關移出失敗的 Pods 詳細資訊。您可以在計劃終止時間之前對訊息執行操作。具體時間根據修補程式的緊迫性而有所不同。到了該時間時，Amazon EKS 會再次嘗試移出 Pods。但是，這次如果移出失敗，則不會傳送新事件。如果移出再次失敗，則會定期刪除現有的 Pods，以便新 Pods 具有最新的修補程式。

以下是當 Pod 移出失敗時收到的範例事件。包含有關叢集、Pod 名稱、Pod 名稱空間、Fargate 設定檔和計劃終止時間的詳細資訊。

```
{
    "version": "0",
    "id": "12345678-90ab-cdef-0123-4567890abcde",
    "detail-type": "EKS Fargate Pod Scheduled Termination",
    "source": "aws.eks",
    "account": "111122223333",
    "time": "2021-06-27T12:52:44Z",
    "region": "region-code",
    "resources": [
        "default/my-database-deployment"
    ],
    "detail": {
        "clusterName": "my-cluster",
        "fargateProfileName": "my-fargate-profile",
        "podName": "my-pod-name",
        "podNamespace": "default",
        "evictErrorMessage": "Cannot evict pod as it would violate the pod's disruption
 budget",
        "scheduledTerminationTime": "2021-06-30T12:52:44.832Z[UTC]"
    }
}
```

此外，將多個 PDB 與 Pod 關聯可能會導致移出失敗事件。此事件傳回以下錯誤訊息。

```
"evictErrorMessage": "This pod has multiple PodDisruptionBudget, which the eviction
 subresource does not support",
```

您可以根據此事件建立所需的動作。例如，您可以調整 Pod 中斷預算 (PDB)，以控制如何移出 Pods。更具體地說，假設您從指定可用 Pods 目標百分比的 PDB 開始。在升級期間強制終止 Pods 之前，您可以將 PDB 調整為不同百分比的 Pods。若要接收此事件，您必須在叢集所屬AWS 帳戶AWS 區域且中建立 Amazon EventBridge 規則。規則必須使用以下自訂模式。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的建立可回應事件的 Amazon EventBridge 規則。

```
{
  "source": ["aws.eks"],
  "detail-type": ["EKS Fargate Pod Scheduled Termination"]
}
```

可以為事件設定合適的目標以進行擷取。如需可用目標的完整清單，請參閱 Amazon EventBridge 使用者指南中的 Amazon EventBridge 目標。您也可以在 AWS 使用者通知中建立通知組態。使用 AWS Management Console 建立通知時，請在事件規則下，選擇 Elastic Kubernetes Service (EKS) 作為 AWS 服務 名稱，並選擇 EKS Fargate Pod 排程終止作為事件類型。如需詳細資訊，請參閱《AWS 使用者通知使用者指南》中的 AWS 使用者通知入門。

Fargate 指標

> **Important**
>
> 採用 Amazon EKS 的 AWS Fargate 不適用於 AWS GovCloud (美國東部) 與 AWS GovCloud (美國西部)。

您可以收集 AWS Fargate 的系統指標和 CloudWatch 用量指標。

應用程式指標

針對在 Amazon EKS 和 AWS Fargate 上執行的應用程式，您可以使用 AWS Distro for OpenTelemetry (ADOT)。ADOT 允許您收集系統指標並將其傳送到 CloudWatch Container Insights 儀表板。若要針對在 Fargate 上執行的應用程式開始使用 ADOT，請參閱 ADOT 文件中的使用 CloudWatch Container Insights 搭配 AWS Distro for OpenTelemetry。

用量指標

您可以使用 CloudWatch 用量指標來提供您帳戶的資源用量可見度。使用這些指標，以 CloudWatch 圖表和儀表板視覺化目前的服務使用狀況。

AWS Fargate 用量指標對應到 AWS 服務配額。您可以設定警示，在您的用量接近服務配額時發出警示。如需 Fargate 的服務配額詳細資訊，請參閱 Amazon EKS 服務配額。

AWS Fargate 在 AWS/Usage 命名空間中發佈下列指標。

指標	描述
ResourceCount	您的帳戶中正在執行的特定資源總數。資源由與指標相關聯的維度定義。

以下維度用於強化 AWS Fargate 發佈的用量指標。

維度	描述
Service	包含該資源的 AWS 服務的名稱。對於 AWS Fargate 用量指標，此維度的值為 Fargate。
Type	正在報告的實體類型。目前，AWS Fargate 用量指標的唯一有效值為 Resource。
Resource	正在執行的資源類型。 目前，AWS Fargate 會傳回您的 Fargate 隨需用量的相關資訊。Fargate 隨需用量的資源值為 OnDemand。 **Note** Fargate 隨需用量結合了使用 Fargate 的 Amazon EKS Pods、使用 Fargate 啟動類型的 Amazon EKS 任務，以及使用 FARGATE 容量供應商的 Amazon ECS 任務。
Class	正在追蹤的資源類別。AWS Fargate 目前不會使用類別維度。

建立 CloudWatch 警示來監控 Fargate 資源用量指標

AWS Fargate 提供 CloudWatch 用量指標，與 Fargate 隨需資源用量的 AWS 服務配額對應。在 Service Quotas 主控台中，您可以在圖表上一目了然地查看使用情況。您也可以設定警示，在您的用量接近服務配額時發出警示。如需詳細資訊，請參閱 Fargate 指標。

使用下列步驟，根據其中一個 Fargate 資源用量指標來建立 CloudWatch 警示。

根據您的 Fargate 用量配額建立警示 (AWS Management Console)

1. 開啟 Service Quotas 主控台，網址為 https://console.aws.amazon.com/servicequotas/。
2. 在左側導覽窗格中，選擇 AWS 服務。
3. 從 AWS services (AWS 服務) 清單中，搜尋並選取 AWS Fargate。
4. 在 Service quotas (服務配額) 清單中，選擇您要為其建立警示的 Fargate 用量配額。
5. 在 Amazon CloudWatch alarms (Amazon CloudWatch 警示) 區段中，選擇 Create (建立)。
6. 針對 Alarm threshold (警示閾值)，選擇您想要多少百分比的套用配額值設為警示值。
7. 針對 Alarm name (警示名稱)，輸入警示的名稱，然後選擇 Create (建立)。

Fargate 記錄

> **Important**
>
> AWS Fargate Amazon EKS 在 AWS GovCloud （美國東部）和 AWS GovCloud （美國西部）中不可用。

Amazon EKS on Fargate 提供了基於 Fluent Bit 的內建日誌路由器。這意味著您沒有明確地執行 Fluent Bit 容器以作為輔助，但 Amazon 會為您執行。您只需設定日誌路由器即可。組態會透過必須符合以下條件的專用 `ConfigMap` 生效：

- 名稱 `aws-logging`
- 在名為 `aws-observability` 的專用命名空間中建立
- 不能超過 5300 個字元。

一旦建立了 ConfigMap，Amazon EKS on Fargate 會使用其自動偵測並設定日誌路由器。Fargate 使用的版本 AWS 為Fluent Bit，這是由Fluent Bit AWS管理的上游相容發行版本。有關更多內容，敬請參閱（AWS 詳Fluent Bit見） GitHub。

日誌路由器允許您使用的服務廣度進行日誌 AWS 分析和存儲。您可以將日誌從 Fargate 直接流式傳輸到 Amazon CloudWatch Amazon OpenSearch 服務。您也可以透過 Amazon 資料 Fire hose，將日誌串流到 Amazon S3、Amazon Kinesis 資料串流和合作夥伴工具等目的地。

必要條件

- 指定您部署 Fargate Pods 之現有 Kubernetes 命名空間的現有 Fargate 設定檔。如需詳細資訊，請參閱 為您的叢集建立 Fargate 描述檔。
- 現有 Fargate Pod 執行角色。如需詳細資訊，請參閱 建立 Fargate Pod 執行角色。

日誌路由器組態

設定日誌路由器

在以下步驟中，使用自己的值取代每一個 *example value*。

1. 建立名為 aws-observability 的專用 Kubernetes 命名空間。

   a. 將下列內容儲存到電腦上名為 *aws-observability-namespace*.yaml 的檔案中。name 的值必須為 aws-observability，且需要 aws-observability: enabled 標籤。

   ```
   kind: Namespace
   apiVersion: v1
   metadata:
     name: aws-observability
     labels:
       aws-observability: enabled
   ```

   b. 建立命名空間。

   ```
   kubectl apply -f aws-observability-namespace.yaml
   ```

2. 建立具有 Fluent Conf 資料值的 ConfigMap，將容器日誌寄送至目的地。Fluent Conf 就是 Fluent Bit，這是一種快速且輕量的日誌處理器組態語言，用於將容器日誌路由至您選擇的日誌目的地。如需詳細資訊，請參閱 Fluent Bit 文件中的組態檔案。

> **Important**
>
> 一般 Fluent Conf 中包含的主要區段是 Service、Input、Filter 和 Output。然而，Fargate 日誌路由器僅接受：
>
> - Filter 和 Output 區段。
> - Parser 區段。
>
> 如果您提供任何其他區段，則這些區段將被拒絕。

Fargate 日誌路由器管理 Service 和 Input 區段。其具有下列 Input 區段，無法修改，也不需要在您的 ConfigMap 中。但是，您可以從中取得洞察，例如記憶體緩衝區限制和套用於日誌的標籤。

```
[INPUT]
    Name tail
    Buffer_Max_Size 66KB
    DB /var/log/flb_kube.db
    Mem_Buf_Limit 45MB
    Path /var/log/containers/*.log
    Read_From_Head On
    Refresh_Interval 10
    Rotate_Wait 30
    Skip_Long_Lines On
    Tag kube.*
```

建立 ConfigMap 時，請考慮下列 Fargate 用來驗證欄位的規則：

- 應該在每個相應的索引鍵下指定 [FILTER]、[OUTPUT] 和 [PARSER]。例如，[FILTER] 必須位於 filters.conf 之下。您可以在 filters.conf 下有一或多個 [FILTER]。[OUTPUT] 和 [PARSER] 區段也應該位於其相應的索引鍵之下。透過指定多個 [OUTPUT] 區段中，您可以同時將日誌路由到不同的目的地。
- Fargate 驗證每個區段所需的索引鍵。Name 和 match 對於每個 [FILTER] 和 [OUTPUT] 都是必要項目。Name 和 format 對於每個 [PARSER] 都是必要項目。索引鍵不區分大小寫。
- ConfigMap 中不允許環境變數 (例如：${ENV_VAR})。

- 在每個 `filters.conf`、`output.conf` 和 `parsers.conf` 中每個指令或鍵值對的縮排必須是相同的。鍵值對必須比指令縮排更多。
- Fargate 會根據以下支援的篩選條件進行驗證：`grep`、`parser`、`record_modifier`、`rewrite_tag`、`throttle`、`nest`、`modify` 和 `kubernetes`。
- Fargate 會根據以下支援的輸出進行驗證：`es`、`firehose`、`kinesis_firehose`、`cloudwatch`、`cloudwatch_logs` 和 `kinesis`。
- 必須在 `ConfigMap` 中提供至少一個支援的 `Output` 外掛程式來啟用記錄。啟用記錄時不需要 `Filter` 和 `Parser`。

您也可以使用所需的組態在 Amazon EC2 上執行 Fluent Bit，以對因驗證而產生的任何問題進行故障診斷。使用下列範例之一建立您的 `ConfigMap`。

> **Important**
>
> Amazon EKS Fargate 記錄不支援 `ConfigMaps` 的動態組態。`ConfigMaps` 的任何變更僅會套用於新的 Pods。變更不會套用至現有 Pods。

使用您所需的日誌目的地範例建立 `ConfigMap`。

> **Note**
>
> 您也可以將 Amazon Kinesis Data Streams 用於您的日誌目的地。如果您使用 Kinesis Data Streams，請確定 Pod 執行角色已獲 `kinesis:PutRecords` 許可。如需詳細資訊，請參閱《Fluent Bit：官方手冊》中的 Amazon Kinesis Data Streams 許可。

CloudWatch

建立適用於 CloudWatch 的 **`ConfigMap`**

使用時有兩個輸出選項 CloudWatch：

- 用 C 編寫的輸出外掛程式
- 用 Golang 編寫的輸出外掛程式

下列範例說明如何使用cloudwatch_logs外掛程式將記錄檔傳送至 CloudWatch。

1. 將下列內容儲存到名為 *aws-logging-cloudwatch-configmap*.yaml 的檔案中。*region-code*以叢集所 AWS 區域 在的位置取代。[OUTPUT] 下的參數為必要項目。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  flb_log_cw: "false"  # Set to true to ship Fluent Bit process logs to
 CloudWatch.
  filters.conf: |
    [FILTER]
        Name parser
        Match *
        Key_name log
        Parser crio
    [FILTER]
        Name kubernetes
        Match kube.*
        Merge_Log On
        Keep_Log Off
        Buffer_Size 0
        Kube_Meta_Cache_TTL 300s
  output.conf: |
    [OUTPUT]
        Name cloudwatch_logs
        Match   kube.*
        region region-code
        log_group_name my-logs
        log_stream_prefix from-fluent-bit-
        log_retention_days 60
        auto_create_group true
  parsers.conf: |
    [PARSER]
        Name crio
        Format Regex
        Regex ^(?<time>[^ ]+) (?<stream>stdout|stderr) (?<logtag>P|F) (?
<log>.*)$
        Time_Key    time
```

```
        Time_Format %Y-%m-%dT%H:%M:%S.%L%z
```

2. 將清單檔案套用至叢集。

```
kubectl apply -f aws-logging-cloudwatch-configmap.yaml
```

3. 將 CloudWatch IAM 政策下載到您的電腦。您也可以在上檢視原則 GitHub。

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-logging-examples/mainline/examples/fargate/cloudwatchlogs/permissions.json
```

Amazon OpenSearch Service

要創建一**ConfigMap**個 Amazon OpenSearch 服務

如果您想將日誌發送到 Amazon OpenSearch 服務，則可以使用 es 輸出，這是一個插件C。下列範例說明如何使用外掛程式將記錄檔傳送至 OpenSearch。

1. 將下列內容儲存到名為 *aws-logging-opensearch-configmap*.yaml 的檔案中。使用您自己的值取代每一個 *example value*。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
      Name  es
      Match *
      Host  search-example-gjxdcilagiprbglqn42jsty66y.region-
code.es.amazonaws.com
      Port  443
      Index example
      Type  example_type
      AWS_Auth On
      AWS_Region region-code
      tls   On
```

2. 將清單檔案套用至叢集。

```
kubectl apply -f aws-logging-opensearch-configmap.yaml
```

3. 將 OpenSearch IAM 政策下載到您的電腦。您也可以在上檢視原則 GitHub。

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-
fluent-logging-examples/mainline/examples/fargate/amazon-elasticsearch/
permissions.json
```

請確定 OpenSearch 儀表板的存取控制已正確設定。 OpenSearch 儀表板`all_access` `role`中必須具有 Fargate Pod 執行角色和對應的 IAM 角色。必須為 `security_manager` 角色進行相同的映射。您可以新增先前的映射，方法是選取 `Menu`、`Security`、`Roles`，然後選取個別的角色。如需詳細資訊，請參閱如何對 CloudWatch 日誌進行疑難排解，以便將其串流到 Amazon ES 網域？ 。

Firehose

若要建立**`ConfigMap`**適用於 Firehose 的步驟

將記錄傳送至 Firehose 時，您有兩個輸出選項：

- `kinesis_firehose`：以 C 編寫的輸出外掛程式。
- `firehose`：以 Golang 編寫的輸出外掛程式。

下列範例說明如何使用`kinesis_firehose`外掛程式將記錄檔傳送至 Firehose。

1. 將下列內容儲存到名為 *aws-logging-firehose-configmap*.yaml 的檔案中。*region-code*以叢集所 AWS 區域 在的位置取代。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
     Name  kinesis_firehose
     Match *
     region region-code
```

```
        delivery_stream my-stream-firehose
```

2. 將清單檔案套用至叢集。

```
kubectl apply -f aws-logging-firehose-configmap.yaml
```

3. 將 Firehose 身分與存取權管理政策下載到您的電腦。您也可以在上檢視原則 GitHub。

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/kinesis-firehose/permissions.json
```

3. 使用上一個步驟中下載的政策檔案，建立 IAM 政策。

```
aws iam create-policy --policy-name eks-fargate-logging-policy --policy-document
 file://permissions.json
```

4. 使用下列命令，將 IAM 政策連接至為 Fargate 設定檔指定的 Pod 執行角色。使用您的帳戶 ID 取代 *111122223333*。將 *AmazonEKSFargatePodExecutionRole* 取代為您的 Pod 執行角色 (如需詳細資訊，請參閱 建立 Fargate Pod 執行角色)。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/eks-fargate-logging-policy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Kubernetes 篩選條件支援

此功能需要符合以下最低的 Kubernetes 版本和平台層級要求，或使用更新的版本。

Kubernetes 版本	平台層級
1.23 版和更新版本	eks.1

Fluent Bit Kubernetes 篩選條件允許您將 Kubernetes 中繼資料新增至您的日誌檔案。如需有關篩選條件的相關資訊，請參閱 Fluent Bit 說明文件中的 Kubernetes。您可以使用 API 伺服器端點來套用篩選條件。

```
filters.conf: |
    [FILTER]
        Name             kubernetes
```

```
        Match             kube.*
        Merge_Log           On
        Buffer_Size         0
        Kube_Meta_Cache_TTL 300s
```

Important

- Kube_URL、Kube_CA_File、Kube_Token_Command 和 Kube_Token_File 是服務擁有的組態參數，您無法指定這些參數。Amazon EKS Fargate 會將這些值填入。
- Kube_Meta_Cache_TTL 是 Fluent Bit 等待直到其可以與 API 伺服器通訊，以取得最新中繼資料所需的時間。若並未指定 Kube_Meta_Cache_TTL，則 Amazon EKS Fargate 會附加預設值 30 分鐘，以減少 API 伺服器上的負載。

將 Fluent Bit 程序日誌傳送至帳戶

您可以選擇 CloudWatch 使用以下命令將Fluent Bit流程日誌運送到 Amazon ConfigMap。傳送 Fluent Bit 處理記錄以 CloudWatch需要額外的記錄擷取和儲存成本。*region-code*以叢集所 AWS 區域 在的位置取代。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
  labels:
data:
  # Configuration files: server, input, filters and output
  # ======================================================
  flb_log_cw: "true"  # Ships Fluent Bit process logs to CloudWatch.

  output.conf: |
    [OUTPUT]
        Name cloudwatch
        Match kube.*
        region region-code
        log_group_name fluent-bit-cloudwatch
        log_stream_prefix from-fluent-bit-
        auto_create_group true
```

記錄位於叢集 AWS 區域 所在的位置 CloudWatch。日誌群組的名稱是 *my-cluster*-fluent-bit-logs，而 Fluent Bit logstream 的名稱是 fluent-bit-*podname*-*pod-namespace*。

> **Note**
>
> - 只有當 Fluent Bit 程序成功啟動時，才會傳送程序日誌。若在啟動 Fluent Bit 時失敗，程序日誌就會遺失。您只能將處理日誌運送到 CloudWatch。
> - 若要對您帳戶的傳送程序日誌除錯，您可以套用先前的 ConfigMap 以取得程序日誌。Fluent Bit 無法啟動通常是由於您的 ConfigMap 在啟動時未被 Fluent Bit 解析或接受。

停止傳送 Fluent Bit 處理日誌

傳送Fluent Bit程序記錄 CloudWatch 需要額外的記錄擷取和儲存費用。若要排除現有 ConfigMap 設定中的處理日誌，請執行下列步驟。

1. 在啟用 Fargate 記 CloudWatch 錄之後，找出為 Amazon EKS 叢集Fluent Bit處理日誌自動建立的日誌群組。它遵循格式 {cluster_name}-fluent-bit-logs。
2. 刪除記 CloudWatch 錄群組中每個Pod's處理程序記錄所建立的現有記 CloudWatch 錄資料流。
3. 編輯 ConfigMap 和設定 flb_log_cw: "false"。
4. 重新啟動叢集中的任何現有 Pods。

測試應用程式

1. 部署範例 Pod。
   a. 將下列內容儲存到電腦上名為 *sample-app*.yaml 的檔案中。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
  namespace: same-namespace-as-your-fargate-profile
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
```

```
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest
          ports:
            - name: http
              containerPort: 80
```

b. 將清單檔案套用至叢集。

```
kubectl apply -f sample-app.yaml
```

2. 使用您在 ConfigMap 中設定的目的地檢視 NGINX 日誌。

大小考量因素

建議您為日誌路由器規劃最多 50 MB 的記憶體。如果您希望應用程式以非常高的輸送量產生日誌，則應該規劃最多 100 MB。

故障診斷

若要確認記錄功能是因某些原因 (例如無效的 ConfigMap，以及無效的原因) 啟用或停用，請使用 **kubectl describe pod *pod_name*** 檢查您的 Pod 事件。輸出可能包含可釐清是否已啟用記錄的 Pod 事件，例如下列範例輸出。

```
[...]
Annotations:          CapacityProvisioned: 0.25vCPU 0.5GB
                      Logging: LoggingDisabled: LOGGING_CONFIGMAP_NOT_FOUND
                      kubernetes.io/psp: eks.privileged
[...]
Events:
  Type     Reason           Age        From
              Message
  ----     ------           ----       ----
              -------
  Warning  LoggingDisabled  <unknown>  fargate-scheduler
               Disabled logging because aws-logging configmap was not found. configmap
 "aws-logging" not found
```

Pod 事件是暫時性的，具體取決於設定的時段。您也可以使用 **kubectl describe pod *pod-name*** 來檢視 Pod's 註釋。在 Pod 註釋中，可參閱是否已啟用或停用記錄功能，及其原因的相關資訊。

選擇 Amazon EC2 執行個體類型

Amazon EC2 為工作節點提供多種執行個體類型選擇。每個執行個體類型皆提供不同的運算、記憶體、儲存與網路功能。每個執行個體也會依照這些功能分組為執行個體系列。如需清單，請參閱 Amazon EC2 使用者指南中的可用執行個體類型和 Amazon EC2 使用者指南中的可用執行個體類型。Amazon EKS 發佈了多種 Amazon EC2 AMI 變體以啟用支援。若要確保您選取的執行個體類型與 Amazon EKS 相容，請考慮以下條件。

- 所有 Amazon EKS AMI 目前不支援 `g5g` 和 `mac` 系列。
- Arm 和非加速的 Amazon EKS AMI 不支援 `g3`、`g4`、`inf` 及 `p` 系列。
- 加速的 Amazon EKS AMI 不支援 `a`、`c`、`hpc`、`m` 及 `t` 系列。
- 對於以 ARM 為基礎的執行個體，Amazon Linux 2023 (AL2023) 僅支援使用Graviton2或更新版本處理器的執行個體類型。AL2023 不支援A1執行個體。

在 Amazon EKS 支援的執行個體類型之間進行選擇時，請考慮每種類型的以下功能。

節點群組中的執行個體數量

一般來說，較少、較大的執行個體更好，特別是在有很多 Daemonsets 的情況下。每個執行個體都需要對 API 伺服器進行 API 呼叫，因此擁有的執行個體越多，API 伺服器上的負載就越大。

作業系統

檢閱 Linux、Windows 以及 Bottlerocket 支援的執行個體類型。在建立 Windows 執行個體之前，請檢閱 為您的 Amazon EKS 叢集啟用 Windows 支援。

硬體架構

您需要 x86 或 Arm？ 您只能在 Arm 上部署 Linux。部署 Arm 執行個體之前，請檢閱 Amazon EKS 最佳化的 Arm Amazon Linux AMI。您是否需要建置在 Nitro System (Linux 或 Windows) 上的執行個體，或具有加速功能的執行個體？ 如果您需要加速功能，則只能將 Linux 與 Amazon EKS 搭配使用。

Pods數目上限

由於每個 Pod 都指派了自己的 IP 地址，因此執行個體類型支援的 IP 地址數量是決定可在執行個體上執行之 Pods 數量的因素。若要手動確定執行個體類型所能支援的 Pods 數量，請參閱 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限。

> **Note**
>
> 如果您使用 Amazon EKS 最佳化的 Amazon Linux 2 AMI (v20220406 或更新版本)，您可以使用新的執行個體類型，而無需升級至最新的 AMI。對於這些 AMI，AMI 會自動計算必要的 max-pods 值 (如果未列於 eni-max-pods.txt 檔案)。預設情況下，Amazon EKS 可能不支援目前處於預覽版中的執行個體類型。此類類型的 max-pods 值仍需新增至我們 AMI 中的 eni-max-pods.txt。

AWS 與非 Nitro 系統執行個體類型相比，Nitro 系統執行個體類型可選擇支援更多的 IP 位址。然而，並非為執行個體指派的所有 IP 地址都可用於 Pods。要為您的執行個體指派大量的 IP 地址，您必須在叢集中安裝並適當設定 Amazon VPC CNI 附加元件 1.9.0 或更新版本。如需詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。要將最大數量的 IP 地址指派給執行個體，您必須在叢集中安裝版本 1.10.1 或更新版本的 Amazon VPC CNI 附加元件，並使用 IPv6 系列部署叢集。

IP 系列

在將 IPv4 系列用於叢集時，您可以使用任何受支援的執行個體類型，這允許您的叢集將私有 IPv4 地址指派給您的 Pods 和服務。但是，如果您想在叢集中使用 IPv6 系列，則必須使用 AWS Nitro 系統執行個體類型或裸機類型。Windows 執行個體僅支援 IPv4。您的叢集必須是執行版本 1.10.1 或更新版本 Amazon VPC CNI 附加元件的叢集。如需有關使用 IPv6 的詳細資訊，請參閱 IPv6叢集的位址Pods、和 services。

您正在執行的 Amazon VPC CNI 附加元件版本

最新版的 Kubernetes 專用 Amazon VPC CNI 外掛程式支援這些執行個體類型。您可能需要更新 Amazon VPC CNI 附加元件版本，才能利用最新支援的執行個體類型。如需詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。最新版本支援與 Amazon EKS 搭配使用的最新功能。舊版不支援所有功能。您可以在 GitHub 上的變更日誌檢視不同版本支援的功能。

AWS 區域 你正在創建你的節點

並非所有 AWS 區域皆提供執行個體類型。

您是否使用 Pods 的安全群組

如果您正在使用 Pods 的安全群組，則僅支援特定的執行個體類型。如需詳細資訊，請參閱 Pods 的安全群組。

Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限

由於每個 Pod 都指派了自己的 IP 地址，因此執行個體類型支援的 IP 地址數量是決定可在執行個體上執行之 Pods 數量的因素。Amazon EKS 提供一個可以下載並執行的指令碼，以便判斷 Amazon EKS 建議在每種執行個體類型上所能執行的 Pods 數量上限。該指令碼會使用每個執行個體的硬體屬性和組態選項來判斷 Pods 數量上限。您可以使用這些步驟中傳回的數字來啟用功能，例如將 IP 地址指派給來自與執行個體不同子網路的 Pods，以及大幅增加執行個體的 IP 地址數量。如果您使用的是具有多個執行個體類型的受管節點群組，請使用適用於所有執行個體類型的值。

1. 下載可用來計算每種執行個體類型的 Pods 數量上限的指令碼。

```
curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/templates/al2/runtime/max-pods-calculator.sh
```

2. 將指令碼標記為電腦上的可執行檔。

```
chmod +x max-pods-calculator.sh
```

3. 執行指令碼，使用您計劃部署的執行個體類型取代 *m5.large*，並使用 Amazon VPC CNI 附加元件版本取代 *1.9.0-eksbuild.1*。若要判斷附加元件版本為何，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件 中的更新程序。

```
./max-pods-calculator.sh --instance-type m5.large --cni-version 1.9.0-eksbuild.1
```

範例輸出如下。

```
29
```

您可以在指令碼中新增下列選項，查看使用選用功能時支援的 Pods 數量上限。

- `--cni-custom-networking-enabled`：想要從不同於執行個體的子網路指派 IP 地址時，請使用此選項。如需詳細資訊，請參閱 Pod 的自訂聯網。將此選項新增至具有相同範例值的前一個指令碼會產生 20。
- `--cni-prefix-delegation-enabled`：想要為每個彈性網路介面指派更大量的 IP 地址時，請使用此選項。此功能需要在 Nitro 系統上執行的 Amazon Linux 執行個體，以及 Amazon VPC CNI 附加元件 `1.9.0` 版或更新版本。如需詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。將此選項新增至具有相同範例值的前一個指令碼會產生 110。

您還可以使用 `--help` 選項執行指令碼，以查看所有可用選項。

> **Note**
>
> 最大 Pods 計算器指令碼會根據 Kubernetes 可擴展性閾值和建議的設定，將傳回值限制為 110。如果您的執行個體類型有超過 30 個 vCPU，則此限制會跳轉至 250，這是根據內部 Amazon EKS 可擴展性團隊測試而定的數字。如需詳細資訊，請參閱 Amazon VPC CNI 外掛程式增加每節點的 Pod 限制部落格文章。

Amazon EKS 最佳化 AMI

您可以使用預先建置的 Amazon EKS 最佳化 Amazon Machine Image (AMI) 或您的自訂 AMI 來部署節點。如需有關各種 Amazon EKS 最佳化 AMI 類型的詳細資訊，請參閱下列其中一個主題。如需如何建立自己的自訂 AMI 的詳細資訊，請參閱 Amazon EKS 最佳化 Amazon Linux AMI 建置指令碼。

主題

- Amazon EKS 已結束對 Dockershim 的支援
- Amazon EKS 最佳化的 Amazon Linux AMI
- Amazon EKS 最佳化 Bottlerocket AMI
- Amazon EKS 優Ubuntu化
- Amazon EKS 最佳化 Windows AMI

Amazon EKS 已結束對 **`Dockershim`** 的支援

Kubernetes 不再支援 `Dockershim`。Kubernetes 團隊已移除 Kubernetes `1.24` 版中的執行階段。如需詳細資訊，請參閱 Kubernetes 部落格中的 Kubernetes 正在從 Dockershim 繼續邁進：承諾和後續步驟。

從推出 Kubernetes `1.24` 版開始，Amazon EKS 也已結束對 `Dockershim` 的支援。從 `1.24` 版開始，正式發布的 Amazon EKS AMI 會包含 `containerd` 作為唯一執行階段。本主題涵蓋了部分詳細資訊，但更多資訊可在「關於遷移至 Amazon EKS 您必須知道的一切」中取得。

您可以使用 `kubectl` 外掛程式來查看哪些 Kubernetes 工作負載已掛載 Docker 通訊端磁碟區。如需詳細資訊，請參閱 GitHub 上的 適用於 Docker 通訊端的偵測器 (DDS)。執行早於 `1.24` 之 Kubernetes 版本的 Amazon EKS AMI 會使用 Docker 作為預設執行階段。然而，這些 Amazon EKS

AMI 具有引導旗標選項，您可將其用於在任何受支援且使用 containerd 的叢集上測試工作負載。如需詳細資訊，請參閱 測試從移轉Docker到 containerd。

我們會繼續為現有的 Kubernetes 版本發佈 AMI，直到其支援日期結束為止。如需詳細資訊，請參閱 Amazon EKS Kubernetes 發佈日曆。若您需要更多時間在 containerd 上測試工作負載，則可繼續使用 1.24 之前的支援版本。但是，當您想將官方 Amazon EKS AMI 升級至 1.24 版或更新版本時，請務必驗證工作負載是否可在 containerd 上執行。

containerd 執行階段提供更可靠的效能和安全性。containerd 是跨 Amazon EKS 並經標準化的執行階段。Fargate 和 Bottlerocket 已經僅使用 containerd。containerd 可協助盡量減少解決 Dockershim 常見漏洞和風險 (CVE) 所需 Amazon EKS AMI 版本的數量。由於 Dockershim 已於內部使用 containerd，您可能不需要進行任何變更。不過，在某些情況下可能需要或必須執行變更：

- 您必須對掛載 Docker 通訊端的應用程式執行變更。例如，使用容器建置的容器映像會受到影響。許多監控工具也掛載 Docker 通訊端。您可能需要等待更新或重新部署工作負載以監控執行階段。
- 您可能需要對依賴特定 Docker 設定的應用程式執行變更。例如，不再支援 HTTPS_PROXY 通訊協定。您必須更新使用此通訊協定的應用程式。如需詳細資訊，請參閱 Docker 文件的 dockerd。
- 如果您使用 Amazon ECR 憑證助手提取映像，則須切換至 kubelet 映像憑證提供者。如需詳細資訊，請參閱 Kubernetes 文件中的 設定 kubelet 映像憑證提供者。
- Amazon EKS 1.24 版不再支援 Docker，因此不再支援先前 Amazon EKS 引導指令碼所支援的某些旗標。在移至 Amazon EKS 1.24 版或更新版本前，您必須移除目前不支援的任何旗標參考：
  - --container-runtime dockerd (containerd 是唯一受支援的值)
  - --enable-docker-bridge
  - --docker-config-json
- 如果您已為 Container Insights 設定了 Fluentd，則必須先將 Fluentd 遷移至 Fluent Bit，才能變更為 containerd。系統會將 Fluentd 剖析器設定為僅剖析 JSON 格式的日誌訊息。與 dockerd 不同的是，containerd 容器執行期具有不是 JSON 格式的日誌訊息。如果您未遷移到 Fluent Bit，則某些已設定的 Fluentd's 剖析器將在 Fluentd 容器內生成大量錯誤。如需有關移轉的詳細資訊，請參閱設定Fluent Bit為將記錄檔傳送 DaemonSet 至 CloudWatch 記錄檔。
- 如果使用自訂 AMI 並且要升級至 Amazon EKS 1.24，您必須確保已為工作節點啟用 IP 轉送。Docker 不需要此設定，但對 containerd 而言，這是必要設定。需針對 Pod 對 Pod、Pod 對外部或 Pod 對 apiserver 網路連線進行疑難排解。

  若要在工作節點上驗證此設定，請執行下列其中一個命令：
  - sysctl net.ipv4.ip_forward
  - cat /proc/sys/net/ipv4/ip_forward

如果輸出為 0，請執行下列其中一個命令來啟用 net.ipv4.ip_forward 核心變數：

- sysctl -w net.ipv4.ip_forward=1
- echo 1 > /proc/sys/net/ipv4/ip_forward

如需有關在 containerd 執行期在 Amazon EKS AMI 上啟用設定的資訊，請參閱 GitHub 上的 install-worker.sh。

Amazon EKS 最佳化的 Amazon Linux AMI

Amazon EKS 優化的 Amazon Linux AMI 建立在 Amazon Linux 2（AL2）和 Amazon Linux 2023（AL2023）之上。此種 AMI 已設定為 Amazon EKS 節點的基礎映像。將 AMI 設定為與 Amazon EKS 搭配使用，其中包含下列元件：

- kubelet
- AWS IAM 身份驗證器
- Docker (Amazon EKS 1.23 版和舊版)
- containerd

> **Note**
>
> - 您可以在 Amazon Linux 安全中心追蹤 AL2 的安全或隱私權事件，或訂閱相關聯的 RSS 摘要。安全與隱私權事件包含問題的概觀、哪些套件受到影響，以及如何更新您的執行個體以修正問題。
> - 部署加速或 Arm AMI 之前，請檢閱 Amazon EKS 最佳化加速 Amazon Linux AMI 和 Amazon EKS 最佳化的 Arm Amazon Linux AMI。
> - 對於Kubernetes版本1.23，您可以使用可選的引導標誌來測試從Docker到的遷移containerd。如需詳細資訊，請參閱 測試從移轉Docker到 containerd。
> - 從 Kubernetes 版本 1.25 開始，您將無法再使用開箱即用的 Amazon EC2 P2 執行個體搭配 Amazon EKS 最佳化加速 Amazon Linux AMI 一起使用。這些適用於 Kubernetes 版本 1.25 或更高版本的 AMI 將支援 NVIDIA 525 系列或更新版本驅動程式，這些與 P2 執行個體不相容。但是，NVIDIA 525 系列或更新版本的驅動程式與 P3 P4 及 P5 執行個體相容，因此您可以在 Kubernetes 版本 1.25 或更新版本將這些執行個體搭配 AMI 一起使用。在 Amazon EKS 叢集升級至版本 1.25 之前，請將任何 P2 執行個體移轉至 P3 P4 和 P5 執

行個體。您也應該主動升級您的應用程式以適用於 NVIDIA 525 系列或更新版本。我們計劃將更新的NVIDIA 525系列或更高Kubernetes版本1.23的驅動程序移植到 2024 年 1 月下旬。1.24

- 在版本1.30或更新版本的叢集中，任何新建立的受管理節點群組都會自動預設使用 AL2023 做為節點作業系統。先前，新節點群組預設為 AL2。建立新節點群組時，您可以選擇 AL2 作為 AMI 類型，繼續使用 AL2。
- 對 AL2 的 Support 將於二零二五年六月三十日結束。如需詳細資訊，請參閱 Amazon Linux 2 常見問答集。

從 AL2 升級到 AL2023

Amazon EKS 優化 AMI 在以 AL2 和 AL2023 為基礎的兩個系列中提供。AL2023 是以 Linux 為基礎的全新作業系統，旨在為您的雲端應用程式提供安全、穩定且高效能的環境。它是 Amazon 網絡服務的下一代 Amazon Linux，可1.24在所有受支持的亞馬遜 EKS 版本中使用，包括版本1.23和擴展支持。基於 AL2023 的 Amazon EKS 加速 AMI 將在以後提供。如果您已經加速了工作負載，則應繼續使用 AL2 加速 AMI 或保加速器。

AL2023 比 AL2 提供了一些改進。有關完整比較，請參閱 Amazon 2023 用戶指南中的比較 AL2 和 Amazon Linux 2023。已從 AL2 新增、升級和移除數個套件。強烈建議您在升級前先使用 AL2023 測試您的應用程式。如需 AL2023 中所有 Package 變更的清單，請參閱 Amazon 2023 版本說明中的套件變更。

除了這些變更之外，您還應注意下列事項：

- AL2023 引入了使用 YAML 組態結構描述nodeadm的新節點初始化程序。如果您使用自我管理的節點群組或 AMI 搭配啟動範本，您現在需要在建立新節點群組時明確提供額外的叢集中繼資料。最小必要參數的範例如下，其中apiServerEndpointcertificateAuthority、和服務現cidr在需要：

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

在 AL2 中，這些參數的中繼資料是從 Amazon EKS `DescribeCluster` API 呼叫中探索到的。使用 AL2023 時，自大型節點擴展期間額外 API 呼叫風險限制以來，此行為已發生變更。如果您使用的是沒有啟動範本的受管節點群組，或您正在使用，則此變更不會影響您Karpenter。如需有關`certificateAuthority`和服務的詳細資訊`cidr`，請參閱 Amazon EKS API 參考*`DescribeCluster`*中的。

- DockerAL2023 不支援所有受支援的 Amazon EKS 版本。在 AL2 中Docker，Amazon EKS 版本`1.24`或更高版本的 Support 已結束，並已移除。如需淘汰的詳細資訊，請參閱 Amazon EKS 已結束的. `Dockershim`
- AL2023 需要 Amazon VPC CNI 版本`1.16.2`或更高版本。
- AL2023 默認情況`IMDSv2`下需要使用。 `IMDSv2`有幾個有助於改善安全狀態的好處。它使用面向會話的身份驗證方法，該方法需要在簡單的 HTTP PUT 請求中創建秘密令牌才能啟動會話。會話的令牌可以在 1 秒到 6 小時之間的任何地方有效。如需如何從`IMDSv1`轉換為的詳細資訊`IMDSv2`，請參閱轉換為使用執行個體中繼資料服務版本 2 和取得 IMDSv2 的完整優點，以及在您的基礎結構中停用 IMDSv1。 AWS 如果您想要使用`IMDSv1`，您仍然可以使用執行個體中繼資料選項啟動屬性來手動覆寫設定。

  > **Note**
  >
  > 對於`IMDSv2`，受管理節點群組的預設躍點計數設定為 1。這表示容器將無法使用 IMDS 存取節點的認證。如果您需要容器存取節點的登入資料，您仍然可以透過手動覆寫自訂 Amazon EC2 啟動範本`HttpPutResponseHopLimit`中的，將範本增加到 2。 或者，您也可以使用 Amazon EKS Pod 身分來提供登入資料，而非使用。`IMDSv2`

- AL2023 具有下一代統一控制群組階層 (`cgroupv2`)。 `cgroupv2`用於實現容器運行時，並通過`systemd`. 雖然 AL2023 仍然包含可以使系統運行使用的代碼`cgroupv1`，但這不是建議或支持的配置。此組態將在 future 的主要發行版本中完全移除。
- `eksctl`需要版本`0.176.0`或更高版本`eksctl`才能支援 AL2023。

對於先前現有的受管理節點群組，您可以執行就地升級或藍/綠升級，視您使用啟動範本的方式而定：

- 如果您將自訂 AMI 與受管節點群組搭配使用，則可以透過交換啟動範本中的 AMI ID 來執行就地升級。在執行此升級策略之前，應確保您的應用程式和任何使用者資料先傳輸到 AL2023。
- 如果您要搭配標準啟動範本或未指定 AMI ID 的自訂啟動範本使用受管節點群組，則必須使用藍/綠策略進行升級。藍/綠升級通常比較複雜，而且需要建立一個全新的節點群組，您可以在其中指定 AL2023 作為 AMI 類型。然後必須仔細設定新節點群組，以確保 AL2 節點群組中的所有自訂資料都

與新作業系統相容。在您的應用程式中測試並驗證新節點群組之後，就Pods可以從舊節點群組移轉到新節點群組。移轉完成後，您可以刪除舊的節點群組。

如果您正在使用Karpenter並希望使用 AL2023，則需要使用 AL2023 修改該`EC2NodeClassamiFamily`字段。依預設，在中啟用「漂移」Karpenter。這表示一旦`amiFamily`欄位變更，Karpenter會在可用時自動將您的 Worker 節點更新為最新的 AMI。

Amazon EKS 最佳化加速 Amazon Linux AMI

> **Note**
>
> 基於 AL2023 的 Amazon EKS 加速 AMI 將在以後提供。如果您已加速工作負載，則應繼續使用 AL2 加速 AMI 或Bottlerocket.

Amazon EKS 最佳化加速 Amazon Linux AMI 是建立在標準 Amazon EKS 最佳化 Amazon Linux AMI 之上。它已設定為做為 Amazon EKS 節點的選用映像，以支援 GPU、推論和基於工作負載的工作負載。

除了採用標準 Amazon EKS 最佳化 AMI 組態，加速 AMI 另包括下列項目：

- NVIDIA 驅動程式
- `nvidia-container-runtime`
- AWS Neuron驅動

如需加速 AMI 中包含的最新元件清單，請參閱上的`amazon-eks-ami`發行GitHub。

> **Note**
>
> - Amazon EKS 最佳化加速 AMI 僅支援 GPU 和 Inferentia 型執行個體類型。請務必在節點 AWS CloudFormation 範本中指定這些執行個體類型。一旦使用 Amazon EKS 最佳化加速 AMI，即表示您同意 NVIDIA 的最終使用者授權合約 (EULA)。
> - Amazon EKS 最佳化加速 AMI 先前稱為支援 GPU 的 Amazon EKS 最佳化 AMI。
> - 舊版 Amazon EKS 最佳化加速 AMI 安裝了 `nvidia-docker` 儲存庫。Amazon EKS AMI 版本 `v20200529` 和更新版本中不再包含該儲存庫。

啟用以 AWS 神經元 (ML 加速器) 為基礎的工作負載

如需 Amazon EKS Neuron 中使用的訓練和推論工作負載的詳細資訊，請參閱下列參考資料：

- 容器-Kubernetes-文件中的入門AWS Neuron
- AWS NeuronEKS 樣本的培訓 GitHub
- 使用 AWS Inferentia 的機器學習推論

若要啟用 GPU 型工作負載

以下程序說明如何使用 Amazon EKS 最佳化加速 AMI，在 GPU 型執行個體上執行工作負載。

1. 將 GPU 節點加入您的叢集之後，您必須套用 Kubernetes 專用 NVIDIA 裝置外掛程式作為叢集上的 DaemonSet。請先以您想要的 NVIDIA/k8s-device-plugin 版本來取代 *vX.X.X*，再執行下列命令。

   ```
   kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
   nvidia-device-plugin.yml
   ```

2. 您可使用以下命令驗證您的節點有否配置 GPU。

   ```
   kubectl get nodes "-o=custom-
   columns=NAME:.metadata.name,GPU:.status.allocatable.nvidia\.com/gpu"
   ```

部署 Pod 以測試 GPU 節點是否設定妥當

1. 使用下列內容建立名為 `nvidia-smi.yaml` 的檔案。以您想要的 nvidia/cuda 標籤取代 *tag*。此清單檔案會啟動要在節點上執行 `nvidia-smi` 的 NVIDIA CUDA 容器。

   ```
   apiVersion: v1
   kind: Pod
   metadata:
     name: nvidia-smi
   spec:
     restartPolicy: OnFailure
     containers:
     - name: nvidia-smi
       image: nvidia/cuda:tag
       args:
       - "nvidia-smi"
   ```

```
    resources:
      limits:
        nvidia.com/gpu: 1
```

2. 執行以下命令，套用此清單檔案。

```
kubectl apply -f nvidia-smi.yaml
```

3. Pod 執行完成後，使用以下命令檢視其日誌。

```
kubectl logs nvidia-smi
```

範例輸出如下。

```
Mon Aug  6 20:23:31 20XX
+-----------------------------------------------------------------------------+
| NVIDIA-SMI XXX.XX                 Driver Version: XXX.XX                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:00:1C.0 Off |                    0 |
| N/A   46C    P0    47W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Amazon EKS 最佳化的 Arm Amazon Linux AMI

Arm 執行個體可為橫向擴展和 Arm 型應用程式節省大量的成本，例如 Web 伺服器、容器化微型服務、快取機群和分散式資料存放區。將 Arm 節點新增至叢集時，請檢閱下列考量事項。

考量事項

- 如果叢集是在 2020 年 8 月 17 日之前部署，您必須對重要的叢集附加元件清單檔案進行一次性升級。如此一來，Kubernetes 就可以為叢集中使用中的每個硬體架構提取正確的映像。如需更新叢集

附加元件的詳細資訊，請參閱 更新 Amazon EKS 叢集的 Kubernetes 版本。如果您在 2020 年 8 月 17 當天或之後部署叢集，則 CoreDNS、kube-proxy 和 Amazon VPC CNI plugin for Kubernetes 附加元件已經具有多架構能力。

- 部署至 Arm 節點的應用程式必須針對 Arm 進行編譯。
- 如果您在現有叢集中部署了 DaemonSets，或者您想將其部署到您也想在其中部署 Arm 節點的新叢集，請確認您的 DaemonSet 是否可以在您叢集中的所有硬體架構上執行。
- 您可以在相同的叢集中執行 Arm 節點群組和 x86 節點群組。如果您這麼做，請考慮將多架構容器映像部署到容器儲存庫 (例如 Amazon Elastic Container Registry)，然後將節點選取器新增至清單檔案，以讓 Kubernetes 知道 Pod 可以部署到哪些硬體架構。如需詳細資訊，請參閱《Amazon ECR 使用者指南》中的推送多架構映像和適用於 Amazon ECR 的多架構容器映像簡介部落格一文。

測試從移轉Docker到 **containerd**

從推出 Kubernetes 1.24 版開始，Amazon EKS 將結束對 Docker 的支援。如需詳細資訊，請參閱 Amazon EKS 已結束對 Dockershim 的支援。

對於Kubernetes版本1.23，您可以使用選用的啟動程序旗標，為 Amazon EKS 最佳化 AL2 AMI 啟用containerd執行階段。此功能為您在更新至 1.24 版或更新版本時遷移至 containerd 的操作提供清晰的路徑。從推出 Kubernetes 1.24 版開始，Amazon EKS 將結束對 Docker 的支援。containerd 執行階段已在 Kubernetes 社群廣泛採用，是 CNCF 的一個畢業專案。您可以將節點群組新增至新叢集或現有叢集來進行測試。

您可以建立下列其中一種節點群組類型，以啟用引導旗標。

自我管理

使用 啟動自我管理的 Amazon Linux 節點 中的指示建立節點群組。為 BootstrapArguments 參數指定一個 Amazon EKS 最佳化 AMI 和以下文字。

```
--container-runtime containerd
```

受管

如果您使用 eksctl，請建立名為 *my-nodegroup*.yaml 的檔案並包含下列內容。使用您自己的值取代每一個 *example value*。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。若要擷取 ami-*1234567890abcdef0* 的最佳化 AMI ID，請參閱 擷取 Amazon EKS 最佳化 Amazon Linux AMI ID。

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
  version: 1.23
managedNodeGroups:
  - name: my-nodegroup
    ami: ami-1234567890abcdef0
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

> **Note**
>
> 如果同時啟動許多節點，您可能還需要指定--apiserver-endpoint、--b64-cluster-ca 及 --dns-cluster-ip 引導參數的值來避免錯誤。如需詳細資訊，請參閱指定 AMI。

執行以下命令建立節點群組。

```
eksctl create nodegroup -f my-nodegroup.yaml
```

如果您偏好使用不同的工具來建立受管節點群組，則必須使用啟動範本來部署節點群組。在您的啟動範本中，指定 Amazon EKS 最佳化 AMI ID，然後使用啟動範本部署節點群組並提供下列使用者資料。此使用者資料會將引數傳遞至 bootstrap.sh 檔案。如需引導檔案的詳細資訊，請參閱 GitHub 上的 bootstrap.sh。

```
/etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

其他資訊

如需使用 Amazon EKS 最佳化 Amazon Linux AMI 的詳細資訊，請參閱下列區段：

- 若要將 Amazon Linux 與受管節點群組搭配使用，請參閱 受管節點群組。
- 若要啟動自我管理的 Amazon Linux 節點，請參閱 擷取 Amazon EKS 最佳化 Amazon Linux AMI ID。

- 如需版本資訊，請參閱Amazon EKS 最佳化 Amazon Linux AMI 版本。
- 若要擷取 Amazon EKS 最佳化 Amazon Linux AMI 的最新 ID，請參閱 擷取 Amazon EKS 最佳化 Amazon Linux AMI ID。
- 如需了解用於建置 Amazon EKS 最佳化 AMI 的開放原始碼指令碼，請參閱 Amazon EKS 最佳化 Amazon Linux AMI 建置指令碼。

Amazon EKS 最佳化 Amazon Linux AMI 版本

Amazon EKS 最佳化 Amazon Linux AMI 是由 Kubernetes 版本和 AMI 發行日期進行版本化，格式如下：

```
k8s_major_version.k8s_minor_version.k8s_patch_version-release_date
```

每個 AMI 發行版本均包含 `kubelet`、Docker、Linux 核心和 `containerd` 的各種版本。加速的 AMI 亦包括 NVIDIA 驅動程序的各種版本。您現在可以在 GitHub 上的變更日誌找到此版本資訊。

擷取 Amazon EKS 最佳化 Amazon Linux AMI ID

您可以透過查詢 AWS Systems Manager 參數存放區 API，以程式設計方式擷取 Amazon EKS 最佳化 AMI 的 Amazon 機器映像 (AMI) 識別碼。此參數讓您無需手動查詢 Amazon EKS 最佳化 AMI ID。如需 Systems Manager 參數存放區 API 的詳細資訊，請參閱GetParameter。

若要擷取 Amazon EKS 最佳化 AMI 的 AMI 識別碼，請使用 AWS CLI

1. 決定節點執行個體將部署在哪個區域，例如`us-west-2`。
2. 確定您需要的 AMI 類型。如需 Amazon EC2 執行個體類型的相關資訊，請參閱執行個體類型。
   - `amazon-linux-2`適用於以 Amazon Linux 2 (AL2) 為`x86`基礎的執行個體。
   - `amazon-linux-2-arm64`適用於 AL2 ARM 例證，例如以AWS 重力子為基礎的例證。
   - `amazon-linux-2-gpu`適用於 AL2 GPU 加速執行個體。
   - `amazon-linux-2023/x86_64/standard`適用於基於 Amazon Linux 2023 (AL2023) 的執行個`x86`體。
   - `amazon-linux-2023/arm64/standard`適用於 AL2023 手臂執行個體。
3. 決定節點將附加到的叢集Kubernetes版本，例如 1.30。
4. 執行下列 AWS CLI 命令以擷取適當的 AMI ID。視需要取代 AWS 區域、Kubernetes版本和平台。您必須 AWS CLI 使用具有 IAM 許可的 IAM 主體登入，才能擷取 Amazon EKS 最佳化 AMI 中繼資料。`ssm:GetParameter`

```
aws ssm get-parameter --name /aws/service/eks/optimized-ami/1.30/amazon-linux-2/
recommended/image_id \
                --region region-code --query "Parameter.Value" --output text
```

範例輸出如下。

```
ami-1234567890abcdef0
```

Amazon EKS 最佳化 Amazon Linux AMI 建置指令碼

Amazon Elastic Kubernetes Service (Amazon EKS) 具備開源的指令碼，可用於建置 Amazon EKS 最佳化的 AMI。這些建置指令碼可於 GitHub 取得。

Amazon EKS 優化 Amazon Linux AMI 建立在 Amazon Linux 2（AL2）和 Amazon Linux 2023（AL2023）之上，專門用於在 Amazon EKS 集群中用作節點。您可以使用此儲存庫來檢視 Amazon EKS 團隊如何設定`kubelet`、Docker AWS IAM 身份驗證器以及從頭開始建置您自己的 Kubernetes Amazon Linux AMI 的詳細資訊。

構建腳本存儲庫包括一個打HashiCorp包器模板和構建腳本來生成 AMI。這些指令碼為 Amazon EKS 最佳化 AMI 建置的真實來源，因此您可遵循 GitHub 儲存庫來監控 AMI 的變更。例如，您可能希望自己的 AMI 使用的 Docker 版本，與 Amazon EKS 團隊用於官方 AMI 的版本相同。

GitHub儲存庫也包含專門的啟動程序指令碼和 nodeadm 指令碼，這些指令碼會在開機時執行，以設定執行個體的憑證資料、控制平面端點、叢集名稱等。

此外，GitHub儲存庫還包含我們的 Amazon EKS 節點 AWS CloudFormation 範本。這些範本讓您更輕鬆運轉執行 Amazon EKS 最佳化 AMI 的執行個體，並將其註冊於叢集。

如需詳細資訊，請至 https://github.com/awslabs/amazon-eks-ami 參閱 GitHub 的儲存庫。

Amazon EKS 最佳化 AL2 包含可選的啟動程序旗標，以啟用執行階段`containerd`。

VT1為您的自訂 Amazon Linux AMI 設定

Amazon EKS 中的自訂 Amazon Linux AMI 可以支援適用於 Amazon Linux 2 (AL2)、Ubuntu 18 和 20 的 VT1 視訊轉碼執行個體系列。Ubuntu VT1支援具有加速 H.264/AVC 和 H.265/HEVC 編解碼器的 Xilinx U30 媒體轉碼卡。若要利用這些加速執行個體的優勢，請務必遵循下列步驟：

1. 從 AL2、Ubuntu 18 或 Ubuntu 20 建立並啟動基礎 AMI。

2. 在啟動基本 AMI 之後，請在節點上安裝 XRT 驅動程式和執行時間。
3. 建立 Amazon EKS 叢集.
4. 在叢集上安裝 Kubernetes FPGA plugin (FPGA 外掛程式)。

```
kubectl apply -f fpga-device-plugin.yml
```

此外掛程式現在將在 Amazon EKS 叢集上為每個節點公告 Xilinx U30 裝置。您可以使用碼 FFMPEG docker 映像在 Amazon EKS 叢集上執行影片轉碼工作負載範例。

DL1為您的自定義 Amazon Linux 2 AMI 配置

Amazon EKS 中的自訂 Amazon Linux 2 (AL2) AMI 可透過其他組態和附加元件大規模支援深度學習工作負載。Kubernetes本文件說明為內部部署設定或做為較大型雲端組態中的基準，設定一般 Kubernetes 解決方案所需的元件。為了支援此功能，您必須在自訂環境中執行下列步驟：

- SynapaseAI® Software系統上載入的驅動程式 — 這些都包含在 Github 上提供的 AMI 中。
- Habana裝置外掛程式 — 可DaemonSet讓您自動啟用Kubernetes叢集中裝Habana置的註冊並追蹤裝置健康狀態。
- Helm 3.x
- 安裝 MPI 運算子的 Helm Chart。
- MPI 運算子

1. 從 AL2、Ubuntu 18 或 Ubuntu 20 建立並啟動基礎 AMI。
2. 依照下列指示設定的環境DL1。

Amazon EKS 最佳化 Bottlerocket AMI

Bottlerocket 是一個由 AWS 贊助和支援的開放原始碼 Linux 發行版。Bottlerocket 專為託管容器工作負載而打造。使用 Bottlerocket，您可以透過自動更新容器基礎設施來提高容器化部署的可用性，並降低營運成本。Bottlerocket 僅包含執行容器的基本軟體，可改善資源用量、減少安全威脅，並降低管理開銷。Bottlerocket AMI 包括 `containerd`、`kubelet` 與 AWS IAM 驗證器。除了受管節點群組和自我管理節點外，Karpenter 亦支援 Bottlerocket。

優點

將 Bottlerocket 與 Amazon EKS 叢集搭配使用具有以下優勢：

- 更長的正常執行時間、更低的營運成本和更低的管理複雜性 – 與其他 Linux 發行版相比，Bottlerocket 具有更低的資源佔用、更短的啟動時間，並且不易受到安全威脅。Bottlerocket's 更小的佔用空間使用更少的儲存、運算和網路資源，有助於降低成本。
- 透過自動作業系統更新提高安全性：對 Bottlerocket 的更新是作為單一單位套用，必要時可復原。這消除了損毀或失敗更新可能使系統處於無法使用狀態的風險。有了 Bottlerocket，可於安全性更新可用時立即以最不造成干擾的方式自動套用，並於發生故障時復原。
- 高級支援 – 在 Amazon EC2 上使用 AWS 提供的 Bottlerocket 建置包含在同樣的 AWS Support 計畫中，這些計畫還涵蓋 Amazon EC2、Amazon EKS、Amazon ECR 等 AWS 服務。

考量事項

將 Bottlerocket 用於 AMI 類型時，請考慮下列事項：

- Bottlerocket 支援具有 `x86_64` 與 `arm64` 處理器的 Amazon EC2 執行個體。Bottlerocket AMI 不建議與採用 Inferentia 晶片的 Amazon EC2 執行個體搭配使用。
- 目前無 AWS CloudFormation 範本可用來部署 Bottlerocket 節點。
- Bottlerocket 映像不包含 SSH 伺服器或 Shell。您可以採用頻外存取方法來允許 SSH。這些方法啟用管理員容器，並傳遞一些引導組態步驟與使用者資料。如需詳細資訊，請參閱 GitHub 上 Bottlerocket OS 中的下列章節：
  - 探勘
  - 管理員容器
  - Kubernetes 設定
- Bottlerocket 使用不同的容器類型：
  - 在預設情況下，已啟用控制容器。此容器會執行 AWS Systems Manager 代理程式，可以讓您在 Amazon EC2 Bottlerocket 執行個體上執行命令或啟動 Shell 工作階段。如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的設定工作階段管理員。
  - 如果在建立節點群組時提供 SSH 金鑰，則啟用管理容器。建議只在開發和測試案例中使用管理容器，不要在生產環境中使用此種容器。如需詳細資訊，請參閱 GitHub 上的管理容器。

其他資訊

如需使用 Amazon EKS 最佳化 Bottlerocket AMI 的詳細資訊，請參閱下列區段：

- 如需有關 Bottlerocket 的詳細資訊，請參閱 GitHub 上的文件和版本。
- 若要將 Bottlerocket 與受管節點群組搭配使用，請參閱 受管節點群組。

- 若要啟動自我管理的 Bottlerocket 節點，請參閱 正在啟動自我管理的 Bottlerocket 節點。
- 若要擷取 Amazon EKS 最佳化 Bottlerocket AMI 的最新 ID，請參閱 擷取 Amazon EKS 最佳化 Bottlerocket AMI ID。
- 如需合規性支援的詳細資訊，請參閱 Bottlerocket 合規性支援。

擷取 Amazon EKS 最佳化 Bottlerocket AMI ID

您可以透過查詢 AWS Systems Manager 參數存放區 API，擷取 Amazon EKS 最佳化 AMI 的 Amazon 機器映像 (AMI) 識別碼。若使用此參數，您無需手動查詢 Amazon EKS 最佳化 AMI ID。如需 Systems Manager 參數存放區 API 的詳細資訊，請參閱GetParameter。您使用的 IAM 主體必須擁有 `ssm:GetParameter` IAM 許可，才能擷取 Amazon EKS 最佳化 AMI 中繼資料。

您可以使用子`image_id`參數，使用下列 AWS CLI 命令擷取最新推薦的 Amazon EKS 最佳化 Bottlerocket AMI 的映像識別碼。將 *`1.30`* 換成支援的版本，以及將 *`region-code`* 換成需要取得 AMI ID 的 Amazon EKS 支援區域。

```
aws ssm get-parameter --name /aws/service/bottlerocket/aws-k8s-1.30/x86_64/latest/
image_id --region region-code --query "Parameter.Value" --output text
```

範例輸出如下。

```
ami-1234567890abcdef0
```

Bottlerocket 合規性支援

Bottlerocket 符合各種組織定義的建議：

- 為 Bottlerocket 定義了 CIS 基準。在預設組態中，Bottlerocket 映像具有 CIS 第 1 級組態設定檔所需的大部分控制項。您可以實作 CIS 第 2 級組態設定檔所需的控制項。如需詳細資訊，請參閱 AWS 部落格上的針對 CIS 基準驗證 Amazon EKS 優化 Bottlerocket AMI。
- 功能集優化和受攻擊面減少表示 Bottlerocket 執行個體只需較少的組態即可滿足 PCI DSS 需求。Bottlerocket 的 CIS 基準是強化指引的極佳資源，可支援您在 PCI DSS 需求 2.2 下對安全組態標準的需求。您也可以利用 Fluent Bit 來支援 PCI DSS 需求 10.2 下作業系統層級稽核日誌記錄的需求。AWS 定期發佈新的 (已修補的) Bottlerocket 執行個體，以協助您符合 PCI DSS 需求 6.2 (適用於 3.2.1 版) 和需求 6.3.3 (適用於 4.0 版)。
- Bottlerocket 是符合 HIPAA 資格的功能，經授權可與 Amazon EC2 和 Amazon EKS 的受管制工作負載搭配使用。如需詳細資訊，請參閱 Amazon EKS 上的 HIPAA 安全和合規架構白皮書。

Amazon EKS 優Ubuntu化

Canonical 已與 Amazon EKS 合作，建立了可供您用於叢集的節點 AMI。

規範提供 built-for-purposeKubernetes節點操作系統映像。此最小化Ubuntu映像檔已針對 Amazon EKS 進行最佳化，並包含與共同開發的自訂 AWS 核心。 AWS如需詳細資訊，請參閱 UbuntuAmazon Elastic Kubernetes Service (E KS) 和. 正在啟動自我管理的 Ubuntu 節點 如需有關支援的資訊，請參閱 AWS Premium Support 常見問答集的第三方軟體章節。

Amazon EKS 最佳化 Windows AMI

Windows Amazon EKS 最佳化 AMI 是根據 Windows Server 2019 和 Windows Server 2022 所建置。這些 AMI 已設定為 Amazon EKS 節點的基礎映像。預設情況下，AMI 會包括以下組件：

- `kubelet`
- `kube-proxy`
- AWS 適用於的 IAM 驗證器 Kubernetes
- `csi-proxy`
- `containerd`

> **Note**
>
> 您可以使用 Microsoft 安全性更新指南，追蹤 Windows Server 的安全性或隱私權事件。

適用於 Amazon EKS 提供針對有下列變化之 Windows 容器最佳化的 AMI：

- Amazon EKS 最佳化 Windows Server 2019 Core AMI
- Amazon EKS 最佳化 Windows Server 2019 Full AMI
- Amazon EKS 最佳化 Windows Server 2022 Core AMI
- Amazon EKS 最佳化 Windows Server 2022 Full AMI

> **Important**
>
> - Amazon ECS 最佳化 Windows Server 20H2 Core AMI 已棄用。不再發佈此 AMI 的新版本。

- 為了確保您預設擁有最新的安全更新，Amazon EKS 會在過去 4 個月維持最佳化的 Windows AMI。從初始發布之日起，每個新 AMI 都可以使用 4 個月。在此期間之後，較舊的 AMI 會設為私有且無法再存取。我們鼓勵使用最新的 AMI 來避免安全漏洞，並失去對已達到支持生命週期結束的舊 AMI 的訪問權限。雖然我們無法保證我們可以提供對已設為私人 AMI 的存取權，但您可以透過提交機票來要求存取權。 AWS Support

版本發佈日曆

下表列出 Amazon EKS 上 Windows 版本的發佈和終止支援日期。如果終止日期為空白，則是因為版本仍然受到支援。

Windows 版本	Amazon EKS 發佈	Amazon EKS 終止支援
Windows Server 2022 Core	10/17/2022	
Windows Server 2022 Full	10/17/2022	
Windows Server 20H2 Core	8/12/2021	8/9/2022
Windows Server 2004 Core	8/19/2020	12/14/2021
Windows Server 2019 Core	10/7/2019	
Windows Server 2019 Full	10/7/2019	
Windows Server 1909 Core	10/7/2019	12/8/2020

引導指令碼組態參數

建立 Windows 節點時，節點上會有允許設定不同參數的指令碼。根據設定，可以在節點上類似於以下位置的地方找到該指令碼：`C:\Program Files\Amazon\EKS\Start-EKSBootstrap.ps1`。您可將自訂參數值指定為引導指令碼的引數，以指定自訂參數值。例如，您可以更新啟動範本中的使用者資料。如需詳細資訊，請參閱 Amazon EC2 使用者資料。

該指令碼包括下列命令列參數：

- `-EKSClusterName`：指定此工作節點要加入的 Amazon EKS 叢集名稱。
- `-KubeletExtraArgs`：指定 `kubelet` 額外的引數 (選用)。

- `-KubeProxyExtraArgs`：指定 kube-proxy 額外的引數 (選用)。
- `-APIServerEndpoint`：指定 Amazon EKS 叢集 API 伺服器端點 (可選)。僅在搭配 `-Base64ClusterCA` 使用時有效。略過呼叫 `Get-EKSCluster`。
- `-Base64ClusterCA`：指定 Base64 編碼的叢集 CA 內容 (可選)。僅在搭配 `-APIServerEndpoint` 使用時有效。略過呼叫 `Get-EKSCluster`。
- `-DNSClusterIP`：覆寫要用於叢集內 DNS 查詢的 IP 地址 (選用)。根據主要介面的 IP 地址，預設為 `10.100.0.10` 或 `172.20.0.10`。
- `-ServiceCIDR` – 覆寫叢集服務從中尋址的 Kubernetes 服務 IP 地址範圍。根據主要介面的 IP 地址，預設為 `172.20.0.0/16` 或 `10.100.0.0/16`。
- `-ExcludedSnatCIDRs`：要從來源網路地址轉譯 (SNAT) 排除的 IPv4 CIDR 清單。這表示 VPC 可定址的 Pod 私有 IP 不會轉譯為傳出流量執行個體 ENI 主要 IPv4 地址的 IP 位址。依預設，會新增用於 Amazon EKS Windows 節點之 VPC 的IPv4 CIDR。將 CIDR 指定給此參數也會另外排除指定的 CIDR。如需詳細資訊，請參閱 適用於 Pods 的 SNAT。

除了命令列參數之外，您還可以指定一些環境變數參數。指定命令列參數時，它的優先順序高於個別的環境變數。環境變數應定義為機器 (或系統) 範圍，因為引導指令碼只會讀取機器範圍的變數。

此指令碼會考慮下列環境變數：

- `SERVICE_IPV4_CIDR`：請參閱 `ServiceCIDR` 命令列參數了解定義。
- `EXCLUDED_SNAT_CIDRS`：應為逗號分隔的字串。請參閱 `ExcludedSnatCIDRs` 命令列參數了解定義。

使用 **eksctl** 啟動自我管理 Windows Server 2022 節點

您可以使用下列 **`test-windows-2022.`**`yaml` 作為將 Windows Server 2022 作為自我管理節點執行的參考。使用您自己的值取代每一個 *`example value`*。

> **Note**
>
> 您必須使用 `eksctl` 版本 `0.116.0` 或更新版本才能執行自我管理的 Windows Server 2022 節點。

```
apiVersion: eksctl.io/v1alpha5
```

```
kind: ClusterConfig

metadata:
  name: windows-2022-cluster
  region: region-code
  version: '1.30'

nodeGroups:
  - name: windows-ng
    instanceType: m5.2xlarge
    amiFamily: WindowsServer2022FullContainer
    volumeSize: 100
    minSize: 2
    maxSize: 3
  - name: linux-ng
    amiFamily: AmazonLinux2
    minSize: 2
    maxSize: 3
```

然後，可以使用以下命令建立節點群組。

```
eksctl create cluster -f test-windows-2022.yaml
```

gMSA 身分驗證支援

Amazon EKS Windows Pods 允許不同類型的群組受管服務帳戶 (gMSA) 身分驗證。

- Amazon EKS 支援 Active Directory 網域身分進行身分驗證。如需有關加入網域的 gMSA 的詳細資訊，請參閱 AWS 部落格上的 Amazon EKS Windowspods 上的 Windows 身分驗證。
- Amazon EKS 提供的外掛程式可讓 non-domain-joinedWindows節點擷取具有可攜式使用者身分的gMSA登入資料。如需有關無網域 gMSA 的詳細資訊，請參閱 AWS 部落格上的 Amazon EKS Windowspods 的無網域 Windows 身分驗證。

快取容器映像

Amazon EKS Windows 優化 AMI 具有針對運行時緩存的某些容器映像。`containerd`使用 Amazon 受管建置元件建置自訂 AMI 時，會快取容器映像。如需詳細資訊，請參閱 使用 Amazon 受管建置元件。

下列的快取容器映像適用於 `containerd` 執行期：

- amazonaws.com/eks/pause-windows
- mcr.microsoft.com/windows/nanoserver
- mcr.microsoft.com/windows/servercore

其他資訊

如需使用 Amazon EKS 最佳化 Windows AMI 的詳細資訊，請參閱下列區段：

- 若要將 Windows 與受管節點群組搭配使用，請參閱 受管節點群組。
- 若要啟動自我管理的 Windows 節點，請參閱 正在啟動自我管理的 Windows 節點。
- 如需版本資訊，請參閱Amazon EKS 最佳化 Windows AMI 版本。
- 若要擷取 Amazon EKS 最佳化 Windows AMI 的最新 ID，請參閱 擷取 Amazon EKS 最佳化 Windows AMI ID。
- 若要使用 Amazon EC2 Image Builder 來建立自訂 Amazon EKS 最佳化 Windows AMI，請參閱 建立自訂 Amazon EKS 優化 Windows AMI。
- 如需最佳實務，請參閱《EKS 最佳實務指南》中的 Amazon EKS 最佳化 Windows AMI 管理。

Amazon EKS 最佳化 Windows AMI 版本

> **Important**
>
> 針對由發佈的 Amazon EKS 最佳化 Windows AMI 的延伸 Support AWS 不適用於Kubernetes版本，1.23但適用於版本1.24及更高Kubernetes版本。

本主題列出 Amazon EKS 最佳化 Windows AMI 的版本及其對應的kubeletcontainerd、和版本。csi-proxy

可透過程式設計方式擷取每個變體的 Amazon EKS 最佳化 AMI 中繼資料 (包括 AMI ID)。如需詳細資訊，請參閱 擷取 Amazon EKS 最佳化 Windows AMI ID。

AMI 是由 Kubernetes 版本和 AMI 發行日期進行版本化，格式如下：

```
k8s_major_version.k8s_minor_version-release_date
```

> **Note**
>
> Amazon EKS 受管節點群組支援 2022 年 11 月和更高版本的 Windows AMI。

Amazon EKS 最佳化 Windows Server 2022 Core AMI

下表列出 Amazon EKS 最佳化 Windows Server 2022 Core AMI 的目前版本和先前版本。

Kubernetes version 1.30

Kubernetes 版本 **1.30**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1.29

Kubernetes 版本 **1.29**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	已將 containerd 升級至 1.7.11。已將 kubelet 升級至 1.29.3。
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	修復了暫停圖像被kubelet垃圾收集過程錯誤刪除的錯誤。
1.29-2024.01.11	1.29.0	1.6.18	1.1.2	Windows伺服器 2022 核心 AMI 上的排除獨立式Windows更新 KB5034439。KB 僅適用於具有獨立WinRE分割區的Windows安裝，而這些分割區不包含在我們的任何 Amazon EKS 最佳化 Windows AMI 中。

Kubernetes version 1.28

Kubernetes 版本 **1.28**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.28.8。
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.11	1.28.5	1.6.18	1.1.2	Windows伺服器 2022 核心 AMI 上的排除獨立式Windows更新 KB5034439。KB 僅適用於具有獨立WinRE分割區的Windows 安裝，而這些分割區不包含在我們的任何 Amazon EKS 最佳化 Windows AMI 中。
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetes 版本 **1.27**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.27.12。
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.27-2024.03.12	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.11	1.27.9	1.6.18	1.1.2	Windows伺服器 2022 核心 AMI 上的排除獨立式Windows更新 KB5034439。KB 僅適用於具有獨立WinRE分割區的Windows 安裝，而這些分割區不包含在我們的任何 Amazon EKS 最佳化 Windows AMI 中。
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
				變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	修正 containers-roadmap 問題 #2042，此為導致節點無法提取私有 Amazon ECR 映像的問題。
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes 版本 **1.26**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.26.15。
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.11	1.26.12	1.6.18	1.1.2	Windows伺服器 2022 核心 AMI 上的排除獨立式Windows更新 KB5034439。KB 僅適用於具有獨立WinRE分割區的Windows 安裝，而這些分割區不包含在我們的任何 Amazon EKS 最佳化 Windows AMI 中。
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.26.9。已新增引導指令碼環境

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
				變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	已將 Kubernetes 升級至 1.26.4。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1.25

Kubernetes 版本 **1.25**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.25-2024.03.12	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.11	1.25.16	1.6.18	1.1.2	Windows伺服器 2022 核心 AMI 上的排除獨立式Windows更新 KB5034439。KB 僅適用於具有獨立WinRE分割區的Windows 安裝，而這些分割區不包含在我們的任何 Amazon EKS 最佳化 Windows AMI 中。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.25.14。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	已將 Kubernetes 升級至 1.25.9。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes 版本 **1.24**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.11	1.24.17	1.6.18	1.1.2	Windows伺服器 2022 核心 AMI 上的排除獨立式Windows更新 KB5034439。KB 僅適用於具有獨立WinRE分割區的Windows 安裝，而這些分割區不包含在我們的任何 Amazon EKS 最佳化 Windows AMI 中。
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.24.17。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	已將 Kubernetes 升級至 1.24.13。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetes 版本降級為 1.24.7，因為 1.24.10 在 kube-proxy 中已報告問題。
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

Amazon EKS 最佳化 Windows Server 2022 Full AMI

下表列出 Amazon EKS 最佳化 Windows Server 2022 Full AMI 的目前版本和先前版本。

Kubernetes version 1.30

Kubernetes 版本 **1.30**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1.29

Kubernetes 版本 **1.29**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	已將 containerd 升級至 1.7.11。已將 kubelet 升級至 1.29.3。
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	修復了暫停圖像被kubelet垃圾收集過程錯誤刪除的錯誤。
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetes 版本 **1.28**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.28.8。
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetes 版本 **1.27**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.27.12。
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.27-2024.03.12	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	修正 containers-roadmap 問題 #2042，此為導致節點無法提取私有 Amazon ECR 映像的問題。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes 版本 **1.26**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.26.15。
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.26.9。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	已將 Kubernetes 升級至 1.26.4。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1.25

Kubernetes 版本 **1.25**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.25-2024.03.12	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.25.14。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	已將 Kubernetes 升級至 1.25.9。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes 版本 **1.24**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.24.17。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	已將 Kubernetes 升級至 1.24.13。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetes 版本降級為 1.24.7，因為 1.24.10 在 kube-proxy 中已報告問題。
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

Amazon EKS 最佳化 Windows Server 2019 Core AMI

下表列出 Amazon EKS 最佳化 Windows Server 2019 Core AMI 的目前版本和先前版本。

Kubernetes version 1.30

Kubernetes 版本 **1.30**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1.29

Kubernetes 版本 **1.29**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	已將 containerd 升級至 1.7.11。已將 kubelet 升級至 1.29.3。
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	修復了暫停圖像被kubelet垃圾收集過程錯誤刪除的錯誤。
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetes 版本 **1.28**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.28.8。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetes 版本 **1.27**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.27.12。
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	修正 containers-roadmap 問題 #2042，此為導致節點無法提取私有 Amazon ECR 映像的問題。
11.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes 版本 **1.26**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.26.15。
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.26.9。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
				API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	已將 Kubernetes 升級至 1.26.4。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1.25

Kubernetes 版本 **1.25**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.25-2024.03.13	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.25.14。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
				API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	已將 Kubernetes 升級至 1.25.9。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes 版本 **1.24**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.24.17。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	已將 Kubernetes 升級至 1.24.13。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetes 版本降級為 1.24.7，因為 1.24.10 在 kube-proxy 中已報告問題。
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.11.08	1.24.7	1.6.6	1.1.1	

Amazon EKS 最佳化 Windows Server 2019 Full AMI

下表列出 Amazon EKS 最佳化 Windows Server 2019 Full AMI 的目前版本和先前版本。

Kubernetes version 1.30

Kubernetes 版本 **1.30**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1.29

Kubernetes 版本 **1.29**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	已將 containerd 升級至 1.7.11。已將 kubelet 升級至 1.29.3。
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	修復了暫停圖像被kubelet垃圾收集過程錯誤刪除的錯誤。
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetes 版本 **1.28**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.28.8。
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetes 版本 **1.27**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.27.12。
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	已修正 kubelet 的 安全性公告。
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	修正 containers-roadmap 問題 #2042，此為導致節點無法提取私有 Amazon ECR 映像的問題。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes 版本 **1.26**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。已將 kubelet 升級至 1.26.15。
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.26.9。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	已將 Kubernetes 升級至 1.26.4。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1.25

Kubernetes 版本 **1.25**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.
1.25-2024.03.13	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.25.14。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	已將 Kubernetes 升級至 1.25.9。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes 版本 **1.24**

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	已將 containerd 升級至 1.6.28。
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	已將 containerd 升級至 1.6.25。重建 CNI 並csi-proxy 使用 golang1.22.1.

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	包含 CVE-2023-5528 的修補程式。
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	已將 containerd 升級至 1.6.18。已將 kubelet 升級至 1.24.17。已新增引導指令碼環境變數 (SERVICE_IPV4_CIDR 和 EXCLUDED_SNAT_CIDRS)。
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	已升級 Amazon VPC CNI 外掛程式以便使用 Kubernetes 連接器二進制，該連接器從 Kubernetes API 伺服器獲取 Pod IP 位址。合併的提取請求#100。
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	包含 CVE-2023-3676 、CVE-2023-3893 和 CVE-2023-3955 的修補程式。
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.21	1.24.13	1.6.6	1.1.1	已解決導致 DNS 字尾搜尋清單填入錯誤的問題。
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	已將 Kubernetes 升級至 1.24.13。加入了對 CNI 中主機連接埠映射的支援。合併提取請求 #93。
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	修正節點重新啟動後，在 Pod 上造成網路連線問題 #1126 的錯誤。引入新的引導指令碼組態參數 (ExcludedSnatCIDRs)。
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	新增了服務當機時 kubelet 和 kube-proxy 的復原機制。
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	安裝無網域 gMSA 外掛程式，以便在 Amazon EKS 上針對 Windows 容器進行 gMSA 身分驗證。
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetes 版本降級為 1.24.7，因為 1.24.10 在 kube-proxy 中已報告問題。
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	

AMI 版本	**kubelet** 版本	**containerd** 版本	**csi-proxy** 版本	版本備註
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.12	1.24.7	1.6.6	1.1.1	

擷取 Amazon EKS 最佳化 Windows AMI ID

您可以透過查詢 AWS Systems Manager 參數存放區 API，以程式設計方式擷取 Amazon EKS 最佳化 AMI 的 Amazon 機器映像 (AMI) 識別碼。此參數讓您無需手動查詢 Amazon EKS 最佳化 AMI ID。如需有關 Systems Manager 參數存放區 API 的詳細資訊，請參閱GetParameter。您使用的 IAM 主體必須擁有 `ssm:GetParameter` IAM 許可，才能擷取 Amazon EKS 最佳化 AMI 中繼資料。

您可以透過使用子參數 `image_id` 的方式，以下列命令擷取最新建議的 Amazon EKS 最佳化 Windows AMI 的映像檔 ID。您可以將 *1.30* 換成任何支援的 Amazon EKS 版本，也可以將 *region-code* 換成需要取得 AMI ID 的 Amazon EKS 支援區域。以 `Full` 取代 *Core* 以查看 Windows Server 完整 AMI ID。對於 Kubernetes `1.24` 版或更新版本，您可以將 *2019* 取代為 2022 以查看 Windows Server 2022 AMI ID。

```
aws ssm get-parameter --name /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-EKS_Optimized-1.30/image_id --region region-code --query "Parameter.Value" --output text
```

範例輸出如下。

```
ami-1234567890abcdef0
```

建立自訂 Amazon EKS 優化 Windows AMI

您可以透過下列其中一種選項，使用 EC2 Image Builder 來建立自訂 Amazon EKS 最佳化 Windows AMI：

- 將 Amazon EKS 最佳化 Windows AMI 用作基礎
- 使用 Amazon 受管建置元件

兩種方法都需要您建立自己的 Image Builder 配方。如需詳細資訊，請參閱《Image Builder 使用者指南》中的建立映像配方的新版本。

> **Important**
>
> 下列 eks 專用的 Amazon 受管元件包含 CVE-2023-5528 的修補程式。
>
> - 1.24.3 和更高版本
> - 1.25.2 和更高版本
> - 1.26.2 和更高版本
> - 1.27.0 和更高版本
> - 1.28.0 和更高版本

將 Amazon EKS 最佳化 Windows AMI 用作基礎

此選項是建立自訂 Windows AMI 的建議方式。相較於 Amazon 受管建置元件，我們提供的 Amazon EKS 最佳化 Windows AMI 會更頻繁地更新。

1. 啟動新的 Image Builder 配方。
   a. 在 https://console.aws.amazon.com/imagebuilder 開啟 EC2 Image Builder 主控台。
   b. 在左側導覽窗格中，選擇映像配方。
   c. 選擇建立映像配方。
2. 在配方詳細資訊區段中，輸入名稱和版本。
3. 在基礎映像區段中指定 Amazon EKS 最佳化 Windows AMI 的 ID。
   a. 選擇輸入自訂 AMI ID。
   b. 擷取所需的 Windows 作業系統版本 AMI ID。如需詳細資訊，請參閱 擷取 Amazon EKS 最佳化 Windows AMI ID。
   c. 輸入自訂 AMI ID。如果找不到 AMI ID，請確定 AWS 區域 AMI ID 與主機右上方 AWS 區域顯示的相符。
4. (選用) 若要取得最新的安全性更新，請在建置元件：區段中新增 update-windows 元件。

a. 從依名稱尋找元件搜尋方塊右側的下拉式清單中，選擇 Amazon 受管。

b. 在依名稱尋找元件搜尋方塊中，輸入 **update-windows**。

c. 選取 **update-windows** 搜尋結果的核取方塊。此元件包含作業系統的最新 Windows 修補程式。

5. 完成具備您所需之組態的剩餘映像配方輸入。如需詳細資訊，請參閱《Image Builder 使用者指南》中的建立映像配方的新版本 (主控台)。

6. 選擇建立配方。

7. 在新的或現有的映像管道中使用新的映像配方。映像管道成功執行後，您的自訂 AMI 將被列為輸出映像並可供使用。如需詳細資訊，請參閱使用 EC2 Image Builder 主控台精靈建立映像管道。

使用 Amazon 受管建置元件

當無法將 Amazon EKS 最佳化 Windows AMI 用作基礎時，您可以改用 Amazon 受管建置元件。在最新支援的 Kubernetes 版本中，此選項可能會延遲。

1. 啟動新的 Image Builder 配方。

a. 在 https://console.aws.amazon.com/imagebuilder 開啟 EC2 Image Builder 主控台。

b. 在左側導覽窗格中，選擇映像配方。

c. 選擇建立映像配方。

2. 在配方詳細資訊區段中，輸入名稱和版本。

3. 在基礎映像區段中，決定您將用來建立自訂 AMI 的選項：

- 選取受管映像：對於您的映像作業系統 (OS) 選擇 Windows。然後對於映像來源，選擇下列其中一個選項：
  - Quick start (Amazon 受管)：在映像名稱下拉式選單中，選擇 Amazon EKS 支援的 Windows Server 版本。如需詳細資訊，請參閱 Amazon EKS 最佳化 Windows AMI。
  - 我擁有的映像：對於映像名稱，請使用您自己的授權選擇您自己的映像 ARN。您提供的映像尚未安裝 Amazon EKS 元件。
- 輸入自訂 AMI ID：對於 AMI ID，請使用您自己的授權輸入 AMI 的 ID。您提供的映像尚未安裝 Amazon EKS 元件。

4. 在建置元件 – 視窗區段中，執行下列操作：

a. 從依名稱尋找元件搜尋方塊右側的下拉式清單中，選擇 Amazon 受管。

b. 在依名稱尋找元件搜尋方塊中，輸入 **eks**。

c. 選取 **eks-optimized-ami-windows** 搜尋結果的核取方塊，即使傳回的結果可能不是您想要的版本。

d. 在依名稱尋找元件搜尋方塊中，輸入 **update-windows**。

e. 選取 update-windows 搜尋結果的核取方塊。此元件包含作業系統的最新 Windows 修補程式。

5. 在選取元件區段中，執行下列操作：

a. 選擇 **eks-optimized-ami-windows** 的版本選項。

b. 選擇指定元件版本。

c. 在元件版本欄位中輸入 ***version.x*** ，以支援 Kubernetes 的版本取代 *version*。輸入版本號碼片段的 *x*，即表示使用最新的元件版本，其亦符合您明確定義的版本片段。請注意控制台輸出，因為它會告知您所需的版本是否可作為受管元件使用。請記住，最新 Kubernetes 版本可能不適用於建置元件。如需有關可用版本的詳細資訊，請參閱 擷取 eks-optimized-ami-windows 元件版本的相關資訊。

> **Note**
>
> 下列 eks-optimized-ami-windows 建置元件版本需要 eksctl 版本 0.129 或更低版本：
>
> - 1.24.0

6. 完成具備您所需之組態的剩餘映像配方輸入。如需詳細資訊，請參閱《Image Builder 使用者指南》中的建立映像配方的新版本 (主控台)。

7. 選擇建立配方。

8. 在新的或現有的映像管道中使用新的映像配方。映像管道成功執行後，您的自訂 AMI 將被列為輸出映像並可供使用。如需詳細資訊，請參閱使用 EC2 Image Builder 主控台精靈建立映像管道。

擷取 **eks-optimized-ami-windows** 元件版本的相關資訊

您可以擷取與每個元件安裝之內容相關的特定資訊。例如，您可以確認安裝的 kubelet 版本。這些元件會在 Amazon EKS 支援的 Windows 作業系統版本上進行功能測試。如需詳細資訊，請參閱 版本發佈日曆。任何其他未列為支援之 Windows 作業系統版本，或已達到終止支援者可能與元件不相容。

1. 在 https://console.aws.amazon.com/imagebuilder 開啟 EC2 Image Builder 主控台。

2. 在左側導覽窗格中選擇元件。
3. 從依名稱尋找元件搜尋方塊右側的下拉式清單中，將我擁有的變更為 Quick start (Amazon 受管)。
4. 在 Find components by name (依名稱尋找元件) 方塊中，輸入 **eks**。
5. (選用) 如果您使用的是最新版本，請選擇兩次，將版本欄以遞減順序排序。
6. 選擇具有所需版本的 **eks-optimized-ami-windows** 鏈接。

結果頁面中的描述會顯示特定資訊。

儲存

本章涵蓋 Amazon EKS 叢集的儲存選項。

主題

Amazon EBS CSI 驅動程式

Amazon Elastic Block Store (Amazon EBS) 容器儲存介面 (CSI) 驅動程式可管理 Amazon EBS 磁碟區的生命週期，作為您所建立Kubernetes磁碟區的儲存空間。Amazon EBS CSI 驅動程式會為下列類型的磁碟區建立 Amazon EBS 磁碟Kubernetes區：一般暫時磁碟區和持續性磁碟區。

以下是使用 Amazon EBS CSI 驅動程序時需要考慮的一些事項。

- Amazon EBS CSI 外掛程式需要 IAM 許可，才能代表您呼叫 AWS API。如需詳細資訊，請參閱 建立 Amazon EBS CSI 驅動程式 IAM 角色。
- 您無法將 Amazon EBS 磁碟區掛載到 Fargate Pods。
- 您可以在 Fargate 節點上執行 Amazon EBS CSI 控制器，但是 Amazon EBS CSI 節點 DaemonSet 僅能在 Amazon EC2 執行個體上執行。

首次建立叢集時，並不會安裝 Amazon EBS CSI 驅動程式。若要使用驅動程式，您必須將其新增為 Amazon EKS 附加元件或自我管理附加元件。

- 如需如何將其新增為 Amazon EKS 附加元件的說明，請參閱 以 Amazon EKS 附加元件形式管理 Amazon EBS CSI 驅動程式。

- 如需如何將其新增為自我管理安裝的說明，請參閱 GitHub 上的 Amazon EBS 容器儲存介面 (CSI) 驅動程式專案。

使用任一方法安裝 CSI 驅動程式後，您可以使用範例應用程式來測試功能。如需詳細資訊，請參閱 部署範例應用程式並確認 CSI 驅動程式正常運作。

建立 Amazon EBS CSI 驅動程式 IAM 角色

Amazon EBS CSI 外掛程式需要 IAM 許可，才能代表您呼叫 AWS API。如需詳細資訊，請參閱 GitHub 上的設定驅動程式許可。

> **Note**
>
> Pods 可以存取指派給 IAM 角色的許可，除非您封鎖對 IMDS 的存取。如需詳細資訊，請參閱 Amazon EKS 的安全最佳實務。

必要條件

- 現有的叢集。
- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。

以下程序說明如何建立 IAM 角色，並將 AWS 受管政策連接到該角色。您可以使用 `eksctl`、 AWS Management Console或 AWS CLI。

> **Note**
>
> 本程序中的特定步驟是針對將驅動程式用作 Amazon EKS 附加元件而編寫。需要執行不同的步驟，才能將驅動程式用作自我管理附加元件。如需詳細資訊，請參閱 GitHub 上的設定驅動程式許可。

eksctl

使用 **eksctl** 建立 Amazon EBS CSI 外掛程式 IAM 角色

1. 建立 IAM 角色並附加政策。 AWS 維護受 AWS 管政策，或者您可以建立自己的自訂原則。您可以使用以下命令建立 IAM 角色並附加 AWS 受管政策。使用您叢集的名稱取代 *my-cluster*。該命令會部署建立 IAM 角色並將 IAM 政策附加到其中的 AWS CloudFormation 堆疊。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov:

```
eksctl create iamserviceaccount \
    --name ebs-csi-controller-sa \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKS_EBS_CSI_DriverRole \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEBSCSIDriverPolicy \
    --approve
```

2. 如果在 Amazon EBS 磁碟區上使用自訂 KMS 金鑰進行加密，請視需要自訂 IAM 角色。例如，請執行以下操作：

   a. 複製以下程式碼並貼到新 *kms-key-for-encryption-on-ebs*.json 檔案中。將 *custom-key-arn* 取代為自訂 KMS 金鑰 ARN。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

b. 建立政策。您可以將 *KMS_Key_For_Encryption_On_EBS_Policy* 變更為不同名稱。但是，如果您這樣做，請務必在稍後的步驟中也進行變更。

```
aws iam create-policy \
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. 使用以下命令將 IAM 政策連接至角色。使用您的帳戶 ID 取代 *111122223333*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

AWS Management Console

若要建立您的 Amazon EBS CSI 外掛程式 IAM 角色 AWS Management Console

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 在 Roles (角色) 頁面上，選擇 Create role (建立角色)。
4. 在 Select trusted entity (選取信任的實體) 頁面上，執行以下作業：

a. 在 Trusted entity type (信任的實體類型) 區段中，選擇 Web identity (Web 身分)。

b. 在 Identity provider (身分提供者) 欄位，為您的叢集選擇 OpenID Connect provider URL (供應商 URL) (如 Amazon EKS Overview (概觀) 下所示)。

c. 針對 Audience (對象)，選擇 `sts.amazonaws.com`。

d. 選擇 Next (下一步)。

5. 在 Add permissions (新增許可) 頁面上，執行以下作業：

a. 在 Filter policies (篩選政策) 方塊中，輸入 `AmazonEBSCSIDriverPolicy`。

b. 勾選在搜尋中傳回的 `AmazonEBSCSIDriverPolicy` 左側的核取方塊。

c. 選擇 Next (下一步)。

6. 在 Name, review, and create (命名、檢閱和建立) 頁面上，執行以下作業：

a. 針對 Role name (角色名稱)，為您的角色輸入唯一名稱 (例如 ***AmazonEKS_EBS_CSI_DriverRole***)。

b. 藉由連接標籤做為鍵值對，在新增標籤 (選用) 下將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的標記 IAM 資源。

c. 選擇建立角色。

7. 建立角色之後，在主控台中選擇角色，以開啟角色進行編輯。

8. 選擇 Trust Relationships (信任關係) 標籤，然後選擇 Edit Trust Relationship (編輯信任政策)。

9. 查找與下列行相似的行：

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

在上一行的末尾新增一個逗號，然後在上一行之後新增下一行。*region-code*以叢集所 AWS 區域 在的位置取代。將 *EXAMPLED539D4633E53DE1B71EXAMPLE* 取代為叢集的 OIDC 供應商 ID。

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:ebs-csi-controller-sa"
```

10. 選擇 Update policy (更新政策) 以完成操作。

11. 如果在 Amazon EBS 磁碟區上使用自訂 KMS 金鑰進行加密，請視需要自訂 IAM 角色。例如，請執行以下操作：

a. 在左側導覽窗格中選擇 Policies (政策)。

b. 在 Policies (政策) 頁面上，選擇 Create a policy (建立政策)。

c. 在建立政策頁面上，選擇 JSON 標籤。

d. 複製以下程式碼並貼到編輯器中，將 *custom-key-arn* 取代為自訂 KMS 金鑰 ARN。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

e. 選擇下一步：標籤。

f. 在 Add tags (Optional) (新增標籤 (選用)) 頁面上，選擇 Next: Review (下一步：檢閱)。

g. 在 Name (名稱) 中輸入您的政策名稱 (例如 ***KMS_Key_For_Encryption_On_EBS_Policy***)。

h. 選擇建立政策。

i. 在左側導覽窗格中，選擇 Roles (角色)。

j. DriverRole在控制台中選擇***### _EBS_CSI_*** 以將其打開以進行編輯。

k. 在新增許可下拉式清單中，選擇連接政策。

l. 在 Filter policies (篩選政策) 方塊中，輸入 *KMS_Key_For_Encryption_On_EBS_Policy*。

m. 勾選在搜尋中傳回的 *KMS_Key_For_Encryption_On_EBS_Policy* 左側的核取方塊。

n. 選擇連接政策。

AWS CLI

若要建立您的 Amazon EBS CSI 外掛程式 IAM 角色 AWS CLI

1. 檢視叢集的 OIDC 提供商 URL。使用您的叢集名稱取代 *my-cluster*。如果來自命令的輸出是 None，則請檢閱先決條件。

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

範例輸出如下。

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. 建立 IAM 角色，授予其 AssumeRoleWithWebIdentity 動作。

a. 將以下內容複製到名為 *aws-ebs-csi-driver-trust-policy*.json 的檔案。使用您的帳戶 ID 取代 *111122223333*。使用前一個步驟傳回的值取代 *EXAMPLED539D4633E53DE1B71EXAMPLE* 和 *region-code*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:ebs-csi-controller-sa"
        }
      }
    }
  ]
}
```

b. 建立角色。您可以將 *AmazonEKS_EBS_CSI_DriverRole* 變更為不同名稱。若要變更名稱，請務必在稍後的步驟中進行變更。

```
aws iam create-role \
  --role-name AmazonEKS_EBS_CSI_DriverRole \
  --assume-role-policy-document file://"aws-ebs-csi-driver-trust-
policy.json"
```

3. 附加策略。 AWS 維護受 AWS 管政策，或者您可以建立自己的自訂原則。使用下列命令將 AWS 受管理的原則附加至角色。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

4. 如果在 Amazon EBS 磁碟區上使用自訂 KMS 金鑰進行加密，請視需要自訂 IAM 角色。例如，請執行以下操作：

a. 複製以下程式碼並貼到新 *kms-key-for-encryption-on-ebs*.json 檔案中。將 *custom-key-arn* 取代為自訂 KMS 金鑰 ARN。

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

b. 建立政策。您可以將 *KMS_Key_For_Encryption_On_EBS_Policy* 變更為不同名稱。但是，如果您這樣做，請務必在稍後的步驟中也進行變更。

```
aws iam create-policy \
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. 使用以下命令將 IAM 政策連接至角色。使用您的帳戶 ID 取代 *111122223333*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
```

```
    --role-name AmazonEKS_EBS_CSI_DriverRole
```

現在，您已建立 Amazon EBS CSI 驅動程式 IAM 角色，即可繼續進行 新增 Amazon EBS CSI 驅動程式附加元件。當您在該程序中部署外掛程式時，其會建立並設定為使用名為 ebs-csi-controller-sa 的服務帳戶。服務帳戶會繫結至指派所需 Kubernetes 許可的 Kubernetes clusterrole。

以 Amazon EKS 附加元件形式管理 Amazon EBS CSI 驅動程式

若要提升安全性並減少工作量，您可以用 Amazon EKS 附加元件形式來管理 Amazon EBS CSI 驅動程式。如需 Amazon EKS 附加元件的詳細資訊，請參閱 Amazon EKS 附加元件。您可以依照 新增 Amazon EBS CSI 驅動程式附加元件 中的步驟來新增 Amazon EBS CSI 附加元件。

如果已新增 Amazon EBS CSI 附加元件，可以依照 將 Amazon EBS CSI 驅動程式作為 Amazon EKS 附加元件進行更新 和 移除 Amazon EBS CSI 附加元件 小節中的步驟加以管理。

必要條件

- 現有的叢集。若要查看所需的平台版本，請執行下列命令。

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver
```

- 叢集的現有 AWS Identity and Access Management (IAM) OpenID Connect (OIDC) 供應商。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。
- Amazon EBS CSI 驅動程式 IAM 角色。如果您不滿足此先決條件，則請嘗試安裝附加元件並執行 kubectl describe pvc 會顯示 failed to provision volume with StorageClass 以及 could not create volume in EC2: UnauthorizedOperation 錯誤。如需詳細資訊，請參閱建立 Amazon EBS CSI 驅動程式 IAM 角色。
- 若您使用叢集範圍受限制的 PodSecurityPolicy，請確認已授予附加元件足夠的許可以供部署。有關每個附加元件所需的權限Pod，請參閱上的相關附加元件資訊清單定義 GitHub。

> **Important**
>
> 若要使用 Amazon EBS CSI 驅動程式的快照功能，您必須在安裝附加元件之前安裝外部快照。必須依照下列順序安裝外部快照元件：
>
> - volumesnapshotclasses、volumesnapshots，和 volumesnapshotcontents 的 CustomResourceDefinition(CRD)
> - RBAC(ClusterRole、ClusterRoleBinding等)

- 控制器部署

如需詳細資訊，請參閱 GitHub 上的 CSI Snapshotter (CSI 快照)。

新增 Amazon EBS CSI 驅動程式附加元件

> **Important**
>
> 將 Amazon EBS 驅動程式新增為 Amazon EKS 附加元件之前，請確認您的叢集上並未安裝自我管理的驅動程式版本。如果是這樣，請參閱 GitHub 上的解除安裝自我管理的 Amazon EBS CSI 驅動程式。

您可以使用 `eksctl`、AWS Management Console 或 AWS CLI 將 Amazon EBS CSI 附加元件新增到您的叢集。

eksctl

使用 **`eksctl`** 來新增 Amazon EBS CSI 附加元件

執行下列命令。使用您的叢集名稱取代 *`my-cluster`*、您的帳戶 ID 取代 *`111122223333`*，再以先前建立的 IAM 角色名稱取代 *`AmazonEKS_EBS_CSI_DriverRole`*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部)AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

```
eksctl create addon --name aws-ebs-csi-driver --cluster my-cluster --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
```

如果您移除 **`--force`** 選項，且任何 Amazon EKS 附加元件設定與現有設定衝突，然後更新 Amazon EKS 附加元件失敗，則您會收到錯誤訊息以協助您解決衝突。指定此選項之前，請確定 Amazon EKS 附加元件未管理您需要管理的設定，因為這些設定會被此選項覆寫。如需有關此設定之其他選項的詳細資訊，請參閱 `eksctl` 文件中的 Addons (附加元件)。如需有關 Amazon EKS Kubernetes 欄位管理的詳細資訊，請參閱 Kubernetes 欄位管理。

AWS Management Console

使用 AWS Management Console 來新增 Amazon EBS CSI 附加元件

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。

2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 選取要為其設定 Amazon EBS CSI 附加元件的叢集名稱。
4. 選擇附加元件索引標籤。
5. 選擇取得更多附加元件。
6. 在選取附加元件頁面上，執行下列動作：
   a. 在 Amazon EKS-外掛程式區段中，選取 Amazon EBS CSI 驅動程式核取方塊。
   b. 選擇下一步。
7. 在設定選取的附加元件設定頁面上，執行以下操作：
   a. 選取您要使用的版本。
   b. 針對選取 IAM 角色，選取已連接 Amazon EBS CSI 驅動程式 IAM 政策的 IAM 角色名稱。
   c. (選用) 您可以展開選用組態設定。對於衝突解決方法，如果您選取覆寫，就可以使用 Amazon EKS 附加元件的設定來覆寫現有附加元件的一種或多種設定。若未啟用此選項，而且有設定與現有設定發生衝突，則操作會失敗。您可以使用產生的錯誤訊息對此衝突進行疑難排解。在選取此選項之前，請確認 Amazon EKS 附加元件未管理您需要自我管理的設定。
   d. 選擇下一步。
8. 在檢閱並新增頁面上，選擇建立。附加元件安裝完成後，您會看到已安裝的附加元件。

AWS CLI

使用 AWS CLI 來新增 Amazon EBS CSI 附加元件

執行下列命令。使用您的叢集名稱取代 *my-cluster*、使用您的帳戶 ID 取代 *111122223333*，再以先前建立的角色名稱取代 *AmazonEKS_EBS_CSI_DriverRole*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部)AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole
```

現在，您已將 Amazon EBS CSI 驅動程式新增為 Amazon EKS 附加元件，可繼續進行 部署範例應用程式並確認 CSI 驅動程式正常運作。該程序包括設定儲存體方案。

將 Amazon EBS CSI 驅動程式作為 Amazon EKS 附加元件進行更新

當新版本發行或當您將叢集更新至新的 Kubernetes 次要版本後，Amazon EKS 不會自動為叢集更新 Amazon EBS CSI。若要更新現有叢集上的 Amazon EBS CSI，您必須啟動更新，然後 Amazon EKS 會為您更新附加元件。

eksctl

使用 **eksctl** 來更新 Amazon EBS CSI 的附加元件

1. 請檢查您 Amazon EBS CSI 附加元件的目前版本。使用您的叢集名稱取代 *my-cluster*。

```
eksctl get addon --name aws-ebs-csi-driver --cluster my-cluster
```

範例輸出如下。

```
NAME                    VERSION                 STATUS  ISSUES  IAMROLE
 UPDATE AVAILABLE
aws-ebs-csi-driver      v1.11.2-eksbuild.1      ACTIVE  0
  v1.11.4-eksbuild.1
```

2. 將附加元件更新為上一步驟輸出中的 UPDATE AVAILABLE 傳回的版本。

```
eksctl update addon --name aws-ebs-csi-driver --version v1.11.4-eksbuild.1 --
cluster my-cluster \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
```

如果您移除 **--force** 選項，且任何 Amazon EKS 附加元件設定與現有設定衝突，然後更新 Amazon EKS 附加元件失敗，則您會收到錯誤訊息以協助您解決衝突。指定此選項之前，請確定 Amazon EKS 附加元件未管理您需要管理的設定，因為這些設定會被此選項覆寫。如需有關此設定之其他選項的詳細資訊，請參閱 eksctl 文件中的 Addons (附加元件)。如需有關 Amazon EKS Kubernetes 欄位管理的詳細資訊，請參閱 Kubernetes 欄位管理。

AWS Management Console

使用 AWS Management Console 來更新 Amazon EBS CSI 的附加元件

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。

2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 選取要更新的 Amazon EBS CSI 附加元件的叢集名稱。
4. 選擇附加元件索引標籤。
5. 選擇 Amazon EBS CSI 驅動程式。
6. 選擇編輯。
7. 在設定 Amazon EBS CSI 驅動程式頁面中，執行下列動作：
   a. 選取您要使用的版本。
   b. 針對選取 IAM 角色，選取已連接 Amazon EBS CSI 驅動程式 IAM 政策的 IAM 角色名稱。
   c. (選用) 您可以展開選用組態設定並視需要修改。
   d. 選擇儲存變更。

AWS CLI

使用 AWS CLI 來更新 Amazon EBS CSI 的附加元件

1. 請檢查您 Amazon EBS CSI 附加元件的目前版本。使用您的叢集名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --query "addon.addonVersion" --output text
```

範例輸出如下。

```
v1.11.2-eksbuild.1
```

2. 判斷哪些版本的 Amazon EBS CSI 附加元件適用於您的叢集版本。

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver --kubernetes-
version 1.23 \
  --query "addons[].addonVersions[].[addonVersion,
 compatibilities[].defaultVersion]" --output text
```

範例輸出如下。

```
v1.11.4-eksbuild.1
True
v1.11.2-eksbuild.1
```

```
False
```

下方具有 True 的版本是建立附加元件時部署的預設版本。建立附加元件時部署的版本可能不是最新的可用版本。在先前的輸出中，建立附加元件時會部署最新版本。

3. 將附加元件更新為上一步驟輸出中傳回的 True 版本。若其在輸出中傳回，您也可以更新至較新的版本。

```
aws eks update-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --addon-version v1.11.4-eksbuild.1 \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --resolve-
conflicts PRESERVE
```

PRESERVE (保留) 選項會保留您為附加元件設定的任何自訂設定。如需有關此設定其他選項的詳細資訊，請參閱 Amazon EKS Command Line Reference (《Amazon EKS 命令列參考》) 中的 update-addon。關於 Amazon EKS 附加元件組態管理的詳細資訊，請參閱 Kubernetes 欄位管理。

移除 Amazon EBS CSI 附加元件

有兩種選項可以用來移除 Amazon EKS 附加元件。

- Preserve add-on software on your cluster (在叢集上保留附加元件軟體)：此選項會移除任何設定的 Amazon EKS 管理。其也會移除 Amazon EKS 通知您更新的功能，並在您啟動更新後自動更新 Amazon EKS 附加元件。不過，該選項會保留您叢集上的附加元件軟體。此選項會使附加元件成為自我管理安裝，而不是 Amazon EKS 附加元件。如果啟用此選項，附加元件就無須停機。本程序中的命令使用此選項。
- Remove add-on software entirely from your cluster (從叢集中完全移除附加元件軟體)：只有叢集上沒有資源依賴於附加元件提供的功能時，我們才會建議您將 Amazon EKS 附加元件從叢集中移除。若要執行此選項，請從您在此程序中使用的命令刪除 `--preserve`。

若附加元件具有與其相關聯的 IAM 帳戶，則不會移除該 IAM 帳戶。

您可以使用 `eksctl`，AWS Management Console，或 AWS CLI以移除 Amazon EBS CSI 附加元件。

eksctl

使用 **eksctl** 移除 Amazon EBS CSI 附加元件

使用您叢集的名稱取代 *my-cluster*，然後執行以下命令。

```
eksctl delete addon --cluster my-cluster --name aws-ebs-csi-driver --preserve
```

AWS Management Console

使用 AWS Management Console 移除 Amazon EBS CSI 附加元件

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 選取要移除 Amazon EBS CSI 附加元件的叢集名稱。
4. 選擇附加元件索引標籤。
5. 選擇 Amazon EBS CSI 驅動程式。
6. 選擇移除。
7. 在「移除: aws-ebs-csi-driver確認」對話方塊中，執行下列動作：
   a. 若希望 Amazon EKS 停止管理附加元件的設定，請選取在叢集上保留。若要在叢集上保留附加元件軟體，請執行此動作。如此一來，您就可以自行管理附加元件的所有設定。
   b. 輸入 **aws-ebs-csi-driver**。
   c. 選取 Remove (移除)。

AWS CLI

使用 AWS CLI 移除 Amazon EBS CSI 附加元件

使用您叢集的名稱取代 *my-cluster*，然後執行以下命令。

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver --
preserve
```

部署範例應用程式並確認 CSI 驅動程式正常運作

您可以使用範例應用程式測試 CSI 驅動程式功能。本主題顯示了一個範例，但您也可以執行下列動作：

- 部署使用外部快照程式的範例應用程式，以建立磁碟區快照。如需詳細資訊，請參閱 GitHub 上的 Volume Snapshots (磁碟區快照)。
- 部署使用磁碟區大小調整的範例應用程式。如需詳細資訊，請參閱 GitHub 上的 Volume Resizing (調整磁碟區大小)。

此程序使用來自 Amazon EBS 容器儲存介面 (CSI) 驅動程式 GitHub 儲存庫的動態磁碟區佈建範例，以取用藉由動態方式佈建的 Amazon EBS 磁碟區。

1. 將 Amazon EBS 容器儲存介面 (CSI) 驅動程式 GitHub 儲存庫複製到您的本機系統。

   ```
   git clone https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git
   ```

2. 導覽至 dynamic-provisioning 範例目錄。

   ```
   cd aws-ebs-csi-driver/examples/kubernetes/dynamic-provisioning/
   ```

3. (選用) manifests/storageclass.yaml 檔案預設佈建 gp2 Amazon EBS 磁碟區。若要改用 gp3 磁碟區，請新增 type: gp3 至 manifests/storageclass.yaml。

   ```
   echo "parameters:
     type: gp3" >> manifests/storageclass.yaml
   ```

4. 從 ebs-sc 目錄部署 ebs-claim 儲存方案、app 持續性磁碟區宣告，以及 manifests 範例應用程式。

   ```
   kubectl apply -f manifests/
   ```

5. 描述 ebs-sc 儲存方案。

   ```
   kubectl describe storageclass ebs-sc
   ```

   範例輸出如下。

   ```
   Name:            ebs-sc
   ```

```
IsDefaultClass:  No
Annotations:     kubectl.kubernetes.io/last-applied-
configuration={"apiVersion":"storage.k8s.io/v1","kind":"StorageClass","metadata":
{"annotations":{},"name":"ebs-
sc"},"provisioner":"ebs.csi.aws.com","volumeBindingMode":"WaitForFirstConsumer"}

Provisioner:           ebs.csi.aws.com
Parameters:            <none>
AllowVolumeExpansion:  <unset>
MountOptions:          <none>
ReclaimPolicy:         Delete
VolumeBindingMode:     WaitForFirstConsumer
Events:                <none>
```

> **Note**
>
> 使用 WaitForFirstConsumer 磁碟區繫結模式的儲存方案。這表示在 Pod 提出持久性磁碟區宣告之前，不會對磁碟區執行動態佈建。如需詳細資訊，請參閱 Kubernetes 文件中的磁碟區繫結模式。

6. 在預設命名空間中監看 Pods。在幾分鐘後，app Pod 的狀態會變更為 Running。

```
kubectl get pods --watch
```

輸入 Ctrl+C 傳回 Shell 提示。

7. 在預設命名空間中列出持續性磁碟區。尋找具有 default/ebs-claim 宣告的持續性磁碟區。

```
kubectl get pv
```

範例輸出如下。

```
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY
   STATUS   CLAIM               STORAGECLASS   REASON   AGE
pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a   4Gi        RWO            Delete
   Bound    default/ebs-claim   ebs-sc                  30s
```

8. 描述持續性磁碟區。使用在上一個步驟中輸出的值取代 pvc-*37717cd6-d0dc-11e9-b17f-06fad4858a5a*。

```
kubectl describe pv pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
```

範例輸出如下。

```
Name:              pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
Labels:            <none>
Annotations:       pv.kubernetes.io/provisioned-by: ebs.csi.aws.com
Finalizers:        [kubernetes.io/pv-protection external-attacher/ebs-csi-aws-com]
StorageClass:      ebs-sc
Status:            Bound
Claim:             default/ebs-claim
Reclaim Policy:    Delete
Access Modes:      RWO
VolumeMode:        Filesystem
Capacity:          4Gi
Node Affinity:
  Required Terms:
    Term 0:        topology.ebs.csi.aws.com/zone in [region-code]
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume source)
    Driver:            ebs.csi.aws.com
    VolumeHandle:      vol-0d651e157c6d93445
    ReadOnly:          false
    VolumeAttributes:      storage.kubernetes.io/
csiProvisionerIdentity=1567792483192-8081-ebs.csi.aws.com
Events:                <none>
```

Amazon EBS 磁碟區 ID 是先前輸出中的 VolumeHandle 值。

9. 確認 Pod 正在將資料寫入磁碟區。

```
kubectl exec -it app -- cat /data/out.txt
```

範例輸出如下。

```
Wed May 5 16:17:03 UTC 2021
Wed May 5 16:17:08 UTC 2021
Wed May 5 16:17:13 UTC 2021
Wed May 5 16:17:18 UTC 2021
```

```
[...]
```

10. 完成後，請刪除此範例應用程式的資源。

```
kubectl delete -f manifests/
```

Amazon EBS CSI 遷移常見問答集

> **Important**
>
> 如果您具有在版本 1.22 或先前版本的叢集上執行的 Pods，則必須先安裝 Amazon EBS CSI 驅動程式，才能將叢集更新至版本 1.23，以避免服務中斷。

Amazon EBS 容器儲存介面 (CSI) 遷移功能，可將處理儲存作業的責任從 Amazon EBS 樹狀內 EBS 儲存佈建程式移至 Amazon EBS CSI 驅動程式。

CSI 驅動程式是什麼？

CSI 驅動程式：

- 更換存在於 Kubernetes 專案原始碼中的 Kubernetes 「樹狀內」儲存體驅動程式。
- 與儲存供應商合作，例如 Amazon EBS。
- 提供簡化的外掛程式模型，讓類似於 AWS 的儲存提供者能更輕鬆地在不依賴 Kubernetes 發行週期的情況下，發行功能與維護支援。

如需詳細資訊，請參閱 Kubernetes CSI 文件中的簡介。

什麼是 CSI 遷移？

Kubernetes CSI 遷移功能將處理儲存操作的責任，從現有的樹狀內儲存外掛程式 (例如 `kubernetes.io/aws-ebs`) 移動至對應的 CSI 驅動程式。只要安裝對應的 CSI 驅動程式，現有的 `StorageClass`、`PersistentVolume` 和 `PersistentVolumeClaim` (PVC) 物件會繼續運作。啟用此功能時：

- 利用 PVC 的現有工作負載會一如既往地運作。
- Kubernetes 將所有儲存管理操作的控制權傳遞給 CSI 驅動程式。

如需詳細資訊，請參閱 Kubernetes 部落格的 Kubernetes1.23：Kubernetes 樹狀內至 CSI 磁碟區遷移狀態更新。

為了幫助您從樹狀內外掛程式遷移到 CSI 驅動程式，Amazon EKS 版本 1.23 和更新版本叢集預設啟用 CSIMigration 和 CSIMigrationAWS 標記。這些標記可讓您的叢集將樹狀內 API 轉換為同等 CSI API。這些標記設定在由 Amazon EKS 管理的Kubernetes 控制平面，以及 Amazon EKS 最佳化 AMI 設定的 kubelet 設定。如果您有在叢集中使用 Amazon EBS 磁碟區的 Pods，您必須先安裝 Amazon EBS CSI 驅動程式，然後再將叢集更新為版本 **1.23**。如果不這樣做，磁碟區操作 (例如佈建和掛載) 可能無法如預期般運作。如需詳細資訊，請參閱Amazon EBS CSI 驅動程式。

> **Note**
>
> 樹狀內 StorageClass 佈建程式名為 kubernetes.io/aws-ebs。Amazon EBS CSI StorageClass 佈建程式名為 ebs.csi.aws.com。

我可以將 **kubernetes.io/aws-ebs StorageClass** 磁碟區掛載於版本 **1.23** 和更新版本的叢集嗎？

可以，只要 Amazon EBS CSI 驅動程式已安裝。對於新建立的版本 1.23 以及更新版本的叢集，我們建議您安裝 Amazon EBS CSI 驅動程式作為叢集建立程序的一部分。我們也建議只使用根據 ebs.csi.aws.com 佈建程式的 StorageClasses。

如果您已將叢集控制平面更新為版本 1.23，並尚未將節點更新為 1.23，則 CSIMigration 和CSIMigrationAWS kubelet 標記沒有啟用。在這種情況下，樹狀內驅動程式用於掛載基於 kubernetes.io/aws-ebs 的磁碟區。不過，仍必須安裝 Amazon EBS CSI 驅動程式，以確保使用基於 kubernetes.io/aws-ebs 磁碟區的 Pods 可以排程。其他磁碟區操作也需要此驅動程式才能成功。

我可以在 Amazon EKS **1.23** 和更新版本的叢集上佈建 **kubernetes.io/aws-ebs StorageClass** 磁碟區嗎？

可以，只要 Amazon EBS CSI 驅動程式已安裝。

`kubernetes.io/aws-ebs StorageClass` 佈建程式有可能從 Amazon EKS 移除嗎？

`kubernetes.io/aws-ebs` `StorageClass` 佈建程式和 `awsElasticBlockStore` 磁碟區類型已不再受支援，但目前尚未計劃移除這些佈建程式和磁碟區類型。這些資源會被視為 Kubernetes API 的一部分。

如何安裝 Amazon EBS CSI 驅動程式？

我們建議安裝 Amazon EBS CSI 驅動程式 Amazon EKS 附加元件。當需要更新 Amazon EKS 附加元件時，您啟動更新後，Amazon EKS 會為您更新附加元件。如果您想自行管理驅動程式，則可以使用開放原始碼 Helm chart 進行安裝。

> **Important**
>
> Kubernetes 樹狀內 Amazon EBS 驅動程式於 Kubernetes 控制平面上執行。其使用指派給 Amazon EKS 叢集 IAM 角色 的 IAM 許可來佈建 Amazon EBS 磁碟區。Amazon EBS CSI 驅動程式在節點上執行。驅動程式需要 IAM 許可才能佈建磁碟區。如需詳細資訊，請參閱建立 Amazon EBS CSI 驅動程式 IAM 角色。

如何檢查我的叢集中是否已安裝 Amazon EBS CSI 驅動程式？

若要判斷驅動程式是否安裝在您的叢集上，請執行下列命令：

```
kubectl get csidriver ebs.csi.aws.com
```

若要檢查該安裝是否由 Amazon EKS 管理，請執行下列命令：

```
aws eks list-addons --cluster-name my-cluster
```

如果我還沒有安裝 Amazon EBS CSI 驅動程式，Amazon EKS 是否會阻止叢集更新到版本 `1.23`？

否。

如果我在將叢集更新為 1.23 版之前忘記安裝 Amazon EBS CSI 驅動程式，該怎麼辦？我可以在更新叢集後安裝驅動程式嗎？

可以，但是叢集更新後，要求 Amazon EBS CSI 驅動程式的磁碟區操作將會失敗，直到安裝驅動程式為止。

什麼是套用於新建立的 Amazon EKS 版本 **1.23** 和更新版本叢集的預設 **StorageClass**？

預設的 StorageClass 行為會保持不變。對於每個新叢集，Amazon EKS 都會套用基於 kubernetes.io/aws-ebs 的 StorageClass,名為 gp2。我們不打算從新建立的叢集永遠刪除此 StorageClass。這和叢集預設 StorageClass 是分開的，如果您在不指定磁碟區類型的情況下，建立以 ebs.csi.aws.com 為基礎的 StorageClass，Amazon EBS CSI 驅動程式將預設使用 gp3。

當我將叢集更新為版本 **1.23** 時，Amazon EKS 是否會對已存在於現有叢集中的 **StorageClasses** 進行任何變更？

否。

如何使用快照將持久性磁碟區從 **kubernetes.io/aws-ebsStorageClass** 遷移至 **ebs.csi.aws.com**？

若要遷移持久性磁碟區，請參閱 AWS 部落格的將 Amazon EKS 叢集從 gp2 遷移到 gp3 EBS 磁碟區。

如何使用註釋修改 Amazon EBS 磁碟區？

從 aws-ebs-csi-driver v1.19.0-eksbuild.2 開始，您可以使用 PersistentVolumeClaim (PVC) 中的註釋修改 Amazon EBS 磁碟區。新的磁碟區修改功能會實作為附屬，稱為 volumemodifier。如需詳細資訊，請參閱 AWS 部落格上的《使用 EBS CSI 驅動程式簡化 Kubernetes 上的 Amazon EBS 磁碟區遷移和修改作業》。

Windows 工作負載是否支援遷移？

是。如果您正在使用開放原始碼 Helm Chart 安裝 Amazon EBS CSI 驅動程式，請把 node.enableWindows 設定為 true。依預設，如果將 Amazon EBS CSI 驅動程式安裝為 Amazon EKS 附加元件。建立 StorageClasses 時，設定 fsType 至 Windows 檔案系統，例如 ntfs。接下來，Windows 工作負載的磁碟區操作會遷移至 Amazon EBS CSI 驅動程式，與 Linux 工作負載相同。

Amazon EFS CSI 驅動程式

Amazon Elastic File System (Amazon EFS) 提供無伺服器、完全彈性的檔案儲存功能，讓您無需佈建或管理儲存容量和效能，即可分享檔案資料。Amazon EFS 容器儲存界面 (CSI) 驅動程式提供 CSI 界面，可讓您在其上 AWS 執行的Kubernetes叢集管理 Amazon EFS 檔案系統的生命週期。本主題旨在說明如何將 Amazon EFS CSI 驅動程式部署到您的 Amazon EKS 叢集。

考量事項

- Amazon EFS CSI 驅動程式與以 Windows 為基礎的容器映像不相容。
- 您無法針對具有 Fargate 節點的持續性磁碟區使用動態佈建，但可以使用靜態佈建。
- 動態佈建需要1.2或更新版本的驅動程式。您可以在任何受支援1.1的 Amazon EKS 叢集版本上，使用驅動程式版本的持續性磁碟區靜態佈建。
- 此驅動程式的 1.3.2 或更新版本支援 Arm64 架構，包括基於 Amazon EC2 Graviton 的執行個體。
- 此驅動程式的 1.4.2 版或更新版本皆支援使用 FIPS 掛載檔案系統。
- 記下 Amazon EFS 的資源配額。例如，每個 Amazon EFS 檔案系統都有建立 1000 個存取點的配額。如需詳細資訊，請參閱無法變更的 Amazon EFS 資源配額。

必要條件

- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。

> **Note**
>
> 在上Pod執行時 AWS Fargate 會自動掛接 Amazon EFS 檔案系統。

建立 IAM 角色

Amazon EFS CSI 驅動程式需要 IAM 許可才能與檔案系統互動。建立 IAM 角色，並將所需的 AWS 受管政策附加至該角色。您可以使用 `eksctl`、 AWS Management Console或 AWS CLI。

> **Note**
>
> 本程序中的特定步驟是針對將驅動程式用作 Amazon EKS 附加元件而編寫。如需有關自我管理安裝的詳細資訊，請參閱 GitHub 上的 Set up driver permission 一節。

eksctl

若要使用 **`eksctl`** 建立您的 Amazon EFS CSI 驅動程式 IAM 角色

執行下列命令以建立 IAM 角色。將 *`my-cluster`* 取代為您的叢集名稱，並將 *`AmazonEKS_EFS_CSI_DriverRole`* 取代為您角色的名稱。

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEFSCSIDriverPolicy \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

AWS Management Console

若要建立您的 Amazon EFS CSI 驅動程式 IAM 角色，請使用 AWS Management Console

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 在 Roles (角色) 頁面上，選擇 Create role (建立角色)。
4. 在 Select trusted entity (選取信任的實體) 頁面上，執行以下作業：
   a. 在 Trusted entity type (信任的實體類型) 區段中，選擇 Web identity (Web 身分)。
   b. 在 Identity provider (身分提供者) 欄位，為您的叢集選擇 OpenID Connect provider URL (供應商 URL) (如 Amazon EKS Overview (概觀) 下所示)。
   c. 針對 Audience (對象)，選擇 `sts.amazonaws.com`。
   d. 選擇 Next (下一步)。
5. 在 Add permissions (新增許可) 頁面上，執行以下作業：
   a. 在 Filter policies (篩選政策) 方塊中，輸入 *AmazonEFSCSIDriverPolicy*。
   b. 勾選在搜尋中傳回的 *AmazonEFSCSIDriverPolicy* 左側的核取方塊。
   c. 選擇 Next (下一步)。
6. 在 Name, review, and create (命名、檢閱和建立) 頁面上，執行以下作業：
   a. 針對 Role name (角色名稱)，為您的角色輸入唯一名稱 (例如 ***AmazonEKS_EFS_CSI_DriverRole***)。
   b. 藉由連接標籤做為鍵值對，在新增標籤 (選用) 下將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的標記 IAM 資源。
   c. 選擇建立角色。
7. 建立角色之後，在主控台中選擇角色，以開啟角色進行編輯。
8. 選擇 Trust Relationships (信任關係) 標籤，然後選擇 Edit Trust Relationship (編輯信任政策)。
9. 查找與下列行相似的行：

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

在前一行上方加入下列行。*region-code*以叢集所 AWS 區域 在的位置取代。將 *EXAMPLED539D4633E53DE1B71EXAMPLE* 取代為叢集的 OIDC 供應商 ID。

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:efs-csi-*",
```

10. 將 Condition 運算子從 "StringEquals" 修改為 "StringLike"。
11. 選擇 Update policy (更新政策) 以完成操作。

AWS CLI

若要建立您的 Amazon EFS CSI 驅動程式 IAM 角色，請使用 AWS CLI

1. 檢視叢集的 OIDC 提供商 URL。使用您的叢集名稱取代 *my-cluster*。如果來自命令的輸出是 None，則請檢閱先決條件。

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

範例輸出如下。

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. 建立授予 AssumeRoleWithWebIdentity 動作許可的 IAM 角色。

a. 將下列內容複製到名為 *aws-efs-csi-driver-trust-policy*.json 的檔案。使用您的帳戶 ID 取代 *111122223333*。使用前一個步驟傳回的值取代 *EXAMPLED539D4633E53DE1B71EXAMPLE* 和 *region-code*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
```

```
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:efs-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. 建立角色。您可以變更 *AmazonEKS_EFS_CSI_DriverRole* 為不同的名稱，但如果這樣做，請務必在稍後的步驟中進行變更。

```
aws iam create-role \
  --role-name AmazonEKS_EFS_CSI_DriverRole \
  --assume-role-policy-document file://"aws-efs-csi-driver-trust-
policy.json"
```

3. 使用下列命令將必要的 AWS 受管理原則附加至角色。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
  --role-name AmazonEKS_EFS_CSI_DriverRole
```

安裝 Amazon EFS CSI 驅動程式

我們建議您透過 Amazon EKS 附加元件安裝 Amazon EFS CSI 驅動程式。若要將 Amazon EKS 附加元件新增至叢集，請參閱 建立附加元件。如需附加元件的詳細資訊，請參閱 Amazon EKS 附加元件。如果您無法使用 Amazon EKS 附加元件，我們建議您提交有關為何無法這麼做的問題至容器藍圖 GitHub 儲存庫。

或者，如果您想要 Amazon EFS CSI 驅動程式的自我管理安裝，請參閱 GitHub 上的安裝。

建立 Amazon EFS 檔案系統

如需建立 Amazon EFS 檔案系統，請參閱 GitHub 上的為 Amazon EKS 建立 Amazon EFS 檔案系統。

部署範例應用程式

您可以部署各種範例應用程式，再視需要進行修改。如需詳細資訊，請參閱 GitHub 上所提供的範例。

Amazon FSx for Lustre CSI 驅動程式

FSx for Lustre 容器儲存介面 (CSI) 驅動程式提供 CSI 介面，允許 Amazon EKS 叢集管理 FSx for Lustre 檔案系統的生命週期。如需詳細資訊，請參閱《FSx for Lustre 使用者指南》。

本主題說明如何將 FSx for Lustre CSI 驅動程式部署到您的 Amazon EKS 叢集，並確認是否正常運作。我們建議您使用最新版的驅動程式。如需可用版本，請參閱（詳見）的 CSI 規格相容性矩陣 GitHub

> **Note**
>
> Fargate 不支援此驅動程式。

如需可用參數的詳細說明，以及示範驅動程式功能的完整範例，請參閱 GitHub 上的 FSx for Lustre Container Storage Interface (CSI) driver (FSx for Lustre 容器儲存介面 (CSI) 驅動程式) 專案。

必要條件

您必須有：

- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本`1.27.160`或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。`2.12.3`若要檢查您目前的版本，請使用 **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到您的主目錄〉。
- 已在裝置或 AWS CloudShell上安裝版本 `0.183.0` 或更新版本的 `eksctl` 命令列工具。如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的安裝一節。
- 已在裝置或 AWS CloudShell上安裝 `kubectl` 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 `1.29`，則

可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。

以下過程可協助您使用 FSx for Lustre CSI 驅動程式建立簡單的測試叢集，以便您查看其運作原理。不建議針對生產工作負載使用測試叢集。在本教學課程中，我們建議您使用 *example values*，除非其註明要替換。完成生產叢集的步驟時，您可以替換任何 *example value*。我們建議您完成同一個終端中的所有步驟，因為變數是在整個步驟中設定和使用，且不會存在於不同的終端中。

將 FSx for Lustre CSI 驅動程式部署至 Amazon EKS 叢集

1. 設定幾個變數以在剩餘步驟中使用。取代*my-csi-fsx-cluster*為您要建立的測試叢集的名稱，以及*region-code*您 AWS 區域 要在其中建立測試叢集的名稱。

```
export cluster_name=my-csi-fsx-cluster
export region_code=region-code
```

2. 建立測試叢集。

```
eksctl create cluster \
  --name $cluster_name \
  --region $region_code \
  --with-oidc \
  --ssh-access \
  --ssh-public-key my-key
```

叢集佈建需要幾分鐘的時間。在叢集建立期間，您會看到數行輸出。輸出的最後一行類似於下面的範例行。

```
[#]  EKS cluster "my-csi-fsx-cluster" in "region-code" region is ready
```

3. 建立驅動程式的Kubernetes服務帳戶，並使用下列命令將 AmazonFSxFullAccess AWS-managed 政策附加至服務帳戶。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov:

```
eksctl create iamserviceaccount \
  --name fsx-csi-controller-sa \
  --namespace kube-system \
  --cluster $cluster_name \
  --attach-policy-arn arn:aws:iam::aws:policy/AmazonFSxFullAccess \
  --approve \
```

```
  --role-name AmazonEKSFSxLustreCSIDriverFullAccess \
  --region $region_code
```

建立服務帳戶時，您會看到數行輸出。輸出的最後一行類似於下列內容。

```
[#]  1 task: {
    2 sequential sub-tasks: {
        create IAM role for serviceaccount "kube-system/fsx-csi-controller-sa",
        create serviceaccount "kube-system/fsx-csi-controller-sa",
    } }
[#]  building iamserviceaccount stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  deploying stack "eksctl-my-csi-fsx-cluster-addon-iamserviceaccount-kube-
system-fsx-csi-controller-sa"
[#]  waiting for CloudFormation stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  created serviceaccount "kube-system/fsx-csi-controller-sa"
```

請記下已部署的 AWS CloudFormation 堆疊名稱。在先前的範例輸出中，堆疊被命名為 eksctl-*my-csi-fsx-cluster*-addon-iamserviceaccount-kube-system-*fsx-csi-controller-sa*。

4. 使用下列命令部署驅動程式。使用所需的分支取代 *release-X.XX*。主要分支不受支援，因為其所包含的即將推出的功能，可能與當前發布的驅動程式穩定版本不相容。我們建議使用最新發行版本。有aws-fsx-csi-driver關活動分支的列表，請參閱（詳見） GitHub。

> **Note**
>
> 您可以在 GitHub 上檢視正在 aws-fsx-csi-driver 中使用的內容。

```
kubectl apply -k "github.com/kubernetes-sigs/aws-fsx-csi-driver/deploy/kubernetes/
overlays/stable/?ref=release-X.XX"
```

範例輸出如下。

```
serviceaccount/fsx-csi-controller-sa created
serviceaccount/fsx-csi-node-sa created
clusterrole.rbac.authorization.k8s.io/fsx-csi-external-provisioner-role created
clusterrole.rbac.authorization.k8s.io/fsx-external-resizer-role created
```

```
clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-external-provisioner-binding
 created
clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-resizer-binding created
deployment.apps/fsx-csi-controller created
daemonset.apps/fsx-csi-node created
csidriver.storage.k8s.io/fsx.csi.aws.com created
```

5. 請注意所建立角色的 ARN 。如果您之前沒有註意到它，並且在 AWS CLI 輸出中不再提供它，則可以執行以下操作以在 AWS Management Console.

   a. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation

   b. 確定主控台已設定為您在其中 AWS 區域 建立 IAM 角色的主控台，然後選取 [堆疊]。

   c. 選取名為 `eksctl-`*`my-csi-fsx-cluster`*`-addon-iamserviceaccount-kube-system-`*`fsx-csi-controller-sa`* 的堆疊。

   d. 選取 Outputs (輸出) 標籤。Role1 ARN 會列在 Output (1) (輸出 (1)) 頁面上。

6. 修補驅動程式部署，以新增您先前使用下列命令建立的服務帳戶。將 ARN 取代為您記下的 ARN。使用您的帳戶 ID 取代 *`111122223333`*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

   ```
   kubectl annotate serviceaccount -n kube-system fsx-csi-controller-sa \
     eks.amazonaws.com/role-
   arn=arn:aws:iam::111122223333:role/AmazonEKSFSxLustreCSIDriverFullAccess --
   overwrite=true
   ```

   範例輸出如下。

   ```
   serviceaccount/fsx-csi-controller-sa annotated
   ```

部署 Kubernetes 儲存方案、持續性磁碟區宣告和範例應用程式，以確認 CSI 驅動程式是否正常運作

此程序使用 FSx for Lustre 容器儲存介面 (CSI) 驅動程式 GitHub 儲存庫，來耗用動態佈建的 FSx for Lustre 磁碟區。

1. 記下叢集的安全群組。您可以在 AWS Management Console 「網絡」部分下或使用以下 AWS CLI 命令中看到它。

```
aws eks describe-cluster --name $cluster_name --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

2. 根據《Amazon FSx for Lustre 使用者指南》中的 Amazon VPC 安全群組中顯示的標準，為您的 Amazon FSx 檔案系統建立安全群組。對於 VPC，選擇叢集的 VPC，如聯網區段所示。對於「與 Lustre 用戶端關聯的安全群組」，請使用您的叢集安全群組。您可以單獨保留傳出規則，以允許 All traffic (所有流量)。
3. 執行以下命令，下載儲存類別清單檔案。

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/storageclass.yaml
```

4. 編輯 `storageclass.yaml` 檔案的參數區段。使用您自己的值取代每一個 *`example value`*。

```
parameters:
  subnetId: subnet-0eabfaa81fb22bcaf
  securityGroupIds: sg-068000ccf82dfba88
  deploymentType: PERSISTENT_1
  automaticBackupRetentionDays: "1"
  dailyAutomaticBackupStartTime: "00:00"
  copyTagsToBackups: "true"
  perUnitStorageThroughput: "200"
  dataCompressionType: "NONE"
  weeklyMaintenanceStartTime: "7:09:00"
  fileSystemTypeVersion: "2.12"
```

- **`subnetId`**：應該在其中建立Amazon FSx for Lustre 檔案系統的子網路 ID。並非所有可用區域都支援 Amazon FSx for Lustre。開啟位於 https://console.aws.amazon.com/fsx/ 的 Amazon FSx for Lustre 主控台，確認您要使用的子網路位於支援的可用區域中。子網路可以包含您的節點，也可以是不同的子網路或 VPC：
  - 您可以選取 [計算] 區段下的 AWS Management Console 節點群組，以檢查中的節點子網路。
  - 若您指定的子網路與您擁有節點的子網路不同，則您的 VPC 必須已連線，且您必須確認安全群組中必要的連接埠已開啟。
- **`securityGroupIds`**：您為檔案系統所建立的安全群組 ID。
- **`deploymentType`** (選用)：檔案系統部署類型。有效值為 `SCRATCH_1`、`SCRATCH_2`、`PERSISTENT_1`、`PERSISTENT_2`。如需部署類型的詳細資訊，請參閱建立您的 Amazon FSx for Lustre 檔案系統。

- 其他參數（可選）— 有關其他參數的更多信息，請參閱編輯 StorageClassGitHub。

5. 建立儲存類別清單檔案。

```
kubectl apply -f storageclass.yaml
```

範例輸出如下。

```
storageclass.storage.k8s.io/fsx-sc created
```

6. 下載持續性磁碟區宣告清單檔案。

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/claim.yaml
```

7. (選用) 編輯 claim.yaml 檔案。根據您的儲存要求和您在上一個步驟中選取的 deploymentType，將 *1200Gi* 變更為下列某一個增量值。

```
storage: 1200Gi
```

- SCRATCH_2 和 PERSISTENT：**1.2 TiB** 或 **2.4 TiB**，超過 2.4 TiB 則以 2.4 TiB 為單位遞增。
- SCRATCH_1：**1.2 TiB**、**2.4 TiB** 或 **3.6 TiB**，超過 3.6 TiB 則以 3.6 TiB 為單位遞增。

8. 建立持續性磁碟區宣告。

```
kubectl apply -f claim.yaml
```

範例輸出如下。

```
persistentvolumeclaim/fsx-claim created
```

9. 確認已佈建檔案系統。

```
kubectl describe pvc
```

範例輸出如下。

```
Name:          fsx-claim
Namespace:     default
```

```
StorageClass:  fsx-sc
Status:        Bound
[...]
```

> **Note**
>
> Status 可能會顯示為 Pending 約 5-10 分鐘，然後變更為 Bound。在 Status 成為 Bound 之前，請勿繼續下一步。如果 Status 顯示 Pending 超過 10 分鐘，請使用 Events 中的警告訊息作為解決任何問題的參考。

10. 部署範例應用程式。

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/examples/kubernetes/dynamic_provisioning/specs/pod.yaml
```

11. 確認範例應用程式正在執行。

```
kubectl get pods
```

範例輸出如下。

```
NAME      READY   STATUS              RESTARTS   AGE
fsx-app   1/1     Running             0          8s
```

12. 驗證應用程式是否正確掛載了檔案系統。

```
kubectl exec -ti fsx-app -- df -h
```

範例輸出如下。

```
Filesystem                  Size  Used Avail Use% Mounted on
overlay                      80G  4.0G   77G   5% /
tmpfs                        64M     0   64M   0% /dev
tmpfs                       3.8G     0  3.8G   0% /sys/fs/cgroup
192.0.2.0@tcp:/abcdef01     1.1T  7.8M  1.1T   1% /data
/dev/nvme0n1p1               80G  4.0G   77G   5% /etc/hosts
shm                          64M     0   64M   0% /dev/shm
tmpfs                       6.9G   12K  6.9G   1% /run/secrets/kubernetes.io/
serviceaccount
tmpfs                       3.8G     0  3.8G   0% /proc/acpi
```

```
tmpfs                           3.8G     0  3.8G   0% /sys/firmware
```

13. 確認範例應用程式是否已將資料寫入 FSx for Lustre 檔案系統。

```
kubectl exec -it fsx-app -- ls /data
```

範例輸出如下。

```
out.txt
```

此範例輸出顯示範例應用程式成功地將 `out.txt` 檔案寫入檔案系統。

> **Note**
>
> 刪除叢集之前，請確認刪除 FSx for Lustre 檔案系統。如需詳細資訊，請參閱《FSx for Lustre 使用者指南》中的清除資源。

適用於 NetApp ONTAP CSI 驅動程式的 Amazon FSx

NetApp'sAstra Trident使用容器儲存介面 (CSI) 相容驅動程式，提供動態儲存協調作業。這可讓 Amazon EKS 叢集管理由 Amazon FSx 支援的 NetApp ONTAP 檔案系統持續性磁碟區 (PV) 的生命週期。若要開始使用，請參閱文件中的將阿斯特拉三叉戟搭配 Amazon FSx 用於 NetApp ONTAP 使用。Astra Trident

適用於 NetApp ONTAP 的 Amazon FSx 是一種儲存服務，可讓您在雲端中啟動和執行全受管ONTAP 檔案系統。 ONTAP是NetApp's檔案系統技術，提供一組廣泛採用的資料存取和資料管理功能。FSx for ONTAP 提供內部部署NetApp檔案系統的功能、效能和 API，同時具備全代管服 AWS 務的敏捷性、可擴充性和簡易性。如需詳細資訊，請參閱《FSx for ONTAP 使用者指南》。

Amazon FSx for OpenZFS CSI 驅動程式

Amazon FSx for OpenZFS 是一項全受管檔案儲存服務，可讓您輕鬆將資料從內部部署 ZFS 或其他以 Linux 為基礎的檔案伺服器移至 AWS。您可以執行此動作，而無需變更應用程式程式碼或資料管理方式。這項服務提供建置在開放原始碼 OpenZFS 檔案系統上的高度可靠、可擴展、高效且功能豐富的檔案儲存。其將這些功能與全受管 AWS 服務的敏捷性、可擴展性和簡易性相結合。如需詳細資訊，請參閱《Amazon FSx for OpenZFS 使用者指南》。

Amazon FSx for OpenZFS 容器儲存介面 (CSI) 驅動程式提供 CSI 介面，允許 Amazon EKS 叢集管理 Amazon FSx for OpenZFS 磁碟區的生命週期。若要將 Amazon FSx for OpenZFS 驅動程式部署到您的 Amazon EKS 叢集，請參閱 GitHub 上的 aws-fsx-openzfs-csi-driver。

mazon File Cache CSI 驅動程式

Amazon 檔案快取是 AWS 上一種全受管的高速快取，可用來處理檔案資料，無論資料存放在何處。Amazon 檔案快取會在資料第一次存取時自動將資料載入快取，並在未使用資料時將其釋放。如需詳細資訊，請參閱 Amazon File Cache 使用者指南。

Amazon File Cache 容器儲存介面 (CSI) 驅動程式提供 CSI 介面，允許 Amazon EKS 叢集管理 Amazon 檔案快取的生命週期。若要將 Amazon File Cache CSI 驅動程式部署到您的 Amazon EKS 叢集，請參閱 GitHub 上的 aws-file-cache-csi-driver。

適用於 Amazon S3 CSI 驅動程式的 Mountpoint

使用適用Mountpoint於 Amazon S3 容器儲存界面 (CSI) 驅動程式，您的Kubernetes應用程式可以透過檔案系統界面存取 Amazon S3 物件，在不變更任何應用程式程式碼的情況下實現高彙總輸送量。CSI 驅動程式以適用於 Amazon S3 的 Mountpoint 為建置基礎，將 Amazon S3 儲存貯體呈現為可透過在 Amazon EKS 中的容器和自我管理 Kubernetes 叢集存取的磁碟區。本主題旨在說明如何將適用於 Amazon S3 CSI 驅動程式的 Mountpoint 部署到您的 Amazon EKS 叢集。

考量事項

- 適用於 Amazon S3 CSI 驅動程式的 Mountpoint 目前與以 Windows 為基礎的容器映像不相容。
- 適用於 Amazon S3 CSI 驅動程式的 Mountpoint 不支援 AWS Fargate。但是，系統支援在 Amazon EC2 (使用 Amazon EKS 或自訂 Kubernetes 安裝) 中執行的容器。
- 適用於 Amazon S3 CSI 驅動程式的 Mountpoint 僅支援靜態佈建。不支援動態佈建或建立新儲存貯體。

> **Note**
>
> 靜態佈建是指使用指定為物件`bucketName`中的現有 Amazon S3 儲存貯`PersistentVolume`體。`volumeAttributes`如需詳細資訊，請參閱 GitHub 上的靜態佈建。

- 使用適用於 Amazon S3 CSI 驅動程式的 Mountpoint 掛載的磁碟區不支援所有 POSIX 檔案系統功能。如需檔案系統行為的詳細資訊，請參閱 GitHub 上的適用於 Amazon S3 的 Mountpoint 之檔案系統行為。

事前準備

- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。
- 您裝置上 AWS CLI 安裝和設定的 2.12.3 版或 AWS CloudShell更新版本。
- 已在裝置或 AWS CloudShell上安裝 `kubectl` 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 `1.29`，則可以搭配使用 `kubectl` `1.28`、`1.29` 或 `1.30` 版。若要安裝或升級 `kubectl`，請參閱 安裝或更新 `kubectl`。

建立 IAM 政策

適用於 Amazon S3 CSI 驅動程式的 Mountpoint 需要 Amazon S3 許可才能與檔案系統互動。此節將介紹如何建立可授予必要許可的 IAM 政策。

下列範例政策遵循 Mountpoint 的 IAM 許可建議。或者，您可以使用 AWS 受管理的策略 AmazonS3FullAccess，但是此受管理策略授與的權限超過所需的權限Mountpoint。

如需 Mountpoint 建議許可的詳細資訊，請參閱 GitHub 上的 Mountpoint IAM 許可。

使用 IAM 主控台建立 IAM 政策

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中選擇 Policies (政策)。
3. 在政策頁面上，選擇建立政策。
4. 對於政策編輯器，選擇 JSON。
5. 在政策編輯器下，複製並貼上以下內容：

> **Important**
>
> 將 `DOC-EXAMPLE-BUCKET1` 取代為您自己 Amazon S3 儲存貯體的名稱。

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Sid": "MountpointFullBucketAccess",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
            ]
        },
        {
            "Sid": "MountpointFullObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ]
        }
   ]
}
```

Amazon S3 Express 單區儲存類別引入的目錄儲存貯體使用與一般用途儲存貯體不同的身份驗證機制。您應該使用`s3:*`動作，而不是使用`s3express:CreateSession`動作。如需目錄儲存貯體的相關資訊，請參閱 Amazon S3 使用者指南中的目錄儲存貯體。

以下是用於目錄值區的最低權限原則範例。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3express:CreateSession",
```

```
            "Resource": "arn:aws:s3express:aws-region:111122223333:bucket/DOC-
EXAMPLE-BUCKET1--az_id--x-s3"
        }
    ]
}
```

6. 選擇下一步。
7. 在檢閱與建立頁面上，為您的政策命名。本範例逐步教學使用的名稱是 `AmazonS3CSIDriverPolicy`。
8. 選擇建立政策。

建立 IAM 角色

適用於 Amazon S3 CSI 驅動程式的 Mountpoint 需要 Amazon S3 許可才能與檔案系統互動。此節將介紹如何建立 IAM 角色以委派這些許可。您可使用`eksctl`、IAM 主控台或 AWS CLI來建立此角色。

> **Note**
>
> IAM 政策 `AmazonS3CSIDriverPolicy` 於上一節建立。

eksctl

使用 **`eksctl`** 建立適用於 Amazon S3 CSI 驅動程式的 Mountpoint IAM 角色

執行以下命令來建立 IAM 角色和 Kubernetes 服務帳戶。這些命令也會將 `AmazonS3CSIDriverPolicy` IAM 政策連接至角色、使用 IAM 角色 Amazon Resource Name (ARN) 標註 Kubernetes 服務帳戶 (`s3-csi-controller-sa`)，並將 Kubernetes 服務帳戶名稱新增至 IAM 角色的信任政策。

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
```

```
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

IAM console

若要建立您Mountpoint的 Amazon S3 CSI 驅動程式 IAM 角色，請使用 AWS Management Console

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 在 Roles (角色) 頁面上，選擇 Create role (建立角色)。
4. 在 Select trusted entity (選取信任的實體) 頁面上，執行以下作業：
   a. 在 Trusted entity type (信任的實體類型) 區段中，選擇 Web identity (Web 身分)。
   b. 在 Identity provider (身分提供者) 欄位，為您的叢集選擇 OpenID Connect provider URL (供應商 URL) (如 Amazon EKS Overview (概觀) 下所示)。

      如果未顯示 URL，請檢視先決條件區段。
   c. 針對 Audience (對象)，選擇 `sts.amazonaws.com`。
   d. 選擇 Next (下一步)。
5. 在 Add permissions (新增許可) 頁面上，執行以下作業：
   a. 在 Filter policies (篩選政策) 方塊中，輸入 **`AmazonS3CSIDriverPolicy`**。

      > **Note**
      > 此政策於上一節建立。

   b. 勾選在搜尋中傳回的 `AmazonS3CSIDriverPolicy` 結果左側的核取方塊。
   c. 選擇 Next (下一步)。
6. 在 Name, review, and create (命名、檢閱和建立) 頁面上，執行以下作業：
   a. 針對 Role name (角色名稱)，為您的角色輸入唯一名稱 (例如 **`AmazonEKS_S3_CSI_DriverRole`**)。
   b. 藉由連接標籤做為鍵值對，在新增標籤 (選用) 下將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的標記 IAM 資源。

c. 選擇建立角色。

7. 建立角色之後，在主控台中選擇角色，以開啟角色進行編輯。
8. 選擇 Trust Relationships (信任關係) 標籤，然後選擇 Edit Trust Relationship (編輯信任政策)。
9. 查找與下列相似的行：

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

在上一行的末尾新增一個逗號，然後在上一行之後新增下一行。*region-code*以叢集所 AWS 區域 在的位置取代。將 *EXAMPLED539D4633E53DE1B71EXAMPLE* 取代為叢集的 OIDC 供應商 ID。

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:s3-csi-*"
```

10. 將 Condition 運算子從 "StringEquals" 變更為 "StringLike"。
11. 選擇 Update policy (更新政策) 以完成操作。

AWS CLI

若要建立您Mountpoint的 Amazon S3 CSI 驅動程式 IAM 角色，請使用 AWS CLI

1. 檢視叢集的 OIDC 提供者 URL。使用您叢集的名稱取代 *my-cluster*。如果來自命令的輸出是 None，則請檢閱先決條件。

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

範例輸出如下。

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. 建立 IAM 角色，將 Kubernetes 服務帳戶授予 AssumeRoleWithWebIdentity 動作。

a. 將下列內容複製到名為 *aws-s3-csi-driver-trust-policy*.json 的檔案。使用您的帳戶 ID 取代 *111122223333*。使用前一個步驟傳回的值取代 *EXAMPLED539D4633E53DE1B71EXAMPLE* 和 *region-code*。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:s3-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. 建立角色。您可以變更 *AmazonEKS_S3_CSI_DriverRole* 為不同的名稱，但如果這樣做，請務必在稍後的步驟中進行變更。

```
aws iam create-role \
  --role-name AmazonEKS_S3_CSI_DriverRole \
  --assume-role-policy-document file://"aws-s3-csi-driver-trust-policy.json"
```

3. 使用以下命令將先前建立的 IAM 政策連接至角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonS3CSIDriverPolicy \
  --role-name AmazonEKS_S3_CSI_DriverRole
```

> **Note**
>
> IAM 政策 AmazonS3CSIDriverPolicy 於上一節建立。

4. 如果您要將驅動程式安裝為 Amazon EKS 附加元件，請略過此步驟。若為驅動程式的自我管理安裝，請建立使用您所建立的 IAM 角色之 ARN 予以註釋的 Kubernetes 服務帳戶。

   a. 將下列內容儲存到名為 *mountpoint-s3-service-account*.yaml 的檔案中。使用您的帳戶 ID 取代 *111122223333*。

```
---
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/name: aws-mountpoint-s3-csi-driver
  name: mountpoint-s3-csi-controller-sa
  namespace: kube-system
  annotations:
    eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
```

   b. 在您的叢集上建立 Kubernetes 服務帳戶。Kubernetes 服務帳戶 (mountpoint-s3-csi-controller-sa) 使用您建立名為 *AmazonEKS_S3_CSI_DriverRole* 的 IAM 角色註釋。

```
kubectl apply -f mountpoint-s3-service-account.yaml
```

> **Note**
>
> 當您在此程序中部署外掛程式時，其會建立並設定為使用名為 s3-csi-driver-sa 的服務帳戶。

安裝適用於 Amazon S3 CSI 驅動程式的 Mountpoint

您可以透過 Amazon EKS 附加元件安裝適用於 Amazon S3 CSI 驅動程式的 Mountpoint。您可以使用eksctl AWS Management Console、或將附 AWS CLI 加元件新增至叢集。

您可以選擇將 Amazon S3 CSI 驅動程式安Mountpoint裝為自我管理安裝。如需執行自我管理安裝的說明，請參閱 GitHub 上的安裝。

eksctl

若要使用新增 Amazon S3 CSI 附加元件 **eksctl**

執行下列命令。使用您的叢集名稱取代 *my-cluster*、您的帳戶 ID 取代 *111122223333*，再以先前建立的 IAM 角色名稱取代 *AmazonEKS_S3_CSI_DriverRole*。

```
eksctl create addon --name aws-mountpoint-s3-csi-driver --cluster my-cluster --
service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
 --force
```

如果您移除 ***--force*** 選項，且任何 Amazon EKS 附加元件設定與現有設定衝突，然後更新 Amazon EKS 附加元件失敗，則您會收到錯誤訊息以協助您解決衝突。指定此選項之前，請確定 Amazon EKS 附加元件未管理您需要管理的設定，因為這些設定會被此選項覆寫。如需有關此設定之其他選項的詳細資訊，請參閱 `eksctl` 文件中的 Addons (附加元件)。如需有關 Amazon EKS Kubernetes 欄位管理的詳細資訊，請參閱 Kubernetes 欄位管理。

AWS Management Console

若要使Mountpoint用新增 Amazon S3 CSI 附加元件 AWS Management Console

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 選擇您要Mountpoint為其設定 Amazon S3 CSI 附加元件的叢集名稱。
4. 選擇附加元件索引標籤。
5. 選擇取得更多附加元件。
6. 在選取附加元件頁面上，執行下列動作：
   a. 在亞馬遜 EKS-外掛程式區段中，選取Mountpoint適用於 Amazon S3 CSI 驅動程式的核取方塊。
   b. 選擇下一步。
7. 在設定選取的附加元件設定頁面上，執行以下操作：
   a. 選取您要使用的版本。
   b. 對於選取 IAM 角色，請選取您附加給 Amazon S3 CSI 驅動程式 IAM 政策的 IAM 角色名稱。Mountpoint
   c. (選用) 您可以展開選用組態設定。對於衝突解決方法，如果您選取覆寫，就可以使用 Amazon EKS 附加元件的設定來覆寫現有附加元件的一種或多種設定。若未啟用此選項，

而且有設定與現有設定發生衝突，則操作會失敗。您可以使用產生的錯誤訊息對此衝突進行疑難排解。在選取此選項之前，請確認 Amazon EKS 附加元件未管理您需要自我管理的設定。

d. 選擇下一步。

8. 在檢閱並新增頁面上，選擇建立。附加元件安裝完成後，您會看到已安裝的附加元件。

AWS CLI

若要使Mountpoint用新增 Amazon S3 CSI 附加元件 AWS CLI

執行下列命令。使用您的叢集名稱取代 *my-cluster*、使用您的帳戶 ID 取代 *111122223333*，再以先前建立的角色名稱取代 *AmazonEKS_S3_CSI_DriverRole*。

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
```

設定適用於 Amazon S3 的 Mountpoint

在大多數情況下，您只能使用儲存貯體名稱為適用於 Amazon S3 的 Mountpoint 設定。如需設定適用於 Amazon S3 的 Mountpoint 之說明，請參閱 GitHub 上的設定適用於 Amazon S3 的 Mountpoint。

部署範例應用程式

您可以將靜態佈建部署到現有 Amazon S3 儲存貯體上的驅動程式。如需詳細資訊，請參閱 GitHub 上的靜態佈建。

刪除 Mountpoint Amazon S3 CSI 驅動程序

有兩種選項可以用來移除 Amazon EKS 附加元件。

- Preserve add-on software on your cluster (在叢集上保留附加元件軟體)：此選項會移除任何設定的 Amazon EKS 管理。其也會移除 Amazon EKS 通知您更新的功能，並在您啟動更新後自動更新 Amazon EKS 附加元件。不過，該選項會保留您叢集上的附加元件軟體。此選項會使附加元件成為自我管理安裝，而不是 Amazon EKS 附加元件。如果啟用此選項，附加元件就無須停機。本程序中的命令使用此選項。

- Remove add-on software entirely from your cluster (從叢集中完全移除附加元件軟體)：只有叢集上沒有資源依賴於附加元件提供的功能時，我們才會建議您將 Amazon EKS 附加元件從叢集中移除。若要執行此選項，請從您在此程序中使用的命令刪除 `--preserve`。

若附加元件具有與其相關聯的 IAM 帳戶，則不會移除該 IAM 帳戶。

您可以使用`eksctl` AWS Management Console、或移 AWS CLI 除 Amazon S3 CSI 附加元件。

eksctl

使用刪除 Amazon S3 CSI 附加組件 **`eksctl`**

使用您叢集的名稱取代 *`my-cluster`*，然後執行以下命令。

```
eksctl delete addon --cluster my-cluster --name aws-mountpoint-s3-csi-driver --
preserve
```

AWS Management Console

若要使用移除 Amazon S3 CSI 附加元件 AWS Management Console

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 選取要移除 Amazon EBS CSI 附加元件的叢集名稱。
4. 選擇附加元件索引標籤。
5. 選擇 MountpointAmazon S3 CSI 驅動程序。
6. 選擇移除。
7. 在「移除: aws-mountpoint-s3-csi 驅動程式確認」對話方塊中，執行下列動作：
   a. 若希望 Amazon EKS 停止管理附加元件的設定，請選取在叢集上保留。若要在叢集上保留附加元件軟體，請執行此動作。如此一來，您就可以自行管理附加元件的所有設定。
   b. 輸入 **`aws-mountpoint-s3-csi-driver`**。
   c. 選取 Remove (移除)。

AWS CLI

若要使用移除 Amazon S3 CSI 附加元件 AWS CLI

使用您叢集的名稱取代 *my-cluster*，然後執行以下命令。

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-driver --preserve
```

CSI 快照控制器

容器儲存介面 (CSI) 快照控制器可讓您在相容的 CSI 驅動程式 (例如 Amazon EBS CSI 驅動程式) 中使用快照拍攝功能。

以下是使用 CSI 快照控制器的一些注意事項。

- 快照控制器必須與具有快照拍攝功能的 CSI 驅動程式一起安裝。Amazon EBS CSI 驅動程式支援建立 Amazon EBS CSI 受管磁碟區的 Amazon EBS 快照。如需安裝指示，請參閱Amazon EBS CSI 驅動程式。
- Kubernetes 不支援透過 CSI 遷移提供的磁碟區快照，例如使用具有佈建程式 `kubernetes.io/aws-ebs` 的 `StorageClass` 之 Amazon EBS 磁碟區。必須使用參照 CSI 驅動程式佈建程式 (`ebs.csi.aws.com`) 的 `StorageClass` 來建立磁碟區。如需 CSI 遷移的詳細資訊，請參閱 Amazon EBS CSI 遷移常見問答集。

我們建議您透過 Amazon EKS 受管附加元件安裝 CSI 快照控制器。若要將 Amazon EKS 附加元件新增至叢集，請參閱 建立附加元件。如需附加元件的詳細資訊，請參閱 Amazon EKS 附加元件。

或者，如果您想要 Amazon EBS CSI 快照控制器的自我管理安裝，請參閱 GitHub 上游 Kubernetes `external-snapshotter` 中的使用。

Amazon EKS 聯網

您的 Amazon EKS 叢集是在 VPC 中建立。Pod 聯網功能由 Amazon VPC 容器網路介面 (CNI) 外掛程式提供。本章包括以下主題，可讓您進一步了解有關叢集聯網。

主題

Amazon EKS VPC 與子網要求和注意事項

當您建立叢集時，您可以指定 VPC 和至少兩個位於不同可用區域的子網。本主題提供您與叢集一起使用的 VPC 和子網的 Amazon EKS 特定需求和注意事項的概觀。如果您沒有 VPC 可與 Amazon EKS 一起使用，則可以使用 Amazon EKS 提供的模板創建一個 VPC。 AWS CloudFormation 如果您要在上建立本機或延伸叢集 AWS Outposts，請參閱Amazon EKS 本機叢集 VPC 與子網路要求和考量事項而非本主題。

VPC 要求和注意事項

當您建立叢集時，您指定的 VPC 必須符合下列要求和注意事項：

- 針對您要建立的叢集、任何節點和其他 Kubernetes 資源，VPC 必須有足夠數目的 IP 地址。如果要使用的 VPC 沒有足夠數目的 IP 地址，請嘗試增加可用 IP 地址的數目。

  您可以更新叢集組態來變更叢集使用的子網路和安全群組，藉此完成此操作。您可以從 AWS Management Console、、和版本的最新版本`v0.164.0-rc.0`或更新`eksctl`版本進行更新。 AWS CLI AWS CloudFormation這麼做才能為子網路提供更多可用 IP 地址，以成功升級叢集版本。

  > **Important**
  >
  > 您新增的所有子網路必須與建立叢集時原本提供的 AZ 位於同一組。新的子網路必須滿足所有其他要求，例如，它們必須具有足夠的 IP 地址。

舉例來說，假設您建立叢集並指定四個子網路。依照您指定它們的順序，第一個子網路位於 us-west-2a 可用區域中，第二個和第三個子網路位於 us-west-2b 可用區域中，而第四個子網路位於 us-west-2c 可用區域中。如果要變更子網路，則必須在三個可用區域中各提供至少一個子網路，而且子網路必須與原始子網路位於相同的 VPC 中。

如果您需要的 IP 地址超過 VPC 中的 CIDR 區塊，您可以將其他無類別域間路由 (CIDR) 區塊與 VPC 產生關聯，以新增其他 CIDR 區塊。您可以在建立叢集的之前或之後，將私有 (RFC 1918) 和公有 (非 RFC 1918) CIDR 區塊與 VPC 關聯。叢集可能最多需要五個小時才能識別您與 VPC 關聯的 CIDR 區塊。

您可以透過使用共享服務 VPC 轉換閘道來節省 IP 地址的使用。如需詳細資訊，請參閱隔離 VPC 與共享服務和混合網路中的 Amazon EKS VPC 可路由 IP 地址保護模式。

- 如果您希望 Kubernetes 指派 IPv6 地址至 Pods 和服務，請將 IPv6 CIDR 區塊與 VPC 建立關聯。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的建立 IPv6 CIDR 區塊與 VPC 的關聯。
- VPC 必須支援 DNS 主機名稱和 DNS 解析。否則，節點無法註冊至您的叢集。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 的 DNS 屬性。
- VPC 可能需要使用 VPC 端點。 AWS PrivateLink如需詳細資訊，請參閱 子網需求和注意事項。

如果您使用 Kubernetes 1.14 或舊版建立叢集，Amazon EKS 會將以下標籤新增至您的 VPC：

金鑰	值
kubernetes.io/cluster/ *my-cluster*	owned

此標籤僅供 Amazon EKS 使用。您可以在不影響服務的情況下移除該標籤。其不與版本 1.15 或更新版本的叢集一起使用。

子網需求和注意事項

當您建立叢集時，Amazon EKS 會在您指定的子網中建立 2-4 個彈性網路介面。這些網路介面啟用您的叢集和 VPC 之間的通訊。這些網路介面還啟用了 Kubernetes 的功能，例如 kubectl exec 和 kubectl logs。每個 Amazon EKS 建立的網路介面都在其說明中包含 Amazon EKS *cluster-name* 字樣。

Amazon EKS 可以在您建立叢集時指定的任何子網中建立其網路介面。建立叢集後，您可以變更 Amazon EKS 要在其中建立網路介面的子網路。當您更新叢集的 Kubernetes 版本時，Amazon EKS 會刪除其建立的原始網路介面，並建立新的網路介面。這些網路介面可能在與原始網路介面相同的子網中建立，或在與原始網路介面不同的子網中建立。若要控制在其中建立的子網路網路介面，可以將建立叢集時指定的子網路數量限制為僅有兩個子網路，或在建立叢集後更新子網路。

叢集的子網路要求

您建立或更新叢集時指定的子網路必須符合下列要求：

- 每個子網必須至少有六個 IP 地址，以供 Amazon EKS 使用。但是，我們建議至少使用 16 個 IP 地址。
- 子網路不能位於 AWS Outposts AWS Wavelength、或 AWS 本機區域中。但是，如果您的 VPC 中有它們，則可以部署自我管理的節點和 Kubernetes 資源到這些類型的子網路。
- 子網可以是公有子網，也可以是私有子網。不過，如果可能的話，我們建議您指定私有子網。公有子網路是一種子網路，其路由表包含到網際網路閘道的路由，而私有子網路是一種子網路，其具有不包括到網際網路閘道路由的路由表。
- 子網路不能位於下列可用區域：

AWS 區域	區域名稱	不允許的可用區域 ID
us-east-1	美國東部 (維吉尼亞北部)	use1-az3
us-west-1	美國西部 (加利佛尼亞北部)	usw1-az2
ca-central-1	加拿大 (中部)	cac1-az3

依元件分類的 IP 位址系列用途

下表包含 Amazon EKS 每個元件所使用的 IP 位址系列。您可以使用網路位址轉譯 (NAT) 或其他相容性系統，從具有表格項目"No"值的系列中的來源 IP 位址連線到這些元件。

功能可能會因叢集的 IP family (`ipFamily`) 設定而有所不同。此設定會變更用於Kubernetes指派給CIDR區塊的 IP 位址類型Services。設定值為的叢集稱IPv4為IPv4 cluster，而設定值為的叢集IPv6則稱為IPv6 cluster。

元件	IPv4僅限位址	IPv6僅限位址	雙堆疊位址
EKS API 公共端點	是	否	否
EKS API VPC 端點	是	否	否
EKS 身份驗證 API 公共端點	是 [1]	是 [1]	是 [1]
EKS 身份驗證 API VPC 端點	是 [1]	是 [1]	是 [1]
EKS 集群公共端點	是	否	否
EKS 群集私有端點	是 [2]	是 [2]	否
EKS 叢集子網路	是 [2]	否	是 [2]
節點主要 IP 位址	是 [2]	否	是 [2]
ServiceIP 位址的叢集CIDR範圍	是 [2]	是 [2]	否
Pod來自 VPC CNI 的 IP 位址	是 [2]	是 [2]	否

Note

[1] 端點是具有IPv4和IPv6位址的雙堆疊。您在外部的應用程式 AWS、叢集節點以及叢集內的網繭都可以透過IPv4或連線到達此端點IPv6。

[2] 當您建立IPv4叢集時，您可以在IPv6叢集的 IP family (ipFamily) 設定中選擇叢集和叢集，且無法變更。相反地，當您建立另一個叢集並移轉工作負載時，您必須選擇不同的設定。

節點的子網路要求

您可以將節點和 Kubernetes 資源部署到您建立叢集時指定的相同子網路。不過，這不是必要的。這是因為您還可以將節點和 Kubernetes 資源部署至建立叢集時未指定的子網路。如果將節點部署到不同的

子網，Amazon EKS 不會在這些子網中建立叢集網路介面。您部署節點和 Kubernetes 資源的任何子網路必須符合以下要求：

- 子網路必須具有足夠的可用 IP 地址，以便將所有節點和 Kubernetes 資源部署至其中。
- 如果您希望 Kubernetes 指派 IPv6 地址至 Pods 和服務，則必須有一個 IPv6 CIDR 區塊和一個與子網路相關聯的 IPv4 CIDR 區塊。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的建立 IPv6 CIDR 區塊與子網的關聯。與子網關聯的路由表必須包含傳送到 IPv4 和 IPv6 地址的路由。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 Routes (路由)。Pod 僅指派一個 IPv6 地址。但是，Amazon EKS 為您的叢集和節點建立的網路介面受指派一個 IPv4 和一個 IPv6 地址。
- 如果您需要從網際網路對您的 Pods 進行傳入存取，請確保至少有一個具有足夠可用 IP 地址的公有子網路來為其部署負載平衡器和傳入。您可以將負載平衡器部署至公有子網。負載均衡器可以對私有子網路或公有子網路中的 Pods 進行負載平衡。如果可能的話，我們建議您將節點部署至私有子網。
- 如果您計畫將節點部署到公有子網路，則該子網路必須自動指派 IPv4 公有地址或 IPv6 地址。如果您將節點部署到具有相關聯 IPv6 CIDR 區塊的子網路，則私有子網路也必須自動指派 IPv6 地址。如果您在 2020 年 3 月 26 日之後使用 Amazon EKS AWS CloudFormation 範本部署 VPC，則會啟用此設定。如果在此日期之前使用範本部署 VPC，或者使用自己的 VPC，則必須手動啟用此設定。如需詳細資訊，請參閱《Amazon VPC 使用者指南》 中的修改公有 IPv4 子網的定址屬性和修改子網路的 IPv6 定址屬性。
- 如果您將節點部署到的子網是私有子網，且其路由表不包含通往網路位址轉譯 (NAT) 設備的路由 (IPv4) 或是僅輸出閘道 (IPv6)，使用 AWS PrivateLink 新增 VPC 端點到您的 VPC。您的所有節點都需要 VPC 端點 AWS 服務 ，並且Pods需要與之通訊。範例包括 Amazon ECR、Elastic Load Balancing CloudWatch AWS Security Token Service、Amazon 和 Amazon Simple Storage Service (Amazon S3)。端點必須包含節點所在的子網。並非所有人都 AWS 服務 支援 VPC 端點。如需詳細資訊，請參閱什麼是 AWS PrivateLink？ 以AWS 及與 AWS PrivateLink. 有關更多 Amazon EKS 要求的列表，請參閱 私有叢集要求。
- 如果要將負載平衡器部署到子網，則子網必須具有以下標籤：
  - 私有子網路

金鑰	值
kubernetes.io/role/internal-elb	1

  - 公有子網路

金鑰	值
kubernetes.io/role/elb	1

如果建立的是版本 Kubernetes 和更早版本的 1.18 叢集，Amazon EKS 會將下列標籤新增至指定的所有子網路。

金鑰	值
kubernetes.io/cluster/ *my-cluster*	shared

現在，當您建立新 Kubernetes 叢集時，Amazon EKS 不會將標籤新增至您的子網路。如果叢集所使用的子網路具有標籤，而且該叢集此前的版本早於 1.19，則當叢集更新為較新版本時，系統不會自動從子網路移除該標籤。2.1.1 版或更早版本的 AWS Load Balancer Controller 需要此標籤。如果您使用較新版本的 Load Balancer Controller，則您無須中斷服務便可移除標籤。

如果您使用eksctl或任何一種 Amazon EKS VPC 範本部署 AWS CloudFormation VPC，則適用以下條件：

- 在 2020 年 3 月 26 日或之後：公有 IPv4 地址會由公有子網自動指派給此公有子網中已部署的新節點。
- 在 2020 年 3 月 26 日之前：公有 IPv4 地址不會由公有子網自動指派給此公有子網路已部署的新節點。

這項變更會影響部署於公有子網中的新節點群組的下列行為：

- 受管節點群組：如果節點群組在 2020 年 4 月 22 日或之後部署到公有子網，此公有子網必須啟用公有 IP 地址的自動指派。如需詳細資訊，請參閱修改子網路的公有 IPv4 定址屬性。
- Linux、Windows 或 Arm 自我管理的節點群組 – 如果節點群組在 2020 年 3 月 26 日或之後部署到公有子網路，此公有子網路必須啟用公有 IP 地址的自動指派。否則，必須改用公有 IP 地址啟動節點。如需詳細資訊，請參閱修改子網路的公有 IPv4 定址屬性或在執行個體啟動期間指派公有 IPv4 地址。

共用子網路需求和注意事項

您可以使用 VPC 共用來與相同 AWS Organizations內的其他 AWS 帳戶共用子網路。您可以在共用子網路中建立 Amazon EKS 叢集，但會有下列考量：

- VPC 子網路的擁有者必須與參與者帳戶共用子網路，該帳戶才能在其中建立 Amazon EKS 叢集。
- 您無法啟動使用 VPC 預設安全群組的資源，因為該資源屬於擁有者。此外，參與者無法使用其他參與者或擁有者所擁有的安全性群組來啟動資源。
- 在共用子網路中，參與者和擁有者會分別控制每個各別帳戶內的安全群組。子網路擁有者可以查看由參與者建立的安全群組，但無法執行任何動作。如果子網路擁有者想要移除或修改這些安全群組，建立安全群組的參與者必須採取動作。
- 如果叢集是由參與者建立，則需要考量下列事項：
  - 必須在該帳戶中建立叢集 IAM 角色和節點 IAM 角色。如需詳細資訊，請參閱 Amazon EKS 叢集 IAM 角色 及 Amazon EKS 節點 IAM 角色。
  - 所有節點都必須由相同的參與者建立，包括受管節點群組。
- 共用 VPC 擁有者無法檢視、更新或刪除參與者在共用子網路中建立的叢集。此為 VPC 資源 (每個帳戶具有不同存取權) 以外。如需更多資訊，請參閱《Amazon VPC 使用者指南》中的擁有者和參與者的責任與許可。
- 如果您使用 Amazon VPC CNI plugin for Kubernetes 的自訂網路功能，您必須使用擁有者帳戶中列出的可用區域 ID 映射來建立每個 `ENIConfig`。如需詳細資訊，請參閱 Pod 的自訂聯網。

如需有關 VPC 子網路共用的詳細資訊，請參閱《Amazon VPC 使用者指南》中的與其他帳戶共用 VPC。

為 Amazon EKS 叢集建立 VPC

您可以使用 Amazon Virtual Private Cloud (Amazon VPC) 將 AWS 資源啟動到您已定義的虛擬網路中。此虛擬網路非常近似於您在自有資料中心內運作的傳統網路。但是，它帶來了使用 Amazon Web 服務的可擴展基礎架構的好處。部署生產 Amazon EKS 叢集前，建議您完整了解 Amazon VPC 服務。如需詳細資訊，請參閱 Amazon VPC 使用者指南。

Amazon EKS 叢集、節點和 Kubernetes 資源已部署至 VPC。如果您要將現有的 VPC 與 Amazon EKS 結合使用，則該 VPC 必須符合 Amazon EKS VPC 與子網要求和注意事項 中描述的要求。本主題說明如何使用 Amazon EKS 提供的範本建立符合 Amazon EKS 需求的 VPC。 AWS CloudFormation 部署範本後，您可以檢視範本所建立的資源，以確切了解其建立的資源以及這些資源的組態。

先決條件

若要為 Amazon EKS 建立 VPC，您必須擁有建立 Amazon VPC 資源所需的 IAM 許可。這些資源包括 VPC、子網、安全群組、路由表和路由，以及網際網路和 NAT 閘道。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用公有子網範例政策建立 VPC，以及服務授權參考中的 Amazon EC2 的動作、資源與條件金鑰的完整清單。

您可以建立含公有和私有子網路、僅公有子網路或僅私有子網路的 VPC。

Public and private subnets

此 VPC 具有兩個公用和兩個私有子網路。公有子網路與具有網際網路閘道路由的路由表相關聯。不過，私有子網的關聯路由表沒有連至網際網路閘道的路由。一個公有子網路和一個私有子網路會部署到相同的可用區域。其他公有和私有子網路會部署到相同 AWS 區域中的第二個可用區域。我們建議大多數部署使用此選項。

使用此選項，您可以將節點部署到私有子網。此選項可讓 Kubernetes 將負載平衡器部署至公有子網路，以便將流量負載平衡至私有子網路中節點上執行的 Pods。公有 IPv4 地址會自動指派給部署到公有子網路的節點，但公有 IPv4 地址不會指派給部署到私有子網路的節點。

您可以將 IPv6 地址指派給公有和私有子網路中的節點。私有子網路中的節點可以與叢集和其他 AWS 服務通訊。Pods 可以透過部署在每個可用區域中的 NAT 閘道 (使用 IPv4 地址) 或僅限傳出之網際網路閘道 (使用 IPv6 地址) 與網際網路通訊。部署的安全群組包含拒絕來自叢集或節點之外的來源的所有傳入流量，但卻允許所有傳出流量的規則。子網路會加上標籤，以讓 Kubernetes 能夠將負載平衡器部署至其中。

建立您的 VPC

1. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation
2. 從導覽列中，選取 AWS 區域 支援 Amazon EKS 的一個。
3. 選擇 Create stack (建立堆疊)、With new resources (standard) (使用新資源 (標準))。
4. 在 Prerequisite - Prepare template (必要條件 - 準備範本) 下，請確定選擇 Template is ready (範本已就緒)，然後在 Specify template (指定範本) 下選取 Amazon S3 URL。
5. 您可以建立僅支援 IPv4 的 VPC，或支援 IPv4 和 IPv6 的 VPC。將下列 URL 之一貼入 Amazon S3 URL 下的文字區域並選擇 Next (下一步)：

   - IPv4

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
amazon-eks-vpc-private-subnets.yaml
```

- IPv4 和 IPv6

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
amazon-eks-ipv6-vpc-public-private-subnets.yaml
```

6. 在 Specify stack details (識別堆疊詳細資訊) 頁面上，輸入參數，然後選擇 Next (下一步)。

   - Stack name：為您的 AWS CloudFormation 堆疊選擇堆疊名稱。例如，您可以使用在先前步驟中使用的範本名稱。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶且您要在中建立叢集。
   - VpcBlock：為您的 VPC 選擇一個 IPv4 CIDR 範圍。您部署的每個節點、Pod 和負載平衡器都會從此區塊指派一個 IPv4 地址。預設 IPv4 值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 和子網路規模調整。您也可以在 VPC 建立後將其他 CIDR 區塊新增至 VPC。如果您正在建立 IPv6 VPC，則會從 Amazon 的全域單播傳送地址空間自動為您指派 IPv6 CIDR 範圍。
   - PublicSubnet01 區塊：為公用子網路 1 指定 IPv4 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。如果建立 IPv6 VPC，則系統會在範本中為您指定此區塊。
   - PublicSubnet02 區塊：為公用子網路 2 指定 IPv4 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。如果建立 IPv6 VPC，則系統會在範本中為您指定此區塊。
   - PrivateSubnet01 區塊：為私有子網路 1 指定 IPv4 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。如果建立 IPv6 VPC，則系統會在範本中為您指定此區塊。
   - PrivateSubnet02 區塊：為私有子網路 2 指定 IPv4 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。如果建立 IPv6 VPC，則系統會在範本中為您指定此區塊。

7. (選用) 在 Configure stack options (設定堆疊選項) 頁面上，為堆疊資源加上標籤，然後選擇 Next (下一步)。
8. 在 Review (檢閱) 頁面上，選擇 Create stack (建立堆疊)。
9. 堆疊建立後，從主控台將其選取，然後選擇 Outputs (輸出)。

10. 記錄VpcId已建立的 VPC。建立叢集和節點時，您需要此值。
11. 記錄已建立SubnetIds的子網路，以及您是否將它們建立為公用子網路或私用子網路。建立叢集和節點時，您至少需要其中兩個。
12. 如果您建立了 IPv4 VPC，請跳過此步驟。如果建立 IPv6 VPC，則必須為範本建立的公有子網路啟用自動指派 IPv6 地址選項。已為私有子網路啟用該設定。若要啟用此設定，請完成下列步驟：

    a. 前往 https://console.aws.amazon.com/vpc/ 開啟 Amazon VPC 主控台。

    b. 在左側導覽窗格中，選擇 Subnets (子網路)。

    c. 選取其中一個公用子網路 (***####/SubnetPublic01 #####***/SubnetPublic02 包含公用字)，然後選擇 [動作]、[編輯子網路設定]。

    d. 勾選啟用自動指派 **IPv6** 地址核取方塊，然後選擇儲存。

    e. 為您的其他公有子網路再次完成上述步驟。

Only public subnets

此 VPC 有三個公有子網路，這些子網路已部署到 AWS 區域中的不同可用區域。所有節點都會自動指派公有 IPv4 地址，並且可以透過網際網路閘道傳送和接收網際網路流量。部署的安全群組會拒絕所有輸入流量，並允許所有輸出流量。子網路會加上標籤，以讓 Kubernetes 能夠將負載平衡器部署至其中。

建立您的 VPC

1. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation
2. 從導覽列中，選取 AWS 區域 支援 Amazon EKS 的一個。
3. 選擇 Create stack (建立堆疊)、With new resources (standard) (使用新資源 (標準))。
4. 在 Prepare template (準備範本) 下，請確定 Template is ready (範本已就緒)，然後在 Template source (範本來源) 下選取 Amazon S3 URL。
5. 將下列 URL 貼入 Amazon S3 URL 下的文字區域並選擇 Next (下一步)：

   ```
   https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-sample.yaml
   ```

6. 在 Specify Details (指定詳細資訊) 頁面上，輸入參數，然後選擇 Next (下一步)。

- Stack name：為您的 AWS CloudFormation 堆疊選擇堆疊名稱。例如，您可以稱它為 ***amazon-eks-vpc-sample***。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
- VpcBlock：為您的虛擬私人雲端選擇一個 CIDR 區塊。您部署的每個節點、Pod 和負載平衡器都會從此區塊指派一個 IPv4 地址。預設 IPv4 值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 和子網路規模調整。您也可以在 VPC 建立後將其他 CIDR 區塊新增至 VPC。
- Subnet01Block：指定適用於子網路 1 的 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。
- Subnet02Block：指定適用於子網路 2 的 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。
- Subnet03Block：指定適用於子網路 3 的 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。

7. (選用) 在 Options (選項) 頁面上，為堆疊資源加上標籤。選擇下一步。
8. 在 Review (檢閱) 頁面上，選擇 Create (建立)。
9. 堆疊建立後，從主控台將其選取，然後選擇 Outputs (輸出)。
10. 記錄VpcId已建立的 VPC。建立叢集和節點時，您需要此值。
11. 記錄已SubnetIds建立之子網路的。建立叢集和節點時，您至少需要其中兩個。
12. (選用) 叢集部署到此 VPC 的任何叢集都能夠為您的 Pods 和 services 指派私有 IPv4 地址。如果要將叢集部署到此 VPC，以指派私有 IPv6 地址給您的 Pods 和 services，則必須對 VPC、子網路、路由表和安全群組進行更新。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的將 IPv4 中現有的 VPC 遷移至 IPv6。Amazon EKS 要求您的子網路啟用 Auto-assign IPv6 地址選項。預設為停用狀態。

Only private subnets

此 VPC 有三個私有子網路，這些子網路部署到 AWS 區域中不同的可用區域。部署到子網的資源無法存取網際網路，網際網路也無法存取子網中的資源。範本會使用 AWS PrivateLink 多個節點通常需要存取 AWS 服務 的節點來建立 VPC 端點。如果您的節點需要傳出網際網路存取權，您可以在建立 VPC 後的每個子網的可用區域中新增公有 NAT 閘道。建立的安全群組會拒絕所有傳入流量，部署到子網中的資源除外。安全群組亦允許所有傳出流量。子網路會加上標籤，以讓 Kubernetes 能夠將內部負載平衡器部署至其中。如果您使用此組態來建立 VPC，請參閱 私有叢集要求 了解其他要求和注意事項。

建立您的 VPC

1. 請在以下位置開啟 AWS CloudFormation 主控台。 https://console.aws.amazon.com/cloudformation
2. 從導覽列中，選取 AWS 區域 支援 Amazon EKS 的一個。
3. 選擇 Create stack (建立堆疊)、With new resources (standard) (使用新資源 (標準))。
4. 在 Prepare template (準備範本) 下，請確定 Template is ready (範本已就緒)，然後在 Template source (範本來源) 下選取 Amazon S3 URL。
5. 將下列 URL 貼入 Amazon S3 URL 下的文字區域並選擇 Next (下一步)：

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-fully-private-vpc.yaml
```

6. 在 Specify Details (指定詳細資訊) 頁面上，輸入參數，然後選擇 Next (下一步)。
   - Stack name：為您的 AWS CloudFormation 堆疊選擇堆疊名稱。例如，您可以稱它為 ***amazon-eks-fully-private-vpc***。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - VpcBlock：為您的虛擬私人雲端選擇一個 CIDR 區塊。您部署的每個節點、Pod 和負載平衡器都會從此區塊指派一個 IPv4 地址。預設 IPv4 值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 和子網路規模調整。您也可以在 VPC 建立後將其他 CIDR 區塊新增至 VPC。
   - PrivateSubnet01 區塊：指定子網路 1 的 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。
   - PrivateSubnet02 區塊：指定子網路 2 的 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。
   - PrivateSubnet03 區塊：指定子網路 3 的 CIDR 區塊。預設值為大多數實作提供足夠的 IP 地址，但如果沒有，則可以進行變更。
7. (選用) 在 Options (選項) 頁面上，為堆疊資源加上標籤。選擇下一步。
8. 在 Review (檢閱) 頁面上，選擇 Create (建立)。
9. 堆疊建立後，從主控台將其選取，然後選擇 Outputs (輸出)。
10. 記錄VpcId已建立的 VPC。建立叢集和節點時，您需要此值。
11. 記錄已SubnetIds建立之子網路的。建立叢集和節點時，您至少需要其中兩個。

12. (選用) 叢集部署到此 VPC 的任何叢集都能夠為您的 Pods 和 services 指派私有 IPv4 地址。如果要將叢集部署到此 VPC，以指派私有 IPv6 地址給您的 Pods 和 services，則必須對 VPC、子網路、路由表和安全群組進行更新。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的將 IPv4 中現有的 VPC 遷移至 IPv6。Amazon EKS 要求您的子網路啟用 Auto-assign IPv6 地址選項 (預設情況下處於停用狀態)。

Amazon EKS 安全群組與考量

本主題說明 Amazon EKS 叢集的安全群組要求。

當您建立叢集時，Amazon EKS 會建立名稱為 eks-cluster-sg-*my-cluster*-*uniqueID* 的安全群組。安全性群組具有以下預設規則：

規則類型	通訊協定	連接埠	來源	目的地
傳入	全部	全部	Self	
傳出	全部	全部		0.0.0.0/0(IPv4) 或 ::/0 (IPv6)

> **Important**
>
> 如果您的叢集不需要傳出規則，則可以將其移除。如果將其移除，您仍必須具有限制叢集流量中列出的最低規則。如果您移除傳入規則，則 Amazon EKS 會在叢集更新時重新建立該規則。

Amazon EKS 會將下列標籤新增至安全群組。如果您移除標籤，則 Amazon EKS 會在叢集更新時將其新增回安全群組。

金鑰	值
kubernetes.io/cluster/ *my-cluster*	owned
aws:eks:cluster-name	*my-cluster*

金鑰	值
Name	eks-cluster-sg- *my-cluster -uniqueid*

Amazon EKS 會自動將此安全群組與以下資源關聯，其資源也會建立：

- 2—4 個彈性網路界面（本文件的其餘部分稱為網路界面），網路界面於叢集建立時建立。
- 您建立的任何受管節點群組中節點的網路界面。

此安全群組的設計目的是允許來自控制平面和 受管節點群組的所有流量彼此之間可以自由流動，以及允許所有的輸出流量流往任何目的地。當您建立叢集時，您可以（選用）指定自己的安全群組。如果您指定自己的安全群組，則 Amazon EKS 還會將您指定的安全群組與為叢集建立的網路界面建立關聯。但是，其不會與您建立的任何節點群組相關聯。

您可以在叢集的 Networking (聯網) 區段下的 AWS Management Console 中判定叢集安全群組的 ID。或者，您可以執行下列 AWS CLI 命令，即可執行這項操作：

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

限制叢集流量

如果您需要限制叢集和節點之間的開放連接埠，則可以移除預設傳出規則，並且新增叢集所需的下列最低規則：如果您移除預設傳入規則，Amazon EKS 會在叢集更新時重新建立該規則。

規則類型	通訊協定	連線埠	目的地
傳出	TCP	443	叢集安全群組
傳出	TCP	10250	叢集安全群組
傳出 (DNS)	TCP 和 UDP	53	叢集安全群組

您還必須為以下流量新增規則：

- 您預期節點用於節點間通訊的任何通訊協定和連接埠。

- 需要對外網際網路存取，以便叢集在啟動時存取 Amazon EKS API，進行叢集自我檢查和節點註冊。如果您的節點沒有網際網路存取，請檢閱 私有叢集要求 以了解其他考量事項。
- 需要虛擬節點存取以從 Amazon ECR 或是從提取映像所需的其他容器登錄檔 API 提取映像，例如 DockerHub。如需詳細資訊，請參閱《AWS 一般參考》中的 AWS IP 地址範圍。
- 節點對 Amazon S3 進行存取。
- 需要 `IPv4` 和 `IPv6` 地址的個別規則。

如果您正在考慮限制規則，建議您先徹底測試 Pods，之後再將變更的規則套用到生產叢集。

如果您最初使用 Kubernetes `1.14` 和 `eks.3` 平台版本或更舊版本部署叢集，則請考慮下列事項：

- 您可能還擁有控制平面和節點安全群組。建立這些群組時，群組包括之前的表格中列出的受限制的規則。不再需要這些安全群組，可以進行移除。但是，您需要確定您的叢集安全群組包含那些群組包含的規則。
- 如果您直接使用 API 部署叢集，或者您使用 AWS CLI 或 AWS CloudFormation 之類的工具建立叢集，且在叢集建立時未指定安全群組，則 VPC 的預設安全群組會套用到 Amazon EKS 建立的叢集網路界面。

Amazon EKS 聯網附加元件

數個聯網附加元件可用於 Amazon EKS 叢集。

內建附加元件

> **Note**
>
> 如果您使用主控台以外的任何方式建立叢集，每個叢集皆會隨附內建附加元件的自我管理版本。自我管理的版本無法從 AWS Management Console AWS Command Line Interface、或 SDK 管理。您可以管理自我管理附加元件的組態和升級。
> 建議將附加元件的 Amazon EKS 類型新增到叢集，而不是使用附加元件的自我管理類型。如果您在主控台中建立叢集，則系統會安裝這些附加元件的 Amazon EKS 類型。

Amazon VPC CNI plugin for Kubernetes

此 CNI 附加元件會建立彈性網路介面並將其連接至 Amazon EC2 節點。附加元件還會將 VPC 中的私有 IPv4 或 IPv6 地址指派給每個 Pod 和服務。您的叢集預設會安裝此附加元件。如需詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。

CoreDNS

CoreDNS 是一個靈活、可擴展的 DNS 伺服器，可作為 Kubernetes 叢集 DNS。CoreDNS 可為叢集中的所有 Pods 提供名稱解析。您的叢集預設會安裝此附加元件。如需詳細資訊，請參閱 使用 CoreDNS Amazon EKS 附加元件。

kube-proxy

此附加元件會維護 Amazon EC2 節點上的網路規則，並啟用與 Pods 的網路通訊。您的叢集預設會安裝此附加元件。如需詳細資訊，請參閱 使用庫伯尼特附kube-proxy加元件。

可選的 AWS 網絡附加

AWS Load Balancer Controller

當您部署類型的Kubernetes服務物件時loadbalancer，控制器會建立 AWS 網路負載平衡器。當您建立Kubernetes輸入物件時，控制器會建立 AWS 應用程式負載平衡器。我們建議您使用此控制器來佈建 Network Load Balancer，而不是使用內建至 Kubernetes 的舊式雲端供應商控制器。如需詳細資訊，請參閱AWS Load Balancer Controller文件。

AWS 閘道器 API 控制器

此控制器可讓您使用 Kubernetes 閘道 API 在多重 Kubernetes 叢集之間連接服務。控制器使用 Amazon VPC Lattice 服務，連接在 Amazon EC2 執行個體、容器和無伺服器功能上執行的 Kubernetes 服務。如需詳細資訊，請參閱 AWS 閘道 API 控制器文件。

如需附加元件的詳細資訊，請參閱 Amazon EKS 附加元件。

使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件

Amazon VPC CNI plugin for Kubernetes 附加元件會部署在 Amazon EKS 叢集中的每個 Amazon EC2 節點上。附加元件會建立彈性網路界面並將其連接到 Amazon EC2 節點。附加元件還會將 VPC 中的私有 IPv4 或 IPv6 地址指派給每個 Pod 和服務。

附加元件的版本會隨叢集中的每個 Fargate 節點一起部署，但您不會在 Fargate 節點上進行更新。其他相容 CNI 外掛程式可在 Amazon EKS 集群上使用，但這是 Amazon EKS 支援的唯一 CNI 外掛程式。

下表列出每個 Kubernetes 版本可用的 Amazon EKS 附加元件類型最新版本。

Kubernetes 版本	1.30	1.29	1.28	1.27	1.26	1.25	1.24	1.23
Amazon EKS 類型的 VPC CNI 版本	v1.18.2-eksbuilc	v1.18.2-eksbuilc	v1.18.2-eksbuilc	v1.18.2-eksbuilc	v1.18.2-eksbuilc	v1.18.2-eksbuilc	v1.18.2-eksbuilc	v1.18.2-eksbuild.1

Important

如果您要自行管理此附加元件，資料表中的版本可能與可用的自我管理版本不同。如需更新此附加元件之自我管理類型的詳細資訊，請參閱 更新自我管理的附加元件。

Important

若要升級至 VPC CNI 1.12.0 或更新版本，您必須先升級至 VPC CNI 1.7.0 版。我們建議您一次更新一個次要版本。

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。
- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。
- 連接有 AmazonEKS_CNI_Policy IAM 政策 (如果您的叢集使用 IPv4 系列) 或 IPv6 政策 (如果您的叢集使用 IPv6 系列) 的 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。
- 如果使用 Amazon VPC CNI plugin for Kubernetes 版本 1.7.0 或更新版本，並使用自訂 Pod 安全政策，請參閱 刪除預設 Amazon EKS Pod 安全政策Pod 安全政策。

- **Important**

  Amazon VPC CNI plugin for Kubernetes版本v1.16.1刪v1.16.0除與版Kubernetes本1.23和更早版本的兼容性。VPC CNI 版本v1.16.2恢復了與版本以1.23及較早Kubernetes版本和 CNI 規範的兼容性。v0.4.0

  Amazon VPC CNI plugin for Kubernetesv1.16.1實作 CNI 規範版本v1.16.0的版v1.0.0本。執行版本v1.24或更新Kubernetes版本的 EKS 叢集支援 CNI 規格v1.0.0。版本或v1.16.0更早版本不支援 VPC CNI Kubernetes 版本v1.16.1v1.23和 CNI 規格v1.0.0。 如需 CNI 規格v1.0.0的詳細資訊，請參閱上的容器網路介面 (CNI) 規格

考量事項

- 版本指定為 major-version.minor-version.patch-version-eksbuild.build-number。
- 檢查每個功能的版本相容性

  每個版本的某些功能Amazon VPC CNI plugin for Kubernetes需要 certian Kubernetes 版本。使用不同的 Amazon EKS 功能時，如果需要特定版本的附加元件，則會備註於功能文件中。除非您有執行舊版本的特定理由，否則建議您執行最新版本。

建立 Amazon EKS 附加元件

建立附加元件的 Amazon EKS 類型。

1. 查看叢集上目前安裝了哪些附加元件版本。

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
    | cut -d : -f 3
   ```

   範例輸出如下。

   ```
   v1.16.4-eksbuild.2
   ```

2. 查看叢集上安裝的附加元件類型。視您用來建立叢集的工具而定，您的叢集上目前可能沒有安裝 Amazon EKS 附加元件類型。使用您叢集的名稱取代 *my-cluster*。

```
$ aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

如果傳回版本編號，則表明已在叢集上安裝 Amazon EKS 類型的附加元件，並且無需完成此程序中的剩餘步驟。如果傳回錯誤，則表明沒有在叢集上安裝 Amazon EKS 類型的附加元件。完成此程序的剩餘步驟以安裝該類型。

3. 儲存您目前安裝的附加元件。

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. 使用 AWS CLI建立附加元件。如果您想要使用 AWS Management Console 或eksctl建立附加元件，請參閱建立附加元件並指定vpc-cni附加元件名稱。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令。

   - 使用您叢集的名稱取代 *my-cluster*。
   - 將 *v1.18.2-eksbuild.1* 取代為叢集版本的最新版本資料表中列出的最新版本。
   - 使用帳戶 ID 取代 *111122223333*，並且使用所建立現有 IAM 角色的名稱取代 *AmazonEKSVPCCNIRole*。若要指定角色，您的叢集需要具有 IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.18.2-eksbuild.1 \
    --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

如果您已將自訂設定套用至與 Amazon EKS 附加元件的預設設定衝突的目前附加元件，建立動作可能會失敗。若建立失敗，您會收到錯誤，其中的訊息有助於您解決問題。或者，您可以將 **--resolve-conflicts OVERWRITE** 新增至上一條命令。這可讓附加元件覆寫任何現有的自訂設定。建立附加元件後，可以使用自訂設定來更新該附加元件。

5. 確認叢集 Kubernetes 版本的附加元件的最新版本已新增至叢集。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

建立附加元件的動作可能需要幾秒鐘的時間才能完成。

範例輸出如下。

```
v1.18.2-eksbuild.1
```

6. 如果您對原始附加元件制定自訂設定，請在建立 Amazon EKS 附加元件之前，使用在上一步中儲存的組態，以您的自訂設定更新 Amazon EKS 附加元件。
7. (選用) 安裝 `cni-metrics-helper` 到您的叢集。它會擷取 elastic network interface 和 IP 地址資訊，在叢集層級彙總，然後將指標發佈到 Amazon。 CloudWatch如需詳細資訊，請參閱中的 CNI 量度協助程式。 GitHub

更新 Amazon EKS 附加元件

更新 Amazon EKS 類型的附加元件。如果您尚未將 Amazon EKS 類型的附加元件新增至叢集，請新增該類型或查看 更新自我管理的附加元件，而不是完成此程序。

1. 查看叢集上目前安裝了哪些附加元件版本。使用您的叢集名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 "addon.addonVersion" --output text
```

範例輸出如下。

```
v1.16.4-eksbuild.2
```

如果傳回的版本與最新版本資料表中的叢集 Kubernetes 版本相同，則表明您已經在叢集上安裝最新版本，不需要完成此程序的剩餘步驟。若您收到錯誤而非版本編號，則表明叢集上沒有安裝 Amazon EKS 類型的附加元件。您必須先建立附加元件，才能使用此程序進行更新。

2. 儲存您目前安裝的附加元件。

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

3. 使用 AWS CLI更新您的附加元件。如果您想要使用 AWS Management Console 或`eksctl`更新附加元件，請參閱更新附加元件。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令。

   - 使用您叢集的名稱取代 *my-cluster*。

- 將 *v1.18.2-eksbuild.1* 取代為叢集版本的最新版本資料表中列出的最新版本。
- 使用帳戶 ID 取代 *111122223333*，並且使用所建立現有 IAM 角色的名稱取代 *AmazonEKSVPCCNIRole*。若要指定角色，您的叢集需要具有 IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。
- **--resolve-conflicts** *PRESERVE* 選項會保留附加元件的現有組態值。如果對於附加元件設定，您已設定自訂值，但未使用此選項，Amazon EKS 會以其預設值覆寫您的值。如果您使用此選項，建議您在更新生產叢集上的附加元件之前，測試非生產叢集上的任何欄位和值變更。如果您將此值變更為 OVERWRITE，則所有設定都會變更為 Amazon EKS 預設值。如果您已設定任何設定的自訂值，則可能會使用 Amazon EKS 預設值覆寫這些值。如果您將此值變更為 none，Amazon EKS 不會變更任何設定的值，但更新可能會失敗。若更新失敗，您會收到錯誤訊息，以協助您解決衝突。
- 如果您不更新組態設定，請從命令中移除 **--configuration-values '{*"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}*}'**。如果您要更新組態設定，請將 *"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}* 取代為您要設定的設定值。在此範例中，AWS_VPC_K8S_CNI_EXTERNALSNAT 環境變數設定為 true。您指定的值必須對組態結構描述有效。如果您不知道配置模式，請運行**aws eks describe-addon-configuration --addon-name vpc-cni --addon-version *v1.18.2-eksbuild.1***，將 *v1.18.2-eksbuild.1* 替換為您要查看其配置的附加元件的版本號碼。結構描述會在輸出中傳回。如果您有任何現有的自訂組態，並且想要全部移除，同時將所有設定的值設定回 Amazon EKS 預設值，請從命令中移除 *"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}*，這樣就會得到空白的 **{}**。如需每個設定的說明，請參閱中的 CNI 組態變數。 GitHub

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.18.2-eksbuild.1 \
    --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
 \
    --resolve-conflicts PRESERVE --configuration-values '{"env":
{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}}'
```

更新動作可能需要幾秒鐘的時間才能完成。

4. 確認附加元件版本已更新。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

更新動作可能需要幾秒鐘的時間才能完成。

範例輸出如下。

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.18.2-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/vpc-
cni/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "serviceAccountRoleArn":
 "arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole",
        "tags": {},
        "configurationValues": "{\"env\":{\"AWS_VPC_K8S_CNI_EXTERNALSNAT\":\"true
\"}}"
    }
}
```

更新自我管理的附加元件

> **Important**
>
> 建議將附加元件的 Amazon EKS 類型新增到叢集，而不是使用附加元件的自我管理類型。如果您不熟悉類型之間的差異，則請參閱 the section called “Amazon EKS 附加元件”。如需將 Amazon EKS 附加元件新增至叢集的詳細資訊，請參閱 the section called “建立附加元件”。如果您無法使用 Amazon EKS 附加元件，我們建議您提交有關為何無法存取容器藍圖GitHub 儲存庫的問題。

1. 確認您的叢集上已安裝 Amazon EKS 類型的附加元件。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

如果傳回錯誤訊息，則表明沒有在叢集上安裝 Amazon EKS 類型的附加元件。若要自行管理附加元件，請完成此程序中的剩餘步驟來更新附加元件。如果傳回版本編號，則表明已在叢集上安裝附加元件的 Amazon EKS 類型。若要將其更新，請使用 更新附加元件 中的程序，而不是使用此程序。如果您不熟悉附加元件類型之間的差異，請參閱 Amazon EKS 附加元件。

2. 查看叢集上目前安裝了哪些容器映像版本。

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

範例輸出如下。

```
v1.16.4-eksbuild.2
```

您的輸出可能不包含建置編號。

3. 備份您目前的設定，以便在更新版本後進行相同的設定。

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. 若要檢閱可用版本並熟悉更新版本中的變更，請參閱 GitHub 上的 releases。請注意，我們建議您更新為相同的major。 minor。 patch列在「最新可用版本」表格中的版本，即使在上提供更新版本 GitHub。表格中列出的組建版本未在上 GitHub列出的自我管理版本中指定。透過以下列其中一種選項完成任務來更新您的版本：

   - 如果您沒有附加元件的任何自訂設定，請針 GitHub對您要更新的發行版本執行To apply this release:標題下的命令。
   - 如果有自訂設定，則請使用以下命令下載清單檔案。將 *https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.18.2/config/master/aws-k8s-cni.yaml* 變更為您要更新的目標 GitHub 發行版本的網址。

     ```
     curl -O https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.18.2/config/
     master/aws-k8s-cni.yaml
     ```

如有必要，請使用您在前一步所建立備份中的自訂設定修改清單檔案，然後將修改後的清單檔案套用至叢集。如果您的節點無法存取從中提取映像的私有 Amazon EKS Amazon ECR 儲存庫 (請參閱清單檔案中開頭為 `image:` 的資料行)，則您必須下載映像，將其複製到您自己的儲存庫，然後修改清單檔案以便從儲存庫中提取映像。如需詳細資訊，請參閱 將容器映像從一個儲存庫複製到另一個儲存庫。

```
kubectl apply -f aws-k8s-cni.yaml
```

5. 確認您的叢集上現在已安裝新版本。

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

範例輸出如下。

```
v1.18.2
```

6. (選用) 安裝 `cni-metrics-helper` 到您的叢集。它會擷取 elastic network interface 和 IP 地址資訊，在叢集層級彙總，然後將指標發佈到 Amazon。 CloudWatch如需詳細資訊，請參閱中的 CNI 量度協助程式。 GitHub

設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)

Amazon VPC CNI plugin for Kubernetes 是 Amazon EKS 叢集中供 Pod 聯網的聯網外掛程式。外掛程式負責將 VPC IP 地址分派給 Kubernetes 節點，以及設定每個節點上 Pods 所需的聯網。外掛程式：

- 需要 AWS Identity and Access Management (IAM) 許可。如果您的叢集使用 IPv4 系列，則會在`AmazonEKS_CNI_Policy` AWS 受管理的策略中指定權限。 如果您的叢集使用 IPv6 系列，則必須將許可新增至您建立的 IAM 政策。您可附加政策至 Amazon EKS 節點 IAM 角色，或附加至單獨 IAM 角色。建議如本主題所述，將其指派給不同的角色。
- 部署時，建立並設定為使用名為 `aws-node` 的 Kubernetes 服務帳戶。服務帳戶綁定到一個名為 `aws-node` 的 Kubernetes `clusterrole`，它被指派了所需的 Kubernetes 許可。

> **Note**
>
> 除非您封鎖 IMDS 的存取，否則 Amazon VPC CNI plugin for Kubernetes 的 Pods 可存取指派給 Amazon EKS 節點 IAM 角色的權限。如需詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。
- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。

步驟 1：建立 Amazon VPC CNI plugin for Kubernetes IAM 角色

建立 IAM 角色

1. 確定您叢集的 IP 系列。

   ```
   aws eks describe-cluster --name my-cluster | grep ipFamily
   ```

   範例輸出如下。

   ```
   "ipFamily": "ipv4"
   ```

   輸出可能會傳回 `ipv6`。

2. 建立 IAM 角色。您可以使用`eksctl`或`kubectl`和 AWS CLI 來建立 IAM 角色。

   eksctl

   建立 IAM 角色，並使用與您叢集 IP 系列相符的命令將 IAM 政策連接至此角色。此命令會建立並部署可建立 IAM 角色的 AWS CloudFormation 堆疊、附加您指定的政策，並使用所建立的 IAM 角色的 ARN 註解現有`aws-node`Kubernetes服務帳戶。

   - `IPv4`

     使用您自己的值取代 *my-cluster*。

```
eksctl create iamserviceaccount \
    --name aws-node \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKSVPCCNIRole \
    --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
    --override-existing-serviceaccounts \
    --approve
```

- IPv6

  使用您自己的值取代 *my-cluster*。使用您的帳戶 ID 取代 *111122223333*，再以您的 IPv6 政策名稱取代 *AmazonEKS_CNI_IPv6_Policy*。如果沒有 IPv6 政策，請參閱 為使用 IPv6 系列的叢集建立 IAM 政策 來建立一個。若要將 IPv6 與您的叢集一起使用，其必須滿足幾個要求。如需詳細資訊，請參閱 IPv6叢集的位址Pods、和 services。

  ```
  eksctl create iamserviceaccount \
      --name aws-node \
      --namespace kube-system \
      --cluster my-cluster \
      --role-name AmazonEKSVPCCNIRole \
      --attach-policy-arn
   arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
      --override-existing-serviceaccounts \
      --approve
  ```

kubectl and the AWS CLI

1. 檢視叢集的 OIDC 提供商 URL。

   ```
   aws eks describe-cluster --name my-cluster --query
    "cluster.identity.oidc.issuer" --output text
   ```

   範例輸出如下。

   ```
   https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
   ```

   如果未傳回任何輸出，則您必須為叢集建立 IAM OIDC 提供商。

2. 將下列內容複製到名為 *vpc-cni-trust-policy.json* 的檔案。使用您的帳戶 ID 取代 *111122223333*，並使用上一個步驟傳回的值取代 *EXAMPLED539D4633E53DE1B71EXAMPLE*。*region-code*以叢集所 AWS 區域 在的位置取代。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-node"
                }
            }
        }
    ]
}
```

3. 建立角色。您可以將 *AmazonEKSVPCCNIRole* 取代為您選擇的任何名稱。

```
aws iam create-role \
  --role-name AmazonEKSVPCCNIRole \
  --assume-role-policy-document file://"vpc-cni-trust-policy.json"
```

4. 將所需的 IAM 政策連接至角色。執行與叢集的 IP 系列相符的命令。

   - IPv4

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
       --role-name AmazonEKSVPCCNIRole
     ```

   - IPv6

使用您的帳戶 ID 取代 *111122223333*，再以您的 IPv6 政策名稱取代 *AmazonEKS_CNI_IPv6_Policy*。如果沒有 IPv6 政策，請參閱 為使用 IPv6 系列的叢集建立 IAM 政策 來建立一個。若要將 IPv6 與您的叢集一起使用，其必須滿足幾個要求。如需詳細資訊，請參閱 IPv6叢集的位址Pods、和 services。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSVPCCNIRole
```

5. 執行下列命令以使用您之前建立之 IAM 角色的 ARN 來標註 aws-node 服務帳戶。以您自己的值取代 *example values*。

```
kubectl annotate serviceaccount \
    -n kube-system aws-node \
    eks.amazonaws.com/role-
arn=arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

3. （可選）設定Kubernetes服務帳戶使用的 AWS Security Token Service 端點類型。如需詳細資訊，請參閱 設定服務帳戶的 AWS Security Token Service 端點。

步驟 2：重新部署 Amazon VPC CNI plugin for KubernetesPods

1. 刪除並重新建立任何與服務帳戶相關聯的現有 Pods，以套用登入資料環境變數。註釋不適用於目前在沒有註釋的情況下執行的 Pods。下列命令會刪除現有的 aws-node DaemonSet Pods，並使用服務帳戶註釋來部署它們。

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

2. 確認 Pods 全部重新啟動。

```
kubectl get pods -n kube-system -l k8s-app=aws-node
```

3. 描述其中一個 Pods 並確認 AWS_WEB_IDENTITY_TOKEN_FILE 和 AWS_ROLE_ARN 環境變數存在。使用前一個步驟輸出中傳回的其中一個 Pods 名稱取代 *cpjw7*。

```
kubectl describe pod -n kube-system aws-node-cpjw7 | grep 'AWS_ROLE_ARN:\|
AWS_WEB_IDENTITY_TOKEN_FILE:'
```

範例輸出如下。

```
AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
      AWS_ROLE_ARN:
 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:           /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

Pod 包含兩個容器，因此會傳回兩組重複結果。兩個容器具有相同的值。

如果您使用的Pod是 AWS 區域 al 端點，則以下行也會在前一個輸出中返回。

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

步驟 3：從節點 IAM 角色移除 CNI 政策

如果您的 Amazon EKS 節點 IAM 角色目前已附加 AmazonEKS_CNI_Policy IAM (IPv4) IPv6 政策或政策，而且您已建立單獨的 IAM 角色，改為將政策附加到該角色，並將其指派給aws-nodeKubernetes服務帳戶，建議您使用符合叢集 IP 系列的 AWS CLI 命令從節點角色中移除該政策。以您的節點角色名稱取代 *AmazonEKSNodeRole*。

- IPv4

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
```

- IPv6

  使用您的帳戶 ID 取代 *111122223333*，再以您的 IPv6 政策名稱取代 *AmazonEKS_CNI_IPv6_Policy*。

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy
```

為使用 **IPv6** 系列的叢集建立 IAM 政策

如果您建立了使用 IPv6 系列的叢集，並且該叢集設定了版本 1.10.1 或更高版本的 Amazon VPC CNI plugin for Kubernetes 附加元件，則需要建立可以指派給 IAM 角色的 IAM 政策。如果您現有的叢

集在建立時未使用 IPv6 系列進行設定，則為了使用 IPv6，必須建立新的叢集。如需搭配使用 IPv6 與叢集的詳細資訊，請參閱 IPv6叢集的位址Pods、和 services。

1. 複製下列文字並將它儲存至名為 *vpc-cni-ipv6-policy*.json 的檔案。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. 建立 IAM 政策。

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

選擇 Pod 聯網使用案例

Amazon VPC CNI plugin for Kubernetes 為 Pods 提供聯網功能。下表協助您了解可以搭配使用的聯網使用案例，以及可與不同 Amazon EKS 節點類型搭配使用的功能和 Amazon VPC CNI plugin for Kubernetes 設定。資料表中的所有資訊僅適用於 Linux IPv4 節點。

Amazon EKS 節點類型	Amazon EC2			Fargate
使用案例	指派給網路介面的個別 IP 地址	指派給網路介面的 IP 字首	Pods 的安全群組	
Pod 的自訂聯網：從節點子網路以外的其他子網路指派 IP 地址	是	是	是	是 (透過 Fargate 設定檔控制的子網路)
適用於 Pods 的 SNAT	是 (預設為 `false`)	是 (預設為 `false`)	是 (僅限 `true`)	是 (僅限 `true`)
功能				
安全群組範圍	節點	節點	Pod (如果已設定 `POD_SECURITY_GROUP_ENFORCING_MODE = standard` 且 `AWS_VPC_K8S_CNI_EXTERNALSNAT = false`，則目的地為 VPC 外部端點的流量會使用節點的安全群組，而不是 Pod's 安全群組)	Pod
Amazon VPC 子網路類型	公有及私有	公有及私有	僅限私有	僅限私有

Amazon EKS 節點類型	Amazon EC2			Fargate
使用案例	指派給網路介面的個別 IP 地址	指派給網路介面的 IP 字首	Pods 的安全群組	
網路政策 (VPC CNI)	相容	相容	相容 僅適用於版本 1.14.0 或更高版本的 Amazon VPC CNI 外掛程式	不支援
每個節點的 Pod 密度	中	高	低	—
Pod 啟動時間	更佳	最佳	好	適中
Amazon VPC CNI 外掛程式設定 (有關每個設定的詳細資訊，請參閱上的 amazon-vpc-cni-k8 秒) GitHub				
WARM_ENI_TARGET	是	不適用	不適用	不適用
WARM_IP_TARGET	是	是	不適用	不適用
MINIMUM_IP_TARGET	是	是	不適用	不適用
WARM_PREFIX_TARGET	不適用	是	不適用	不適用

Note

- 您無法使用 IPv6 搭配自訂聯網。
- 未轉換 IPv6 地址，因此 SNAT 不適用。

- 具有關聯安全群組的 Pods 的流量不受 Calico 網路政策強制執行的約束，且僅限於 Amazon VPC 安全群組強制執行。
- 如果您使用Calico網路原則強制執行，建議您將環境變數設定ANNOTATE_POD_IPtrue為避免與的已知問題Kubernetes。若要使用此功能，您必須將網繭的patch權限新增至 aws-nodeClusterRole. 請注意，新增修補程式權限會aws-nodeDaemonSet增加外掛程式的安全性範圍。如需詳細資訊，請參閱上的 VPC CN I 存放庫中的。 GitHub
- IP 字首和 IP 地址與標準 Amazon EC2 彈性網路介面相關聯。需要特定安全群組的 Pod 會被指派分支網路介面的主要 IP 地址。您可以把取得 IP 地址的 Pods，或從 IP 字首的 IP 地址與在相同節點上取得分支網路介面的 Pods 混合。

Windows 節點

每個節點僅支援一個網路介面。您可以使用次要 IPv4 地址和 IPv4 字首。依預設，節點上的可用 IPv4 地址數量等於您可以指派給每個彈性網路介面的次要 IPv4 地址數量減去一。不過，您可以啟用 IP 字首來增加節點上的可用 IPv4 地址和 Pod 數量。如需詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。

Windows 支援 Calico 網路政策。您無法在 Windows 上使用 Pods 的安全群組或自訂聯網。

IPv6叢集的位址Pods、和 services

預設情況下，Kubernetes 會為您的 Pods 和 services 指派 IPv4 地址。您可以將叢集設定為對 Pods 和 services 指派 IPv6 地址，而不是指派 IPv4 地址。Amazon EKS 不支援雙堆疊 Pods 或 services，即使 Kubernetes 在版本 1.23 和更新版本提供支援。因此，您無法同時將 IPv4 和 IPv6 地址指派給 Pods 和 services。

您可以在建立叢集時選擇要用於此叢集的 IP 系列。建立叢集後無法變更系列。

將IPv6系列用於叢集的考量

- 您必須建立新叢集，並指定要為該叢集使用 IPv6 系列。無法為從早期版本更新的叢集啟用 IPv6 系列。如需如何建立新叢集的指示，請參閱 建立 Amazon EKS 叢集。
- 您部署到叢集的 Amazon VPC CNI 附加元件的版本必須是 1.10.1 或更高版本。依預設，將部署此版本或更新版本。部署附加元件後，您就無法將 Amazon VPC CNI 附加元件降級為低於 1.10.1 的版本，且不先移除叢集中所有節點群組中的所有節點。
- 不支援 Windows Pods 和 services。

- 如果您使用 Amazon EC2 節點，則必須使用 IP 字首委派和 IPv6 設定 Amazon VPC CNI 附加元件。如果您在建立叢集時選擇 IPv6 系列，則附加元件的 1.10.1 版本預設為此設定。自我管理或 Amazon EKS 附加元件都是這種情況。如需 IP 字首委派的詳細資訊，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。
- 建立叢集時，您指定的 VPC 和子網必須具有指派給所指定 VPC 和子網的 IPv6 CIDR 區塊。此外，還必須為其指派 IPv4 CIDR 區塊。這是因為，即使您只想使用 IPv6，VPC 仍然需要 IPv4 CIDR 區塊才能正常工作。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的建立 IPv6 CIDR 區塊與 VPC 的關聯。
- 建立叢集和節點時，必須指定設定為自動指派 IPv6 地址的子網。否則，您就無法部署自己的叢集和節點。根據預設，會停用此設定。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的修改子網的公有 IPv6 定址屬性。
- 指派給子網的路由表必須具有 IPv6 地址的路由。如需詳細資訊，請參閱 《Amazon VPC 使用者指南》中的遷移至 IPv6。
- 您的安全群組必須允許 IPv6 地址。如需詳細資訊，請參閱 《Amazon VPC 使用者指南》中的遷移至 IPv6。
- 您只能IPv6與以 AWS 硝基為基礎的 Amazon EC2 或 Fargate 節點搭配使用。
- 您無法將 IPv6 與 Pods 的安全群組 和 Amazon EC2 節點搭配使用。但是，您可以將其與 Fargate 節點搭配使用。如果您需要為單個 Pods 設定單獨的安全群組，請繼續將 IPv4 系列與 Amazon EC2 節點搭配使用，或者改用 Fargate 節點。
- 如果您以前使用自訂聯網來幫助緩解 IP 地址耗盡情況，則可以改用 IPv6。您無法將自訂聯網與 IPv6 搭配使用。如果您使用自訂聯絡進行網路隔離，則可能需要繼續為叢集使用自訂聯網和 IPv4 系列。
- 您無法將 IPv6 與 AWS Outposts 搭配使用。
- 僅將一個 IPv6 地址指派給 Pods 和 services。沒有為其指派 IPv4 地址。Pods 能夠透過執行個體本身上的 NAT 與 IPv4 端點進行通訊，因此並不需要 DNS64 和 NAT64。如果流量需要公有 IP 地址，則流量將轉換為公有 IP 的來源網路地址。
- 在 VPC 外部通訊時，Pod 的來源 IPv6 地址不是轉換為節點 IPv6 地址的來源網路地址。它使用網際網路閘道或僅限輸出的網際網路閘道進行路由。
- 為所有節點指派 IPv4 和 IPv6 地址。
- 系統不支援 Amazon FSx for Lustre CSI 驅動程式。
- 您可以使用 AWS Load Balancer 控制器的版本2.3.1或更新版本，IPv6Pods在 IP 模式下負載平衡應用程式或網路流量，但不能平衡執行個體模式。如需詳細資訊，請參閱 什麼是 AWS Load Balancer Controller？。

- 您必須將 IPv6 IAM 政策連接至節點 IAM 或 CNI IAM 角色。在兩者之間，建議您將其連接至 CNI IAM 角色。如需更多詳細資訊，請參閱 為使用 IPv6 系列的叢集建立 IAM 政策 及 步驟 1：建立 Amazon VPC CNI plugin for Kubernetes IAM 角色。
- 每個 Fargate Pod 接收來自 CIDR 的 IPv6 地址，為其部署在其中的子網指定此地址。執行 Fargate Pods 的基礎硬體單位從 CIDR 取得唯一的 IPv4 和 IPv6 地址，這些地址指派給部署硬體單位的子網。
- 我們建議您在部署IPv6叢集之前，對整合的應用程式、Amazon EKS 附加元件和 AWS 服務執行全面評估。這是為了確保使用 IPv6 時一切都能正常工作。
- Amazon EKS 不支援使用 Amazon EC2 執行個體中繼資料服務 IPv6 端點。
- 在使用 IPv6 系列的叢集中建立自我管理的節點群組時，使用者資料必須包含在節點啟動時執行的 bootstrap.sh 檔案的下列 BootstrapArguments。使用叢集 VPC 的 IPv6 CIDR 範圍取代 *your-cidr*。

```
--ip-family ipv6 --service-ipv6-cidr your-cidr
```

如果您不知道叢集的IPv6CIDR範圍，可以使用下列命令 (需要 AWS CLI 版本2.4.9或更新版本) 來查看它。

```
aws eks describe-cluster --name my-cluster --query
 cluster.kubernetesNetworkConfig.serviceIpv6Cidr --output text
```

部署 **IPv6** 叢集和受管 Amazon Linux 節點

在本教學課程中，您將部署 IPv6 Amazon VPC、含 IPv6 系列的 Amazon EKS 叢集，以及含有 Amazon EC2 Amazon Linux 節點的受管節點群組。您無法在 IPv6 叢集中部署 Amazon EC2 Windows 節點。您也可以將 Fargate 節點部署到自己的叢集，但為了簡單起見，本主題中未提供這些指示。

建立叢集供生產使用之前，建議您先熟悉所有設定，再使用符合需求的設定來部署叢集。如需詳細資訊，請參閱此主題的 建立 Amazon EKS 叢集、受管節點群組 與考量。您只可在建立叢集時啟用某些設定。

必要條件

開始此教學之前，您必須先安裝和設定下列在建立和管理 Amazon EKS 叢集所需的工具和資源。

- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。
- 您使用的 IAM 安全主體必須具有使用 Amazon EKS IAM 角色、服務連結角色 AWS CloudFormation、VPC 和相關資源的許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 Amazon Elastic Kubernetes Service 的動作、資源和條件索引鍵以及使用服務連結角色。

提供使用 eksctl 或 AWS CLI建立資源的程序。您也可以使用部署資源 AWS Management Console，但為了簡單起見，本主題未提供這些指示。

eksctl

先決條件

安裝在您的電腦上的 eksctl 0.183.0 或更新版本。如需有關安裝或更新的指示，請參閱 eksctl 文件中的 Installation 一節。

使用 **eksctl** 部署 **IPv6** 叢集

1. 建立 ipv6-cluster.yaml 檔案。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令：
   - 使用您的叢集名稱取代 *my-cluster*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - 使用 Amazon EKS 支援的任何 AWS 區域 取代 *region-code*。如需清單 AWS 區域，請參閱 AWS 一般參考指南中的 Amazon EKS 端點和配額。
   - 包含叢集版本的 version 值。如需詳細資訊，請參閱支援的 Amazon EKS Kubernetes 版本。
   - 將 *my-nodegroup* 取代為您的節點群組名稱。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。
   - 將 *t3.medium* 取代為任何 AWS Nitro 系統執行個體類型。

```
cat >ipv6-cluster.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
```

```
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "X.XX"

kubernetesNetworkConfig:
  ipFamily: IPv6

addons:
  - name: vpc-cni
    version: latest
  - name: coredns
    version: latest
  - name: kube-proxy
    version: latest

iam:
  withOIDC: true

managedNodeGroups:
  - name: my-nodegroup
    instanceType: t3.medium
EOF
```

2. 建立叢集。

```
eksctl create cluster -f ipv6-cluster.yaml
```

叢集建立需要幾分鐘的時間。在看到最後一行輸出之前，請勿繼續操作，該行看起來類似於以下輸出。

```
[...]
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

3. 確認已為預設 Pods 指派 IPv6 地址。

```
kubectl get pods -n kube-system -o wide
```

範例輸出如下。

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

4. 確認已為預設服務指派 IPv6 地址。

```
kubectl get services -n kube-system -o wide
```

範例輸出如下。

```
NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
 SELECTOR
kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
 k8s-app=kube-dns
```

5. (選用) 部署範例應用程式或部署 AWS Load Balancer Controller 和範例應用程式，將應用程式或網路流量負載平衡至 IPv6 Pods。
6. 在完成為本教學建立的叢集和節點後，您應該使用如下命令清除所建立的資源。

```
eksctl delete cluster my-cluster
```

AWS CLI

先決條件

您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。如果您使用 AWS CloudShell，則可能需要安裝的版本2.12.3或更高版本1.27.160或更高版本 AWS CLI，因為安裝在中的預設 AWS CLI 版本 AWS CloudShell 可能是較早的版本。

> **Important**
>
> - 您必須以同一位使用者的身分完成本程序中的所有步驟。若要檢查目前使用者，請執行以下命令：
>
>   ```
>   aws sts get-caller-identity
>   ```
>
> - 必須在同一 shell 中完成此程序中的所有步驟。若干步驟使用前面步驟中設定的變數。如果在其他 shell 中設定變數值，則使用此變數的步驟將無法正常工作。如果您使用 AWS CloudShell 完成以下程序，請記住，如果您在大約 20—30 分鐘內沒有使用鍵盤或指標與其互動，則 shell 工作階段將結束。正在執行的進程不算作互動。
> - 這些指示是針對 Bash shell 編寫的，在其他 shell 中可能需要進行調整。

若要建立叢集 AWS CLI

將程序此步驟中的所有 *example values* 取代為您自己的值。

1. 執行下列命令以設定稍後步驟中使用的某些變數。*region-code*替換為您 AWS 區域 要在其中部署資源的。該值可以 AWS 區域 是 Amazon EKS 支持的任何值。如需清單 AWS 區域，請參閱 AWS 一般參考指南中的 Amazon EKS 端點和配額。使用您的叢集名稱取代 *my-cluster*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。將 *my-nodegroup* 取代為您的節點群組名稱。節點群組名稱不可超過 63 個字

元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。使用您的帳戶 ID 取代 *111122223333*。

```
export region_code=region-code
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
```

2. 使用符合 Amazon EKS 和 IPv6 要求的公有和私有子網建立 Amazon VPC。

   a. 運行以下命令為 AWS CloudFormation 堆棧名稱設置一個變量。您可以使用您選擇的任何名稱取代 *my-eks-ipv6-vpc*。

   ```
   export vpc_stack_name=my-eks-ipv6-vpc
   ```

   b. 使用 AWS CloudFormation 範本建立 IPv6 VPC。

   ```
   aws cloudformation create-stack --region $region_code --stack-name
    $vpc_stack_name \
     --template-url https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-
   subnets.yaml
   ```

   堆疊需要幾分鐘的時間建立。執行下列命令。在命令輸出為 CREATE_COMPLETE 之前，請勿繼續下一步。

   ```
   aws cloudformation describe-stacks --region $region_code --stack-name
    $vpc_stack_name --query Stacks[].StackStatus --output text
   ```

   c. 檢索已建立的公有子網路的 ID。

   ```
   aws cloudformation describe-stacks --region $region_code --stack-name
    $vpc_stack_name \
       --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
   output text
   ```

   範例輸出如下。

   ```
   subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE
   ```

   d. 為已建立的公有子網啟用自動指派 IPv6 地址選項。

```
aws ec2 modify-subnet-attribute --region $region_code --
subnet-id subnet-0a1a56c486EXAMPLE --assign-ipv6-address-on-
creation
aws ec2 modify-subnet-attribute --region $region_code --subnet-id
 subnet-099e6ca77aEXAMPLE --assign-ipv6-address-on-creation
```

e. 從已部署的 AWS CloudFormation 堆疊擷取由範本建立的子網路和安全群組的名稱，並將其儲存在變數中，以便在稍後的步驟中使用。

```
security_groups=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SecurityGroups`].OutputValue' --
output text)

public_subnets=$(aws cloudformation describe-stacks --region $region_code --
stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
output text)

private_subnets=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPrivate`].OutputValue' --
output text)

subnets=${public_subnets},${private_subnets}
```

3. 建立叢集 IAM 角色，並將必要的 Amazon EKS IAM 受管政策附加到該角色。KubernetesAmazon EKS 管理的叢集代表您呼叫其他 AWS 服務，以管理您搭配服務使用的資源。

a. 執行下列命令以建立 `eks-cluster-role-trust-policy.json` 檔案。

```
cat >eks-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
```

```
    }
  ]
}
EOF
```

b. 執行下列命令，以設定角色名稱的變數。您可以使用您選擇的任何名稱取代 *myAmazonEKSClusterRole*。

```
export cluster_role_name=myAmazonEKSClusterRole
```

c. 建立角色。

```
aws iam create-role --role-name $cluster_role_name --assume-role-policy-
document file://"eks-cluster-role-trust-policy.json"
```

d. 擷取 IAM 角色的 ARN 並將其儲存在變數中，以便在稍後的步驟中使用。

```
cluster_iam_role=$(aws iam get-role --role-name $cluster_role_name --
query="Role.Arn" --output text)
```

e. 將必要的 Amazon EKS 受管 IAM 政策連接到角色。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name $cluster_role_name
```

4. 建立叢集。

```
aws eks create-cluster --region $region_code --name $cluster_name --kubernetes-
version 1.XX \
   --role-arn $cluster_iam_role --resources-vpc-config subnetIds=
$subnets,securityGroupIds=$security_groups \
   --kubernetes-network-config ipFamily=ipv6
```

-

> **Note**
>
> 您可能會收到錯誤，表示在請求中的其中一個可用區域沒有足夠的容量可建立 Amazon EKS 叢集。如果發生這種情況，錯誤輸出包含的可用區域可支援新的叢集。使用至少兩個位於帳戶的支援可用區域子網路來建立您的叢集。如需詳細資訊，請參閱 容量不足。

建立叢集需要幾分鐘才能完成。執行下列命令。在命令輸出為 ACTIVE 之前，請勿繼續下一步。

```
aws eks describe-cluster --region $region_code --name $cluster_name --query
 cluster.status
```

5. 為叢集建立或更新 kubeconfig 檔案，以便能夠與您的叢集通訊。

```
aws eks update-kubeconfig --region $region_code --name $cluster_name
```

根據預設，已在 ~/.kube 建立了 config 檔案，或者已將新叢集的組態新增至現有 ~/.kube 中的 config 檔案。

6. 建立節點 IAM 角色。

a. 執行下列命令以建立 vpc-cni-ipv6-policy.json 檔案。

```
cat >vpc-cni-ipv6-policy <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
```

```
    ]
}
EOF
```

b. 建立 IAM 政策。

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

c. 執行下列命令以建立 node-role-trust-relationship.json 檔案。

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

d. 執行下列命令，以設定角色名稱的變數。您可以使用您選擇的任何名稱取代 *AmazonEKSNodeRole*。

```
export node_role_name=AmazonEKSNodeRole
```

e. 建立 IAM 角色。

```
aws iam create-role --role-name $node_role_name --assume-role-policy-
document file://"node-role-trust-relationship.json"
```

f. 將 IAM 政策連接至 IAM 角色。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy \
    --role-name $node_role_name
```

> **Important**
>
> 為實現本教學的簡單性，將政策連接至此 IAM 角色。但是，在生產叢集中，我們建議將政策連接至單獨的 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

g. 將兩個所需的 IAM 受管政策連接到 IAM 角色。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
```

h. 擷取 IAM 角色的 ARN 並將其儲存在變數中，以便在稍後的步驟中使用。

```
node_iam_role=$(aws iam get-role --role-name $node_role_name --
query="Role.Arn" --output text)
```

7. 建立受管節點群組。

a. 查看您在上一個步驟中建立的子網路的 ID。

```
echo $subnets
```

範例輸出如下。

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE,subnet-
0377963d69EXAMPLE,subnet-0c05f819d5EXAMPLE
```

b. 建立節點群組。將 *0a1a56c486EXAMPLE*、*099e6ca77aEXAMPLE*、*0377963d69EXAMPLE*，以及 *0c05f819d5EXAMPLE* 取代為前一個步驟輸出中傳回的值。請確保從以下命令中的上一個輸出刪除子網路 ID 之間的逗號。您可以將 *t3.medium* 取代為任何 AWS Nitro 系統執行個體類型。

```
aws eks create-nodegroup --region $region_code --cluster-name $cluster_name
 --nodegroup-name $nodegroup_name \
```

```
    --subnets subnet-0a1a56c486EXAMPLE subnet-099e6ca77aEXAMPLE
 subnet-0377963d69EXAMPLE subnet-0c05f819d5EXAMPLE \
    --instance-types t3.medium --node-role $node_iam_role
```

此節點群組需要幾分鐘的時間建立。執行下列命令。在傳回的輸出為 ACTIVE 之前，請勿進行下一個步驟。

```
aws eks describe-nodegroup --region $region_code --cluster-name
 $cluster_name --nodegroup-name $nodegroup_name \
    --query nodegroup.status --output text
```

8. 確認在 IP 欄中已為預設 Pods 指派 IPv6 地址。

```
kubectl get pods -n kube-system -o wide
```

範例輸出如下。

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

9. 確認在 IP 欄中已為預設服務指派 IPv6 地址。

```
kubectl get services -n kube-system -o wide
```

範例輸出如下。

```
NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
 SELECTOR
kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
 k8s-app=kube-dns
```

10. (選用) 部署範例應用程式或部署 AWS Load Balancer Controller 和範例應用程式，將應用程式或網路流量負載平衡至 IPv6 Pods。
11. 在完成為本教學建立的叢集和節點後，您應該使用如下命令清除所建立的資源。在刪除之前，請確保您沒有使用本教程以外的任何資源。

    a. 如果您在與完成之前步驟不同的 shell 中完成此步驟，請設定之前步驟中使用的所有變數的值，並將 *example values* 取代為在完成之前步驟時指定的值。如果您要在與完成之前步驟相同的 shell 中完成此步驟，請跳至下一步。

    ```
    export region_code=region-code
    export vpc_stack_name=my-eks-ipv6-vpc
    export cluster_name=my-cluster
    export nodegroup_name=my-nodegroup
    export account_id=111122223333
    export node_role_name=AmazonEKSNodeRole
    export cluster_role_name=myAmazonEKSClusterRole
    ```

    b. 刪除節點群組。

    ```
    aws eks delete-nodegroup --region $region_code --cluster-name $cluster_name
     --nodegroup-name $nodegroup_name
    ```

    刪除需要幾分鐘的時間。執行下列命令。如果傳回任何輸出，請勿繼續下一步。

    ```
    aws eks list-nodegroups --region $region_code --cluster-name $cluster_name
     --query nodegroups --output text
    ```

    c. 刪除叢集。

    ```
    aws eks delete-cluster --region $region_code --name $cluster_name
    ```

叢集需要幾分鐘的時間刪除。在繼續之前，請務必使用下列命令來確定叢集已被刪除。

```
aws eks describe-cluster --region $region_code --name $cluster_name
```

在輸出與下列輸出類似之前，請不要進行下一個步驟。

```
An error occurred (ResourceNotFoundException) when calling the
 DescribeCluster operation: No cluster found for name: my-cluster.
```

d. 刪除您建立的 IAM 資源。將 *AmazonEKS_CNI_IPv6_Policy* 取代為您選擇的名稱 (如果您選擇的名稱與之前步驟中使用的名稱不同)。

```
aws iam detach-role-policy --role-name $cluster_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-role --role-name $cluster_role_name
aws iam delete-role --role-name $node_role_name
```

e. 刪除建立 VPC 的 AWS CloudFormation 堆疊。

```
aws cloudformation delete-stack --region $region_code --stack-name
 $vpc_stack_name
```

適用於 Pods 的 SNAT

如果您使用 IPv6 系列部署叢集，則本主題中的資訊並不適用於您的叢集，因為 IPv6 地址未經過網路轉換。如需搭配使用 IPv6 與叢集的詳細資訊，請參閱 IPv6叢集的位址Pods、和 services。

預設情況下，叢集中的每個 Pod 均從與部署 Pod 的 VPC 關聯的無類別域間路由 (CIDR) 區指派一個私有 IPv4 地址。同一 VPC 中的 Pods 使用這些私有 IP 地址作為端點相互通訊。當 Pod 與任何不在與您的 VPC 關聯的 CIDR 區塊內的 IPv4 地址通訊時，預設情況下，Amazon VPC CNI 外掛程式 (適用

於 Linux 或 Windows) 會將 Pod's IPv4 地址轉譯為正在執行 Pod 的節點的主要彈性網路界面的主要私有 IPv4 地址 -*。

> **Note**
>
> 對於 Windows 節點，還有其他詳細資訊需要考慮。依預設，Windows 的 VPC CNI 外掛程式為使用網路組態定義，其中，相同 VPC 內通往目的地的流量會從 SNAT 中排除。這表示內部 VPC 通訊已停用 SNAT，且配置給 Pod 的 IP 地址可在 VPC 內路由。但是通往 VPC 外目的地的流量會將來源 Pod IP SNAT 到執行個體 ENI 的主要 IP 地址。Windows 的此預設組態可確保 Pod 可以使用與主機執行個體相同的方式存取 VPC 外部的網路。

由於此行為：

- 僅當 Pod 執行的節點獲指派公有或彈性 IP 地址，且位於公有子網路時，您的 Pods 才能與網際網路資源進行通訊。公有子網路與具有網際網路閘道路由的路由表相關聯。我們建議儘可能將節點部署到私有子網。
- 對於版本比 1.8.0 更早的外掛程式，使用 VPC 對等互連、傳輸 VPC 或 AWS Direct Connect 連線到叢集 VPC 的網路或 VPC 的資源，無法針對輔助彈性網路介面背後的 Pods 啟動通訊。您的 Pods 可以啟動與這些資源的通訊並接收來自資源的回應。

如果在您的環境中下列任一陳述式成立，請使用下列命令變更預設組態。

- 您在網路或 VPC 擁有資源，且其使用 VPC 對等互連、傳輸 VPC 或 AWS Direct Connect 連線到叢集，且其需使用 IPv4 位址與您的 Pods 啟動通訊，而您的外掛程式版本早於 1.8.0。
- 您的 Pods 位於私有子網路中，需要與網際網路進行對外通訊。子網路包含指向 NAT 閘道的路由。

```
kubectl set env daemonset -n kube-system aws-node AWS_VPC_K8S_CNI_EXTERNALSNAT=true
```

> **Note**
>
> AWS_VPC_K8S_CNI_EXTERNALSNAT 與 AWS_VPC_K8S_CNI_EXCLUDE_SNAT_CIDRS CNI 組態變數不適用 Windows 節點。Windows 不支援停用 SNAT。對於從 SNAT 排除 IPv4 CIDR，您可透過在 Windows 引導指令碼中指定 ExcludedSnatCIDRs 參數來進行定義。如需使用此參數的詳細資訊，請參閱 引導指令碼組態參數。

* 如果 Pod's 規格包含 `hostNetwork=true` (預設為 `false`)，則其 IP 地址不會被轉換為其他地址。預設情況下，在您的叢集上執行的 `kube-proxy` 和 Amazon VPC CNI plugin for Kubernetes Pods 即為這種情況。對於這些 Pods，IP 地址與節點的主 IP 地址相同，因此未轉譯 Pod's IP 地址。如需有關 Pod's`hostNetwork`設定的詳細資訊，請參閱 Kubernetes API 參考資料中的 PodSpec v1 核心。

為 Kubernetes 網路政策設定您的叢集

根據預設，Kubernetes 中沒有任何限制施加於 IP 地址、連接埠，或叢集中任何 Pods 之間或您的 Pods 與任何其他網路中的資源之間的連結。您可以使用 Kubernetes 網路政策來限制往返 Pods 之間的網路流量。如需詳細資訊，請參閱 Kubernetes 文件中的網路政策。

如果您在叢集上有版本 `1.13` 或更早版本的 Amazon VPC CNI plugin for Kubernetes，您需要實作第三方解決方案才能將 Kubernetes 套用至叢集的網路政策。外掛程式的 `1.14` 版本或更新版本可以實作網絡政策，因此您不需要使用第三方解決方案。在本主題中，您將學習如何設定叢集以在叢集上使用 Kubernetes 網路政策，而不需要使用第三方附加元件。

Amazon VPC CNI plugin for Kubernetes 中的網路政策在以下組態中受到支援。

- Amazon EKS 叢集 `1.25` 版本及更新版本。
- 在您叢集上版本 1.14 或更高版本的 Amazon VPC CNI plugin for Kubernetes。
- 設定為 `IPv4` 或 `IPv6` 地址的叢集。
- 您可以搭配 Pods 的安全群組來使用網路政策。有了網路政策，您就可以控制所有叢集內通訊。使用的安全性群組Pods，您可以控制 AWS 服務 從Pod.
- 您可以搭配自訂聯網和字首委派來使用網路政策。

考量事項

- 使用 Amazon VPC CNI plugin for Kubernetes 將 Amazon VPC CNI plugin for Kubernetes 網路政策套用至您的叢集時，您只能將這些政策套用至 Amazon EC2 Linux 節點。您無法將這些政策套用至 Fargate 或 Windows 節點。
- 如果您的叢集目前正在使用第三方解決方案來管理 Kubernetes 網路政策，您可以搭配 Amazon VPC CNI plugin for Kubernetes 來使用那些相同的政策。然而，您必須移除現有的解決方案，如此才不會管理相同的政策。
- 您可以將多個網路政策套用至相同的 Pod。當設定了兩個以上選取相同 Pod 的政策，所有政策皆會套用至 Pod。
- 網路原則中每個通訊協定`ingress:`或`egress:`選取器中每個通訊協定的唯一連接埠組合數目上限為 24 個。

- 對於任何一種您的 Kubernetes 服務，服務連接埠必須與容器連接埠相同。如果您使用的是已命名連接埠，請也要在服務規格中使用相同的名稱。
- Pod啟動時執行原則

  Amazon VPC CNI plugin for Kubernetes 會在 Pod 佈建的同時設定 Pod 的網路政策。在為新網繭設定所有原則之前，新網繭中的容器將以預設允許原則啟動。這就是所謂的標準模式。預設允許原則表示允許所有輸入和輸出流量進出新網繭。

  您可以`strict`在 VPC CNI `DaemonSet` 的`aws-node`容器中將環境變數設定`NETWORK_POLICY_ENFORCING_MODE`為，以變更此預設網路原則。

```
env:
  - name: NETWORK_POLICY_ENFORCING_MODE
    value: "strict"
```

  將`NETWORK_POLICY_ENFORCING_MODE`變數設定為時`strict`，使用 VPC CNI 的網繭會以預設拒絕原則開始，然後設定原則。這稱為嚴格模式。在嚴格模式下，您必須針對網繭需要在叢集中存取的每個端點建立網路原則。請注意，此需求適用於CoreDNS網繭。未針對具有主機網路的網繭設定預設拒絕原則。
- 網路政策功能會建立且需要稱為 `policyendpoints.networking.k8s.aws` 的 `PolicyEndpoint` 自訂資源定義 (CRD)。自訂資源的 `PolicyEndpoint` 物件是由 Amazon EKS 管理。您不應修改或刪除這些資源。
- 如果您執行使用執行個體角色 IAM 憑證的 Pod 或連線至 EC2 IMDS，請小心檢查是否有會封鎖 EC2 IMDS 存取的網路政策。您可能需要新增網路政策以允許 EC2 IMDS 的存取權。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的執行個體中繼資料和使用者資料。

  使用服務帳戶的 IAM 角色的 Pod 不會存取 EC2 IMD。
- Amazon VPC CNI plugin for Kubernetes 不會將網路政策套用至每個 Pod 的其他網路介面，而只會套用每個 Pod 的主要介面 (`eth0`)。這會影響下列架構：
  - `ENABLE_V4_EGRESS` 變數設定為 `true` 的 `IPv6` Pod。此變數可讓 `IPv4` 輸出功能將 IPv6 Pod 連線到 `IPv4` 端點，例如叢集外部的端點。`IPv4` 輸出功能的運作方式是使用本機迴路 IPv4 地址建立額外的網路介面。
  - 當使用鏈接的網絡插件，Multus如. 由於這些外掛程式會將網路介面新增至每個 Pod，因此網路政策不會套用至鏈結的網路外掛程式。

- 依預設，網路政策功能會使用節點的連接埠 8162 做為指標。此外，該功能使用了連接埠 8163 進行健康狀態探查。如果您在需要使用這些連接埠的節點或 Pod 內部執行另一應用程式，則該應用程式將無法執行。在 VPC CNI 版本 `v1.14.1` 或更新版本，您可在下列位置變更這些連接埠：

  AWS Management Console

  1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
  2. 在左側導覽窗格中，選取叢集，然後選取您要為其設定 Amazon VPC CNI 附加元件的叢集名稱。
  3. 選擇附加元件索引標籤。
  4. 選取附加元件方塊右上方的方塊，然後選擇 Edit (編輯)。
  5. 在 Configure ***name of addon*** (設定 name of addon) 頁面中：

     a. 在版本下拉式清單中，選取 `v1.14.0-eksbuild.3` 或更高版本。

     b. 展開選用組態設定。

     c. 在組態值中輸入 JSON 金鑰 `"enableNetworkPolicy":` 和值 `"true"`。產生的文字必須是有效的 JSON 物件。如果此金鑰和值是文字方塊中唯一的資料，請以大括號 `{}` 括住該金鑰和值。

        下列範例已啟用網路政策功能、啟用網路政策日誌、傳送至 Amazon Logs 的網路政策 CloudWatch 日誌，以及指標和健康狀態探查設定為預設連接埠號碼：

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true",
        "enableCloudWatchLogs": "true",
        "healthProbeBindAddr": "8163",
        "metricsBindAddr": "8162"
    }
}
```

  Helm

  如果您已透過 `helm` 安裝 Amazon VPC CNI plugin for Kubernetes，您可更新組態來變更連接埠。

- 執行下列命令來變更連接埠。分別在金鑰 nodeAgent.metricsBindAddr 或金鑰 nodeAgent.healthProbeBindAddr 值設定連接埠號碼。

```
helm upgrade --set nodeAgent.metricsBindAddr=8162 --set
 nodeAgent.healthProbeBindAddr=8163 aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. 在您的編輯器中開啟 aws-node DaemonSet。

```
kubectl edit daemonset -n kube-system aws-node
```

2. 在 VPC CNI aws-node daemonset 清單檔案的 aws-network-policy-agent 容器，取代 args: 中下列命令引數的連接埠號碼。

```
    - args:
            - --metrics-bind-addr=:8162
            - --health-probe-bind-addr=:8163
```

必要條件

- 最低叢集版本

現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。叢集必須是 Kubernetes 版 1.25 或更高版本。叢集必須執行下表所列的其中一種 Kubernetes 版本和平台版本。請注意，也支援比上列任何 Kubernetes 與平台版本更新的版本。您可使用叢集名稱取代下列命令的 *my-cluster*，然後執行修改的命令來檢查目前 Kubernetes 版本：

```
aws eks describe-cluster
              --name my-cluster --query cluster.version --output
              text
```

Kubernetes 版本	平台版本
1.27.4	eks.5

Kubernetes 版本	平台版本
1.26.7	eks.6
1.25.12	eks.7

- 最低 VPC CNI 版本

  在您叢集上版本 1.14 或更高版本的 Amazon VPC CNI plugin for Kubernetes。您可以使用下列命令來查看您目前擁有哪個版本。

  ```
  kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
   cut -d : -f 3
  ```

  如果您的版本早於 1.14，請查看 更新 Amazon EKS 附加元件 以升級到版本 1.14 或更高版本。

- 最低 Linux 核心版本

  您的節點必須具有 Linux 核心版本 5.10 或更高版本。您可以使用 uname -r 來檢查您的核心版本。如果您使用的是最新版本的 Amazon EKS 最佳化 Amazon Linux、Amazon EKS 最佳化加速 Amazon Linux AMI 和 Bottlerocket AMI，則它們已經具有所需的核心版本。

  Amazon EKS 最佳化加速 Amazon Linux AMI 版本 v20231116 或具有核心版本 5.10 的更新版本。

若要設定叢集來使用 Kubernetes 網路政策

1. 掛載 BPF 檔案系統

   > **Note**
   >
   > 如果您的叢集是版本 1.27 或更高版本，您可以跳過此步驟，因為所有 Amazon EKS 最佳化的 Amazon Linux 和 1.27 或更高版本的 Bottlerocket AMI 均已擁有此功能。
   > 對於所有其他叢集版本，如果您將 Amazon EKS 最佳化的 Amazon Linux 升級到版本 v20230703 或者更高版本，或者您將 Bottlerocket AMI 升級到版本 v1.0.2 或更高版本，則可以跳過此步驟。

   a. 在每個節點上掛載 Berkeley Packet Filter (BPF) 檔案系統。

```
sudo mount -t bpf bpffs /sys/fs/bpf
```

b. 接著，將相同的命令新增至您 Amazon EC2 Auto Scaling 群組的啟動範本中的使用者資料。

2. 在 VPC CNI 中啟用網路政策

a. 查看叢集上安裝的附加元件類型。視您用來建立叢集的工具而定，您的叢集上目前可能沒有安裝 Amazon EKS 附加元件類型。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

如果傳回版本編號，則表明已在叢集上安裝 Amazon EKS 類型的附加元件，並且無需完成此程序中的剩餘步驟。如果傳回錯誤，則表明沒有在叢集上安裝 Amazon EKS 類型的附加元件。

b. • Amazon EKS 附加元件

AWS Management Console

a. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

b. 在左側導覽窗格中，選取叢集，然後選取您要為其設定 Amazon VPC CNI 附加元件的叢集名稱。

c. 選擇附加元件索引標籤。

d. 選取附加元件方塊右上方的方塊，然後選擇 Edit (編輯)。

e. 在 Configure ***name of addon*** (設定 name of addon) 頁面中：

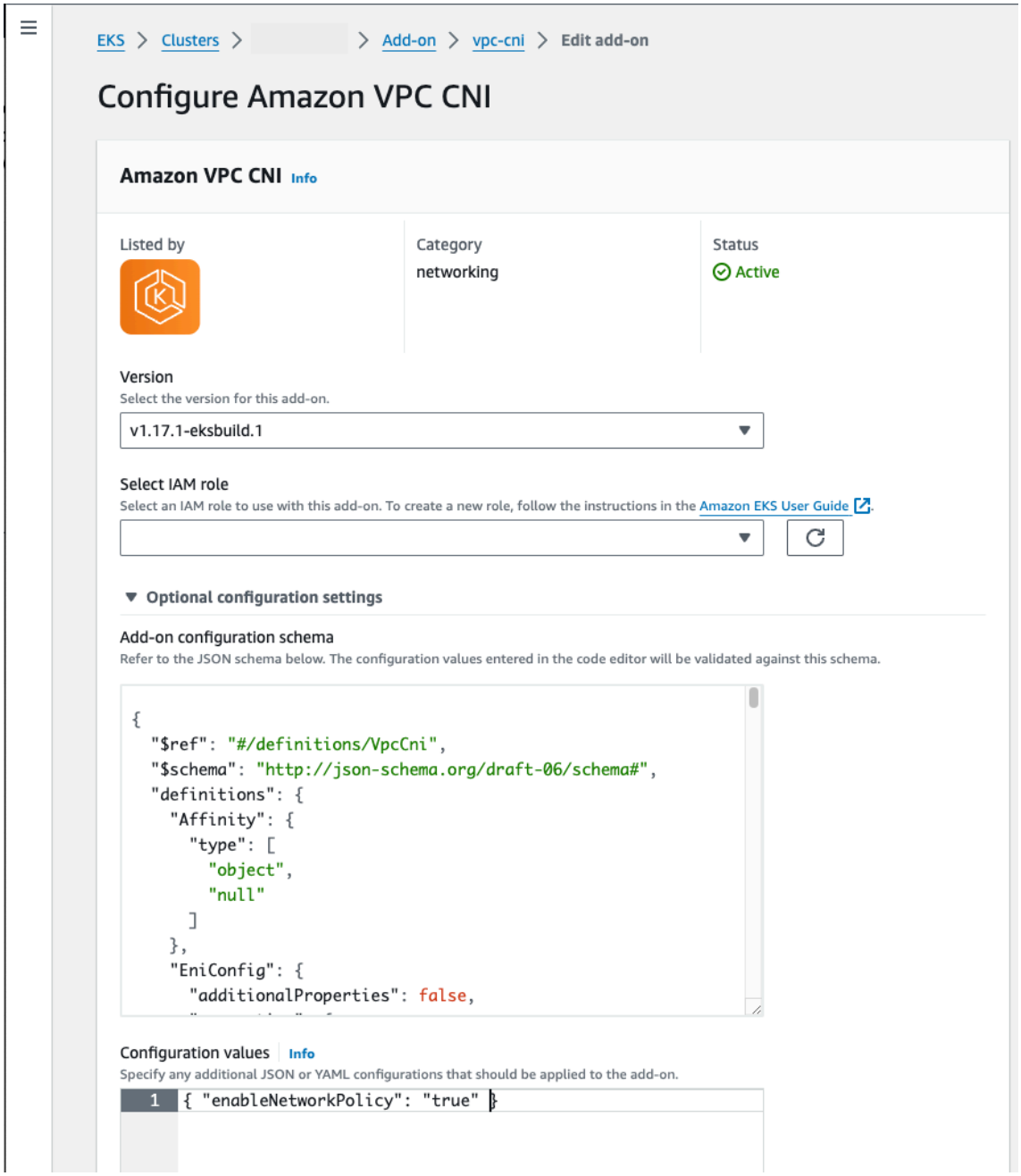
i. 在版本下拉式清單中，選取 `v1.14.0-eksbuild.3` 或更高版本。

ii. 展開選用組態設定。

iii. 在組態值中輸入 JSON 金鑰 `"enableNetworkPolicy":` 和值 `"true"`。產生的文字必須是有效的 JSON 物件。如果此金鑰和值是文字方塊中唯一的資料，請以大括號 `{}` 括住該金鑰和值。下列範例顯示網路原則已啟用：

```
{ "enableNetworkPolicy": "true" }
```

下列螢幕擷取畫面展示了案例的範例。



AWS CLI

- 執行下列 AWS CLI 命令。將 `my-cluster` 取代為您的叢集名稱，並將 IAM 角色 ARN 取代為您正在使用的角色。

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni
 --addon-version v1.14.0-eksbuild.3 \
```

```
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values
 '{"enableNetworkPolicy": "true"}'
```

- 自我管理附加元件

  Helm

  如果您已透過 `helm` 安裝 Amazon VPC CNI plugin for Kubernetes，您可以更新組態以啟用網路政策。

  - 執行下列命令以啟用網路政策。

    ```
    helm upgrade --set enableNetworkPolicy=true aws-vpc-cni --namespace
     kube-system eks/aws-vpc-cni
    ```

  kubectl

  a. 在您的編輯器中開啟 `amazon-vpc-cni` ConfigMap。

     ```
     kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
     ```

  b. 在 ConfigMap 中，加入下列行至 `data`。

     ```
     enable-network-policy-controller: "true"
     ```

     一旦您新增此行，您的 ConfigMap 看起來應該會像下列範例。

     ```
     apiVersion: v1
      kind: ConfigMap
      metadata:
       name: amazon-vpc-cni
       namespace: kube-system
      data:
       enable-network-policy-controller: "true"
     ```

  c. 在您的編輯器中開啟 `aws-node` DaemonSet。

```
kubectl edit daemonset -n kube-system aws-node
```

d. 在 VPC CNI aws-node daemonset 清單檔案的 aws-network-policy-agent 容器，以 true 取代 args: 中命令引數 --enable-network-policy=false 裡的 false。

```
    - args:
       - --enable-network-policy=true
```

3. 確認 aws-node Pod 正在您的叢集上執行。

```
kubectl get pods -n kube-system | grep 'aws-node\|amazon'
```

範例輸出如下。

```
aws-node-gmqp7                                          2/2     Running   1 (24h
 ago)   24h
aws-node-prnsh                                          2/2     Running   1 (24h
 ago)   24h
```

如果啟用網路政策，則 aws-node Pod 中會有 2 個容器。在先前版本中，如果網路政策已停用，則 aws-node Pod 中只會有單一容器。

您現在可以將 Kubernetes 網路政策部署到您叢集。如需詳細資訊，請參閱 Kubernetes 網路政策。

網路政策的星型展示

此示範會在 Amazon EKS 叢集上建立前端、後端和用戶端服務。示範中亦將建立管理圖形使用者介面，其顯示各服務之間可用的輸入和輸出路徑。我們建議您在不執行生產工作負載的叢集上完成示範。

在您建立任何網路政策前，所有服務都可以雙向通訊。在您套用網路政策後，便可看到用戶端只能與前端服務通訊，同時後端只能接受來自前端的流量。

執行星政策示範

1. 套用前端、後端、用戶端和管理使用者介面服務：

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
```

2. 檢視叢集上的所有 Pods。

```
kubectl get pods -A
```

範例輸出如下。

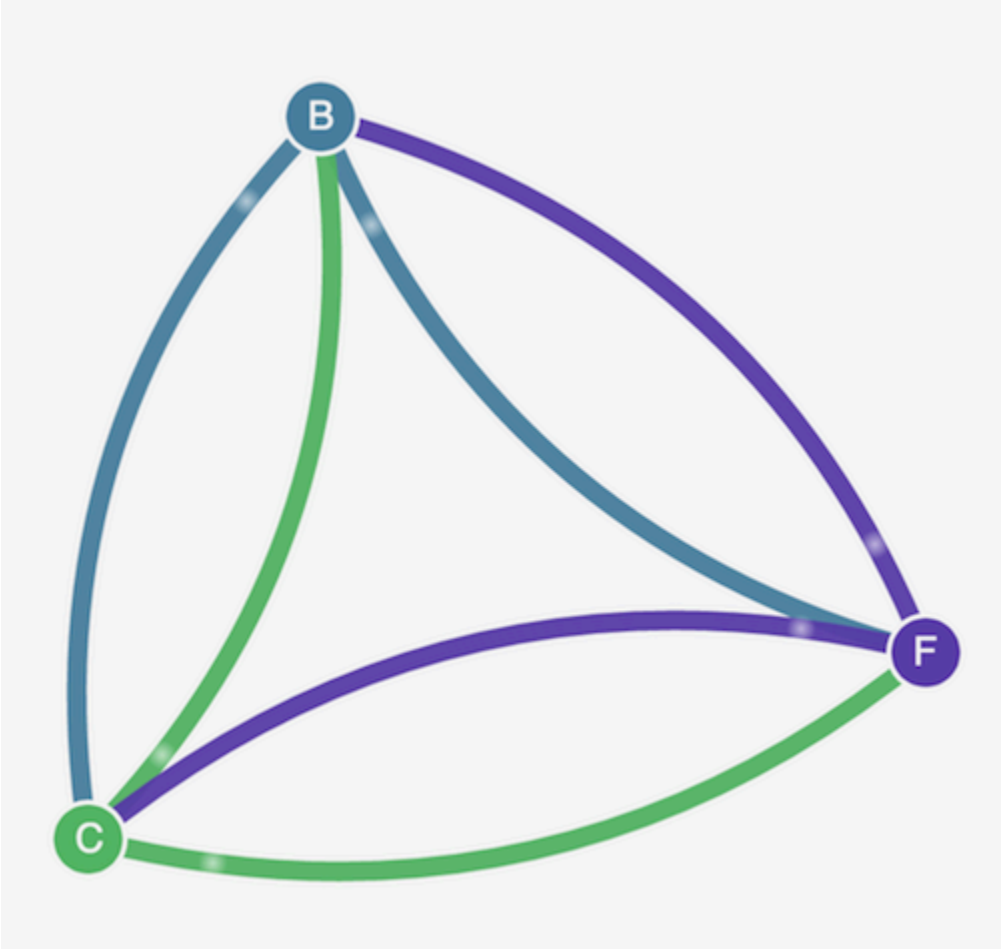
在輸出中，您應該在以下輸出中顯示的命名空間中看到 pod。*NAMES* 和 READY 直欄中的 Pod 數量與下列輸出中的數量不同。不要繼續，直到您看到具有相似名稱的 Pod 並且其在 STATUS 資料欄都有 Running 狀態。

```
NAMESPACE         NAME                                       READY   STATUS
 RESTARTS   AGE
[...]
client            client-xlffc                               1/1     Running   0
       5m19s
[...]
management-ui     management-ui-qrb2g                        1/1     Running   0
       5m24s
stars             backend-sz87q                              1/1     Running   0
       5m23s
stars             frontend-cscnf                             1/1     Running   0
       5m21s
[...]
```

3. 若要連接到管理使用者介面，請連接到在叢集上執行之服務的 EXTERNAL-IP：

```
kubectl get service/management-ui -n management-ui
```

4. 打開瀏覽器到上一個步驟的位置。您應該會看到管理使用者介面。C 節點是用戶端服務，而 F 節點是前端服務，且 B 節點是後端服務。每個節點都有對所有其他節點的完整通訊存取權 (如粗體、上顏色行的文字所指示)。



5. 在 `stars` 和 `client` 命名空間中套用以下網路政策來將服務彼此隔離：

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny
spec:
  podSelector:
    matchLabels: {}
```

您可以使用下列命令將政策同時套用至兩個命名空間：

```
kubectl apply -n stars -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
```

```
kubectl apply -n client -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
```

6. 重新整理您的瀏覽器。您將看到管理使用者介面不再到達任何節點，所以各節點均不會顯示在使用者介面中。
7. 套用下列不同的網路政策，以允許管理使用者介面存取服務。套用此政策以允許 UI：

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

套用此政策以允許用戶端：

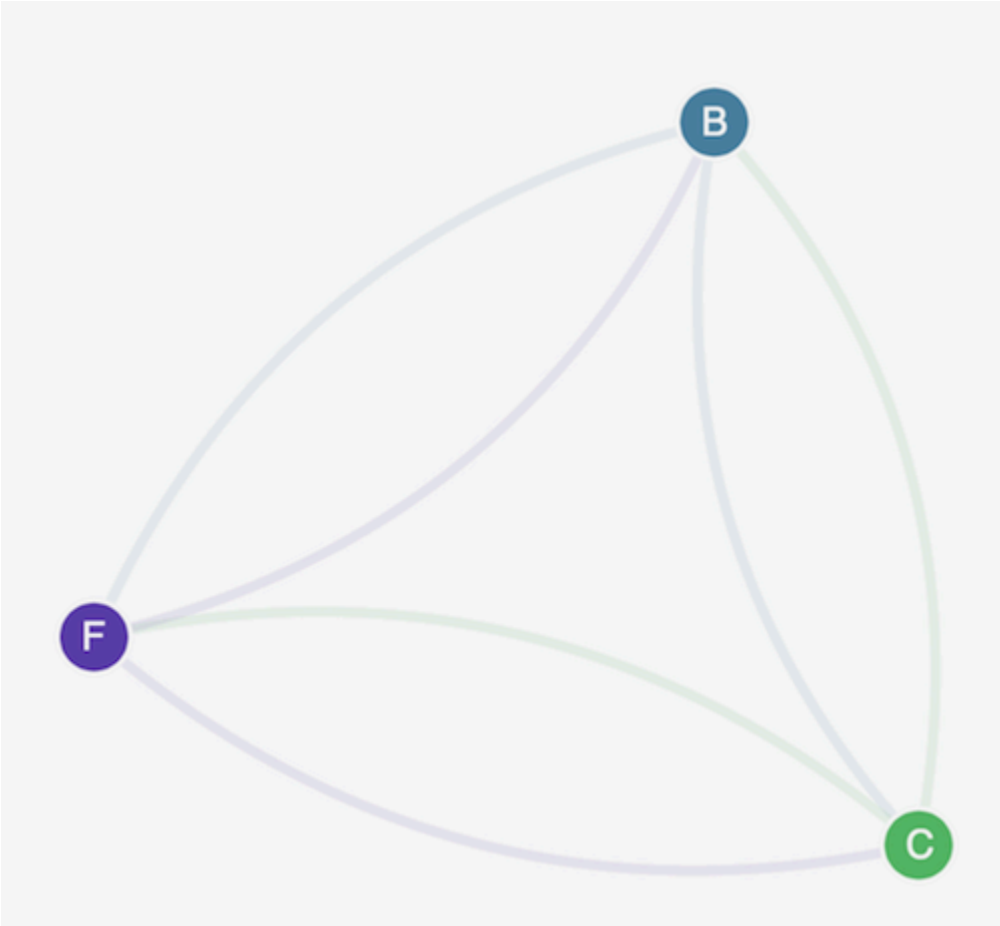
```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: client
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

您可以使用下列命令來同時套用兩個政策：

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui.yaml
```

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui-client.yaml
```

8. 重新整理您的瀏覽器。您將看到管理使用者介面再次可到達各節點，但各節點無法互相通訊。

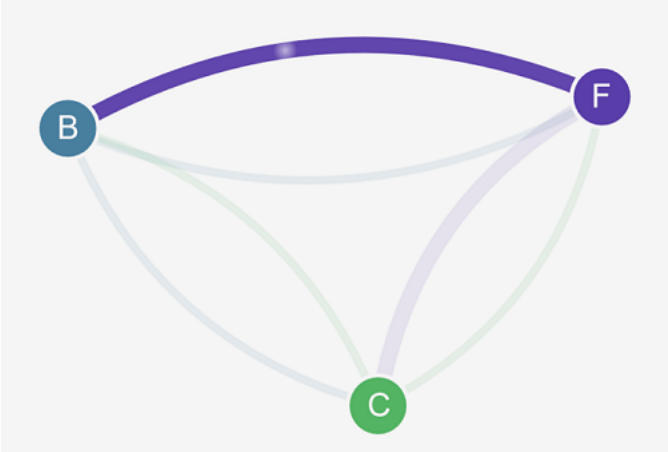


9. 套用以下網路政策以允許流量從前端服務流向後端服務：

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: backend-policy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
        - podSelector:
            matchLabels:
```

```
            role: frontend
    ports:
      - protocol: TCP
        port: 6379
```

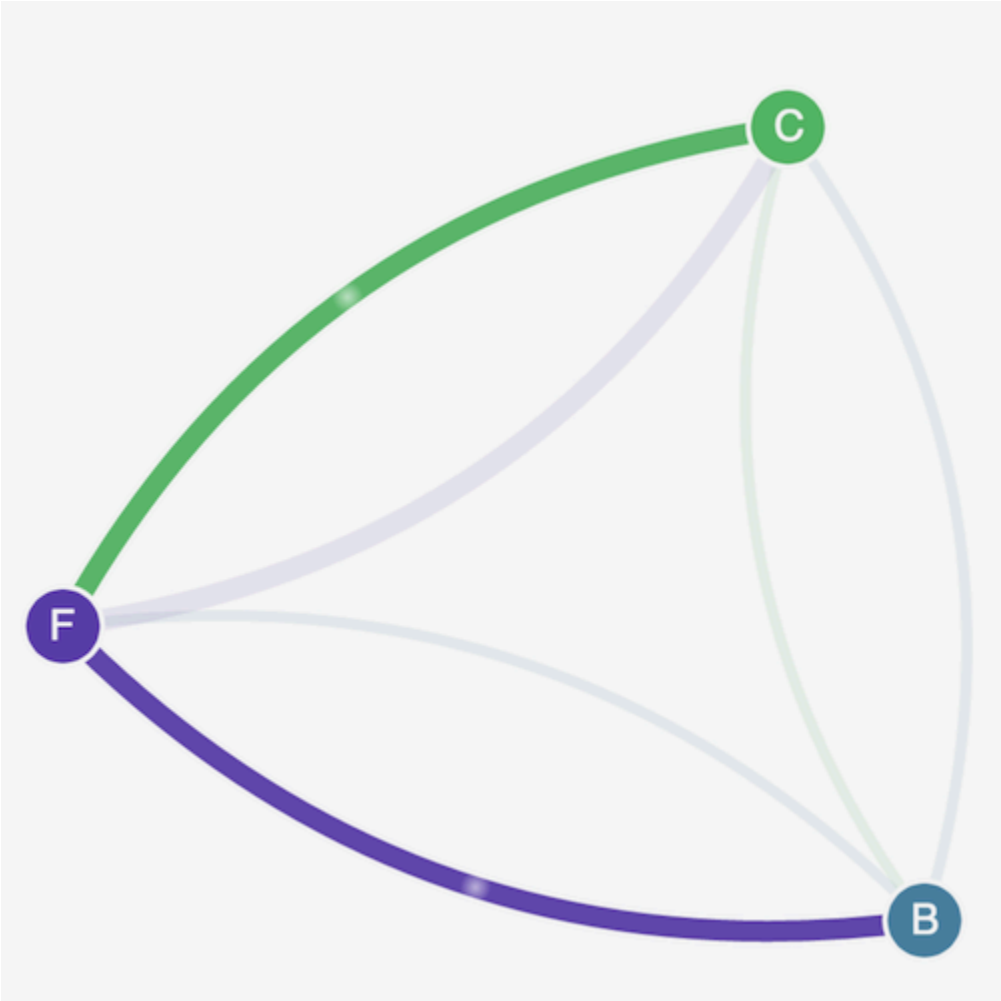
10. 重新整理您的瀏覽器。您可以看到前端服務可以與後端服務通訊。



11. 套用以下網路政策以允許流量從用戶端流向後端服務：

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
    matchLabels:
      role: frontend
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: client
      ports:
        - protocol: TCP
          port: 80
```

12. 重新整理您的瀏覽器。您可以看到用戶端可以與前端服務通訊。前端服務仍然可以與後端服務通訊。



13. (選用) 在完成示範後，您可以刪除其資源。

```
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
```

即使刪除資源之後，節點上仍可能有網路政策端點，這些端點可能會以非預期的方式干擾叢集中的聯網。移除這些規則的唯一確定方法，是重新啟動節點或終止所有節點並將其回收。若要終止所有節點，請將 Auto Scaling 群組所需的計數設定為 0，然後備份到所需的數字，或是只終止節點。

疑難排解網路政策

閱讀 網路政策日誌 並透過執行來自 eBPF SDK 的工具，藉此對使用網路政策的網路連線進行疑難排解與調查。

網路政策日誌

網路政策是否允許或拒絕連線會記錄在流程日誌中。每個節點上的網路政策日誌均包含具有網路政策的每個 Pod 之流程日誌。網路政策日誌會儲存於 `/var/log/aws-routed-eni/network-policy-agent.log`。以下範例來自於 `network-policy-agent.log` 檔案：

```
{"level":"info","timestamp":"2023-05-30T16:05:32.573Z","logger":"ebpf-
client","msg":"Flow Info: ","Src
IP":"192.168.87.155","Src Port":38971,"Dest IP":"64.6.160","Dest
Port":53,"Proto":"UDP","Verdict":"ACCEPT"}
```

網路原則記錄檔預設為停用狀態。如果要啟用網路原則記錄檔，請依照下列步驟執行：

> **Note**
>
> 網路原則記錄檔需要額外的 1 個 vCPU 供 VPC CNI `aws-node` 精靈集資訊清單中的`aws-network-policy-agent`容器使用。

Amazon EKS 附加元件

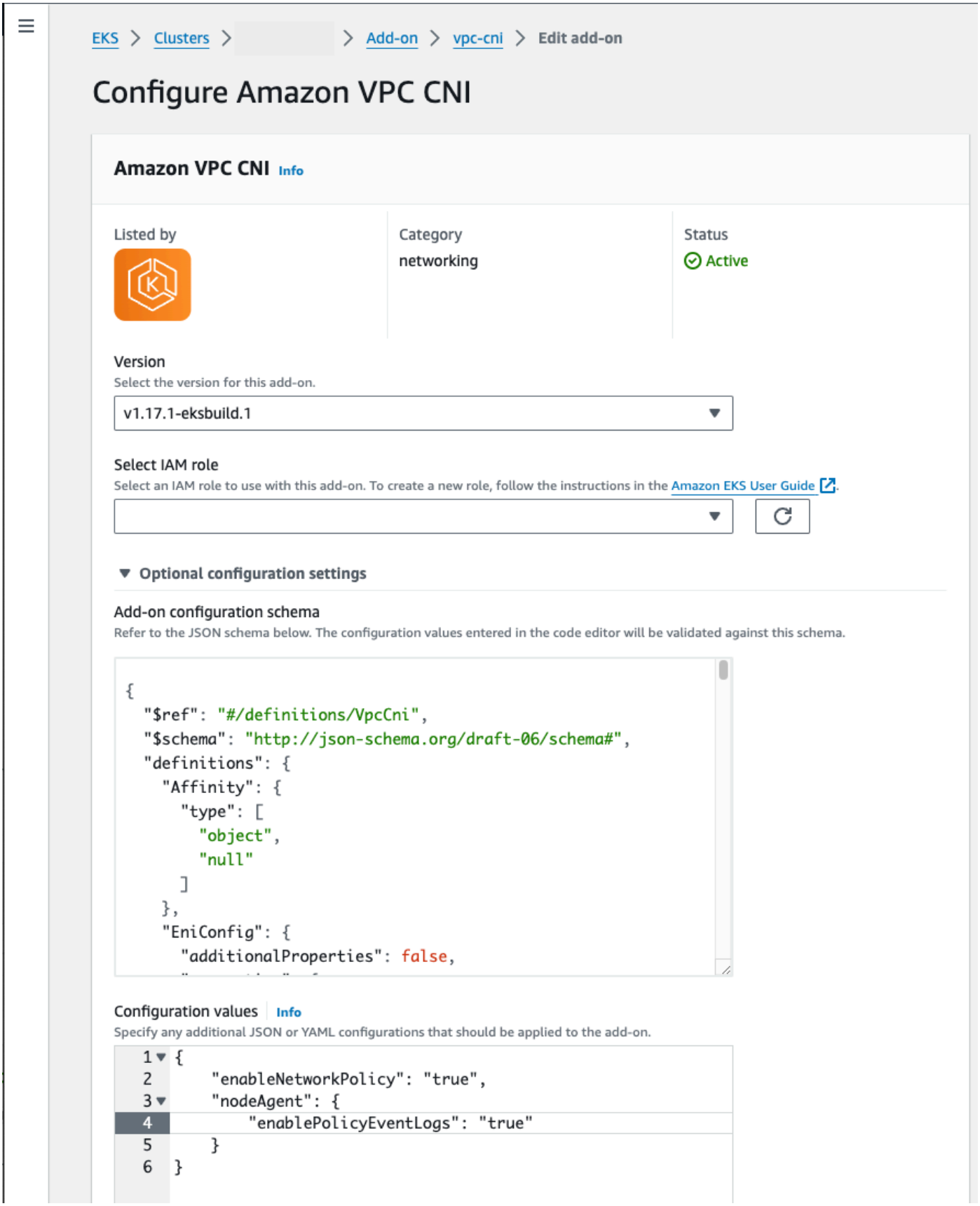
AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取叢集，然後選取您要為其設定 Amazon VPC CNI 附加元件的叢集名稱。
3. 選擇附加元件索引標籤。
4. 選取附加元件方塊右上方的方塊，然後選擇 Edit (編輯)。
5. 在 Configure ***name of addon*** (設定 name of addon) 頁面中：
   a. 在版本下拉式清單中，選取 `v1.14.0-eksbuild.3` 或更高版本。
   b. 展開選用組態設定。

c. 輸入最上層 JSON 金鑰 "nodeAgent":，且值為具有組態值中的金鑰 "enablePolicyEventLogs": 與 "true" 的值之物件。產生的文字必須是有效的 JSON 物件。下列範例顯示網路原則和網路原則記錄已啟用，以及網路原則記錄檔會傳送至 CloudWatch 記錄檔：

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true"
    }
}
```

下列螢幕擷取畫面展示了案例的範例。


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true"
5 }
6 }

AWS CLI

- 執行下列 AWS CLI 命令。將 `my-cluster` 取代為您的叢集名稱，並將 IAM 角色 ARN 取代為您正在使用的角色。

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true"}}'
```

自我管理附加元件

Helm

如果您已安裝Amazon VPC CNI plugin for Kubernetes透過`helm`，則可以更新組態以寫入網路原則記錄檔。

- 執行下列命令以啟用網路政策。

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true aws-vpc-cni --namespace
 kube-system eks/aws-vpc-cni
```

kubectl

如果您已安裝Amazon VPC CNI plugin for Kubernetes透過`kubectl`，則可以更新組態以寫入網路原則記錄檔。

1. 在您的編輯器中開啟 `aws-node DaemonSet`。

```
kubectl edit daemonset -n kube-system aws-node
```

2. 在 VPC CNI `aws-node` daemonset 清單檔案的 `aws-network-policy-agent` 容器，以 `true` 取代 `args:` 中命令引數 `--enable-policy-event-logs=false` 裡的 `false`。

```
    - args:
       - --enable-policy-event-logs=true
```

將網路政策日誌傳送到 Amazon CloudWatch 日誌

您可以使用 Amazon 日誌等服務監控網路政策 CloudWatch 日誌。您可以使用下列方法將網路原則記錄檔傳送至記 CloudWatch 錄檔。

若為 EKS 叢集，政策日誌會放置在 /aws/eks/*cluster-name*/cluster/ 下；若為自我管理的 K8S 群集，日誌會放置在 /aws/k8s-cluster/cluster/ 下。

透過 Amazon VPC CNI plugin for Kubernetes 傳送網路政策日誌

如果您啟用網路政策，系統會將第二個容器新增至節點代理程式的 aws-node Pod。此節點代理程式可以將網路原則記錄檔傳送至 CloudWatch 記錄檔。

> **Note**
>
> 網路政策日誌僅會由節點代理程式傳送。其他由 VPC CNI 製作的日誌不包含在內。

必要條件

- 將下列許可作為一節或個別政策新增至您用於 VPC CNI 的 IAM 角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

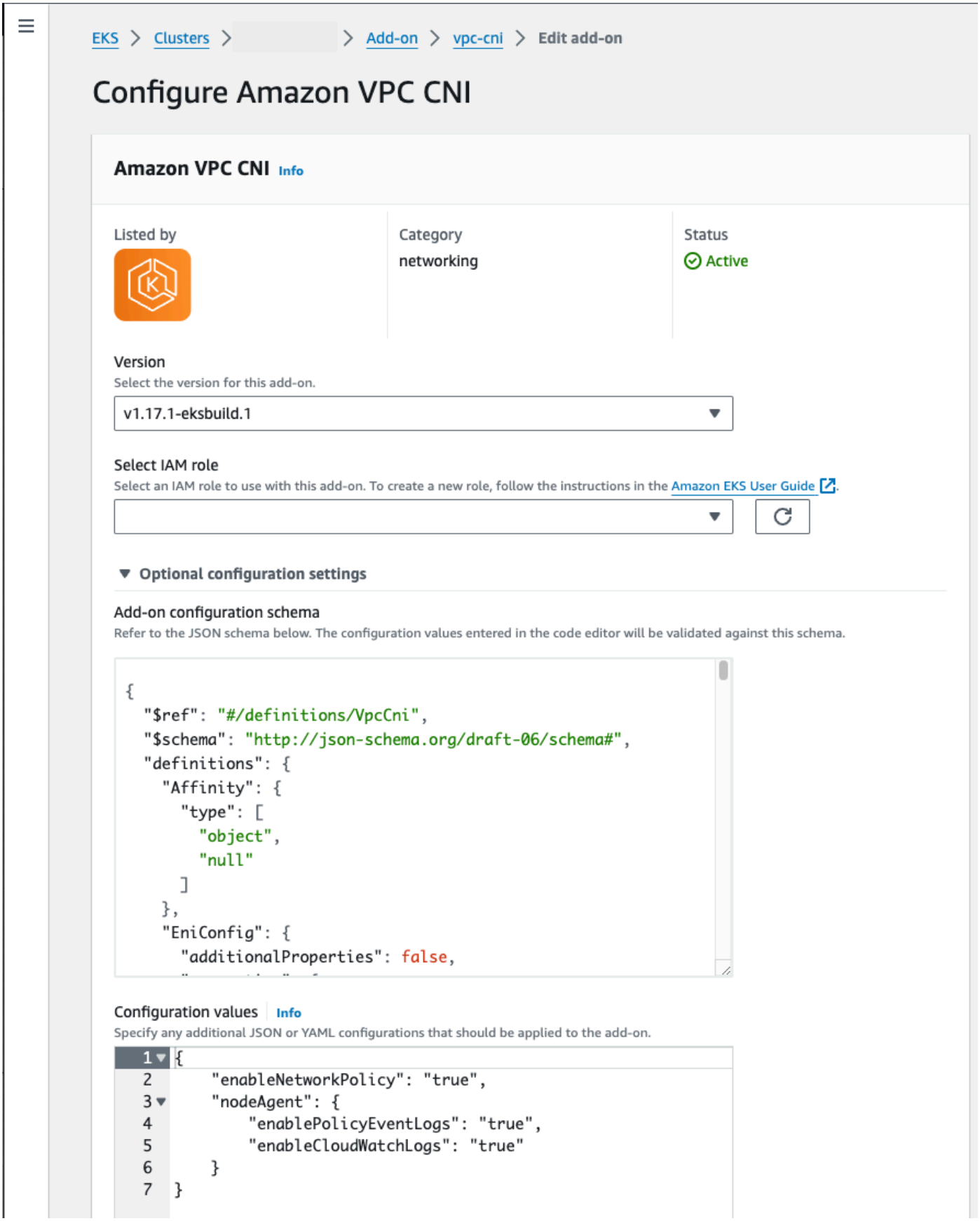
Amazon EKS 附加元件

AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取叢集，然後選取您要為其設定 Amazon VPC CNI 附加元件的叢集名稱。
3. 選擇附加元件索引標籤。
4. 選取附加元件方塊右上方的方塊，然後選擇 Edit (編輯)。
5. 在 Configure ***name of addon*** (設定 name of addon) 頁面中：
   a. 在版本下拉式清單中，選取 `v1.14.0-eksbuild.3` 或更高版本。
   b. 展開選用組態設定。
   c. 輸入最上層 JSON 金鑰 `"nodeAgent":`，且值為具有組態值中的金鑰 `"enableCloudWatchLogs":` 與 `"true"` 的值之物件。產生的文字必須是有效的 JSON 物件。下列範例顯示網路原則和網路原則記錄已啟用，以及記錄檔會傳送至 CloudWatch 記錄檔：

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true",
        "enableCloudWatchLogs": "true",
    }
}
```

下列螢幕擷取畫面展示了案例的範例。


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true",
5 "enableCloudWatchLogs": "true"
6 }
7 }

AWS CLI

- 執行下列 AWS CLI 命令。將 `my-cluster` 取代為您的叢集名稱，並將 IAM 角色 ARN 取代為您正在使用的角色。

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true", "enableCloudWatchLogs": "true"}}'
```

自我管理附加元件

Helm

如果您已安裝Amazon VPC CNI plugin for Kubernetes到`helm`，則可以更新配置以將網路策略記錄檔傳送至 CloudWatch 記錄檔。

- 執行下列命令以啟用網路原則記錄檔，並將其傳送至 CloudWatch 記錄檔。

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true --set
 nodeAgent.enableCloudWatchLogs=true aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. 在您的編輯器中開啟 `aws-node DaemonSet`。

```
kubectl edit daemonset -n kube-system aws-node
```

2. `true`在兩個命令引數中以`--enable-policy-event-logs=false`及 `--enable-cloudwatch-logs=false` VPC CNI `aws-node` 守護`args:`進程集資訊清單中的`aws-network-policy-agent`容器中取代為。`false`

```
    - args:
       - --enable-policy-event-logs=true
       - --enable-cloudwatch-logs=true
```

透過 Fluent Bit daemonset 傳送網路政策日誌

如果您正在 daemonset 使用 Fluent Bit 從節點傳送日誌，則可新增組態以便包括來自網路政策的網路政策日誌。您可以使用下列範例組態：

```
    [INPUT]
        Name              tail
        Tag               eksnp.*
        Path              /var/log/aws-routed-eni/network-policy-agent*.log
        Parser            json
        DB                /var/log/aws-routed-eni/flb_npagent.db
        Mem_Buf_Limit     5MB
        Skip_Long_Lines   On
        Refresh_Interval  10
```

已包含 eBPF SDK

Amazon VPC CNI plugin for Kubernetes 會在節點上安裝eBPF SDK 工具集。您可以使用 eBPF SDK 來找出網路政策的問題。例如，下列命令會列出目前在節點上執行的程式。

```
sudo /opt/cni/bin/aws-eks-na-cli ebpf progs
```

若要執行此命令，您可以使用任何方法連接到節點。

Kubernetes 網路政策

若要實作 Kubernetes 網路政策，您需要建立 Kubernetes `NetworkPolicy` 物件並將其部署到您的叢集。`NetworkPolicy` 物件的範圍是一個命名空間。您實施政策以允許或拒絕根據標籤選取工具、命名空間及 IP 地址範圍的 Pods 之間的流量。如需有關建立 `NetworkPolicy` 物件的詳細資訊，請參閱 Kubernetes 文件中的網路政策。

強制執行 Kubernetes `NetworkPolicy` 物件是使用 Extended Berkeley Packet Filter (eBPF) 來實作。相對於 `iptables` 型實作，它提供了更低的延遲和效能特性，包含降低 CPU 使用率與避免循序查詢。此外，eBPF 探查可讓您存取內容豐富的資料，協助除錯複雜的核心層級問題並改善可觀測性。Amazon EKS 支援 eBPF型匯出工具，此工具利用探查來記錄每個節點上的政策結果，並將資料匯出到外部日誌收集器以協助除錯。如需詳細資訊，請參閱 eBPF 文件。

Pod 的自訂聯網

預設情況下，當 Amazon VPC CNI plugin for Kubernetes 為您的 Amazon EC2 節點建立次要彈性網路介面 (網路介面) 時，會在與節點的主網路介面相同的子網中建立這些介面。它還會將與主網路介面關

聯的相同安全群組關聯至次要網路介面。由於以下一或多個原因，您可能希望外掛程式在不同子網中建立次要網路介面，或者希望將不同的安全群組與次要網路介面關聯 (或者兩者兼具)：

- 主網路介面所在的子網中可用的 IPv4 地址數量有限。這可能會限制您可以在子網中建立的 Pods 數量。如果為次要網路介面使用不同的子網，您可以增加可用於 Pods 的 IPv4 地址的數量。
- 基於安全考量，您的 Pods 可能需使用節點之主要網路介面以外的其他子網或安全群組。
- 將節點設定在公有子網，並將 Pods 放置在私有子網。與公有子網相關聯的路由表包括網際網路閘道的路由。與私有子網相關聯的路由表不會包含網際網路閘道的路由。

考量事項

- 啟用自訂聯網後，不會將指派給主網路介面的 IP 地址指派給 Pods。僅有來自次要網路介面的 IP 地址會指派給 Pods。
- 如果您的叢集使用 IPv6 系列，您無法使用自訂聯網。
- 如果您計劃僅使用自訂聯網來協助緩解 IPv4 地址耗盡的情況，則可以改為使用 IPv6 系列來建立叢集。如需詳細資訊，請參閱 IPv6叢集的位址Pods、和 services。
- 即使部署到為次要網路介面指定的子網的 Pods 可以使用與節點的主網路介面不同的子網和安全群組，但子網和安全群組仍須與節點位於同一 VPC 中。

必要條件

- 熟悉 Amazon VPC CNI plugin for Kubernetes 如何建立輔助網路介面並將 IP 地址指派給 Pods。如需詳細資訊，請參閱 GitHub 上的 ENI 分配。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。

- 建議您在 Bash shell 中完成本主題中的步驟。如果您不使用 Bash shell，則某些指令碼命令 (如行接續字元以及設定和使用變數的方式) 需要針對 shell 進行調整。此外，您的 Shell 的引用及轉義規則可能會有所不同。若要取得更多資訊，請參閱《使用指南》中的〈〉 AWS CLI中的〈使用引號搭配字串〉。 AWS Command Line Interface

在本教學課程中，我們建議您使用 *example values*，除非其註明要替換。完成生產叢集的步驟時，您可以替換任何 *example value*。我們建議您在同一終端中完成所有步驟。這是因為變數是在整個步驟中設定和使用，並且不會存在於不同的終端中。

本主題中的指令會使用使用範例中列出的慣AWS CLI 例來格式化。如果您從命令列針對與您正在使用的設定 AWS CLI 檔中 AWS 區域 定義的預設值 AWS 區域 不同的資源執行命令，則您需要新增**--region *region-code***至命令。

如果要將自訂聯網部署到生產叢集，請跳至 步驟 2：設定 VPC。

步驟 1：建立測試 VPC 和叢集

建立叢集

下列程序可幫助您建立測試 VPC 和叢集，並為該叢集設定自訂聯網。我們不建議對生產工作負載使用測試叢集，因為本主題未涵蓋您可能在生產叢集上使用的幾個不相關功能。如需詳細資訊，請參閱 建立 Amazon EKS 叢集。

1. 定義幾個變數以在剩餘步驟中使用。

```
export cluster_name=my-custom-networking-cluster
account_id=$(aws sts get-caller-identity --query Account --output text)
```

2. 建立 VPC。

   1. 使用 Amazon EKS AWS CloudFormation 範本建立 VPC。

```
aws cloudformation create-stack --stack-name my-eks-custom-networking-vpc \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --parameters ParameterKey=VpcBlock,ParameterValue=192.168.0.0/24 \
  ParameterKey=PrivateSubnet01Block,ParameterValue=192.168.0.64/27 \
  ParameterKey=PrivateSubnet02Block,ParameterValue=192.168.0.96/27 \
  ParameterKey=PublicSubnet01Block,ParameterValue=192.168.0.0/27 \
  ParameterKey=PublicSubnet02Block,ParameterValue=192.168.0.32/27
```

AWS CloudFormation 堆疊需要幾分鐘的時間來建立。若要檢查堆疊的部署狀態，請執行下列命令。

```
aws cloudformation describe-stacks --stack-name my-eks-custom-networking-vpc --
query Stacks\[\].StackStatus  --output text
```

在命令輸出為 CREATE_COMPLETE 之前，請勿繼續下一步。

2. 使用以範本建立的私有子網 ID 的值來定義變數。

```
subnet_id_1=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet01'].PhysicalResourceId" --output text)
subnet_id_2=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet02'].PhysicalResourceId" --output text)
```

3. 使用上一步中擷取的子網的可用區域來定義變數。

```
az_1=$(aws ec2 describe-subnets --subnet-ids $subnet_id_1 --query
 'Subnets[*].AvailabilityZone' --output text)
az_2=$(aws ec2 describe-subnets --subnet-ids $subnet_id_2 --query
 'Subnets[*].AvailabilityZone' --output text)
```

3. 建立叢集 IAM 角色。

a. 執行下列命令以建立 IAM 信任政策 JSON 檔案。

```
cat >eks-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
```

```
}
EOF
```

b. 建立 Amazon EKS 叢集 IAM 角色。如有必要，請在 `eks-cluster-role-trust-policy.json` 前面加上您在上一步中寫入檔案的電腦的路徑。命令會將您在上一步驟中建立的信任策略與角色相關聯。若要建立 IAM 角色，必須為建立角色的 IAM 主體指派以下 `iam:CreateRole` 動作 (許可)。

```
aws iam create-role --role-name myCustomNetworkingAmazonEKSClusterRole --
assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
```

c. 將名為 `AmazonEKSClusterPolicy` 的 Amazon EKS 受管 IAM 政策連接到角色。若要將 IAM 政策連接至 IAM 主體，必須為連接政策的 IAM 實體指派以下 IAM 動作之一 (許可)：`iam:AttachUserPolicy` 或 `iam:AttachRolePolicy`。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name myCustomNetworkingAmazonEKSClusterRole
```

4. 建立 Amazon EKS 叢集，並設定您的裝置以便與其進行通訊。

a. 建立叢集。

```
aws eks create-cluster --name my-custom-networking-cluster \
   --role-arn arn:aws:iam::$account_id:role/
myCustomNetworkingAmazonEKSClusterRole \
   --resources-vpc-config subnetIds=$subnet_id_1","$subnet_id_2
```

> **Note**
>
> 您可能會收到錯誤，表示在請求中的其中一個可用區域沒有足夠的容量可建立 Amazon EKS 叢集。如果發生這種情況，錯誤輸出包含的可用區域可支援新的叢集。使用至少兩個位於帳戶的支援可用區域子網路來建立您的叢集。如需詳細資訊，請參閱 容量不足。

b. 建立叢集需要幾分鐘才能完成。若要檢查叢集的部署狀態，請執行下列命令。

```
aws eks describe-cluster --name my-custom-networking-cluster --query
 cluster.status
```

在命令輸出為 "ACTIVE" 之前，請勿繼續下一步。

c. 設定 kubectl 以便與叢集通訊。

```
aws eks update-kubeconfig --name my-custom-networking-cluster
```

步驟 2：設定 VPC

本教學課程需要在 步驟 1：建立測試 VPC 和叢集 中建立的 VPC。對於生產叢集，透過將所有 *example values* 取代為您自己的值，為您的 VPC 相應地調整步驟。

1. 確認目前安裝的 Amazon VPC CNI plugin for Kubernetes 是最新版本。若要確定 Amazon EKS 附加元件類型的最新版本並更新您的版本，請參閱 更新附加元件。若要確定自我管理附加元件類型的最新版本並更新您的版本，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。
2. 擷取叢集 VPC 的 ID，並將其存放在變數中以供稍後步驟使用。對於生產叢集，請將 *my-custom-networking-cluster* 取代為您的叢集名稱。

```
vpc_id=$(aws eks describe-cluster --name my-custom-networking-cluster --query
 "cluster.resourcesVpcConfig.vpcId" --output text)
```

3. 將額外的無類別域間路由 (CIDR) 區塊與叢集的 VPC 相關聯。CIDR 區塊不能與任何現有關聯的 CIDR 區塊重疊。

   1. 檢視目前與您的 VPC 關聯的 CIDR 區塊。

```
aws ec2 describe-vpcs --vpc-ids $vpc_id \
    --query 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

   範例輸出如下。

```
----------------------------------
|          DescribeVpcs          |
+-----------------+--------------+
|    CIDRBlock    |    State     |
+-----------------+--------------+
|  192.168.0.0/24 |  associated  |
```

```
+-----------------+-------------+
```

2. 將其他 CIDR 區塊與 VPC 建立關聯。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的將 IPv4 CIDR 區塊與 VPC 建立關聯。

```
aws ec2 associate-vpc-cidr-block --vpc-id $vpc_id --cidr-block 192.168.1.0/24
```

3. 確認新區塊已關聯。

```
aws ec2 describe-vpcs --vpc-ids $vpc_id --query
 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

範例輸出如下。

```
---------------------------------
|          DescribeVpcs          |
+-----------------+--------------+
|    CIDRBlock    |    State     |
+-----------------+--------------+
|  192.168.0.0/24 |  associated  |
|  192.168.1.0/24 |  associated  |
+-----------------+--------------+
```

在新 CIDR 區塊 State 為 associated 之前，請勿進行下一個步驟。

4. 在現有子網所在的每個可用區域中建立任意數量的子網。指定在上一步中與 VPC 關聯的 CIDR 區塊內的 CIDR 區塊。

1. 建立新子網。子網必須在不同於現有子網所在的 VPC CIDR 區塊中建立，但與現有子網位於相同的可用區域中。在此範例中，會在目前私有子網所在的每個可用區域的新 CIDR 區塊中，建立一個子網。建立的子網 ID 會儲存在變數中，以便在稍後步驟中使用。Name 值與指派給在上一步中使用 Amazon EKS VPC 範本建立的子網的值相符。名稱並非必要資訊。您可以使用不同的名稱。

```
new_subnet_id_1=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_1 --cidr-block 192.168.1.0/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet01},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
```

```
    --query Subnet.SubnetId --output text)
new_subnet_id_2=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_2 --cidr-block 192.168.1.32/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet02},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
```

> **Important**
>
> 預設情況下，您的新子網與 VPC 的主路由表具有隱式關聯。此路由表允許在 VPC 中部署的所有資源之間進行通訊。但是，不允許與其 IP 地址在與 VPC 關聯的 CIDR 區塊之外的資源進行通訊。您可以將自己的路由表與子網相關聯以變更此行為。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的子網由表。

2. 檢視 VPC 中的目前子網。

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" \
    --query 'Subnets[*].{SubnetId: SubnetId,AvailabilityZone:
 AvailabilityZone,CidrBlock: CidrBlock}' \
    --output table
```

範例輸出如下。

```
-------------------------------------------------------------------
|                          DescribeSubnets                        |
+------------------+--------------------+--------------------------+
| AvailabilityZone |     CidrBlock      |         SubnetId         |
+------------------+--------------------+--------------------------+
|  us-west-2d      |  192.168.0.0/27    |     subnet-example1      |
|  us-west-2a      |  192.168.0.32/27   |     subnet-example2      |
|  us-west-2a      |  192.168.0.64/27   |     subnet-example3      |
|  us-west-2d      |  192.168.0.96/27   |     subnet-example4      |
|  us-west-2a      |  192.168.1.0/27    |     subnet-example5      |
|  us-west-2d      |  192.168.1.32/27   |     subnet-example6      |
+------------------+--------------------+--------------------------+
```

您可以看到您建立的 192.168.1.0 CIDR 區塊中的子網，與 192.168.0.0 CIDR 區塊中的子網位於相同的可用區域中。

步驟 3：設定 Kubernetes 資源

若要設定 Kubernetes 資源

1. 在 aws-node DaemonSet 中，將 AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG 環境變數設定為 true。

```
kubectl set env daemonset aws-node -n kube-system
 AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG=true
```

2. 擷取叢集安全群組的 ID，並將其存放在變數中以供下一步驟使用。在您建立叢集時，Amazon EKS 會自動建立此安全群組。

```
cluster_security_group_id=$(aws eks describe-cluster --name $cluster_name --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId --output text)
```

3. 為每個您要為其部屬 Pods 的子網建立 ENIConfig 自訂資源。

   a. 為每個網路介面組態建立唯一的檔案。

   以下命令為在上一步中建立的兩個子網建立單獨的 ENIConfig 檔案。name 的值必須是唯一的值。此名稱與子網所在的可用區域相同。叢集安全群組指派給 ENIConfig。

```
cat >$az_1.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_1
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_1
EOF
```

```
cat >$az_2.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_2
spec:
  securityGroups:
```

```
    - $cluster_security_group_id
  subnet: $new_subnet_id_2
EOF
```

對於生產叢集，您可以對前述命令進行以下變更：

- 將 *$cluster_security_group_id* 取代為您想要用於每個 ENIConfig 的現有安全群組的 ID。
- 我們建議盡可能將您的 ENIConfigs 命名為與您將使用 ENIConfig 的可用區域相同的名稱。由於各種原因，您可能需要為您的 ENIConfigs 使用與可用區域名稱不同的名稱。例如，如果您在同一可用區域中有兩個以上的子網，並且希望將其與自訂聯網一起使用，則需要多個 ENIConfigs 用於相同的可用區域。由於每個 ENIConfig 都需要一個唯一的名稱，因此您不能使用可用區域名稱命名多個 ENIConfigs。

  如果您的 ENIConfig 名稱與可用區域名稱並不完全相同，則將 *$az_1* 和 *$az_2* 取代為在前面命令中您自己的名稱，並使用在本教學課程後的 ENIConfig 註釋您的節點。

> **Note**
>
> 如果您未指定用於生產叢集的有效安全群組，並且您正在使用：
>
> - 1.8.0 版或更高版本的 Amazon VPC CNI plugin for Kubernetes，則會使用與節點的主要彈性網路介面關聯的安全群組。
> - Amazon VPC CNI plugin for Kubernetes 版本低於 1.8.0，則 VPC 的預設安全群組將指派給次要網路介面。

> **Important**
>
> - AWS_VPC_K8S_CNI_EXTERNALSNAT=false 是適用於 Kubernetes 的 Amazon VPC CNI 外掛程式組態中的預設設定。如果您使用預設設定，則送往不在與 VPC 關聯的其中一個 CIDR 區塊內之 IP 地址的流量，將使用節點主網路介面的安全群組和子網。在您的 ENIConfigs 中定義、用於建立次要網路介面的子網和安全群組不用於此流量。如需有關此設定的詳細資訊，請參閱 適用於 Pods 的 SNAT。

- 如果您還為 Pods 使用安全群組，則會使用 SecurityGroupPolicy 中指定的安全群組，而不是 ENIConfigs 中指定的安全群組。如需詳細資訊，請參閱 Pods 的安全群組。

b. 使用下列命令，將您建立的每個自訂資源檔案套用到叢集。

```
kubectl apply -f $az_1.yaml
kubectl apply -f $az_2.yaml
```

4. 確認您的 ENIConfigs 已建立。

```
kubectl get ENIConfigs
```

範例輸出如下。

```
NAME         AGE
us-west-2a   117s
us-west-2d   105s
```

5. 如果要在生產叢集上啟用自訂聯網，並將您的 ENIConfigs 命名為所用可用區域以外的名稱，則請跳到下一步以部署 Amazon EC2 節點。

啟用 Kubernetes 以將可用區域的 ENIConfig 自動應用至叢集中建立的任何新 Amazon EC2 節點。

1. 對於本教學課程中的測試叢集，請跳至下一步。

對於生產叢集，請檢查 aws-node DaemonSet 的容器規格中是否存在含有 ENI_CONFIG_ANNOTATION_DEF 環境變數的金鑰 k8s.amazonaws.com/eniConfig 的 annotation。

```
kubectl describe daemonset aws-node -n kube-system | grep
 ENI_CONFIG_ANNOTATION_DEF
```

如果傳回輸出，則表示註釋已存在。如果未傳回輸出，表示變數尚未設定。對於生產叢集，您可以使用此設定或以下步驟中的設定。如果使用此設定，將會覆寫以下步驟中的設定。在本教學課程中，將使用下一步中的設定。

2. 更新您的 aws-node DaemonSet 以將可用區域的 ENIConfig 自動應用至叢集中建立的任何新 Amazon EC2 節點。

```
kubectl set env daemonset aws-node -n kube-system
 ENI_CONFIG_LABEL_DEF=topology.kubernetes.io/zone
```

步驟 4：部署 Amazon EC2 節點

部署 Amazon EC2 節點

1. 建立節點 IAM 角色。

   a. 執行下列命令以建立 IAM 信任政策 JSON 檔案。

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

   b. 執行下列命令，以設定角色名稱的變數。您可以使用您選擇的任何名稱取代 *myCustomNetworkingAmazonEKSNodeRole*。

```
export node_role_name=myCustomNetworkingAmazonEKSNodeRole
```

   c. 建立 IAM 角色，並將其傳回的 Amazon Resource Name (ARN) 儲存在變數中，以便在稍後步驟中使用。

```
node_role_arn=$(aws iam create-role --role-name $node_role_name --assume-role-
policy-document file://"node-role-trust-relationship.json" \
    --query Role.Arn --output text)
```

   d. 將三個所需的 IAM 受管政策連接到 IAM 角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
    --role-name $node_role_name
```

> **Important**
>
> 在本教學課程中，為簡單起見，將 AmazonEKS_CNI_Policy 政策連接至節點 IAM 角色。但是，在生產叢集中，我們建議將政策連接至僅與 Amazon VPC CNI plugin for Kubernetes 一起使用的單獨 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

2. 建立下列其中一種節點群組類型。若要確定您想要部署的執行個體類型，請參閱 選擇 Amazon EC2 執行個體類型。在本教學課程中，請完成受管、沒有啟動範本或未指定 AMI ID 的啟動範本選項。如果要將節點群組用於生產工作負載，我們建議您熟悉所有受管和自我管理節點群組選項，然後再部署節點群組。

   - 受管：使用下列其中一個選項，部署節點群組：
     - 沒有啟動範本，或有啟動範本，但沒有指定 AMI ID：執行下列命令。在本教學課程中，使用 *example values*。對於生產節點群組，請將所有 *example values* 取代為您自己的值。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。

       ```
       aws eks create-nodegroup --cluster-name $cluster_name --nodegroup-name my-
       nodegroup \
           --subnets $subnet_id_1 $subnet_id_2 --instance-types t3.medium --node-role
        $node_role_arn
       ```

     - 使用具有指定 AMI ID 的啟動範本

       1. 確定 Amazon EKS 為您的節點推薦的 Pods 數目上限。請遵循 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限 中的指示，將 **--cni-custom-networking-enabled** 新增至該主題的步驟 3。請記下輸出，以便在下一個步驟中使用。

2. 在啟動範本中，指定 Amazon EKS 最佳化 AMI ID 或基於 Amazon EKS 最佳化 AMI 的自訂 AMI，然後使用啟動範本部署節點群組，並在啟動範本中提供下列使用者資料。此使用者資料會將引數傳遞至 `bootstrap.sh` 檔案。如需引導檔案的詳細資訊，請參閱 GitHub 上的 bootstrap.sh。您可將 *20* 取代為上一個步驟的值 (建議) 或您自己的值。

```
/etc/eks/bootstrap.sh my-cluster --use-max-pods false --kubelet-extra-args
 '--max-pods=20'
```

如果您建立的自訂 AMI 並非基於 Amazon EKS 最佳化 AMI，則需要自行自訂建立組態。

- 自我管理

1. 確定 Amazon EKS 為您的節點推薦的 Pods 數目上限。請遵循 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限 中的指示，將 **--cni-custom-networking-enabled** 新增至該主題的步驟 3。請記下輸出，以便在下一個步驟中使用。
2. 使用 啟動自我管理的 Amazon Linux 節點 中的指示部署節點群組。指定BootstrapArguments參數的下列文字。您可將 *20* 取代為上一個步驟的值 (建議) 或您自己的值。

```
--use-max-pods false --kubelet-extra-args '--max-pods=20'
```

> **Note**
>
> 如果您希望生產叢集中的節點支援更多數量的 Pods，請再次在 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限 中執行指令碼。此外，將 **--cni-prefix-delegation-enabled** 選項新增制命令中。例如：`m5.large` 執行個體類型會傳回 *110*。如需有關如何啟用此功能的說明，請參閱 增加 Amazon EC2 節點的可用 IP 地址數量。您可將此功能與自訂聯網搭配使用。

建立節點群組需要幾分鐘的時間。您可以使用以下命令檢查受管節點群組的建立狀態。

```
aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-
nodegroup --query nodegroup.status --output text
```

在傳回的輸出為 `ACTIVE` 之前，請勿繼續進行下一個步驟。

3. 針對本教學課程，您可以略過此步驟。

對於生產叢集，如果您未將 ENIConfigs 命名為與您使用它們的可用區域相同的名稱，則必須使用應與節點一起使用的 ENIConfig 名稱來註釋您的節點。如果您在每個可用區域中只有一個子網，且將 ENIConfigs 命名為與可用區域相同的名稱，則無需執行此步驟。這是因為當您在上一步中啟用時，Amazon VPC CNI plugin for Kubernetes 會自動為您將正確的 ENIConfig 與節點關聯起來。

a. 取得您叢集中的節點清單。

```
kubectl get nodes
```

範例輸出如下。

```
NAME                                             STATUS   ROLES    AGE     VERSION
ip-192-168-0-126.us-west-2.compute.internal   Ready    <none>   8m49s
 v1.22.9-eks-810597c
ip-192-168-0-92.us-west-2.compute.internal    Ready    <none>   8m34s
 v1.22.9-eks-810597c
```

b. 確定每個節點所在的可用區域。針對先前步驟傳回的每個節點執行下列命令。

```
aws ec2 describe-instances --filters Name=network-interface.private-dns-
name,Values=ip-192-168-0-126.us-west-2.compute.internal \
--query 'Reservations[].Instances[].{AvailabilityZone:
 Placement.AvailabilityZone, SubnetId: SubnetId}'
```

範例輸出如下。

```
[
    {
        "AvailabilityZone": "us-west-2d",
        "SubnetId": "subnet-Example5"
    }
]
```

c. 使用您為子網 ID 和可用區域建立的 ENIConfig 註釋每個節點。您只能使用一個 ENIConfig 註釋一個節點，但可以使用相同的 ENIConfig 註釋多個節點。使用自己的取代 *example values*。

```
kubectl annotate node ip-192-168-0-126.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName1
kubectl annotate node ip-192-168-0-92.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName2
```

4. 如果在切換到使用自訂聯網功能之前，生產叢集中的節點正在執行 Pods，請先完成以下任務：

   a. 確保您具有使用自訂聯網功能的可用節點。

   b. 封鎖並排空節點以正常關閉 Pods。如需詳細資訊，請參閱 Kubernetes 文件中的安全地耗盡節點。

   c. 終止節點。如果節點位於現有的受管節點群組中，則可以刪除該節點群組。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令：

      - 將 *my-cluster* 取代為您的叢集名稱。
      - 將 *my-nodegroup* 取代為您的節點群組名稱。

```
aws eks delete-nodegroup --cluster-name my-cluster --nodegroup-name my-
nodegroup
```

   只有使用 k8s.amazonaws.com/eniConfig 標籤註冊的新節點能使用自訂聯網功能。

5. 確認 Pods 從 CIDR 區塊 (與您在上一步中建立的子網之一相關聯) 指派 IP 地址。

```
kubectl get pods -A -o wide
```

   範例輸出如下。

```
NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE     IP
           NODE                                            NOMINATED NODE   READINESS
 GATES
kube-system   aws-node-2rkn4             1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   aws-node-k96wp             1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
```

```
kube-system   coredns-657694c6f4-smcgr   1/1     Running   0          56m
 192.168.1.23    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   coredns-657694c6f4-stwv9   1/1     Running   0          56m
 192.168.1.28    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-jgshq           1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-wx9vk           1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
```

您可以看到 coredns Pods 被指派了 IP 地址來自您新增至 VPC 的 192.168.1.0 CIDR 區塊的 IP 地址。如果沒有自訂網路，他們將被指派來自 192.168.0.0 CIDR 區塊的地址，因為其是最初與 VPC 關聯的唯一 CIDR 區塊。

如果 Pod's spec 包含 hostNetwork=true，則會為其指派節點的主 IP 地址。它不會從您新增的子網中指派地址。依預設，此值是設為 false。針對在您的叢集上執行的 kube-proxy 和 Amazon VPC CNI plugin for Kubernetes (aws-node) Pods，將此值設定為 true。這就是為什麼在前面的輸出中沒有為 kube-proxy 和外掛程式的 aws-node Pods 指派 192.168.1.*x* 地址的原因。如需有關Pod'shostNetwork設定的詳細資訊，請參閱 Kubernetes API 參考資料中的 PodSpec v1 核心。

步驟 5：刪除教學課程資源

完成本教學課程後，建議您刪除您建立的資源。然後，您可以調整步驟以啟用生產叢集的自訂聯網。

刪除教學課程資源

1. 如果您建立的節點群組僅用於測試，請將其刪除。

```
aws eks delete-nodegroup --cluster-name $cluster_name --nodegroup-name my-nodegroup
```

即使 AWS CLI 輸出顯示叢集已刪除，刪除程序實際上可能並未完成。刪除程序需要幾分鐘。執行下列命令來確認已完成。

```
aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-
nodegroup --query nodegroup.status --output text
```

在傳回與下列輸出類似的輸出之前，請勿繼續執行。

```
An error occurred (ResourceNotFoundException) when calling the DescribeNodegroup
 operation: No node group found for name: my-nodegroup.
```

2. 如果您建立的節點群組僅用於測試，則請刪除節點 IAM 角色。

   a. 將策略與角色分離。

   ```
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
   policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
   policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
   policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
   ```

   b. 刪除角色。

   ```
   aws iam delete-role --role-name myCustomNetworkingAmazonEKSNodeRole
   ```

3. 刪除叢集。

   ```
   aws eks delete-cluster --name $cluster_name
   ```

   請使用下列命令來確認刪除叢集。

   ```
   aws eks describe-cluster --name $cluster_name --query cluster.status --output text
   ```

   如果傳回類似以下的輸出，則已成功刪除叢集。

   ```
   An error occurred (ResourceNotFoundException) when calling the DescribeCluster
    operation: No cluster found for name: my-cluster.
   ```

4. 刪除叢集 IAM 角色。

   a. 將策略與角色分離。

   ```
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSClusterRole
    --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
   ```

b. 刪除角色。

```
aws iam delete-role --role-name myCustomNetworkingAmazonEKSClusterRole
```

5. 刪除您在上一個步驟中建立的子網。

```
aws ec2 delete-subnet --subnet-id $new_subnet_id_1
aws ec2 delete-subnet --subnet-id $new_subnet_id_2
```

6. 刪除您建立的 VPC。

```
aws cloudformation delete-stack --stack-name my-eks-custom-networking-vpc
```

增加 Amazon EC2 節點的可用 IP 地址數量

每個 Amazon EC2 執行個體都支援最大數量的彈性網路介面，以及可指派給每個網路介面的最大數量的 IP 地址。每個節點的每個網路介面都需要一個 IP 地址。所有其他可用的 IP 地址都可以指派給 Pods。每個 Pod 都需要專屬的 IP 地址。因此，您的節點可能具有可用的運算和記憶體資源，但無法容納其他 Pods，因為節點已耗盡要指派給 Pods 的 IP 地址。

本主題旨在介紹如何透過指派 IP 字首 (而不是將個別的次要 IP 地址指派給節點)，大幅增加節點可指派給 Pods 的 IP 地址數量。每個字首都包含多個 IP 地址。如果您未針對 IP 字首指派設定叢集，則叢集必須進行更多 Amazon EC2 應用程式介面 (API) 呼叫，才能設定 Pod 連線所需的網路介面和 IP 地址。隨著叢集規模不斷擴大，這些 API 呼叫的頻率可能會導致 Pod 和執行個體的啟動時間變得更長。這樣一來，會導致擴展延遲以滿足大型和高峰工作負載的需求，並增加成本和管理負荷，因為您需要佈建額外的叢集和 VPC 以符合擴展需求。如需詳細資訊，請參閱上的Kubernetes延展性閾值 GitHub。

考量事項

- 每種 Amazon EC2 執行個體類型都支援最大數量的 Pods。如果受管節點群組包含多個執行個體類型，叢集中執行個體的最小 Pods 數量上限會套用至叢集中的所有節點。
- 依預設，您最多可在節點上執行 110 個 Pods，但您可以變更該數量。如果您變更該數量且擁有現有受管節點群組，則節點群組的下一個 AMI 或啟動範本更新會導致新的工作節點出現變更後的值。
- 從指派 IP 地址轉換為指派 IP 字首時，建議您建立新的節點群組來增加可用 IP 地址的數量，而不是對現有節點執行輪流取代。在同時指派 IP 地址和字首的節點上執行 Pods，可能會導致公告的 IP 地址容量不一致，進而影響節點日後的工作負載。如需執行轉換的建議方式，請參閱《Amazon EKS 最佳實務指南》中的在次要 IP 模式與字首委派模式相互遷移期間取代所有節點。
- 僅適用於具有 Linux 節點的叢集。

- 設定附加元件以將字首指派給網路介面後，如果不移除叢集中所有節點群組中的所有節點，您就無法將 Amazon VPC CNI plugin for Kubernetes 附加元件降級為低於 `1.9.0` (或 `1.10.1`) 的版本。
- 如果您也使用 Pods 的安全群組，且 `POD_SECURITY_GROUP_ENFORCING_MODE = standard`、`AWS_VPC_K8S_CNI_EXTERNALSNAT = false`，則當 Pods 與 VPC 外部端點通訊時，將使用節點的安全群組，而不會使用您指派給 Pods 的任何安全群組。

  如果您也使用 Pods 的安全群組，且 `POD_SECURITY_GROUP_ENFORCING_MODE = strict`，則當 Pods 與 VPC 外部端點通訊時，將使用 `Pod's` 安全群組。

必要條件

- 現有的叢集。若要部署叢集，請參閱 建立 Amazon EKS 叢集。
- Amazon EKS 節點所在的子網路必須具有足夠的連續 `/28` (針對 `IPv4` 叢集) 或 `/80` (針對 `IPv6` 叢集) 無類別域間路由 (CIDR) 區塊。`IPv6` 叢集中只能包含 Linux 節點。如果 IP 地址分散在子網路 CIDR 中，則使用 IP 字首可能會失敗。建議您進行下列動作：
  - 使用子網路 CIDR 保留，如此一來，即使保留範圍內的任何 IP 地址仍在使用中，發佈後也不會重新指派 IP 地址。此舉可確保字首無需分割，即可用於配置。
  - 使用專門用於執行指派 IP 字首之工作負載的新子網路。指派 IP 字首時，Windows 和 Linux 工作負載可在同一子網路中同時執行。
- 若要將 IP 前置詞指派給節點，您的節點必須是以 AWS Nitro 為基礎。非 Nitro 型執行個體會繼續分配個別的次要 IP 地址，但要指派給 Pods 的 IP 地址數量遠低於 Nitro-based 執行個體。
- 僅適用於具有 Linux 節點的叢集：如果您的叢集針對 `IPv4` 系列設定，則必須安裝 Amazon VPC CNI plugin for Kubernetes 附加元件的版本 `1.9.0` 或更新版本。您可以使用下列命令來檢查目前版本：

  ```
  kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/" -f 2
  ```

  如果您的叢集針對 `IPv6` 系列設定，則必須安裝附加元件的版本 `1.10.1`。如果您的外掛程式版本早於所需版本，則必須對其進行更新。如需詳細資訊，請參閱更新的 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件 區段。
- 僅適用於具有 Windows 節點的叢集
  - 您的叢集及其平台版本必須與下表中的版本相同或為更新版本。若要升級叢集版本，請參閱 更新 Amazon EKS 叢集 Kubernetes 版本。如果叢集並非最低平台版本，則在 Amazon EKS 更新平台版本之前，您無法將 IP 字首指派給節點。

Kubernetes 版本	平台版本
1.27	eks.3
1.26	eks.4
1.25	eks.5

若要檢查目前的 Kubernetes 和平台版本，您可以使用叢集名稱取代以下命令中的 *my-cluster*，然後執行修改後的命令：**aws eks describe-cluster --name *my-cluster* --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'**。

- 已為叢集啟用 Windows 支援。如需詳細資訊，請參閱 為您的 Amazon EKS 叢集啟用 Windows 支援。

若要增加 Amazon EC2 節點的可用 IP 地址數量

1. 設定叢集以將 IP 地址字首指派給節點。完成與節點作業系統相符的索引標籤上的步驟。

   Linux

   1. 啟用參數以為 Amazon VPC CNI DaemonSet 的網路介面指派字首。當您部署 1.21 或更高版本的叢集時，Amazon VPC CNI plugin for Kubernetes 附加元件的版本 1.10.1 或更高版本將與其一起部署。如果您使用 IPv6 系列建立叢集，則此設定會預設設為 true。如果您使用 IPv4 系列建立叢集，則此設定會預設設為 false。

      ```
      kubectl set env daemonset aws-node -n kube-system
       ENABLE_PREFIX_DELEGATION=true
      ```

      Important

      即使子網路有可用的 IP 地址，如果子網路沒有任何可用的連續 /28 區塊，您也會在 Amazon VPC CNI plugin for Kubernetes 日誌中看到下列錯誤：

      ```
      InsufficientCidrBlocks: The specified subnet does not have enough free
       cidr blocks to satisfy the request
      ```

這可能是由分散在子網路上的現有次要 IP 地址的分段造成。若要解決此錯誤，請建立新的子網並在該處啟動 Pods，或使用 Amazon EC2 子網 CIDR 保留來保留子網內的空間，以便與字首指派搭配使用。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的子網路 CIDR 保留。

2. 如果計劃在沒有啟動範本的情況下或使用未在其中指定 AMI ID 的啟動範本來部署受管節點群組，且您使用的是先決條件中列出的 Amazon VPC CNI plugin for Kubernetes 版本或更新版本，則請跳至下一個步驟。受管節點群組會自動為您計算 Pods 的數量上限。

   如果要部署的是自我管理節點群組，或使用其中具有指定 AMI ID 的啟動範本來部署受管節點群組，則必須決定 Amazon EKS 為節點建議的 Pods 數量上限。請遵循 Amazon EKS 為每種 Amazon EC2 執行個體類型建議 Pods 數量上限 中的指示，將 **--cni-prefix-delegation-enabled** 新增至步驟 3。請記下輸出，以便在稍後步驟中使用。

   Important

   受管節點群組會強制對 maxPods 的值執行數量上限。對於少於 30 個 vCPU 的執行個體，數量上限為 110；對於所有其他執行個體，數量上限為 250。無論是否啟用字首委派，都會套用此數量上限。

3. 如果您使用針對 IPv6 設定 1.21 或更新版本叢集，請跳到下一個步驟。

   在下列其中一個選項中指定參數。若要判斷哪個選項適合您，以及為其提供哪些價值，請參閱 GitHub 上的 WARM_PREFIX_TARGET、WARM_IP_TARGET 以及 MINIMUM_IP_TARGET。

   您可以用大於零的值取代 *example values*。

   - WARM_PREFIX_TARGET

     ```
     kubectl set env ds aws-node -n kube-system WARM_PREFIX_TARGET=1
     ```

   - WARM_IP_TARGET 或 MINIMUM_IP_TARGET：如果此值已設定，則其會覆寫為 WARM_PREFIX_TARGET 設定的任何值。

     ```
     kubectl set env ds aws-node -n kube-system WARM_IP_TARGET=5
     ```

     ```
     kubectl set env ds aws-node -n kube-system MINIMUM_IP_TARGET=2
     ```

4. 使用至少一個 Amazon EC2 Nitro Amazon Linux 2 執行個體類型，建立下列類型的節點群組之一。如需 Nitro 執行個體類型的清單，請參閱 Amazon EC2 使用者指南中的在 Nitro 系統上建置的執行個體。Windows 不支援這項功能。對於包含 *110* 的選項，使用步驟 3 的值 (建議) 或您自己的值將其取代。

   - 自我管理：使用 啟動自我管理的 Amazon Linux 節點 中的指示部署節點群組。指定BootstrapArguments參數的下列文字。

     ```
     --use-max-pods false --kubelet-extra-args '--max-pods=110'
     ```

     如果使用 `eksctl` 來建立節點群組，便可使用以下命令。

     ```
     eksctl create nodegroup --cluster my-cluster --managed=false --max-pods-per-node 110
     ```

   - 受管：使用下列其中一個選項，部署節點群組：

     - 沒有啟動範本，或有啟動範本，但沒有指定 AMI ID：完成 建立受管節點群組 中的程序。受管節點群組會自動為您計算 Amazon EKS 建議的 `max-pods` 值。
     - 使用具有指定 AMI ID 的啟動範本：在啟動範本中，指定 Amazon EKS 最佳化 AMI ID 或基於 Amazon EKS 最佳化 AMI 的自訂 AMI，然後使用啟動範本部署節點群組，並在啟動範本中提供下列使用者資料。此使用者資料會將引數傳遞至 `bootstrap.sh` 檔案。如需引導檔案的詳細資訊，請參閱 GitHub 上的 bootstrap.sh。

       ```
       /etc/eks/bootstrap.sh my-cluster \
         --use-max-pods false \
         --kubelet-extra-args '--max-pods=110'
       ```

       如果使用 `eksctl` 來建立節點群組，便可使用以下命令。

       ```
       eksctl create nodegroup --cluster my-cluster --max-pods-per-node 110
       ```

       如果您建立的自訂 AMI 並非基於 Amazon EKS 最佳化 AMI，則需要自行自訂建立組態。

> **Note**
>
> 如果您還想要將 IP 地址指派給來自與執行個體不同子網的 Pods，則需要在此步驟中啟用該功能。如需詳細資訊，請參閱 Pod 的自訂聯網。

Windows

1. 啟用 IP 字首指派。

   a. 開啟 amazon-vpc-cni ConfigMap 進行編輯。

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. 將下行新增至 data 區段：

   ```
     enable-windows-prefix-delegation: "true"
   ```

   c. 儲存檔案並關閉編輯器。

   d. 確認行已新增至 ConfigMap。

   ```
   kubectl get configmap -n kube-system amazon-vpc-cni -o
    "jsonpath={.data.enable-windows-prefix-delegation}"
   ```

   如果傳回的輸出不是 true，則可能會出現錯誤。請嘗試再次完成該步驟。

   > **Important**
   >
   > 即使子網路有可用的 IP 地址，如果子網路沒有任何可用的連續 /28 區塊，您也會在節點事件中看到下列錯誤：
   >
   > ```
   > "failed to allocate a private IP/Prefix address:
   >  InsufficientCidrBlocks: The specified subnet does not have enough
   >  free cidr blocks to satisfy the request"
   > ```
   >
   > 這可能是由分散在子網路上的現有次要 IP 地址的分段造成。若要解決此錯誤，請建立新的子網並在該處啟動 Pods，或使用 Amazon EC2 子網 CIDR 保留來保

留子網內的空間，以便與字首指派搭配使用。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的子網路 CIDR 保留。

2. (選用) 指定其他組態來控制叢集的預先擴展和動態擴展行為。如需詳細資訊，請參閱開啟前置碼委派模式的Windows組態選項 GitHub。

   a. 開啟 amazon-vpc-cni ConfigMap 進行編輯。

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. 使用大於零的值取代 *example values*，再將所需項目新增至 ConfigMap 的 data 區段。如果您為 warm-ip-target 或 minimum-ip-target 設定了值，則該值會覆寫任何為 warm-prefix-target 設定的值。

   ```
   warm-prefix-target: "1"
   warm-ip-target: "5"
   minimum-ip-target: "2"
   ```

   c. 儲存檔案並關閉編輯器。

3. 建立 Windows 節點群組，其中至少包含一個 Amazon EC2 Nitro 執行個體類型。Nitro如需執行個體類型的清單，請參閱 Amazon Amazon EC2 使用者指南中的在Nitro系統上建置的執行個體。依預設，您最多可將 110 個 Pods 部署到節點。如果您要增加或減少該數量，請在引導組態的使用者資料中指定以下內容：使用 Pod 最大值取代 *max-pods-quantity*。

   ```
   -KubeletExtraArgs '--max-pods=max-pods-quantity'
   ```

   如果您要部署受管節點群組，則需要在啟動範本中新增此組態。如需詳細資訊，請參閱 使用啟動範本自訂受管節點。如需有關 Windows 引導指令碼組態參數的詳細資訊，請參閱 引導指令碼組態參數。

2. 部署您的節點後，請檢視叢集中的節點。

```
kubectl get nodes
```

範例輸出如下。

```
NAME                                                STATUS     ROLES    AGE   VERSION
ip-192-168-22-103.region-code.compute.internal   Ready      <none>   19m   v1.XX.X-eks-6b7464
```

```
ip-192-168-97-94.region-code.compute.internal    Ready      <none>   19m   v1.XX.X-
eks-6b7464
```

3. 描述其中一個節點，判斷該節點的 max-pods 值以及可用 IP 地址的數量。使用上一個輸出中傳回的其中一個節點名稱中的 IPv4 地址取代 *192.168.30.193*。

```
kubectl describe node ip-192-168-30-193.region-code.compute.internal | grep 'pods\|
PrivateIPv4Address'
```

範例輸出如下。

```
pods:                                       110
vpc.amazonaws.com/PrivateIPv4Address:  144
```

在先前的輸出中，110 是 Kubernetes 將部署至節點的 Pods 數量上限，即使有 *144* 個可用的 IP 地址。

Pods 的安全群組

Pods 的安全群組整合了 Amazon EC2 安全群組與 Kubernetes Pods。您可以使用 Amazon EC2 安全群組定義規則，允許您部署的 Pods 到許多 Amazon EC2 執行個體類型和 Fargate 上執行的節點之間的對內和對外網路流量。如需此功能的詳細說明，請參閱 Pods 的安全群組簡介部落格文章。

考量事項

- 部署 Pods 的安全群組之前，請考慮下列限制和條件：
- Pods 的安全群組無法與 Windows 節點搭配使用。
- Pods 的安全群組可與為包含 Amazon EC2 節點的 IPv6 系列設定的叢集搭配使用，方法是使用 1.16.0 版或更新的 Amazon VPC CNI 外掛程式。Pods 的安全群組可與為僅包含 Fargate 節點的 IPv6 系列設定的叢集搭配使用，方法是使用 1.7.7 版或更新的 Amazon VPC CNI 外掛程式。如需更多資訊，請參閱IPv6叢集的位址Pods、和 services
- Pods 的安全群組受大多數 Nitro 型 Amazon EC2 執行個體系列支援，但並非受所有系列世代皆支援。例如，支援m5c5、r5、m6gc6g、和r6g例證族群和層代。不支援 t 系列中的執行個體類型。如需完整的支援執行個體類型清單，請參閱 GitHub 上的 limits.go 檔案。您的節點必須為該檔案列出的具有 IsTrunkingCompatible: true 的執行個體類型之一。
- 如果您也使用 Pod 安全政策來限制對 Pod 變動的存取權，則必須針對 psp 被指派到的 role 在 Kubernetes ClusterRoleBinding 中指定 eks:vpc-resource-controller Kubernetes

使用者。如果您使用的是預設 Amazon EKS psp、role 及 ClusterRoleBinding，則為 eks:podsecuritypolicy:authenticated ClusterRoleBinding。例如，您可以將使用者新增至 subjects: 區段，如下列範例所示：

```
[...]
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: system:authenticated
  - apiGroup: rbac.authorization.k8s.io
    kind: User
    name: eks:vpc-resource-controller
  - kind: ServiceAccount
    name: eks-vpc-resource-controller
```

- 如果您使用的是自訂聯網和 Pods 的安全群組，則會使用 Pods 安全群組指定的安全群組，而不是 ENIConfig 指定的安全群組。
- 如果您使用的是版本 1.10.2 或更舊版本的 Amazon VPC CNI 外掛程式，且 terminationGracePeriodSeconds 規格包括 Pod 設定，則該設定的值不能為零。
- 如果您使用的是版本 1.10 或更舊版本的 Amazon VPC CNI 外掛程式，或預設設定為 POD_SECURITY_GROUP_ENFORCING_MODE=strict 的版本 1.11，則使用 externalTrafficPolicy 設定為 Local 的執行個體目標的 NodePort 和 LoadBalancer 類型 Kubernetes 服務，就不支援指派安全群組至其 Pods。如需將負載平衡器與執行個體目標搭配使用的詳細資訊，請參閱 Amazon EKS 上的網路負載平衡
- 如果您使用的是版本 1.10 或更舊版本的 Amazon VPC CNI 外掛程式或預設設定為 POD_SECURITY_GROUP_ENFORCING_MODE=strict 的版本 1.11，來自具有指派安全群組之 Pods 的對外流量會停用來源 NAT，以便套用輸出安全群組規則。若要存取網際網路，必須在部署在使用 NAT 閘道或執行個體設定的私有子網中的節點上啟動具有指派安全群組的 Pods。已指派安全群組部署至公有子網的 Pods 無法存取網際網路。

  如果您使用的是版本 1.11 或更新版本的搭配 POD_SECURITY_GROUP_ENFORCING_MODE=standard 的外掛程式，則以 VPC 外部為目標的 Pod 流量會轉換為執行個體主要網路介面的 IP 地址。對於此流量，將使用主要網路介面的安全群組中的規則，而不是使用 Pod's 安全群組中的規則。
- 若要將 Calico 網路政策與和安全群組關聯的 Pods 搭配使用，則必須使用版本 1.11.0 或更新版本的 Amazon VPC CNI 外掛程式，並設定為 POD_SECURITY_GROUP_ENFORCING_MODE=standard。否則，具有關聯的安全群組之 Pods 的流量流出和流向不受 Calico 網路政策強制執行的約束，且僅限於 Amazon EC2 安全群組強制執行。

若要更新 Amazon VPC CNI 版本，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件

- Pods 在 Amazon EC2 節點上執行，這些節點使用位於使用 Nodelocal DNSCache 的叢集內的安全群組，僅受 Amazon VPC CNI 外掛程式版本 `1.11.0` 或更新版本以及 `POD_SECURITY_GROUP_ENFORCING_MODE=standard` 支援。若要更新您的 Amazon VPC CNI 外掛程式版本，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件
- Pods 的安全群組可能會導致具有高流失率的 Pods 產生更高的 Pod 啟動延遲。原因是資源控制器的速率限制。

為 Pods 設定安全群組的 Amazon VPC CNI plugin for Kubernetes

若要部署 Pods 的安全群組

如果您僅使用 Fargate Pods 的安全群組，且叢集中沒有任何 Amazon EC2 節點，請跳至步驟 部署範例應用程式。

1. 使用以下命令查看您目前的 Amazon VPC CNI plugin for Kubernetes 版本：

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
    | cut -d : -f 3
   ```

   範例輸出如下。

   ```
   v1.7.6
   ```

   如果您的 Amazon VPC CNI plugin for Kubernetes 版本早於 `1.7.7`，則請將外掛程式更新至版本 `1.7.7` 或更新版本。如需更多資訊，請參閱使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件

2. 將 `AmazonEKSVPCResourceController` 受管 IAM 政策新增至與您 Amazon EKS 叢集相關聯的叢集角色。此政策允許角色管理網路介面、其私有 IP 地址以及其與網路執行個體之間的連接和分離。

   a. 擷取叢集 IAM 角色的名稱，並存放在變數中。使用您叢集的名稱取代 *my-cluster*。

      ```
      cluster_role=$(aws eks describe-cluster --name my-cluster --query
       cluster.roleArn --output text | cut -d / -f 2)
      ```

   b. 將政策連接到角色。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSVPCResourceController --role-name $cluster_role
```

3. 啟用 Amazon VPC CNI 外掛程式來管理 Pods 的網路介面，方法是將 ENABLE_POD_ENI 變數設定為 aws-node DaemonSet 中的 true。一旦此設定設定為 true，對於叢集中的每個節點，該附加元件會建立一個 cninode 自訂資源。VPC 資源控制器會建立並連接一個稱為幹線網路介面與描述 aws-k8s-trunk-eni 的特殊網路介面。

```
kubectl set env daemonset aws-node -n kube-system ENABLE_POD_ENI=true
```

> **Note**
>
> 幹線網路介面包含在執行個體類型所支援的最大網路介面數量中。如需每種執行個體類型支援的最大網路界面數目清單，請參閱 Amazon EC2 使用者指南中每個執行個體類型每個網路界面的 IP 地址。如果您的節點已經連接到標準網路介面的最大數量，那麼 VPC 資源控制器將保留一個空間。您將不得不縮減正在執行的 Pods 的規模，以便控制器分開並刪除標準網路介面，建立幹線網路介面，並將其連接到執行個體。

4. 您可以使用以下命令查看您的哪些節點擁有 CNINode 自訂資源。如果傳回 No resources found，則等待幾秒鐘後再重試一次。上一步驟要求重新啟動 Amazon VPC CNI plugin for Kubernetes Pods，這需要幾秒鐘的時間。

```
$ kubectl get cninode -A
     NAME FEATURES
     ip-192-168-64-141.us-west-2.compute.internal
 [{"name":"SecurityGroupsForPods"}]
     ip-192-168-7-203.us-west-2.compute.internal [{"name":"SecurityGroupsForPods"}]
```

如果您使用的 VPC CNI 版本低於 1.15，則會使用節點標籤而非 CNINode 自訂資源。您可以使用以下命令查看您的哪些節點已將節點標籤 aws-k8s-trunk-eni 設定為 true。如果傳回 No resources found，則等待幾秒鐘後再重試一次。上一步驟要求重新啟動 Amazon VPC CNI plugin for Kubernetes Pods，這需要幾秒鐘的時間。

```
kubectl get nodes -o wide -l vpc.amazonaws.com/has-trunk-attached=true
-
```

一旦建立幹線網路介面，Pods 就可以從幹線或標準網路介面指派次要 IP 地址。如果刪除節點，則會自動刪除幹線介面。

當您在稍後的步驟中為 Pod 部署安全群組時，VPC 資源控制器會建立稱為分支網路介面與 描述 `aws-k8s-branch-eni`的特殊網路介面，並將安全群組與其關聯。除了連接至節點的標準和幹線網路介面之外，還會建立分支網路介面。

如果您使用的是存活或整備探查，則還需要停用 TCP 早期的 demux，以便 `kubelet` 可以透過 TCP 連接到分支網路介面上的 Pods。若要停用 TCP 早期的 Demux，請執行下列命令：

```
kubectl patch daemonset aws-node -n kube-system \
  -p '{"spec": {"template": {"spec": {"initContainers": [{"env":
[{"name":"DISABLE_TCP_EARLY_DEMUX","value":"true"}],"name":"aws-vpc-cni-
init"}]}}}}'
```

> **Note**
>
> 如果您使用的是 `1.11.0` 或更新版本的 Amazon VPC CNI plugin for Kubernetes 附加元件和設定 `POD_SECURITY_GROUP_ENFORCING_MODE=standard`，如下一個步驟所述，則不需要執行之前的命令。

5. 如果您的叢集使用 `NodeLocal DNSCache`，或者您希望將 Calico 網路政策與具有自己的安全群組的 Pods 搭配使用，或者您的 Kubernetes 服務類型為 `NodePort` 和 `LoadBalancer`，其使用的執行個體目標中您欲指派安全群組的 `externalTrafficPolicy` 的 `Local` 設定為 Pods，則必須使用版本 `1.11.0` 或更新版本的 Amazon VPC CNI plugin for Kubernetes 附加元件，且您必須啟用下列設定：

```
kubectl set env daemonset aws-node -n kube-system
 POD_SECURITY_GROUP_ENFORCING_MODE=standard
```

> **Important**
>
> - Pod 安全群組規則不會套用至介於Pods 或 介於 Pods 和 services 之間的流量，例如位於同一個節點上的 `kubelet` 或 `nodeLocalDNS`。在相同節點上使用不同安全群組的 Pod 無法進行通訊，這是因為它們在不同子網路中設定，且這些子網路之間已停用路由。

- 來自 Pods 的輸出流量傳輸到 VPC 外部的地址是轉換為執行個體主要網路介面的 IP 地址的網路地址 (除非您還設定了AWS_VPC_K8S_CNI_EXTERNALSNAT=true)。對於此流量，將使用主要網路介面的安全群組中的規則，而不是使用 Pod's 安全群組中的規則。
- 若要將此設定套用至現有的 Pods，您必須重新啟動 Pods 或 Pods 正在運行之節點。

部署範例應用程式

若要將安全群組用於 Pods，您必須擁有現有的安全群組和部署 Amazon EKS SecurityGroupPolicy至叢集，如下列步驟所述。下列步驟介紹如何使用 Pod 的安全群組政策。除非另有備註，請從同一終端機完成所有步驟，因為變數用於不會保留跨終端機的以下步驟。

使用安全群組部署範例 Pod

1. 建立 Kubernetes 命名空間以部署資源。您可以將 *my-namespace* 取代為要使用的命名空間的名稱。

```
kubectl create namespace my-namespace
```

2. 將 Amazon EKS SecurityGroupPolicy 部署至您的叢集。

   a. 將以下內容複製到您的裝置。若您希望根據服務帳戶標籤選取 Pods，則您可以使用 **serviceAccountSelector** 取代 *podSelector*。您必須指定其中一個選取器或其他工具。空白 podSelector (例如：podSelector: {}) 選取命名空間中的所有 Pods。您可以將 *my-role* 變更為您的角色名稱。空白 serviceAccountSelector 會選取命名空間中的所有服務帳戶。您可以將 *my-security-group-policy* 取代為您的 SecurityGroupPolicy 的名稱，將 *my-namespace* 取代為要在其中建立 SecurityGroupPolicy 的命名空間。

   您必須將 *my_pod_security_group_id* 取代為現有安全群組的 ID。如果您沒有現有的安全群組，則必須先建立一個。如需詳細資訊，請參閱《Amazon EC2 使用者指南》https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/中的適用於 Linux 執行個體的 Amazon EC2 安全群組一節。您可以指定 1-5 個安全群組 ID。如果您指定多個 ID，則所有安全群組中的所有規則的組合對選取的 Pods 都有效。

```
cat >my-security-group-policy.yaml <<EOF
apiVersion: vpcresources.k8s.aws/v1beta1
kind: SecurityGroupPolicy
```

```
metadata:
  name: my-security-group-policy
  namespace: my-namespace
spec:
  podSelector:
    matchLabels:
      role: my-role
  securityGroups:
    groupIds:
      - my_pod_security_group_id
EOF
```

> **Important**
>
> 您指定的 Pod 的一個或多個安全群組 必須符合下列條件：
>
> - 它們必須存在。如果安全群組不存在，那麼當您部署符合選取器的 Pod 時，您的 Pod 仍會停留在建立程序中。如果您描述 Pod，您會看到類似下列的錯誤訊息：`An error occurred (InvalidSecurityGroupID.NotFound) when calling the CreateNetworkInterface operation: The securityGroup ID 'sg-05b1d815d1EXAMPLE' does not exist`。
> - 安全群組必須允許透過您已設定偵測的任何連接埠，而來自套用於您節點的安全群組的傳入通訊 (適用於 kubelet)。
> - 其必須允許透過 TCP 和 UDP 連接埠 53 與指派給執行 CoreDNS 的 Pods 的安全群組 (或 Pods 在其上執行的節點) 進行對外通訊。您的 CoreDNS Pods 的安全群組必須允許來自您所指定安全群組的傳入 TCP 和 UDP 連接埠 53 流量。
> - 其必須具備必要的傳入和傳出規則，從而和其需要與之溝通的其他 Pods 溝通。
> - 如果您使用具有 Fargate 的安全群組，其必須具有允許 Pods 與 Kubernetes 控制平面通訊的規則。執行此動作最簡單的方法是指定叢集安全群組為其中一個安全群組。
>
> 安全群組政策僅適用於新排程的 Pods。它們不會影響執行中的 Pods。

b. 部署政策。

```
kubectl apply -f my-security-group-policy.yaml
```

3. 部署範例應用程式，其中標籤符合您在上一步驟指定的 *podSelector* 的 *my-role* 值。

a. 將以下內容複製到您的裝置。使用您自己的值取代*###*，然後執行修改後的命令。如果您取代 *my-role*，請確保其與您在上一步中為選取器指定的值相同。

```
cat >sample-application.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deployment
  namespace: my-namespace
  labels:
    app: my-app
spec:
  replicas: 4
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        role: my-role
    spec:
      terminationGracePeriodSeconds: 120
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: my-app
  namespace: my-namespace
  labels:
    app: my-app
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
```

```
      targetPort: 80
EOF
```

b. 使用下列命令部署應用程式。當您部署應用程式時，Amazon VPC CNI plugin for Kubernetes 符合您在上一步驟中指定的 `role` 標籤和安全群組，已套用至 Pod。

```
kubectl apply -f sample-application.yaml
```

4. 檢視與範例應用程式一起部署的 Pods。對於本主題的餘數，該終端機稱為 `TerminalA`。

```
kubectl get pods -n my-namespace -o wide
```

範例輸出如下。

```
NAME                             READY   STATUS    RESTARTS   AGE     IP
       NODE                                            NOMINATED NODE   READINESS
 GATES
my-deployment-5df6f7687b-4fbjm   1/1     Running   0          7m51s   192.168.53.48
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
my-deployment-5df6f7687b-j9fl4   1/1     Running   0          7m51s
  192.168.70.145   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-rjxcz   1/1     Running   0          7m51s
  192.168.73.207   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-zmb42   1/1     Running   0          7m51s   192.168.63.27
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
```

> **Note**
>
> - 若有任何 Pods 卡在 `Waiting` 狀態，則請執行 `kubectl describe pod my-deployment-xxxxxxxxxx-xxxxx -n my-namespace`。若您看到 `Insufficient permissions: Unable to create Elastic Network Interface.`，請確認您已在上一步驟中將 IAM 政策新增至 IAM 叢集角色。
> - 如果有任何 Pods 卡在 `Pending` 狀態，請確認您的節點執行個體類型已列在 limits.go 中，並且尚未達到該執行個體類型支援的分支網路界面數量上限乘以節點群組中節點數量的乘積。例如，`m5.large` 執行個體支援九個分支網路介面。如果您的節點群組有五個節點，則最多可以為節點群組建立 45 個分支網路介面。您嘗試部署的

第 46 個 Pod 將位於 Pending 狀態，直到刪除具有相關聯安全群組的另一個 Pod 為止。

如果您執行 kubectl describe pod ***my-deployment-xxxxxxxxxx-xxxxx*** **-n** ***my-namespace***，並看到類似下列訊息的訊息，則可以放心地將其忽略。如果在系統建立網路介面時 Amazon VPC CNI plugin for Kubernetes 嘗試設定主機聯網並失敗，可能會出現此訊息。外掛程式會記錄此事件，直到網路介面建立為止。

```
Failed to create Pod sandbox: rpc error: code = Unknown desc = failed to set up
 sandbox container
 "e24268322e55c8185721f52df6493684f6c2c3bf4fd59c9c121fd4cdc894579f" network for Pod
 "my-deployment-5df6f7687b-4fbjm": networkPlugin
cni failed to set up Pod "my-deployment-5df6f7687b-4fbjm-c89wx_my-namespace"
 network: add cmd: failed to assign an IP address to container
```

您不能超過執行個體類型上可以執行的 Pods 數量上限。關於每種執行個體類型上可執行的 Pods 數量上限清單，請參閱 GitHub 上的 eni-max-pods.txt。當您刪除具有關聯安全群組的 Pod，或刪除 Pod 執行所在的節點時，VPC 資源控制器會刪除分支網路介面。如果您使用安全群組的 Pods 來刪除具有 Pods 的叢集，則控制器不會刪除分支網路介面，因此您需要自行刪除它們。如需如何刪除網路界面的詳細資訊，請參閱 Amazon EC2 使用者指南中的刪除網路界面。

5. 在單獨的終端機中，shell 轉換為 Pods。對於本主題的餘數，該終端機稱為 TerminalB。使用從上一步驟傳回的輸出中的其中一個 Pods 的 ID 取代 *5df6f7687b-4fbjm*。

```
kubectl exec -it -n my-namespace my-deployment-5df6f7687b-4fbjm -- /bin/bash
```

6. 從 TerminalB 中的 shell，確認範例應用程式是否正常運作。

```
curl my-app
```

範例輸出如下。

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

您收到了輸出，因為所有執行該應用程式的 Pods 都與您建立的安全群組關聯。該群組包含一個規則，其允許與安全群組關聯的所有 Pods 之間的流量。允許從該安全群組傳出 DNS 流量到與節點關聯的叢集安全群組。這些節點正在執行 CoreDNS Pods，您的 Pods 對此進行了名稱查詢。

7. 從 TerminalA 中，移除允許從安全群組與叢集進行 DNS 通訊的安全群組規則。如果在上一步驟並未新增 DNS 規則至叢集安全群組，則用您在其中建立規則的安全群組的 ID 取代 *$my_cluster_security_group_id*。

```
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_tcp_rule_id
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_udp_rule_id
```

8. 從 TerminalB，再次嘗試存取該應用程式。

```
curl my-app
```

範例輸出如下。

```
curl: (6) Could not resolve host: my-app
```

嘗試失敗，因為 Pod 不再能存取 CoreDNS Pods，其具有與之關聯的叢集安全群組。叢集安全群組不再具有允許從與 Pod 關聯的安全群組進行 DNS 通訊的安全群組規則。

如果您在上一步驟中，嘗試存取使用其中一個 Pods 傳回的 IP 地址的應用程式，您仍會接收到回應，因為已允許具有與之關聯的安全群組的 Pods 之間的所有連接埠，並且不需要進行名稱查詢。

9. 在實驗完成後，您可以移除自己建立的範例安全群組政策、應用程式和安全群組。從 TerminalA 執行下列命令。

```
kubectl delete namespace my-namespace
aws ec2 revoke-security-group-ingress --group-id $my_pod_security_group_id --
security-group-rule-ids $my_inbound_self_rule_id
wait
sleep 45s
aws ec2 delete-security-group --group-id $my_pod_security_group_id
```

Pods 的多個網路介面

Multus CNI 是適用於 Amazon EKS 的容器網路介面 (CNI) 外掛程式，可將多個網路介面連接到 Pod。如需詳細資訊，請參閱 GitHub 上的 Multus-CNI 文件。

在 Amazon EKS 中，每個 Pod 都有一個由 Amazon VPC CNI 外掛程式指派的網路介面。使用 Multus，您可以建立具有多個介面的多重主目錄 Pod。這是由 Multus 充當「中繼外掛程式」完成的；一個 CNI 外掛程式，可以呼叫多個其他 CNI 外掛程式。AWS 對 Multus 的支援設定了 Amazon VPC CNI 外掛程式作為預設委派外掛程式。

考量事項

- Amazon EKS 將無法建置和發佈單一根 I/O 虛擬化 (SR-IOV) 和資料平面開發套件 (DPDK) CNI 外掛程式。不過，您可以透過 Multus 受管主機裝置和 `ipvlan` 外掛程式直接連接至 Amazon EC2 彈性網路介面 (ENA)，進而達成封包加速。
- Amazon EKS 支援 Multus，它提供了一個通用的程序，可以簡單地鏈接其他 CNI 外掛程式。支援 Multus 和鏈接過程，但 AWS 不會支援所有可鏈接的相容 CNI 外掛程式，或者與鏈接組態無關的 CNI 外掛程式中可能出現的問題。
- Amazon EKS 為 Multus 外掛程式提供支援和生命週期管理，但不負責與其他網路介面相關聯的任何 IP 地址或其他管理。使用 Amazon VPC CNI 外掛程式之預設網路介面的 IP 地址和管理維持不變。
- mazon VPC CNI 外掛程式僅獲預設委派外掛程式的正式支援。如果您選擇不將 Amazon VPC CNI 外掛程式用於主要聯網，則需要修改已發佈的 Multus 安裝清單檔案，以將預設委派外掛程式重新設定為替代 CNI。
- 只有在使用 Amazon VPC CNI 作為主要 CNI 時，才支援 Multus。用於高階、次要或其他介面時，我們不支援 Amazon VPC CNI。
- 若要防止 Amazon VPC CNI 外掛程式嘗試管理指派給 Pods 的其他網路介面，請將下列標籤新增至網路介面。

  key (索引鍵)：`node.k8s.amazonaws.com/no_manage`

  value (值)：`true`
- Multus 與網路政策相容，但必須加強政策才能包含連接埠和 IP 地址，其中這些連接埠和 IP 地址可能是連接至 Pods 的其他網路介面的一部分。

如需實作逐步示範，請參閱 GitHub 上的 Multus 設定指南。

替代相容的 CNI 外掛程式

Amazon VPC CNI plugin for Kubernetes 是 Amazon EKS 支援的唯一 CNI 外掛程式。Amazon EKS 會執行上游 Kubernetes，因此您可以將替代相容 CNI 外掛程式安裝到叢集中的 Amazon EC2 節點。如果您的叢集中有 Fargate 節點，則表示 Amazon VPC CNI plugin for Kubernetes 已在 Fargate 節點上。這是唯一可以與 Fargate 節點搭配使用的 CNI 外掛程式。嘗試在 Fargate 節點上安裝替代 CNI 外掛程式失敗。

如果您打算在 Amazon EC2 節點中使用替代 CNI 外掛程式，那麼建議您取得外掛程式的商業支援，或擁有內部專業知識來排除問題，並為 CNI 外掛程式專案提供修正方法。

Amazon EKS 會持續與提供替代相容 CNI 外掛程式支援的合作夥伴網路建立關係。請參閱下列合作夥伴文件，獲取有關版本、資格和所執行測試的詳細資訊。

合作夥伴	產品	文件
Tigera	Calico	安裝說明
Isovalent	Cilium	安裝說明
Juniper	雲端原生 Contrail Networking (CN2)	安裝說明
VMware	Antrea	安裝說明

Amazon EKS 旨在為您提供各種選擇來涵蓋所有使用案例。

替代兼容的網絡策略插件

Calico 是一種廣泛採用的容器網路和安全解決方案。Calico在 EKS 上使用可為您的 EKS 叢集提供完全合規的網路原則強制執行。此外，您可以選擇使用Calico的網路，以節省基礎 VPC 的 IP 位址。印花布雲增強了的功能Calico Open Source，提供了進階的安全性和可觀察性功能。

什麼是 AWS Load Balancer Controller？

AWS Load Balancer Controller管理Kubernetes叢集的 AWS 彈性負載平衡器。您可以使用控制器將叢集應用程式公開至網際網路。控制器佈建指向叢集服務或 Ingress 資源的 AWS 負載平衡器。換句話說，控制器會建立指向叢集中多個網繭的單一 IP 位址或 DNS 名稱。

控制器監視KubernetesIngress或Service資源。作為回應，它會建立適當的 E AWS lastic Load Balancing 資源。您可以透過將註釋套用至Kubernetes資源來設定負載平衡器的特定行為。例如，您可以使用註釋將 AWS 安全群組附加至負載平衡器。

控制器會佈建以下資源：

Kubernetes `Ingress`

當您建立AWS 應用程式負載平衡器 (ALB) 時，LBC 會建立. Kubernetes `Ingress` 檢閱您可以套用至 Ingress 資源的註釋。

`LoadBalancer` 類型的 Kubernetes 服務

當您建立類型的Kubernetes服務時，LBC 會建立 AWS Network Load Balancer (NLB)。`LoadBalancer`檢閱您可以套用至服務資源的註釋。

過去，Kubernetes網路負載平衡器用於執行個體目標，但 LBC 已用於 IP 目標。使用 AWS Load Balancer Controller `2.3.0` 版或更新版本，您可以使用任一目標類型建立 NLB。如需 NLB 目標類型的詳細資訊，請參閱《Network Load Balancer 使用者指南》中的目標類型。

控制器是一個管理的開源項目GitHub。

在部署控制器之前，我們建議您檢閱 Amazon EKS 上的應用程式負載平衡 和 Amazon EKS 上的網路負載平衡 中的先決條件和考量事項。在這些主題中，您將部署包含 AWS 負載平衡器的範例應用程式。

安裝控制器

- 瞭解如何the section called “使用頭盔安裝”。如果您是 Amazon EKS 的新手，請使用此程序。此程序使用 Helm (套件管理員)Kubernetes，並`eksctl`簡化 LBC 的安裝程序。
- 或者，the section called “使用資訊清單安裝”. 此程序適用於進階叢集配置。這包括對公用容器登錄具有限制網路存取權的叢集。

從過時的控制器版本遷移

- 如果您已淘汰AWS Load Balancer Controller安裝的版本，請瞭解如何the section called “從棄用的控制器遷移”。
- 已取代的版本無法升級。它們必須被刪除並AWS Load Balancer Controller安裝的當前版本。
- 已過時的版本包括：

- AWS ALB 入口控制器 Kubernetes (「入口控制器」)，前身是. AWS Load Balancer Controller
- 任何`0.1.x`版本的 AWS Load Balancer Controller

傳統雲端供應商

Kubernetes包括的舊版雲端供應商 AWS。舊版雲端提供者能夠佈建 AWS 負載平衡器，類似於AWS Load Balancer Controller. 舊版雲端提供者會建立傳統負載平衡器。如果您未安裝AWS Load Balancer Controller，Kubernetes將預設為使用舊版雲端提供者。您應該安裝AWS Load Balancer Controller並避免使用舊版雲端供應商。

> **Important**
>
> 在 2.5 及更新版本中，AWS Load Balancer Controller會成為Kubernetes服務資源的預設控制器，`type: LoadBalancer`並為每個服務建立 AWS Network Load Balancer (NLB)。它透過為服務製作變異的 webhook 來做到這一點，此 webhook 會為 `type: LoadBalancer` 的新服務將 `spec.loadBalancerClass` 欄位設定至 `service.k8s.aws/nlb`。您可以關閉此功能並恢復為使用舊式雲端供應商作為預設控制器，方法是將 Helm Chart 值 `enableServiceMutatorWebhook` 設定為 `false`。除非您關閉此功能，否則叢集不會為您的服務佈建新的 Classic Load Balancer。現有的 Classic Load Balancer 會繼續運作。

安裝使AWS Load Balancer Controller用頭盔

本主題說明如何AWS Load Balancer Controller使用 Helm (適Kubernetes用於和的套件管理員) 來安裝`eksctl`。控制器以預設選項安裝。如需有關控制器的詳細資訊，包括使用註解設定控制器的詳細資訊，請參閱上的AWS Load Balancer Controller文件GitHub。

在下列步驟中，使用自己的值取代 *`example values`*。

必要條件

開始此教學之前，您必須先安裝和設定下列在建立和管理 Amazon EKS 叢集所需的工具和資源。

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。
- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。
- 確認您的 Amazon VPC CNI plugin for Kubernetes、`kube-proxy` 及 CoreDNS 附加元件為服務帳戶字符中列出的最低版本。

- 熟悉 E AWS lastic Load Balancing 如需詳細資訊，請參閱《Elastic Load Balancing 使用者指南》。
- 熟悉 Kubernetes 服務和傳入資源。
- 頭盔安裝在本地。

步驟 1：使用以下方式建立 IAM 角色 **eksctl**

> **Note**
>
> 您只需為每個 AWS 帳戶建立一AWS Load Balancer Controller個 IAM 角色。檢查 IAM 主控台中是否AmazonEKSLoadBalancerControllerRole存在。如果此角色存在，請跳至the section called “步驟 2：安裝 AWS Load Balancer Controller”。

建立 IAM 政策。

1. 下載適用於 AWS Load Balancer Controller 的 IAM 政策，允許它代表您呼叫 AWS API。

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. 使用上一個步驟中下載的政策，建立 IAM 政策。

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

> **Note**
>
> 如果您在中檢視原則 AWS Management Console，主控台會顯示 ELB 服務的警告，但不會顯示 ELB v2 服務的警告。發生這種情況是由於政策中存在適用於 ELB v2 的某些動作，但不適用於 ELB。您可以忽略這些對 ELB 發出的警告。

使用建立 IAM 角色 **eksctl**

- 使用叢集名稱取代 *my-cluster*，並使用帳戶 ID 取代 *111122223333*，然後執行命令。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

```
$ eksctl create iamserviceaccount \
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

步驟 2：安裝 AWS Load Balancer Controller

AWS Load Balancer Controller使用頭盔 V3 進行安裝

1. 添加`eks-charts`頭盔圖庫。 AWS 維護此儲存庫於 GitHub。

```
$ helm repo add eks https://aws.github.io/eks-charts
```

2. 更新您的本機儲存庫，以確定您擁有最新的圖表。

```
$ helm repo update eks
```

3. 安裝AWS Load Balancer Controller。

   使用您叢集的名稱取代 *my-cluster*。在下列命令中，`aws-load-balancer-controller` 是您在上一個步驟中建立的 Kubernetes 服務帳戶。

有關如何配置舵圖的更多內容，敬請參閱 values.yaml（詳見） GitHub。

```
$ helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=my-cluster \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

a. 如果您要將控制器部署到限制存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的 Amazon EC2 節點，或者如果您正在部署到 Fargate，則請將以下旗標新增到以下 helm 命令中：

   - **--set region=*region-code***
   - **--set vpcId=*vpc-xxxxxxxx***

b. 若要檢視 Helm 圖和 Load Balancer 控制器的可用版本，請使用下列命令：

```
helm search repo eks/aws-load-balancer-controller --versions
```

> **Important**
>
> 部署的圖表不會自動接收安全性更新。當圖表可用時，您需要手動升級到較新的圖表。升級時，請在上一個指令**upgrade**中變更*install*為。
> 此指helm install令會自動安裝控制器的自訂資源定義 (CRDs)。該helm upgrade命令沒有。如果helm upgrade,您使用，則必須手動安裝CRDs. 執行下列命令來安裝 CRDs：
>
> ```
> wget https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-controller/crds/crds.yaml
> kubectl apply -f crds.yaml
> ```

步驟 3：確認控制器已安裝

1. 確認控制器已安裝。

```
$ kubectl get deployment -n kube-system aws-load-balancer-controller
```

範例輸出如下。

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

如果使用 Helm 進行部署，則會收到先前的輸出。如果使用 Kubernetes 清單檔案進行部署，則您只有一個複本。

2. 使用控制器佈建 AWS 資源之前，您的叢集必須符合特定需求。如需更多詳細資訊，請參閱 Amazon EKS 上的應用程式負載平衡 及 Amazon EKS 上的網路負載平衡。

使用資Kubernetes訊清單安裝AWS Load Balancer Controller附加元件

本主題說明如何透過下載和套用資Kubernetes訊清單來安裝控制器。您可以檢視 GitHub 上的完整文件。

在下列步驟中，使用自己的值取代 *example values*。

必要條件

開始此教學之前，您必須先安裝和設定下列在建立和管理 Amazon EKS 叢集所需的工具和資源。

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。
- 叢集的現有 AWS Identity and Access Management OpenID Connect (IAMOIDC) () 提供者。若要判定您是否已經擁有一個，或是要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。
- 確認您的 Amazon VPC CNI plugin for Kubernetes、`kube-proxy` 及 CoreDNS 附加元件為服務帳戶字符中列出的最低版本。
- 熟悉 E AWS lastic Load Balancing 如需詳細資訊，請參閱《Elastic Load Balancing 使用者指南》。
- 熟悉 Kubernetes 服務和傳入資源。

步驟 1：設定身分與存取

> **Note**
>
> 您只需為每個 AWS 帳戶建立一AWS Load Balancer Controller個 IAM 角色。檢查 IAM 主控台中是否`AmazonEKSLoadBalancerControllerRole`存在。如果此角色存在，請跳至the section called “步驟 2：安裝 `cert-manager`”。

建立 IAM 政策。

1. 下載適用於 AWS Load Balancer Controller 的 IAM 政策，允許它代表您呼叫 AWS API。

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. 使用上一個步驟中下載的政策，建立 IAM 政策。

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

   > **Note**
   >
   > 如果您在中檢視原則 AWS Management Console，主控台會顯示 ELB 服務的警告，但不會顯示 ELB v2 服務的警告。發生這種情況是由於政策中存在適用於 ELB v2 的某些動作，但不適用於 ELB。您可以忽略這些對 ELB 發出的警告。

eksctl

使用建立 IAM 角色 **eksctl**

- 使用叢集名稱取代 *my-cluster*，並使用帳戶 ID 取代 *111122223333*，然後執行命令。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:`

  ```
  $ eksctl create iamserviceaccount \
    --cluster=my-cluster \
  ```

```
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

AWS CLI and kubectl

使用 AWS CLI 和建立 IAM 角色 **kubectl**

1. 擷取叢集的 OIDC 供應商 ID 並將其存放在變數中。

```
oidc_id=$(aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
```

2. 判斷您的帳戶中是否已經有擁有叢集 ID 的 IAM OIDC 供應商。您需要OIDC針對叢集和 IAM 進行設定。

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

如果有輸出傳回，則表示叢集已具備 IAM OIDC 提供者。如果未傳回任何輸出，則您必須為叢集建立 IAM OIDC 供應商。如需詳細資訊，請參閱 為您的叢集建立 IAM OIDC 提供者。

3. 將以下內容複製到您的裝置。使用您的帳戶 ID 取代 *111122223333*。*region-code*以叢集所 AWS 區域 在的位置取代。使用前一個步驟傳回的輸出取代 *EXAMPLED539D4633E53DE1B71EXAMPLE*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov: 取代文字後，請執行已修改的命令以建立 load-balancer-role-trust-policy.json 檔案。

```
cat >load-balancer-role-trust-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
```

```
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-load-balancer-controller"
                }
            }
        }
    ]
}
EOF
```

4. 建立 IAM 角色。

```
aws iam create-role \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --assume-role-policy-document file://"load-balancer-role-trust-policy.json"
```

5. 將必要的 Amazon EKS 受管 IAM 政策連接到 IAM 角色。使用您的帳戶 ID 取代 *111122223333*。

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --role-name AmazonEKSLoadBalancerControllerRole
```

6. 將以下內容複製到您的裝置。使用您的帳戶 ID 取代 *111122223333*。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:` 取代文字後，請執行已修改的命令以建立 `aws-load-balancer-controller-service-account.yaml` 檔案。

```
cat >aws-load-balancer-controller-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
```

```
  annotations:
    eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/AmazonEKSLoadBalancerControllerRole
EOF
```

7. 在您的叢集上建立 Kubernetes 服務帳戶。名為 aws-load-balancer-controller 的 Kubernetes 服務帳戶使用您建立名為 *AmazonEKSLoadBalancerControllerRole* 的 IAM 角色進行註釋。

```
$ kubectl apply -f aws-load-balancer-controller-service-account.yaml
```

步驟 2：安裝 **cert-manager**

使用以下其中一個方法安裝 cert-manager，將憑證組態注入 Webhook。如需詳細資訊，請參閱 *cert-manager*cert-manager 文件上的入門。

我們建議您使用quay.io容器登錄進行安裝cert-manager。如果您的節點無法存取quay.io容器登錄，請cert-manager使用 Amazon ECR 進行安裝 (請參閱以下說明)。

奎伊大作战

cert-manager使用 Quay.io 進行安裝

- 如果您的節點可以存取 quay.io 容器登錄檔，請安裝 cert-manager 以將憑證組態注入 Webhook。

```
$ kubectl apply \
    --validate=false \
    -f https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-
manager.yaml
```

Amazon ECR

cert-manager使用 Amazon ECR 安裝

1. 使用以下其中一個方法安裝 cert-manager，將憑證組態注入 Webhook。如需詳細資訊，請參閱 *cert-manager*cert-manager 文件上的入門。
2. 下載清單檔案。

```
curl -Lo cert-manager.yaml https://github.com/jetstack/cert-manager/releases/
download/v1.13.5/cert-manager.yaml
```

3. 提取以下映像並將其推送到節點可存取的儲存庫。有關如何提取、標記映像並將其推送到您儲存庫的詳細資訊，請參閱 將容器映像從一個儲存庫複製到另一個儲存庫。

```
quay.io/jetstack/cert-manager-cainjector:v1.13.5
quay.io/jetstack/cert-manager-controller:v1.13.5
quay.io/jetstack/cert-manager-webhook:v1.13.5
```

4. 使用您的登錄檔名稱取代清單檔案中三個映像的 quay.io。以下命令假定私有儲存庫的名稱與來源儲存庫的名稱相同。將 *111122223333.dkr.ecr.region-code.amazonaws.com* 取代為您的私有登錄檔。

```
$ sed -i.bak -e 's|quay.io|111122223333.dkr.ecr.region-code.amazonaws.com|' ./
cert-manager.yaml
```

5. 套用清單檔案。

```
$ kubectl apply \
    --validate=false \
    -f ./cert-manager.yaml
```

步驟 3：安裝 AWS Load Balancer Controller

AWS Load Balancer Controller使用Kubernetes資訊清單安裝

1. 下載控制器規格。如需控制器的詳細資訊，請參閱 GitHub 上的文件。

```
curl -Lo v2_7_2_full.yaml https://github.com/kubernetes-sigs/aws-load-balancer-
controller/releases/download/v2.7.2/v2_7_2_full.yaml
```

2. 對檔案進行以下編輯。

   a. 如果您已下載 v2_7_2_full.yaml 檔案，請執行下列命令以移除清單檔案中的 ServiceAccount 區段。如果您未移除此區段，則會覆寫您在前一個步驟中對服務帳戶所做的必要註解。如果您刪除控制器，移除此區段還可保留您在上一個步驟中建立的服務帳戶。

```
$ sed -i.bak -e '596,604d' ./v2_7_2_full.yaml
```

如果您下載不同的檔案版本，請在編輯器中開啟檔案並移除下列各行。

```
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
---
```

b. 使用叢集名稱取代 *my-cluster*，從而使用叢集名稱取代檔案的 Deployment spec 區段中的 your-cluster-name。

```
$ sed -i.bak -e 's|your-cluster-name|my-cluster|' ./v2_7_2_full.yaml
```

c. 如果您的節點無法存取 Amazon EKS Amazon ECR 映像儲存庫，則需要提取以下映像並將其推送到節點可存取的儲存庫。有關如何提取、標記映像並將其推送到您儲存庫的詳細資訊，請參閱 將容器映像從一個儲存庫複製到另一個儲存庫。

```
public.ecr.aws/eks/aws-load-balancer-controller:v2.7.2
```

將您的登錄檔名稱新增至清單檔案。以下命令假定私有儲存庫的名稱與來源儲存庫的名稱相同，並將您的私有登錄檔名稱新增至檔案中。將 *111122223333.dkr.ecr.region-code.amazonaws.com* 取代為您的登錄檔。此行假定您將私有儲存庫命名為與來源儲存庫相同的名稱。如果沒有，請將私有登錄檔名稱後的 eks/aws-load-balancer-controller 文字變更為儲存庫名稱。

```
$ sed -i.bak -e 's|public.ecr.aws/eks/aws-load-balancer-
controller|111122223333.dkr.ecr.region-code.amazonaws.com/eks/aws-load-
balancer-controller|' ./v2_7_2_full.yaml
```

d. （僅適用於 Fargate 或受限制 IMDS）

如果您要將控制器部署到限制存取 Amazon EC2 執行個體中繼資料服務 (IMDS) 的 Amazon EC2 節點，或者如果您正在部署到 Fargate，那麼請在 - args: 下新增 **following parameters**。

```
[...]
spec:
      containers:
        - args:
            - --cluster-name=your-cluster-name
            - --ingress-class=alb
            - --aws-vpc-id=vpc-xxxxxxxx
            - --aws-region=region-code


[...]
```

3. 套用檔案。

```
$ kubectl apply -f v2_7_2_full.yaml
```

4. 將 IngressClass 和 IngressClassParams 清單檔案下載至叢集。

```
$ curl -Lo v2_7_2_ingclass.yaml https://github.com/kubernetes-sigs/aws-load-balancer-controller/releases/download/v2.7.2/v2_7_2_ingclass.yaml
```

5. 將清單檔案套用至叢集。

```
$ kubectl apply -f v2_7_2_ingclass.yaml
```

步驟 4：確認控制器已安裝

1. 確認控制器已安裝。

```
$ kubectl get deployment -n kube-system aws-load-balancer-controller
```

範例輸出如下。

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

如果使用 Helm 進行部署，則會收到先前的輸出。如果使用 Kubernetes 清單檔案進行部署，則您只有一個複本。

2. 使用控制器佈建 AWS 資源之前，您的叢集必須符合特定需求。如需更多詳細資訊，請參閱 Amazon EKS 上的應用程式負載平衡 及 Amazon EKS 上的網路負載平衡。

從棄用的控制器遷移

本主題說明如何從已取代的控制器版本移轉。更具體地說，它描述了如何刪除已過時的版本AWS Load Balancer Controller。

- 已取代的版本無法升級。必須將它們移除並安裝目前版本的 LBC。
- 已過時的版本包括：
  - AWS ALB 入口控制器 Kubernetes (「入口控制器」)，這是. AWS Load Balancer Controller
  - 任何`0.1.x`版本的 AWS Load Balancer Controller

移除已淘汰的控制器

> **Note**
>
> 您可能已使用 Helm 或使用Kubernetes清單手動安裝已過時的版本。請使用最初使用安裝的工具來完成此程序。

使用頭盔移除入口控制器

1. 如果您安裝了 `incubator/aws-alb-ingress-controller` Helm Chart，請將其解除安裝。

```
$ helm delete aws-alb-ingress-controller -n kube-system
```

2. 如果您安裝了第 `0.1.x` 版的 `eks-charts/aws-load-balancer-controller` 圖表，請將其解除安裝。由於與 Webhook API 版本不相容，無法從 `0.1.x` 升級至第 `1.0.0` 版。

```
$ helm delete aws-load-balancer-controller -n kube-system
```

使用資訊清單移除入口控制器 Kubernetes

1. 檢查目前是否已安裝控制器。

```
$ kubectl get deployment -n kube-system alb-ingress-controller
```

如果未安裝控制器，則此為輸出。

伺服器發生錯誤 (NotFound): 找不到部署 .apps "" alb-ingress-controller

如果已安裝控制器，則此為輸出。

```
NAME                   READY UP-TO-DATE AVAILABLE AGE
alb-ingress-controller 1/1   1          1         122d
```

2. 輸入下列命令以移除控制器。

```
$ kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
ingress-controller/v1.1.8/docs/examples/alb-ingress-controller.yaml
kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
ingress-controller/v1.1.8/docs/examples/rbac-role.yaml
```

遷移至 AWS Load Balancer Controller

要從 ALB 入口控制器遷移Kubernetes到AWS Load Balancer Controller，您需要：

1. 移除 ALB 入口控制器 (請參閱上文)。
2. 安裝AWS Load Balancer Controller.
3. 將其他政策新增至 LBC 使用的 IAM 角色。此原則允許 LBC 管理由 ALB 入口控制器建立的資源。Kubernetes

將移轉政策新增至 AWS Load Balancer Controller IAM 角色。

1. 下載 IAM 政策。此原則允許 LBC 管理由 ALB 入口控制器建立的資源。Kubernetes您也可以檢視政策。

```
$ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/v2.7.2/docs/install/iam_policy_v1_to_v2_additional.json
```

2. 如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代`arn:aws:`為. `arn:aws-us-gov:` 將檔案為`arn:aws:`。

```
$ sed -i.bak -e 's|arn:aws:|arn:aws-us-gov:|' iam_policy_v1_to_v2_additional.json
```

3. 建立 IAM 政策並記下傳回的 ARN。

```
$ aws iam create-policy \
  --policy-name AWSLoadBalancerControllerAdditionalIAMPolicy \
  --policy-document file://iam_policy_v1_to_v2_additional.json
```

4. 將 IAM 政策附加到 LBC 使用的 IAM 角色。以角*your-role-name*色的名稱取代，例如AmazonEKSLoadBalancerControllerRole。

   如果您使用創建角色eksctl，然後找到創建的角色名稱，打開AWS CloudFormation 控制台並選擇 eksctl-*###*--addon-iamserviceaccount-kube-system 堆棧。aws-load-balancer-controller 選取 Resources (資源) 標籤。角色名稱位於 Physical ID (實體 ID) 欄。如果您的叢集位於 AWS GovCloud (美國東部) 或 AWS GovCloud (美國西部) AWS 區域，請取代arn:aws:為. arn:aws-us-gov:

```
$ aws iam attach-role-policy \
  --role-name your-role-name \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerAdditionalIAMPolicy
```

使用 CoreDNS Amazon EKS 附加元件

CoreDNS 是一個靈活、可擴展的 DNS 伺服器，可用來做為 Kubernetes 叢集 DNS。當您啟動具有至少一個節點的 Amazon EKS 叢集時，依預設會部署兩個 CoreDNS 映像複本，而不論叢集中部署的節點數量為何。CoreDNS Pods 為叢集中的所有 Pods 提供名稱解析。如果您的叢集包含使用命名空間與 CoreDNS deployment 的命名空間相匹配的 AWS Fargate 設定檔，則可將 CoreDNS Pods 部署至 Fargate 節點。如需 CoreDNS 的詳細資訊，請參閱 Kubernetes 文件中的使用 CoreDNS 進行服務探索。

下表列出適用於每個 Kubernetes 版本的 Amazon EKS 附加元件類型最新版本。

Kubernetes 版本	**1.30**	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
	v1.11.1-eksbuild	v1.11.1-eksbuild	v1.10.1-eksbuild.1	v1.10.1-eksbuild.1	v1.9.3-eksbuild.	v1.9.3-eksbuild.	v1.9.3-eksbuild.	v1.8.7-eksbuild.10

> **Important**
>
> 如果您要自行管理此附加元件，資料表中的版本可能與可用的自我管理版本不同。如需更新此附加元件之自我管理類型的詳細資訊，請參閱 更新自我管理的附加元件。

重要的 CoreDNS 升級考量

- 為了提高 CoreDNS Deployment 的穩定性與可用性，v1.9.3-eksbuild.6 及更新版本和 v1.10.1-eksbuild.3 隨 PodDisruptionBudget 一起部署。如果您已部署現有 PodDisruptionBudget，則可能無法升級至上述版本。如果升級失敗，完成下列其中一項任務即可解決問題：
  - 升級 Amazon EKS 附加元件時，選擇將覆寫現有設定作為衝突解決方案選項。如果您已對 Deployment 制定自訂設定，則務必在升級前備份設定，以便在升級後重新套用其他自訂設定。
  - 移除現有的 PodDisruptionBudget，然後再次嘗試升級。
- 在 EKS 附加元件版本以v1.9.3-eksbuild.3及更新版本v1.10.1-eksbuild.6和更新版本中，將CoreDNSDeployment設定readinessProbe為使用/ready端點。此端點已在的Corefile組態檔案中啟用CoreDNS。

  如果您使用 CustomCorefile，則必須將ready外掛程式新增至組態，以便在CoreDNS中使用該/ready端點才能使用探查。
- 在 EKS 附加元件版本以v1.9.3-eksbuild.7及更新版本v1.10.1-eksbuild.4和更新版本中，您可以變更 PodDisruptionBudget。您可以使用下列範例的欄位，在選擇性組態設定中編輯附加元件並變更這些設定。此範例顯示預設值PodDisruptionBudget。

```
{
    "podDisruptionBudget": {
        "enabled": true,
        "maxUnavailable": 1
        }
}
```

  您可以設置 maxUnavailable 或 minAvailable，但不能同時在單一PodDisruptionBudget進行設置。如需 PodDisruptionBudgets 的詳細資訊，請在Kubernetes 文件參閱指定 PodDisruptionBudget。

請注意，如果您設enabled定為false，則PodDisruptionBudget不會移除。將此欄位設定為false之後，您必須刪除PodDisruptionBudget物件。同樣地，如果您在升級至具有的版本之後編輯附加元件以使用較舊版本的附加元件 (降級附加元件)PodDisruptionBudget，則PodDisruptionBudget不會移除。執行下列命令以刪除 PodDisruptionBudget：

```
kubectl delete poddisruptionbudget coredns -n kube-system
```

- 在 EKS 附加元件版本v1.10.1-eksbuild.5及更新版本中，將預設容許範圍從變更node-role.kubernetes.io/master:NoSchedulenode-role.kubernetes.io/control-plane:NoSchedule為符合 KEP 2067。如需 KEP 2067 的詳細資訊，請參閱 GitHub 上 Kubernetes Enhancement Proposals (KEP) 中的 KEP-2067：重新命名 kubeadm「主要」標籤和污點。

  在 EKS 附加版本以v1.8.7-eksbuild.8及更新版本v1.9.3-eksbuild.9和更新版本中，兩個容許設置為與每個Kubernetes版本兼容。

- 在 EKS 附加元件版本v1.9.3-eksbuild.11v1.10.1-eksbuild.7和更新版本中，會CoreDNSDeployment設定的預設值。topologySpreadConstraints如果有多個可用區域中CoreDNSPods的節點可用，預設值可確保分散到可用區域。您可以設定將使用的自訂值，而非預設值。預設值如下：

```
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

CoreDNSv **1.11** 升級考量

- 在 EKS 附加版本v1.11.1-eksbuild.4和更高版本中，容器映像是以 Amazon EKS Distro 維護的最小基本映像為基礎，該映像包含最少的套件且沒有殼層。如需詳細資訊，請參閱 Amazon EKS Distro。CoreDNS圖像的使用和故障排除保持不變。

建立 Amazon EKS 附加元件

建立附加元件的 Amazon EKS 類型。Check

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。

1. 查看叢集上目前安裝了哪些附加元件版本。

```
kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
d : -f 3
```

範例輸出如下。

```
v1.10.1-eksbuild.11
```

2. 查看叢集上安裝的附加元件類型。視您用來建立叢集的工具而定，您的叢集上目前可能沒有安裝 Amazon EKS 附加元件類型。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

如果傳回版本編號，則表明已在叢集上安裝 Amazon EKS 類型的附加元件，並且無需完成此程序中的剩餘步驟。如果傳回錯誤，則表明沒有在叢集上安裝 Amazon EKS 類型的附加元件。完成此程序的剩餘步驟以安裝該類型。

3. 儲存您目前安裝的附加元件。

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

4. 使用 AWS CLI建立附加元件。如果您想要使用 AWS Management Console 或eksctl建立附加元件，請參閱建立附加元件並指定coredns附加元件名稱。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令。

   - 使用您叢集的名稱取代 *my-cluster*。
   - 將 *v1.11.1-eksbuild.9* 取代為您的叢集版本的最新版本表格中所列的最新版本。

```
aws eks create-addon --cluster-name my-cluster --addon-name coredns --addon-
version v1.11.1-eksbuild.9
```

如果您已將自訂設定套用至與 Amazon EKS 附加元件的預設設定衝突的目前附加元件，建立動作可能會失敗。若建立失敗，您會收到錯誤，其中的訊息有助於您解決問題。或者，您可以將 **--resolve-conflicts OVERWRITE** 新增至上一條命令。這可讓附加元件覆寫任何現有的自訂設定。建立附加元件後，可以使用自訂設定來更新該附加元件。

5. 確認叢集 Kubernetes 版本的附加元件的最新版本已新增至叢集。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

建立附加元件的動作可能需要幾秒鐘的時間才能完成。

範例輸出如下。

```
v1.11.1-eksbuild.9
```

6. 如果您對原始附加元件制定自訂設定，請在建立 Amazon EKS 附加元件之前，使用在上一步中儲存的組態，以您的自訂設定更新 Amazon EKS 附加元件。

更新 Amazon EKS 附加元件

更新 Amazon EKS 類型的附加元件。如果您尚未將 Amazon EKS 類型的附加元件新增至叢集，請新增該類型或查看 更新自我管理的附加元件，而不是完成此程序。

1. 查看叢集上目前安裝了哪些附加元件版本。使用您的叢集名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 "addon.addonVersion" --output text
```

範例輸出如下。

```
v1.10.1-eksbuild.11
```

如果傳回的版本與最新版本資料表中的叢集 Kubernetes 版本相同，則表明您已經在叢集上安裝最新版本，不需要完成此程序的剩餘步驟。若您收到錯誤而非版本編號，則表明叢集上沒有安裝 Amazon EKS 類型的附加元件。您必須先建立附加元件，才能使用此程序進行更新。

2. 儲存您目前安裝的附加元件。

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

3. 使用 AWS CLI更新您的附加元件。如果您想要使用 AWS Management Console 或eksctl更新附加元件，請參閱更新附加元件。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令。

   - 使用您叢集的名稱取代 *my-cluster*。
   - 將 *v1.11.1-eksbuild.9* 取代為您的叢集版本的最新版本表格中所列的最新版本。
   - **--resolve-conflicts** *PRESERVE* 選項會保留附加元件的現有組態值。如果對於附加元件設定，您已設定自訂值，但未使用此選項，Amazon EKS 會以其預設值覆寫您的值。如果您使用此選項，建議您在更新生產叢集上的附加元件之前，測試非生產叢集上的任何欄位和值變更。如果您將此值變更為 OVERWRITE，則所有設定都會變更為 Amazon EKS 預設值。如果您已設定任何設定的自訂值，則可能會使用 Amazon EKS 預設值覆寫這些值。如果您將此值變更為 none，Amazon EKS 不會變更任何設定的值，但更新可能會失敗。若更新失敗，您會收到錯誤訊息，以協助您解決衝突。
   - 如果您不更新組態設定，請從命令中移除 **--configuration-values '{*"replicaCount":3*}'**。如果您要更新組態設定，請以您要進行的設定取代 *"replicaCount":3*。在此範例中，CoreDNS 的複本數目設定為 3。您指定的值必須對組態結構描述有效。如果您不知道配置模式，請運行**aws eks describe-addon-configuration --addon-name coredns --addon-version *v1.11.1-eksbuild.9***並將 *v1.11.1-eksbuild.9* 替換為您要查看其配置的附加元件的版本號碼。結構描述會在輸出中傳回。如果您有任何現有的自訂組態，並且想要全部移除，同時將所有設定的值設定回 Amazon EKS 預設值，請從命令中移除 *"replicaCount":3*，這樣就會得到空白的 **{}**。如需有關 CoreDNS 設定的詳細資訊，請參閱 Kubernetes 文件中的自訂 DNS 服務。

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns --addon-
   version v1.11.1-eksbuild.9 \
       --resolve-conflicts PRESERVE --configuration-values '{"replicaCount":3}'
   ```

   更新動作可能需要幾秒鐘的時間才能完成。

4. 確認附加元件版本已更新。使用您叢集的名稱取代 *my-cluster*。

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns
   ```

   更新動作可能需要幾秒鐘的時間才能完成。

範例輸出如下。

```
{
    "addon": {
        "addonName": "coredns",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.11.1-eksbuild.9",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/coredns/
d2c34f06-1111-2222-1eb0-24f64ce37fa4",
        "createdAt": "2023-03-01T16:41:32.442000+00:00",
        "modifiedAt": "2023-03-01T18:16:54.332000+00:00",
        "tags": {},
        "configurationValues": "{\"replicaCount\":3}"
    }
}
```

更新自我管理的附加元件

> **Important**
>
> 建議將附加元件的 Amazon EKS 類型新增到叢集，而不是使用附加元件的自我管理類型。如果您不熟悉類型之間的差異，則請參閱 the section called “Amazon EKS 附加元件”。如需將 Amazon EKS 附加元件新增至叢集的詳細資訊，請參閱 the section called “建立附加元件”。如果您無法使用 Amazon EKS 附加元件，我們建議您提交有關為何無法存取容器藍圖GitHub 儲存庫的問題。

1. 確認您的叢集上已安裝附加元件的自我管理類型。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

如果傳回錯誤訊息，則表明您的叢集上安裝了附加元件的自我管理類型。完成此程序的剩餘步驟。如果傳回版本編號，則表明已在叢集上安裝附加元件的 Amazon EKS 類型。若要更新附加元件的

Amazon EKS 類型，請使用 更新 Amazon EKS 附加元件 中的程序，而不是使用此程序。如果您不熟悉附加元件類型之間的差異，請參閱 Amazon EKS 附加元件。

2. 查看叢集上目前安裝了哪些容器映像版本。

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

範例輸出如下。

```
v1.8.7-eksbuild.2
```

3. 如果您目前的 CoreDNS 版本為 `v1.5.0` 或更新版本，但比 CoreDNS 版本表格中列出的版本還要舊，則略過此步驟。如果您目前的版本比 `1.5.0` 還要舊，則需要修改 CoreDNS 的 `ConfigMap`，以使用正向附加元件，而不是代理附加元件。

   1. 使用下列命令開啟 ConfigMap。

      ```
      kubectl edit configmap coredns -n kube-system
      ```

   2. 使用 `forward` 取代為下列行中的 `proxy`：儲存檔案，然後退出編輯器。

      ```
      proxy . /etc/resolv.conf
      ```

4. 若您原先在 Kubernetes `1.17` 或舊版本上部署叢集，則您可能需要從您的 CoreDNS 清單檔案中移除已終止的行。

   > **Important**
   >
   > 您必須在升級至 CoreDNS 版本 `1.7.0` 之前完成此步驟，即使您正在升級至較舊版本，也建議您先完成此步驟。

   1. 檢查您的 CoreDNS 清單檔案是否有這一行。

      ```
      kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
       grep upstream
      ```

      如果沒有傳回任何輸出，則清單檔案沒有該行，您可以跳至下一個步驟來更新 CoreDNS。如果傳回輸出，那麼您需要移除該行。

2. 使用下列命令編輯 ConfigMap，移除檔案中有文字 upstream 的行。不要變更檔案中的任何其他內容。移除行之後，儲存變更。

```
kubectl edit configmap coredns -n kube-system -o yaml
```

5. 擷取您目前的 CoreDNS 映像版本：

```
kubectl describe deployment coredns -n kube-system | grep Image
```

範例輸出如下。

```
602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.8.7-eksbuild.2
```

6. 若要更新至 CoreDNS 1.8.3 或更新版本，您必須將 endpointslices 許可新增至 system:coredns Kubernetes clusterrole。

```
kubectl edit clusterrole system:coredns -n kube-system
```

在檔案的 rules 區段的現有許可行下，新增以下行。

```
[...]
- apiGroups:
  - discovery.k8s.io
  resources:
  - endpointslices
  verbs:
  - list
  - watch
[...]
```

7. 使用上一步中傳回的輸出值取代 *602401143452* 和 *region-code* 來更新 CoreDNS 附加元件。將 *v1.11.1-eksbuild.9* 取代為 Kubernetes 版本的最新版本資料表中列出的 CoreDNS 版本。

```
kubectl set image deployment.apps/coredns -n kube-system
 coredns=602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.11.1-
eksbuild.9
```

範例輸出如下。

```
deployment.apps/coredns image updated
```

8. 再次檢查容器映像版本，以確認其已更新至您在上一步中指定的版本。

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

範例輸出如下。

```
v1.11.1-eksbuild.9
```

自動調度 CoreDNS

當您啟動具有至少一個節點的 Amazon EKS 叢集時，依預設會部署CoreDNS映像的兩個Deployment複本中的其中一個，無論叢集中部署的節點數目為何。CoreDNS網繭會為叢集中的所有網繭提供名稱解析。應用程式會使用名稱解析來連線至叢集中的網繭和服務，以及連線至叢集外部的服務。隨著來自網繭的名稱解析 (查詢) 要求數量增加，CoreDNS網繭可能會不堪重負並減慢速度，並拒絕網繭無法處理的要求。

若要處理CoreDNS網繭上增加的負載，請考慮使用CoreDNS的自動調度資源系統。Amazon EKS 可以在的 EKS 附加元件版本中管理CoreDNS部署的自動調度資源。CoreDNS此CoreDNS自動配置器會持續監控叢集狀態，包括節點和 CPU 核心數。根據該資訊，控制器會動態調整 EKS 叢集中CoreDNS部署的複本數目。此功能適用於CoreDNSv1.9和 EKS 發行版本1.25及更高版本。如需有關哪些版本與CoreDNS自動調度資源相容的詳細資訊，請參閱下一節。

我們建議將此功能與其他 EKS 叢集自動調度資源最佳做法搭配使用，以改善整體應用程式可用性和叢集延展性。

必要條件

若要讓 Amazon EKS 擴展您的CoreDNS部署，有三個先決條件：

- 您必須使用的 EKS 附加元件版本的CoreDNS.
- 您的叢集至少必須執行最低叢集版本和平台版本。
- 您的叢集必須至少執行的 EKS 附加元件的最低版本。CoreDNS

最低叢集版本

的自動調度CoreDNS資源由叢集控制平面中的新元件完成，該元件由 Amazon EKS 管理。因此，您必須將叢集升級至支援具有新元件之最低平台版本的 EKS 版本。

新的 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。叢集必須是 Kubernetes 版 1.25 或更高版本。叢集必須執行下表或更新版Kubernetes本中列出的其中一個版本和平台版本。請注意，也支援比上列任何 Kubernetes 與平台版本更新的版本。您可使用叢集名稱取代下列命令的 *my-cluster*，然後執行修改的命令來檢查目前 Kubernetes 版本：

```
aws eks describe-cluster
              --name my-cluster --query cluster.version --output
              text
```

Kubernetes 版本	平台版本
1.29.3	eks.7
1.28.8	eks.13
1.27.12	eks.17
1.26.15	eks.18
1.25.16	eks.19

> **Note**
>
> 還支持更高版Kubernetes本的每個平台版本，例如來1.30自eks.1和開始的Kubernetes版本。

最低 EKS 附加版本

Kubernetes 版本	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**
	v1.11.1-e	v1.10.1-e	v1.10.1-e	v1.9.3-ek	v1.9.3-ek

Kubernetes 版本	1.29	1.28	1.27	1.26	1.25
	ksbuild.9	ksbuild.11	ksbuild.11	sbuild.15	sbuild.15

在中設定CoreDNS自動調度資源 AWS Management Console

1. 確定您的叢集已達到或高於最低叢集版本。

   Amazon EKS 會自動升級相同Kubernetes版本的平台版本之間的叢集，而您無法自行啟動此程序。相反地，您可以將叢集升級到下一個Kubernetes版本，叢集將升級至該 K8S 版本和最新的平台版本。例如，如果您從升級1.25到1.26，叢集將升級到1.26.15 eks.18。

   新的 Kubernetes 版本有時會加入重大變更。因此，建議您在更新生產叢集之前，先使用新Kubernetes版本的個別叢集來測試應用程式的行為。

   若要將叢集升級至新Kubernetes版本，請遵循中的程序更新 Amazon EKS 叢集 Kubernetes 版本。

2. 確保您擁有的 EKS 附加元件CoreDNS，而不是自我管理CoreDNS的部署。

   視您用來建立叢集的工具而定，您的叢集上目前可能沒有安裝 Amazon EKS 附加元件類型。若要查看叢集上安裝的附加元件類型，您可以執行下列命令。使用您叢集的名稱取代 my-cluster。

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   如果傳回版本號碼，您已在叢集上安裝附加元件的 Amazon EKS 類型，並且可以繼續執行下一個步驟。如果傳回錯誤，則表明沒有在叢集上安裝 Amazon EKS 類型的附加元件。完成程序的剩餘步驟，建立 Amazon EKS 附加元件以 Amazon EKS 附加元件取代自我管理版本。

3. 確保您的 EKS 附加元件CoreDNS的版本與最低 EKS 附加元件版本相同或更高。

   查看叢集上目前安裝了哪些附加元件版本。您可以簽入 AWS Management Console 或執行下列命令：

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

範例輸出如下。

```
v1.10.1-eksbuild.11
```

將此版本與上一節中的最低 EKS 附加版本進行比較。如有需要，請依照下列程序更新 Amazon EKS 附加元件將 EKS 附加元件升級至更高版本。

4. 將自動調度資源組態新增至 EKS 附加元件的選用組態設定。

   a. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。

   b. 在左側導覽窗格中，選取 Clusters (叢集)，然後選取您要為其設定附加元件之叢集的名稱。

   c. 選擇附加元件索引標籤。

   d. 選取CoreDNS附加元件方塊右上角的方塊，然後選擇 [編輯]。

   e. 在「設定 CoreDNS」頁面上：

      i. 選取您要使用的 Version (版本)。我們建議您保留與上一步相同的版本，並以單獨的動作更新版本和配置。

      ii. 展開選用組態設定。

      iii. 在組態值中輸入含索引鍵`"autoscaling":`和值的巢狀 JSON 物件的 JSON 金鑰`"enabled":`和值。產生的文字必須是有效的 JSON 物件。如果此金鑰和值是文字方塊中唯一的資料，請以大括號 `{}` 括住該金鑰和值。下列範例顯示已啟用自動調度資源：

```
{
  "autoScaling": {
    "enabled": true
  }
}
```

      iv. (選擇性) 您可以提供自動調度資源可以調整CoreDNS網繭數目的最小值和最大值。

      下列範例顯示已啟用自動調度資源，且所有選用索引鍵都有值。我們建議最小CoreDNS網繭數目永遠大於 2，以提供叢集中 DNS 服務的恢復能力。

```
{
  "autoScaling": {
    "enabled": true,
    "minReplicas": 2,
```

```
    "maxReplicas": 10
  }
}
```

f. 若要透過取代CoreDNS網繭套用新組態，請選擇 [儲存變更]。

Amazon EKS 會使用 CoreDNS 的Kubernetes部署部署，將變更套用至 EKS 附加元件。您可以在 AWS Management Console 和中的附加元件的更新歷史記錄中追蹤推出的狀態。`kubectl rollout status deployment/coredns --namespace kube-system`

`kubectl rollout`具有以下命令：

```
$ kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

如果部署時間太長，Amazon EKS 將撤銷推出，並在附加元件的更新歷史記錄中新增附加元件更新類型和失敗狀態的訊息。若要調查任何問題，請從部署的歷程記錄開始，然後在CoreDNS網繭`kubectl logs`上執行以查看的CoreDNS記錄。

5. 如果更新歷程記錄中的新項目狀態為「成功」，表示推出已完成，且附加元件正在使用所有CoreDNS網繭中的新組態。當您變更叢集中節點的節點數量和 CPU 核心時，Amazon EKS 會擴展部署的複本數量。CoreDNS

在中設定CoreDNS自動調度資源 AWS Command Line Interface

1. 確定您的叢集已達到或高於最低叢集版本。

Amazon EKS 會自動升級相同Kubernetes版本的平台版本之間的叢集，而您無法自行啟動此程序。相反地，您可以將叢集升級到下一個Kubernetes版本，叢集將升級至該 K8S 版本和最新的平台版本。例如，如果您從升級`1.25`到`1.26`，叢集將升級到`1.26.15 eks.18`。

新的 Kubernetes 版本有時會加入重大變更。因此，建議您在更新生產叢集之前，先使用新Kubernetes版本的個別叢集來測試應用程式的行為。

若要將叢集升級至新Kubernetes版本，請遵循中的程序更新 Amazon EKS 叢集 Kubernetes 版本。

2. 確保您擁有的 EKS 附加元件CoreDNS，而不是自我管理CoreDNS的部署。

   視您用來建立叢集的工具而定，您的叢集上目前可能沒有安裝 Amazon EKS 附加元件類型。若要查看叢集上安裝的附加元件類型，您可以執行下列命令。使用您叢集的名稱取代 `my-cluster`。

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   如果傳回版本編號，則表明已在叢集上安裝附加元件的 Amazon EKS 類型。如果傳回錯誤，則表明沒有在叢集上安裝 Amazon EKS 類型的附加元件。完成程序的剩餘步驟，建立 Amazon EKS 附加元件以 Amazon EKS 附加元件取代自我管理版本。

3. 確保您的 EKS 附加元件CoreDNS的版本與最低 EKS 附加元件版本相同或更高。

   查看叢集上目前安裝了哪些附加元件版本。您可以簽入 AWS Management Console 或執行下列命令：

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   範例輸出如下。

   ```
   v1.10.1-eksbuild.11
   ```

   將此版本與上一節中的最低 EKS 附加版本進行比較。如有需要，請依照下列程序更新 Amazon EKS 附加元件將 EKS 附加元件升級至更高版本。

4. 將自動調度資源組態新增至 EKS 附加元件的選用組態設定。

   執行下列 AWS CLI 命令。將 `my-cluster` 取代為您的叢集名稱，並將 IAM 角色 ARN 取代為您正在使用的角色。

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true}}'
   ```

Amazon EKS 會使用 CoreDNS 的Kubernetes部署部署，將變更套用至 EKS 附加元件。您可以在 AWS Management Console 和中的附加元件的更新歷史記錄中追蹤推出的狀態。`kubectl rollout status deployment/coredns --namespace kube-system`

`kubectl rollout`具有以下命令：

```
kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

如果部署時間太長，Amazon EKS 將撤銷推出，並在附加元件的更新歷史記錄中新增附加元件更新類型和失敗狀態的訊息。若要調查任何問題，請從部署的歷程記錄開始，然後在CoreDNS網繭`kubectl logs`上執行以查看的CoreDNS記錄。

5. (選擇性) 您可以提供自動調度資源可以調整CoreDNS網繭數目的最小值和最大值。

   下列範例顯示已啟用自動調度資源，且所有選用索引鍵都有值。我們建議最小CoreDNS網繭數目永遠大於 2，以提供叢集中 DNS 服務的恢復能力。

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true}, "minReplicas": 2, "maxReplicas": 10}'
   ```

6. 執行下列命令，以檢查附加元件的更新狀態：

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns \
   ```

   如果您看到此行:`"status": "ACTIVE"`，則表示推出已完成，且附加元件正在使用所有CoreDNS網繭中的新組態。當您變更叢集中節點的節點數量和 CPU 核心時，Amazon EKS 會擴展部署的複本數量。CoreDNS

CoreDNS 指標

CoreDNS 作為 EKS 附加元件，會在 kube-dns 服務中以 Prometheus 格式公開來自連接埠 9153 上 CoreDNS 的指標。您可以使用 Prometheus、Amazon CloudWatch 代理程式或任何其他相容的系統來湊集 (收集) 這些指標。

如需與 Prometheus 和 CloudWatch 代理程式相容的範例湊集組態，請參閱《Amazon CloudWatch 使用者指南》中的適用於 Prometheus 的 CloudWatch 代理程式組態。

使用庫伯尼特附kube-proxy加元件

> **Important**
>
> 建議將附加元件的 Amazon EKS 類型新增到叢集，而不是使用附加元件的自我管理類型。如果您不熟悉類型之間的差異，則請參閱 the section called “Amazon EKS 附加元件”。如需將 Amazon EKS 附加元件新增至叢集的詳細資訊，請參閱 the section called “建立附加元件”。如果您無法使用 Amazon EKS 附加元件，我們建議您提交有關為何無法存取容器藍圖GitHub 儲存庫的問題。

kube-proxy 附加元件會部署在 Amazon EKS 叢集中的每個 Amazon EC2 節點上。其會維護節點上的網路規則，並啟用與 Pods 的網路通訊。附加元件未部署至叢集中的 Fargate 節點。如需詳細資訊，請參閱 Kubernetes 文件中的 kube-proxy。

下表列出適用於每個 Kubernetes 版本的 Amazon EKS 附加元件類型最新版本。

Kubernetes 版本	1.30	1.29	1.28	1.27	1.26	1.25	1.24	1.23
	v1.30.0-eksbuild	v1.29.3-eksbuild	v1.28.8-eksbuild	v1.27.16-eksbuild.5	v1.26.15-eksbuild.5	v1.25.16-eksbuild.8	v1.24.17-eksbuild.8	v1.23.17-eksbuild.9

> **Important**
>
> 較早版本的文件不正確。 kube-proxy版本v1.28.5v1.27.9、和v1.26.12不可用。
> 如果您要自行管理此附加元件，資料表中的版本可能與可用的自我管理版本不同。

有兩種類型的 kube-proxy 容器映像可供每個 Amazon EKS 叢集版本使用：

- 預設：此映像類型的基礎為 Kubernetes 上游社群維護的 Debian 型 Docker 映像檔。
- 最低：此映像類型的基礎為 Amazon EKS Distro 維護的最低基礎映像，其中包含最少套件，且沒有 Shell。如需詳細資訊，請參閱 Amazon EKS Distro。

每個 Amazon EKS 叢集版本的最新可用自我管理 **kube-proxy** 容器映像版本

Image type (映像類型)	**1.30**	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
kube-proxy (預設類型)	只有最小的類型可用	只有最小的類型可用	只有最小的類型可用	只有最小的類型可用	只有最小的類型可用	只有最小的類型可用	v1.24.1-eksbuild.2	v1.23.16-eksbuild.2
kube-proxy (最小型)	v1.30.0-minimal-eksbuild.	v1.29.3-minimal-eksbuild.	v1.28.8-minimal-eksbuild.	v1.27.1-minimal-eksbuild	v1.26.1-minimal-eksbuild	v1.25.1-minimal-eksbuild	v1.24.1-minimal-eksbuild	v1.23.17-minimal-eksbuild.5

> **Important**
>
> - 預設影像類型不適用於 Kubernetes 版本 1.25 和更新版本。您必須使用最小的映像類型。
> - 更新 Amazon EKS 附加元件類型時，您必須指定有效的 Amazon EKS 附加元件版本，該版本可能不是此資料表中列出的版本。這是因為 Amazon EKS 附加元件版本不一定與更新此附加元件的自我管理類型時指定的容器映像版本相符。當您更新此附加元件的自我管理類型時，請指定此資料表中列出的有效容器映像版本。

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。

考量事項

- Amazon EKS 叢集上的 Kube-proxy 有與 Kubernetes 相同的相容性和差異政策。了解如何 檢索插件版本兼容性。
- Kube-proxy 的次要版本必須與 Amazon EC2 節點上的 kubelet 次要版本相同。
- Kube-proxy 不能比叢集控制平面的次要版本還要新。
- 如果您最近已將叢集更新至新的 Kubernetes 次要版本，請先將 Amazon EC2 節點更新至相同的次要版本，然後再將 kube-proxy 更新至與您的節點相同的次要版本。

更新 **kube-proxy** 自我管理的附加元件

1. 確認您的叢集上已安裝附加元件的自我管理類型。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name kube-proxy --query
 addon.addonVersion --output text
```

   如果傳回錯誤訊息，則表明您的叢集上安裝了附加元件的自我管理類型。本主題中的其餘步驟用於更新附加元件的自我管理類型。如果傳回版本編號，則表明已在叢集上安裝附加元件的 Amazon EKS 類型。若要將其更新，請使用 更新附加元件 中的程序，而不是使用本主題中的程序。如果您不熟悉附加元件類型之間的差異，請參閱 Amazon EKS 附加元件。

2. 查看叢集上目前安裝了哪些容器映像版本。

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image
```

   範例輸出如下。

```
Image:    602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.29.1-
eksbuild.2
```

   在範例輸出中，*v1.29.1-eksbuild.2* 是安裝在叢集上的版本。

3. 透過將 *602401143452* 和 *region-code* 取代為上一步輸出中的值來更新 kube-proxy 附加元件。將 *v1.30.0-eksbuild.3* 取代為每個 Amazon EKS 叢集版本的最新可用自我管理 kube-proxy 容器映像版本表中列出的 kube-proxy 版本。您可以為預設值或者最小影像類型指定版本編號。

```
kubectl set image daemonset.apps/kube-proxy -n kube-system kube-
proxy=602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.30.0-
eksbuild.3
```

範例輸出如下。

```
daemonset.apps/kube-proxy image updated
```

4. 確認您的叢集上現在已安裝新版本。

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image | cut -d ":" -f 3
```

範例輸出如下。

```
v1.30.0-eksbuild.3
```

5. 如果您在同一個叢集中使用 x86 和 Arm 節點，而且您的叢集是在 2020 年 8 月 17 日之前部署的。然後，編輯您的 kube-proxy 清單檔案，以包含多個硬體架構的節點選取器，並使用以下命令。這是一次性操作。將選取器新增至清單檔案後，不需要在每次更新附加元件時進行新增。如果您的叢集是在 2020 年 8 月 17 日或之後部署的，則 kube-proxy 已經具備多架構能力。

```
kubectl edit -n kube-system daemonset/kube-proxy
```

將下列節點選取器新增至編輯器中的檔案，然後儲存檔案。如需在編輯器中包含此文字的範例，請參閱 GitHub 上的 CNI manifest (CNI 清單檔案) 檔案。這樣一來，可讓 Kubernetes 根據節點的硬體架構提取正確的硬體映像。

```
- key: "kubernetes.io/arch"
  operator: In
  values:
  - amd64
  - arm64
```

6. 如果您的叢集最初是使用 Kubernetes 版本 1.14 或更新版本建立的，那麼您可以略過此步驟，原因在於 kube-proxy 已包含此 Affinity Rule。如果您最初使用 Kubernetes 版本 1.13 或更舊版本建立 Amazon EKS 叢集，並打算使用叢集中的 Fargate 節點，則請編輯 kube-proxy 清單檔案以包含 NodeAffinity 規則，進而防止 kube-proxy Pods 在 Fargate 節點上排程。這是

一次性編輯。將 Affinity Rule 新增至清單檔案後，不需要在每次更新附加元件時進行新增。編輯您的 kube-proxy DaemonSet。

```
kubectl edit -n kube-system daemonset/kube-proxy
```

新增下列項目 Affinity Rule 到編輯器的檔案中的 DaemonSet spec 部分，然後儲存檔案。如需在編輯器中包含此文字的範例，請參閱 GitHub 上的 CNI manifest (CNI 清單檔案) 檔案。

```
- key: eks.amazonaws.com/compute-type
  operator: NotIn
  values:
  - fargate
```

使用介面端點 (AWS PrivateLink) 存取 Amazon Elastic Kubernetes Service

您可以使 AWS PrivateLink 用在 VPC 和 Amazon Elastic Kubernetes Service 之間建立私有連接。您可以像在 VPC 中一樣存取 Amazon EKS，而無需使用網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線。VPC 中的執行個體無需公有 IP 地址，即可存取 Amazon EKS。

您可以建立由 AWS PrivateLink提供支援的介面端點來建立此私有連線。我們會在您為介面端點啟用的每個子網中建立端點網路介面。這些是請求者管理的網路介面，可作為目的地為 Amazon EKS 之流量的進入點。

如需詳細資訊，請參閱《AWS PrivateLink 指南》中的透過 AWS PrivateLink存取 AWS 服務。

Amazon EKS 的考量事項

- 在您為 Amazon EKS 設定介面端點之前，請檢閱《AWS PrivateLink 指南》中的「考量事項」。
- Amazon EKS 支援透過介面端點呼叫其所有 API 動作，但不支援呼叫 Kubernetes API。Kubernetes API 伺服器已支援私有端點。Kubernetes API 伺服器私有端點會為您用於與叢集進行通訊的 Kubernetes API 伺服器建立私有端點 (使用 Kubernetes 管理工具，例如 kubectl)。您可以啟用 Kubernetes API 伺服器的私有存取權，以便節點與 API 伺服器之間的所有通訊都保持在 VPC 中。AWS PrivateLink Amazon EKS API 可協助您從 VPC 呼叫 Amazon EKS API，而不會將流量暴露到公用網際網路。
- 您無法將 Amazon EKS 設定為只能透過介面端點存取。

- 適用於的標準定價 AWS PrivateLink 適用於 Amazon EKS 的界面端點。對於在每個可用區域中佈建的介面端點，以及透過介面端點處理的資料，都會按每小時向您收費。如需詳細資訊，請參閱 AWS PrivateLink 定價。
- Amazon EKS 不支援 VPC 端點政策。依預設，允許透過介面端點完整存取 Amazon EKS。或者，您也可以將安全群組與端點網路介面相關聯，以控制透過介面端點傳輸至 Amazon EKS 的流量。
- 您可以使用 VPC 流量日誌來擷取傳入和傳出網路介面之 IP 流量的相關資訊，包括介面端點。您可以將流程日誌資料發佈到 Amazon CloudWatch 或 Amazon S3。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用 VPC 流量日誌來記錄 IP 流量。
- 您可以將 Amazon EKS API 連線到擁有介面端點的 VPC，藉以從內部部署資料中心存取 Amazon EKS API。您可以使用 AWS Direct Connect 或 AWS Site-to-Site VPN 將內部部署網站連線到 VPC。
- 您可以使用 AWS Transit Gateway 或 VPC 對等互連，將其他 VPC 連線到具有介面端點的 VPC。VPC 對等互連是兩個 VPC 之間的網路連線。您可以在自己的 VPC 之間建立 VPC 對等互連，或與其他帳戶的 VPC 建立對等互連。VPC 可以是不同 AWS 區域的。對等 VPC 之間的流量會保留在網路上。 AWS 流量不會周遊公有網際網路。傳輸閘道是網路傳輸中樞，您可以用於互相連接 VPC。VPC 和傳輸閘道之間的流量保持在 AWS 全域私有網路上。流量不會向公有網際網路公開。
- Amazon EKS 的 VPC 介面端點只能透過 `IPv4` 存取。不支援 `IPv6`。
- AWS PrivateLink 亞太區域 (海德拉巴)、亞太區域 (墨爾本)、亞太區域 (大阪)、加拿大西部 (卡加利)、歐洲 (西班牙)、歐洲 (蘇黎世) 或中東 (阿拉伯聯合大公國) 不提供支援服務 AWS 區域。

建立 Amazon VPC 的介面端點

您可以使用 Amazon VPC 主控台或 AWS Command Line Interface ()AWS CLI為 Amazon EKS 建立介面端點。如需詳細資訊，請參閱《AWS PrivateLink 指南》中的建立 VPC 端點。

使用以下服務名稱為 Amazon EKS 建立介面端點：

```
com.amazonaws.region-code.eks
```

建立 Amazon EKS 和其他 AWS 服務的介面端點時，預設為啟用私有 DNS 功能。但是，您必須確保將下列 VPC 屬性設為 `true`：`enableDnsHostnames` 和 `enableDnsSupport`。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的檢視和更新 VPC 的 DNS 屬性。在介面端點啟用私有 DNS 功能時：

- 您可以使用其預設區域 DNS 名稱對 Amazon EKS 進行任何 API 請求。例如 `eks.region.amazonaws.com`。如需 API 清單，請參閱《Amazon EKS API 參考》中的動作。

- 您無需對呼叫 EKS API 的應用程式進行任何變更。
- 對 Amazon EKS 預設服務端點進行的任何呼叫，都會透過私有網路透過界面端點自動 AWS 路由傳送。

工作負載

您的工作負載會部署在容器中，這些容器會部署在 Kubernetes 的 Pods 中。Pod 包含一或多個容器。一般而言，提供相同服務的一或多個 Pods 會部署在 Kubernetes 服務中。部署多個提供相同服務的 Pods 後，您可以：

- 使用 AWS Management Console 檢視在每個叢集上執行之工作負載的相關資訊。
- 使用 Kubernetes Vertical Pod Autoscaler 將 Pods 縱向擴展或縮減規模。
- 水平調整所需的 Pods 數量，以滿足使用 Kubernetes Horizontal Pod Autoscaler 進行縮放的需求。
- 建立外部 (針對網際網路可存取的 Pods) 或內部 (針對私有 Pods) Network Load Balancer 來平衡 Pods 間的網路流量。負載平衡器會在 OSI 模型的第 4 層路由流量。
- 建立 Amazon EKS 上的應用程式負載平衡 來平衡 Pods 間的應用程式流量。Application Load Balancer 會在 OSI 模型的第 7 層路由流量。
- 若您是初次使用 Kubernetes ，則此主題可協助您 部署範例應用程式。
- 您可以使用 `externalIPs` 限制可指派給服務的 IP 地址。

部署範例應用程式

在這個主題中，您要部署範例應用程式至叢集。

必要條件

- 一個現有的 Kubernetes 叢集至少會有一個節點。如果尚無現有 Amazon EKS 叢集，可以使用 Amazon EKS 入門 指南之一部署叢集。如果您正在部署 Windows 應用程式，則必須為叢集啟用 Windows 支援，以及啟用至少一個 Amazon EC2 Windows 節點。
- 安裝在電腦上的 `Kubectl`。如需詳細資訊，請參閱 安裝或更新 `kubectl`。
- 設定好的 `Kubectl`，以便與叢集通訊。如需詳細資訊，請參閱 建立或更新 Amazon EKS 叢集的 `kubeconfig` 檔案。
- 如果打算將範例工作負載部署至 Fargate，則必須擁有現有的 Fargate 設定檔，其中包含在本教學課程中建立的相同命名空間，也就是 `eks-sample-app`，除非您變更名稱。如果使用其中一個入門指南來建立叢集，則必須建立新的設定檔，或將命名空間新增至現有設定檔，因為在入門指南中建立的設定檔不會指定本教學課程中使用的命名空間。您的 VPC 也必須至少有一個私有子網路。

部署範例應用程式

雖然許多變數可在下列步驟中變更，但建議只在指定的位置變更變值。更加了解 Kubernetes Pods、部署和服務後，您可以嘗試變更其他值。

1. 建立 命名空間。命名空間可讓您將 Kubernetes 中的資源分組。如需詳細資訊，請參閱 Kubernetes 文件中的命名空間。如果打算將範例應用程式部署至 AWS Fargate，請確定您 AWS Fargate 設定檔 中的 namespace 值是 eks-sample-app。

   ```
   kubectl create namespace eks-sample-app
   ```

2. 建立 Kubernetes 部署。此範例部署會從公有儲存庫提取容器映像，並將其三個複本 (個別 Pods) 部署至您的叢集。若要進一步了解，請參閱 Kubernetes 文件中的部署。您可以將應用程式部署至 Linux 或 Windows 節點。如果要部署至 Fargate，則只能部署 Linux 應用程式。

   a. 將下列內容儲存到名為 eks-sample-deployment.yaml 的檔案中。範例應用程式中的容器不會使用網路儲存，但您可能有需要使用網路儲存的應用程式。如需詳細資訊，請參閱 儲存。

      Linux

      在 kubernetes.io/arch 金鑰下方的 amd64 或 arm64 values 代表可以部署應用程式至其中一個硬體架構 (如果叢集中有兩個)。此操作可行，因為此映像是一個多架構映像，但並非所有映像都是。透過檢視要從中提取映像的儲存庫中的映像詳細資訊，可以確定支援該映像的硬體架構。部署不支援硬體架構類型的映像時，或者不希望部署映像時，請從清單檔案中移除該類型。如需詳細資訊，請參閱 Kubernetes 文件中的已知標籤、註釋和污點。

      kubernetes.io/os: linux nodeSelector 表示，如果叢集中有 Linux 和 Windows 節點 (舉例來說)，則該映像只會部署至 Linux 節點。如需詳細資訊，請參閱 Kubernetes 文件中的已知標籤、註釋和污點。

      ```
      apiVersion: apps/v1
      kind: Deployment
      metadata:
        name: eks-sample-linux-deployment
        namespace: eks-sample-app
        labels:
          app: eks-sample-linux-app
      spec:
        replicas: 3
      ```

```
  selector:
    matchLabels:
      app: eks-sample-linux-app
  template:
    metadata:
      labels:
        app: eks-sample-linux-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: kubernetes.io/arch
                operator: In
                values:
                - amd64
                - arm64
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
      nodeSelector:
        kubernetes.io/os: linux
```

Windows

`kubernetes.io/os: windows` nodeSelector 表示，如果叢集中有 Windows 和 Linux 節點 (舉例來說)，則該映像只會部署至 Windows 節點。如需詳細資訊，請參閱 Kubernetes 文件中的已知標籤、註釋和污點。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-windows-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  replicas: 3
```

```
  selector:
    matchLabels:
      app: eks-sample-windows-app
  template:
    metadata:
      labels:
        app: eks-sample-windows-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: beta.kubernetes.io/arch
                operator: In
                values:
                - amd64
      containers:
      - name: windows-server-iis
        image: mcr.microsoft.com/windows/servercore:ltsc2019
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
        command:
        - powershell.exe
        - -command
        - "Add-WindowsFeature Web-Server; Invoke-WebRequest -UseBasicParsing
 -Uri 'https://dotnetbinaries.blob.core.windows.net/servicemonitor/2.0.1.6/
ServiceMonitor.exe' -OutFile 'C:\\ServiceMonitor.exe'; echo
 '<html><body><br/><br/><marquee><H1>Hello EKS!!!<H1><marquee></body><html>'
 > C:\\inetpub\\wwwroot\\default.html; C:\\ServiceMonitor.exe 'w3svc'; "
      nodeSelector:
        kubernetes.io/os: windows
```

b. 將部署清單檔案套用至叢集。

```
kubectl apply -f eks-sample-deployment.yaml
```

3. 建立服務。服務可讓您透過單一 IP 地址或名稱存取所有複本。如需詳細資訊，請參閱 Kubernetes 文件中的服務。雖然未在範例應用程式中實作，但如果您的應用程式需要與其他 AWS 服務互動，建議您為您建立Kubernetes服務帳戶Pods，並將其關聯至 AWS IAM 帳戶。指定服務帳戶後，您

的 Pods 僅能擁有您為它們指定與其他服務互動所需的最低許可。如需詳細資訊，請參閱 服務帳戶的 IAM 角色。

a. 將下列內容儲存到名為 `eks-sample-service.yaml` 的檔案。Kubernetes 會為服務指派自己的 IP 地址，這些地址只能從叢集內存取。若要從叢集外部存取服務，請部署 AWS Load Balancer Controller，以將應用程式或網路流量負載平衡至服務。

Linux

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-linux-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-linux-app
spec:
  selector:
    app: eks-sample-linux-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Windows

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-windows-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  selector:
    app: eks-sample-windows-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

b. 將服務清單檔案套用至叢集。

```
kubectl apply -f eks-sample-service.yaml
```

4. 檢視 eks-sample-app 命名空間中存在的所有資源。

```
kubectl get all -n eks-sample-app
```

範例輸出如下。

如果部署了 Windows 資源，則下列輸出中的所有 *linux* 執行個體都為 windows。其他 *example values* 可能與輸出不同。

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/eks-sample-linux-deployment-65b7669776-m6qxz     1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-mmxvd     1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-qzn22     1/1     Running   0          27m

NAME                                  TYPE         CLUSTER-IP      EXTERNAL-IP
 PORT(S)   AGE
service/eks-sample-linux-service      ClusterIP    10.100.74.8     <none>        80/
TCP    32m

NAME                                            READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/eks-sample-linux-deployment     3/3     3            3           27m

NAME                                                       DESIRED   CURRENT   READY
   AGE
replicaset.apps/eks-sample-linux-deployment-776d8f8fd8     3         3         3
   27m
```

在輸出中，您會看到在先前步驟中所部署範例清單檔案中指定的服務和部署。您也會看到三個 Pods。這是因為範例清單檔案中指定了 3 個 replicas。如需有關 Pods 的詳細資訊，請參閱 Kubernetes 文件中的 Pod。Kubernetes 會自動創建 replicaset 資源，即使未在範例清單檔案指定也一樣。如需相關資訊ReplicaSets，請參閱Kubernetes文件ReplicaSet中的。

> **Note**
>
> Kubernetes 會維護清單檔案中指定的複本數量。如果這是生產部署，並且您希望 Kubernetes 橫向擴展複本數量或垂直擴展 Pods 的運算資源，請使用 Horizontal Pod Autoscaler 與 Vertical Pod Autoscaler 來執行此作業。

5. 檢視已部署服務的詳細資訊。如果已部署 Windows 服務，請使用 **windows** 取代 *linux*。

```
kubectl -n eks-sample-app describe service eks-sample-linux-service
```

範例輸出如下。

如果部署了 Windows 資源，則下列輸出中的所有 *linux* 執行個體都為 windows。其他 *example values* 可能與輸出不同。

```
Name:              eks-sample-linux-service
Namespace:         eks-sample-app
Labels:            app=eks-sample-linux-app
Annotations:       <none>
Selector:          app=eks-sample-linux-app
Type:              ClusterIP
IP Families:       <none>
IP:                10.100.74.8
IPs:               10.100.74.8
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         192.168.24.212:80,192.168.50.185:80,192.168.63.93:80
Session Affinity:  None
Events:            <none>
```

在先前的輸出中，IP: 值是可從叢集內的任一個節點或 Pod 連線的唯一 IP 地址，但無法從叢集外部連線。Endpoints 值是從 VPC 內指派至服務一部分的 Pods 的 IP 地址。

6. 在先前步驟中檢視命名空間時，請檢視輸出中列出的其中一個 Pods 詳細資訊。如果部署了 Windows 應用程式，請使用 **windows** 取代 *linux*，並用您的 Pods 中一個返回的值取代 *776d8f8fd8-78w66*。

```
kubectl -n eks-sample-app describe pod eks-sample-linux-deployment-65b7669776-m6qxz
```

縮寫輸出

如果部署了 Windows 資源，則下列輸出中的所有 *linux* 執行個體都為 windows。其他 *example values* 可能與輸出不同。

```
Name:         eks-sample-linux-deployment-65b7669776-m6qxz
Namespace:    eks-sample-app
Priority:     0
```

```
Node:         ip-192-168-45-132.us-west-2.compute.internal/192.168.45.132
[...]
IP:           192.168.63.93
IPs:
  IP:           192.168.63.93
Controlled By:  ReplicaSet/eks-sample-linux-deployment-65b7669776
[...]
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
[...]
Events:
  Type    Reason     Age    From
 Message
  ----    ------     ----   ----
 -------
  Normal  Scheduled  3m20s  default-scheduler
 Successfully assigned eks-sample-app/eks-sample-linux-deployment-65b7669776-m6qxz
 to ip-192-168-45-132.us-west-2.compute.internal
[...]
```

在先前的輸出中，IP: 的值是唯一的 IP，從指派給節點所在子網路的 CIDR 區塊指派給 Pod。如果希望從不同的 CIDR 區塊指派 Pods IP 地址，則可以變更預設行為。如需詳細資訊，請參閱 Pod 的自訂聯網。您也可以看到 Kubernetes 排程器使用 IP 地址 *192.168.45.132* 排定了 Node 上的 Pod。

> **Tip**
>
> 若不使用命令列，您可以在 AWS Management Console中檢視有關 Pods、服務、部署和其他 Kubernetes 資源的許多詳細資訊。如需詳細資訊，請參閱 檢視 Kubernetes 資源。

7. 在先前步驟中描述的 Pod 上執行 Shell，使用您 Pods 中的一個 ID 取代 *65b7669776-m6qxz*。

   Linux

   ```
   kubectl exec -it eks-sample-linux-deployment-65b7669776-m6qxz -n eks-sample-app
    -- /bin/bash
   ```

Windows

```
kubectl exec -it eks-sample-windows-deployment-65b7669776-m6qxz -n eks-sample-app -- powershell.exe
```

8. 在 Pod Shell 中，檢視在先前步驟中與部署一起安裝的 Web 伺服器輸出。您只需要指定服務名稱。CoreDNS 會將服務名稱解析為服務的 IP 地址；其預設會與 Amazon EKS 叢集一起部署。

   Linux

   ```
   curl eks-sample-linux-service
   ```

   範例輸出如下。

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```

   Windows

   ```
   Invoke-WebRequest -uri eks-sample-windows-service/default.html -UseBasicParsing
   ```

   範例輸出如下。

   ```
   StatusCode        : 200
   StatusDescription : OK
   Content           : < h t m l > < b o d y > < b r / > < b r / > < m a r q u e e
    > < H 1 > H e l l o
                       E K S ! ! ! < H 1 > < m a r q u e e > < / b o d y > < h t
    m l >
   ```

9. 從 Pod Shell 中檢視 Pod 的 DNS 伺服器。

   Linux

   ```
   cat /etc/resolv.conf
   ```

範例輸出如下。

```
nameserver 10.100.0.10
search eks-sample-app.svc.cluster.local svc.cluster.local cluster.local us-
west-2.compute.internal
options ndots:5
```

在先前的輸出中，10.100.0.10 會自動指派為讓所有 Pods 部署至叢集的 nameserver。

Windows

```
Get-NetIPConfiguration
```

縮寫輸出

```
InterfaceAlias       : vEthernet
[...]
IPv4Address          : 192.168.63.14
[...]
DNSServer            : 10.100.0.10
```

在先前的輸出中，10.100.0.10 會自動指派為讓所有 Pods 部署至叢集的 DNS 伺服器。

10. 透過輸入 exit 中斷從 Pod 的連線。
11. 完成範例應用程式後，您可以使用以下命令來移除範例命名空間、服務和部署。

```
kubectl delete namespace eks-sample-app
```

後續步驟

部署範例應用程式之後，您可能想要嘗試下列其中一些練習：

- the section called “應用程式負載平衡”
- the section called “網路負載平衡”

Vertical Pod Autoscaler

KubernetesVertical Pod Autoscaler 可自動調整保留給 Pods 的 CPU 和記憶體，以協助保持應用程式的「適當大小」。這項調整可以改善叢集資源使用率，並釋放 CPU 和記憶體給其他 Pods。本主題協助您將 Vertical Pod Autoscaler 部署到您的叢集，並驗證是否正常運作。

必要條件

- 您擁有現有的 Amazon EKS 叢集。如果您沒有，則請參閱 Amazon EKS 入門。
- 您已安裝 Kubernetes 指標伺服器。如需詳細資訊，請參閱安裝 Kubernetes 指標伺服器。
- 您所使用的 `kubectl` 用戶端已設定將與您的 Amazon EKS 叢集通訊。
- 裝置上已安裝 OpenSSL `1.1.1` 或更新版本。

部署 Vertical Pod Autoscaler

在本節中，您會將 Vertical Pod Autoscaler 部署到叢集。

部署 Vertical Pod Autoscaler

1. 開啟終端機視窗，導覽至您要下載 Vertical Pod Autoscaler 原始程式碼的目錄。
2. 複製 kubernetes/autoscaler GitHub 儲存庫。

```
git clone https://github.com/kubernetes/autoscaler.git
```

3. 切換至 `vertical-pod-autoscaler` 目錄。

```
cd autoscaler/vertical-pod-autoscaler/
```

4. (選用) 如果您已部署另一個版本的 Vertical Pod Autoscaler，請使用下列命令將其移除。

```
./hack/vpa-down.sh
```

5. 如果您的節點對 `registry.k8s.io` 容器登錄檔無網際網路存取權，則您需要提取以下映像，並將其推送到您的私有儲存庫。如需有關如何提取映像以及將映像推送到您的私有儲存庫的詳細資訊，請參閱 將容器映像從一個儲存庫複製到另一個儲存庫。

```
registry.k8s.io/autoscaling/vpa-admission-controller:0.10.0
registry.k8s.io/autoscaling/vpa-recommender:0.10.0
```

```
registry.k8s.io/autoscaling/vpa-updater:0.10.0
```

如果要將映像推送到私有 Amazon ECR 儲存庫，請使用您的登錄檔取代清單檔案中的 `registry.k8s.io`。使用您的帳戶 ID 取代 *111122223333*。使用叢集所在的 AWS 區域 取代 *region-code*。以下命令假定您將自己的儲存庫命名為與清單檔案中儲存庫相同的名稱。如果您將自己的儲存庫命名為其他名稱，您也需要變更清單檔案中儲存庫的名稱。

```
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/admission-controller-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/recommender-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/updater-deployment.yaml
```

6. 使用下列命令，將 Vertical Pod Autoscaler 部署到您的叢集。

```
./hack/vpa-up.sh
```

7. 確認已成功建立 Vertical Pod Autoscaler Pods。

```
kubectl get pods -n kube-system
```

範例輸出如下。

```
NAME                                        READY   STATUS    RESTARTS   AGE
[...]
metrics-server-8459fc497-kfj8w              1/1     Running   0          83m
vpa-admission-controller-68c748777d-ppspd   1/1     Running   0          7s
vpa-recommender-6fc8c67d85-gljpl            1/1     Running   0          8s
vpa-updater-786b96955c-bgp9d                1/1     Running   0          8s
```

測試您的 Vertical Pod Autoscaler 安裝

在本節中，您將部署範例應用程式，以驗證 Vertical Pod Autoscaler 是否正常運作。

測試您的 Vertical Pod Autoscaler 安裝

1. 使用以下命令部署 `hamster.yaml` Vertical Pod Autoscaler 範例。

```
kubectl apply -f examples/hamster.yaml
```

2. 從 hamster 範例應用程式取得 Pods。

```
kubectl get pods -l app=hamster
```

範例輸出如下。

```
hamster-c7d89d6db-rglf5   1/1     Running   0          48s
hamster-c7d89d6db-znvz5   1/1     Running   0          48s
```

3. 描述其中一個 Pods，以檢視其 cpu 和 memory 的保留情況。使用在上一個步驟傳回的輸出中其中一個 ID 來取代 *c7d89d6db-rglf5*。

```
kubectl describe pod hamster-c7d89d6db-rglf5
```

範例輸出如下。

```
[...]
Containers:
  hamster:
    Container ID:  docker://
e76c2413fc720ac395c33b64588c82094fc8e5d590e373d5f818f3978f577e24
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:35:16 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        100m
      memory:     50Mi
```

```
[...]
```

您可以看到原始 Pod 保留 100 millicpu 的 CPU 和 50 MiB 的記憶體。在這個範例應用程式中，100 millicpu 小於 Pod 執行所需，因此 CPU 受到限制。保留的記憶體也遠少於所需。Vertical Pod Autoscaler `vpa-recommender` 部署會分析 `hamster` Pods，以檢查 CPU 和記憶體需求是否合適。若需要調整，`vpa-updater` 會以更新的值重新啟動 Pods。

4. 等待 `vpa-updater` 啟動新的 `hamster` Pod。這需要一兩分鐘。您可以使用下列命令來監控 Pods。

> **Note**
>
> 若您不確定新的 Pod 是否已啟動，請將 Pod 名稱與之前的清單互相比較。當新的 Pod 啟動時，您會看到新的 Pod 名稱。

```
kubectl get --watch Pods -l app=hamster
```

5. 當新的 `hamster` Pod 啟動時，請對其做出描述並檢視更新的 CPU 和記憶體保留。

```
kubectl describe pod hamster-c7d89d6db-jxgfv
```

範例輸出如下。

```
[...]
Containers:
  hamster:
    Container ID:
 docker://2c3e7b6fb7ce0d8c86444334df654af6fb3fc88aad4c5d710eac3b1e7c58f7db
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:37:08 -0700
```

```
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        587m
      memory:     262144k
[...]
```

在先前的輸出中，您可以看到 cpu 保留已增加到 587 millicpu，這已是原始值的 5 倍。memory 已增加到 262,144 KB，大約是 250 MiB 或原始值的 5 倍。此 Pod 的資源不足，Vertical Pod Autoscaler 已依照更適當的值來更正預估值。

6. 描述 hamster-vpa 資源以檢視新的建議。

```
kubectl describe vpa/hamster-vpa
```

範例輸出如下。

```
Name:         hamster-vpa
Namespace:    default
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"autoscaling.k8s.io/
v1beta2","kind":"VerticalPodAutoscaler","metadata":{"annotations":
{},"name":"hamster-vpa","namespace":"d...
API Version:  autoscaling.k8s.io/v1beta2
Kind:         VerticalPodAutoscaler
Metadata:
  Creation Timestamp:  2019-09-27T18:22:51Z
  Generation:          23
  Resource Version:    14411
  Self Link:           /apis/autoscaling.k8s.io/v1beta2/namespaces/default/
verticalpodautoscalers/hamster-vpa
  UID:                 d0d85fb9-e153-11e9-ae53-0205785d75b0
Spec:
  Target Ref:
    API Version:  apps/v1
    Kind:         Deployment
    Name:         hamster
Status:
  Conditions:
    Last Transition Time:  2019-09-27T18:23:28Z
    Status:                True
```

```
    Type:                   RecommendationProvided
  Recommendation:
    Container Recommendations:
      Container Name:  hamster
      Lower Bound:
        Cpu:     550m
        Memory:  262144k
      Target:
        Cpu:     587m
        Memory:  262144k
      Uncapped Target:
        Cpu:     587m
        Memory:  262144k
      Upper Bound:
        Cpu:     21147m
        Memory:  387863636
Events:          <none>
```

7. 當您完成範例應用程式的實驗後，您可以使用下列命令將其刪除。

```
kubectl delete -f examples/hamster.yaml
```

Horizontal Pod Autoscaler

Kubernetes Horizontal Pod Autoscaler 會根據資源的 CPU 使用率，自動調整部署、複寫控制器或複本集中的 Pods 裝置數量。這可協助您的應用程式水平擴展以滿足增加的需求，或在不需要資源時縮減，從而釋出節點供其他應用程式使用。當您設定目標 CPU 使用率百分比時，Horizontal Pod Autoscaler 會擴展或縮減您的應用程式，以嘗試滿足該目標。

Horizontal Pod Autoscaler 是 Kubernetes 中的標準 API 資源，只需要在 Amazon EKS 叢集上安裝指標來源 (例如 Kubernetes 指標伺服器) 即可運作。您不需要在叢集上部署或安裝 Horizontal Pod Autoscaler，即可開始擴展您的應用程式。如需詳細資訊，請參閱 Kubernetes 文件中的 Horizontal Pod Autoscaler。

使用此主題為您的 Amazon EKS 叢集準備 Horizontal Pod Autoscaler，並以範例應用程式驗證是否正常運作。

> **Note**
>
> 此主題是根據 Kubernetes 文件中的 Horizontal Pod autoscaler 演練。

必要條件

- 您擁有現有的 Amazon EKS 叢集。如果您沒有，則請參閱 Amazon EKS 入門。
- 您已安裝 Kubernetes 指標伺服器。如需詳細資訊，請參閱安裝 Kubernetes 指標伺服器。
- 您所使用的 `kubectl` 用戶端已設定將與您的 Amazon EKS 叢集通訊。

執行 Horizontal Pod Autoscaler 測試應用程式

在本節中，您將部署範例應用程式，以驗證 Horizontal Pod Autoscaler 是否正常運作。

> **Note**
>
> 此範例是根據 Kubernetes 文件中的 Horizontal Pod Autoscaler 演練。

測試您的 Horizontal Pod Autoscaler 安裝

1. 使用下列命令部署簡單的 Apache Web 伺服器應用程式。

   ```
   kubectl apply -f https://k8s.io/examples/application/php-apache.yaml
   ```

   此 Apache Web 伺服器 Pod 獲得 500 millicpu CPU 的限制，而在連接埠 80 上提供服務。
2. 建立 `php-apache` 部署的 Horizontal Pod Autoscaler 資源。

   ```
   kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
   ```

   此命令會建立自動擴展器，其目標為 50% CPU 使用率用於部署，以及至少一個 Pod 和最多十個 Pods。當平均 CPU 負載低於 50% 時，自動擴展器會嘗試降低部署中的 Pods 裝置數量，最少降低至一個。當負載大於 50% 時，自動擴展器會嘗試增加部署中的 Pods 裝置數量，最多十個。如需詳細資訊，請參閱如何 HorizontalPodAutoscaler 運作？ 在文Kubernetes檔中。
3. 使用以下命令描述自動擴展器，以檢視其詳細資訊。

   ```
   kubectl get hpa
   ```

   範例輸出如下。

   ```
   NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
   ```

```
php-apache   Deployment/php-apache   0%/50%    1         10        1          51s
```

如您所見，目前的 CPU 負載是 0%，因為伺服器上還沒有負載。Pod 數量已在其最低底限 (一)，因此無法縮減。

4. 透過執行容器來建立 Web 伺服器的負載。

```
kubectl run -i \
    --tty load-generator \
    --rm --image=busybox \
    --restart=Never \
    -- /bin/sh -c "while sleep 0.01; do wget -q -O- http://php-apache; done"
```

5. 若要觀察向外擴展部署，請定期在不同的終端機中，從您執行上一個步驟的終端機執行下列命令。

```
kubectl get hpa php-apache
```

範例輸出如下。

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   250%/50%   1         10        5
 4m44s
```

複本計數可能需要一分多鐘的時間才會增加。只要實際 CPU 百分比高於目標百分比，複本計數就會增加，最多可增加至 10。在這種情況下，它會是 250%，所以 REPLICAS 的數量會繼續增加。

> **Note**
>
> 您可能需要幾分鐘才會看到複本計數達到其最大值。例如，如果 CPU 負載只需要 6 個複本維持在 50% 或以下，則負載將不會擴展到 6 個複本以外。

6. 停止該負載。在您正在產生負載的終端窗口中，按住 Ctrl+C 鍵來停用負載。透過在正在觀察縮減的終端窗口中再次執行以下命令，您會看到複本縮減回 1。

```
kubectl get hpa
```

範例輸出如下。

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
```

```
php-apache   Deployment/php-apache   0%/50%    1         10        1          25m
```

> **Note**
>
> 縮減的預設時間範圍為 5 分鐘，因此需要一些時間才會看到複本計數再次達到 1，即使目前的 CPU 百分比為 0。時間範圍是可修改的。如需詳細資訊，請參閱 Kubernetes 文件中的 Horizontal Pod Autoscaler。

7. 當您完成範例應用程式的實驗後，請刪除 php-apache 資源。

```
kubectl delete deployment.apps/php-apache service/php-apache
 horizontalpodautoscaler.autoscaling/php-apache
```

Amazon EKS 上的網路負載平衡

網路流量在 OSI 模型的 L4 處於負載平衡狀態。若要在負載平衡應用程式流量L7，請部署佈建 Kubernetesingress AWS 應用程式負載平衡器的應用程式流量。如需詳細資訊，請參閱 Amazon EKS 上的應用程式負載平衡。若要深入瞭解這兩種負載平衡類型之間的差異，請參閱 AWS 網站上的 E lastic Load Balancing 功能。

當您建立KubernetesService類型時LoadBalancer， AWS 雲端提供者負載平衡器控制器預設會建立 AWS 傳統負載平衡器，但也可以建立 AWS 網路負載平衡器。此控制器將來僅接收關鍵錯誤修正。如需使用 AWS 雲端提供者負載平衡器的詳細資訊，請參閱Kubernetes文件中的AWS 雲端提供者負載平衡器控制器。本主題中不涉及其使用方式。

建議您使用 AWS Load Balancer Controller 的版本 2.7.2 或更新版本，而非 AWS 雲端提供者負載平衡器控制器。建AWS Load Balancer Controller立 AWS 網路負載平衡器，但不會建立 AWS 傳統負載平衡器。本主題的其餘部分是關於使用 AWS Load Balancer 控制器。

AWS Network Load Balancer 可以負載平衡Pods部署到 Amazon EC2 IP 和執行個體目標或 AWS Fargate IP 目標的網路流量。如需詳細資訊，請參閱 GitHub 上的 AWS Load Balancer Controller。

必要條件

在使用 AWS Load Balancer Controller 將網路流量負載平衡前，您必須先達到以下要求。

- 擁有現有的叢集。如果沒有現有叢集，請參閱 Amazon EKS 入門。如果您需要更新現有叢集的版本，請參閱 更新 Amazon EKS 叢集 Kubernetes 版本。

- 將 AWS Load Balancer Controller 部署在您的叢集上。如需詳細資訊，請參閱 什麼是 AWS Load Balancer Controller？。我們建議使用 2.7.2 版或更新版本。
- 至少有一個子網路。若在可用區域中找到多個標記的子網路，控制器會根據其子網路 ID 依詞典編纂順序來選擇第一個子網路。子網路必須至少有 8 個可用的 IP 地址。
- 若您正在使用 2.1.1 版本或更舊版的 AWS Load Balancer Controller，則子網必須按如下方式加上標籤。如果使用版本 2.1.2 或更新版本，則此標籤是選用的。如果您有多個叢集在同一個 VPC 中執行，或在 VPC 中共用多個 AWS 服務的子網路，並希望對每個叢集佈建負載平衡器的位置進一步控制，則可能需要標記子網路。如果明確指定子網路 ID 作為服務物件上的註釋，則 Kubernetes 和 AWS Load Balancer Controller 會直接使用這些子網路來建立負載平衡器。如果選擇使用此方法來佈建負載平衡器，而且可以略過下列私有和公有子網路標記要求，則不需要標記子網路。使用您的叢集名稱取代 *my-cluster*。
  - 索引鍵：kubernetes.io/cluster/*my-cluster*
  - 值：shared 或 owned
- 您的公有和私有子網路必須符合下列要求，除非您明確指定子網路 ID 作為服務或傳入物件上的註釋。如果透過明確指定子網路 ID 作為服務或傳入物件上的註釋來佈建負載平衡器，則 Kubernetes 和 AWS Load Balancer Controller 會直接使用這些子網路來建立負載平衡器，且不需要以下標籤。
  - 私有子網路：必須使用下列格式進行標記。這Kubernetes樣，Load Bal AWS ancer 控制器就知道子網路可用於內部負載平衡器。如果您在 2020 年 3 月 26 日之後使用eksctl或 Amazon EKS AWS AWS CloudFormation 範本建立 VPC，則子網路在建立時會適當地標記這些子網路。如需 Amazon EKS AWS AWS CloudFormation VPC 範本的詳細資訊，請參閱 為 Amazon EKS 叢集建立 VPC。
    - 索引鍵：kubernetes.io/role/internal-elb
    - 值：1
  - 公有子網路：必須使用下列格式進行標記。這是為了讓 Kubernetes 僅以這些子網路用於外部負載平衡器，而非在每個可用區域 (依據子網路 ID 的字典順序) 中選擇公有子網路。如果您在 2020 年 3 月 26 日之後使用eksctl或 Amazon EKS AWS CloudFormation 範本建立 VPC，則子網路在建立時會適當地標記這些子網路。如需 Amazon EKS AWS CloudFormation VPC 範本的詳細資訊，請參閱。為 Amazon EKS 叢集建立 VPC
    - 索引鍵：kubernetes.io/role/elb
    - 值：1

  如果沒有明確新增子網路角色標籤，則 Kubernetes 服務控制器會檢查叢集 VPC 子網路的路由表，以判定子網路是私有還是公有。建議您不要依賴此行為，而是明確地新增私有或公有角色標籤。AWS Load Balancer Controller 不會檢查路由表，並要求私有和公有標籤出現，才能成功自動探索。

考量事項

- 負載平衡器的組態是由加入至該服務清單檔案的註釋所控制。使用時的服務註解與使用 AWS 雲端提供者負載平衡器控制器時的服務註解不同。AWS Load Balancer Controller部署服務之前，請務必檢閱 AWS Load Balancer Controller 的 annotations (註釋)。
- 當使用 Amazon VPC CNI plugin for Kubernetes 時，AWS Load Balancer Controller 可以針對 Amazon EC2 IP 或執行個體目標和 Fargate IP 目標進行負載平衡。當使用替代相容 CNI 外掛程式時，控制器僅能針對執行個體目標進行負載平衡。如需 Network Load Balancer 目標類型的詳細資訊，請參閱 Network Load Balancer 使用者指南中的目標類型。
- 如果要在建立負載平衡器時或之後將標籤新增至負載平衡器，請在服務規格中新增下列註釋。如需詳細資訊，請參閱 AWS Load Balancer Controller 文件中的 AWS Resource Tags。

```
service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags
```

- 您可以透過新增以下注釋將彈性 IP 地址指派至 Network Load Balancer。使用彈性 IP 地址的 `Allocation IDs` 取代 *`example values`*。`Allocation IDs` 的數量必須與負載平衡器所使用的子網路數量一致。如需詳細資訊，請參閱 AWS Load Balancer Controller 文件。

```
service.beta.kubernetes.io/aws-load-balancer-eip-allocations:
 eipalloc-xxxxxxxxxxxxxxxxx,eipalloc-yyyyyyyyyyyyyyyyy
```

- Amazon EKS 會為用戶端流量新增一個傳入規則到節點的安全群組，並為 VPC 中的每個負載平衡器子網路新增一個規則，以對您所建立的每個 Network Load Balancer 進行運作狀態檢查。如果 Amazon EKS 嘗試建立超過安全群組允許之規則數量上限配額的規則，則 `LoadBalancer` 類型的服務部署可能會失敗。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中 Amazon VPC 配額的安全群組。請考慮下列選項，將超過安全群組規則數量上限的機會降至最低：
  - 請求提高每個安全群組規則的配額。如需詳細資訊，請參閱《Service Quotas 使用者指南》中的請求提高配額。
  - 使用 IP 目標，而不是執行個體目標。使用 IP 目標時，您可以共用相同目標連接埠的規則。您可以使用註釋手動指定負載平衡器子網路。如需詳細資訊，請參閱 GitHub 上的註釋。
  - 使用傳入，而不是類型服務 `LoadBalancer` 將流量傳送至您的服務。 AWS 應用程式負載平衡器所需的規則比網路負載平衡器少。您可在多個輸入之間共用 ALB。如需詳細資訊，請參閱 Amazon EKS 上的應用程式負載平衡。您不能在多個服務之間共用 Network Load Balancer。
  - 將叢集部署到多個帳戶。
- 如果您的 Pods 在 Amazon EKS 叢集中的 Windows 上執行，則具有負載平衡器的單一服務最多可支援 1024 個後端 Pods。每個 Pod 都有自己的唯一 IP 地址。

- 建議只使用 AWS Load Balancer Controller 建立新的 Network Load Balancer。嘗試取代使用 AWS 雲端提供者負載平衡器控制器建立的現有網路負載平衡器，可能會導致多個網路負載平衡器造成應用程式停機。

建立 Network Load Balancer

您可以建立具有 IP 或執行個體目標的 Network Load Balancer。

IP targets

您可以將 IP 目標與部署到 Amazon EC2 節點或 Fargate 的 Pods 搭配使用。您的 Kubernetes 服務必須以 `LoadBalancer` 類型建立。如需詳細資訊，請參閱Kubernetes文件 LoadBalancer中的 Type。

若要建立使用 IP 目標的負載平衡器，請將下列註解新增至服務清單檔案，然後部署您的服務。的`external`值`aws-load-balancer-type`是導致雲端提供者負載平衡器控制器 (而非 AWS 雲端提供者負載平衡器控制器) 建立 Network Load Balancer 的原因。AWS Load Balancer Controller您可以檢視具有注釋的範例服務清單檔案。

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
```

> **Note**
>
> 如果您正在負載平衡至 IPv6 Pods，請新增以下註釋。您只能在 IPv6 上負載平衡至 IP 目標，而不能負載平衡執行個體目標。如果沒有此註釋，負載平衡將在 IPv4 上執行。
>
> ```
> service.beta.kubernetes.io/aws-load-balancer-ip-address-type: dualstack
> ```

依預設，使用 `internal` `aws-load-balancer-scheme` 建立 Network Load Balancer。您可以在叢集 VPC 的任何子網路中啟動 Network Load Balancer，包括建立叢集時未指定的子網路。

Kubernetes 會檢查您子網路的路由表，以判別其為公有或私有。公有子網路具備使用網際網路閘道直接通向網際網路的路由，而私有子網路則不然。

若要在公有子網路中建立 Network Load Balancer 對 Amazon EC2 節點進行負載平衡 (Fargate 只能為私有)，請指定具有以下註釋的 `internet-facing`：

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Note**
>
> 仍支援 `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"` 註釋以提供回溯相容性。不過，建議您針對新的負載平衡器使用先前的註釋，而不是 `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"`。

> **Important**
>
> 建立服務後，請勿編輯註釋。如果需要進行修改，請刪除服務物件，並使用此註釋所需的值重新建立服務物件。

Instance targets

AWS 雲端提供者負載平衡器控制器僅建立具有執行個體目標的網路負載平衡器。版本 `2.2.0` 和更新版本的 AWS Load Balancer Controller 也會建立具有執行個體目標的 Network Load Balancer。我們建議您使用它，而不是 AWS 雲端提供者負載平衡器控制器來建立新的網路負載平衡器。您可以將 Network Load Balancer 執行個體目標與部署至 Amazon EC2 節點 (而非 Fargate) 的 Pods 搭配使用。若要在部署到 Fargate 的 Pods 之間負載平衡網路流量，則必須使用 IP 目標。

若要將 Network Load Balancer 部署至私有子網路，則您的服務規格必須具有下列註釋。您可以檢視具有注釋的範例服務清單檔案。的`external`值`aws-load-balancer-type`是造成 AWS Load Balancer 控制器 (而非 AWS 雲端提供者負載平衡器控制器) 建立 Network Load Balancer 的原因。

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "instance"
```

依預設，使用 `internal` `aws-load-balancer-scheme` 建立 Network Load Balancer。若為內部 Network Load Balancer，您的 Amazon EKS 叢集必須設定至少使用 VPC 中的一個私有子網路。Kubernetes 會檢查子網路的路由表，以識別其為公有或私有。公有子網路具備使用網際網路閘道直接通向網際網路的路由，而私有子網路則不然。

若要在公有子網路中建立 Network Load Balancer 對 Amazon EC2 節點進行負載平衡，請指定具有以下註釋的 `internet-facing`：

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Important**
>
> 建立服務後，請勿編輯註釋。如果需要進行修改，請刪除服務物件，並使用此註釋所需的值重新建立服務物件。

(選用) 部署範例應用程式

必要條件

- 叢集 VPC 中必須至少有一個公有或私有子網路。
- 將 AWS Load Balancer Controller 部署在您的叢集上。如需詳細資訊，請參閱 什麼是 AWS Load Balancer Controller？。我們建議使用 2.7.2 版或更新版本。

部署範例應用程式

1. 若要部署至 Fargate，請確保 VPC 中有一個可用的私有子網路，並建立一個 Fargate 設定檔。如果您沒有部署到 Fargate，請略過此步驟。您可以透過執行下列命令來建立描述檔，也可以使用命令中 name 和 namespace 相同的值藉由 AWS Management Console 進行。使用自己的取代 *example values*。

   ```
   eksctl create fargateprofile \
       --cluster my-cluster \
       --region region-code \
       --name nlb-sample-app \
       --namespace nlb-sample-app
   ```

2. 部署範例應用程式。

   a. 建立應用程式的命名空間。

      ```
      kubectl create namespace nlb-sample-app
      ```

   b. 將下列內容儲存到電腦上名為 *sample-deployment*.yaml 的檔案中。

      ```
      apiVersion: apps/v1
      kind: Deployment
      ```

```
metadata:
  name: nlb-sample-app
  namespace: nlb-sample-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: public.ecr.aws/nginx/nginx:1.23
          ports:
            - name: tcp
              containerPort: 80
```

c. 將清單檔案套用至叢集。

```
kubectl apply -f sample-deployment.yaml
```

3. 使用負載平衡至 IP 目標的網際網路取向的 Network Load Balancer 建立服務。

a. 將下列內容儲存到電腦上名為 *sample-service*.yaml 的檔案中。若要部署至 Fargate 節點，請移除 `service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing` 行。

```
apiVersion: v1
kind: Service
metadata:
  name: nlb-sample-service
  namespace: nlb-sample-app
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 80
```

```
    protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

b. 將清單檔案套用至叢集。

```
kubectl apply -f sample-service.yaml
```

4. 確認已部署服務。

```
kubectl get svc nlb-sample-service -n nlb-sample-app
```

範例輸出如下。

```
NAME             TYPE           CLUSTER-IP         EXTERNAL-IP
                                                  PORT(S)        AGE
sample-service   LoadBalancer   10.100.240.137
 k8s-nlbsampl-nlbsampl-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
  80:32400/TCP    16h
```

> **Note**
>
> *10.100.240.137* AWS 區域

5. 打開 Amazon EC2 AWS Management Console。選取左側導覽窗格中的 Target Groups (目標群組) (在 Load Balancing (負載平衡) 下)。在 Name (名稱) 資料欄中選取目標群組名稱，其 Load Balancer (負載平衡器) 資料欄中的值必須與上一步驟中 EXTERNAL-IP 資料欄的部分輸出值名稱相符。例如，若輸出值名稱與先前的輸出值名稱相同，請選取名為 k8s-default-samplese-*xxxxxxxxxx* 的目標群組。Target type (目標類型) 為 IP，因為其已在範例服務清單檔案中指定。

6. 選取 Target group (目標群組)，然後選擇 Targets (目標) 索引標籤。在 Registered targets (已註冊目標) 下，您應該會看到在上一個步驟中部署之三個複本的三個 IP 地址。待所有目標狀態變為 healthy (狀態良好) 再繼續。可能需要幾分鐘的時間，所有目標才能變為 healthy。目標在變更為 healthy 狀態前，可能處於 unhealthy 狀態。

7. 將流量傳送至服務，使用 EXTERNAL-IP 在上一個步驟的輸出中傳回的值，來取代 *xxxxxxxxxx-xxxxxxxxxxxxxxxx* 和 *us-west-2*。如果您部署至私有子網路，則需要從 VPC 內的裝置 (例如堡壘主機) 檢視頁面。如需詳細資訊，請參閱 AWS上的 Linux 堡壘主機。

```
curl k8s-default-samplese-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
```

範例輸出如下。

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

8. 在完成範例部署、服務和命名空間後，請將其移除。

```
kubectl delete namespace nlb-sample-app
```

Amazon EKS 上的應用程式負載平衡

建立時 Kubernetes`ingress`，系統會佈建 Ap AWS plication Load Balancer (ALB) 來平衡應用程式流量的負載。如需進一步了解，請參閱《Application Load Balancer 使用者指南》中的什麼是 Application Load Balancer？和 Kubernetes 文件中的傳入。ALB 可與部署至節點或 Pods 的 AWS Fargate搭配使用。您可以將 ALB 部署到公有或私有子網路。

應用程式流量會在 OSI 模型的 L7 平衡。若要負載平衡 L4 的網路流量，您可以部署 `LoadBalancer` 類型的 Kubernetes `service`。此類型提供 AWS Network Load Balancer。如需詳細資訊，請參閱 Amazon EKS 上的網路負載平衡。若要進一步了解兩種負載平衡之間的差異，請參閱 AWS 網站上的 Elastic Load Balancing 功能。

必要條件

在將應用程式流量負載平衡到應用程式之前，您必須符合以下要求。

- 擁有現有的叢集。如果沒有現有叢集，請參閱 Amazon EKS 入門。如果您需要更新現有叢集的版本，請參閱 更新 Amazon EKS 叢集 Kubernetes 版本。
- 將 AWS Load Balancer Controller 部署在您的叢集上。如需詳細資訊，請參閱 什麼是 AWS Load Balancer Controller？。我們建議使用 `2.7.2` 版或更新版本。
- 至少兩個子網路位於不同的可用區域。AWS Load Balancer Controller 會從每個可用區域選擇一個子網。當在可用區域中找到多個標記的子網路時，控制器會根據其子網路 ID 依詞典編纂順序來選擇子網路。每個子網路必須至少有 8 個可用的 IP 地址。

如果使用連接至工作節點的多個安全群組，則必須依如下方式標記一個安全群組。使用您的叢集名稱取代 *my-cluster*。

- 索引鍵：kubernetes.io/cluster/*my-cluster*
- 值：shared 或 owned

- 如果正在使用 AWS Load Balancer Controller 版本 2.1.1 或更早版本，則子網必須依以下格式加上標籤。如果使用版本 2.1.2 或更新版本，則不一定要加上標籤。不過，如果出現下列任何情況，我們建議您標記子網路。您有多個叢集在同一個 VPC 中執行，或者有多個共用 VPC 中子網路的 AWS 服務。或者，您希望更好地控制為每個叢集佈建負載平衡器的位置。使用您的叢集名稱取代 *my-cluster*。
  - 索引鍵：kubernetes.io/cluster/*my-cluster*
  - 值：shared 或 owned
- 您的公有和私有子網路必須符合下列要求，除非明確指定子網路 ID 作為服務或傳入物件上的註釋。假設您透過明確指定子網路 ID 作為服務或傳入物件上的註釋來佈建負載平衡器。在這種情況下，Kubernetes 和 AWS Load Balancer 控制器會直接使用這些子網路來建立負載平衡器，而且不需要下列標記。
  - 私有子網路：必須使用下列格式進行標記。這Kubernetes樣， AWS 負載平衡器控制器就知道子網路可用於內部負載平衡器。如果您在 2020 年 3 月 26 日之後使用eksctl或 Amazon EKS AWS CloudFormation 範本建立 VPC，則子網路在建立時會正確標記。如需 Amazon EKS AWS CloudFormation VPC 範本的詳細資訊，請參閱。為 Amazon EKS 叢集建立 VPC
    - 索引鍵：kubernetes.io/role/internal-elb
    - 值：1
  - 公有子網路：必須使用下列格式進行標記。這是為了讓 Kubernetes 知道只使用為外部負載平衡器指定的子網路。如此一來，Kubernetes 不會在每個可用區域中選擇公有子網路 (根據其子網路 ID 依詞典編纂順序)。如果您在 2020 年 3 月 26 日之後使用eksctl或 Amazon EKS AWS CloudFormation 範本建立 VPC，則子網路在建立時會正確標記。如需 Amazon EKS AWS CloudFormation VPC 範本的詳細資訊，請參閱。為 Amazon EKS 叢集建立 VPC
    - 索引鍵：kubernetes.io/role/elb
    - 值：1

  如果沒有明確新增子網路角色標籤，則 Kubernetes 服務控制器會檢查叢集 VPC 子網路的路由表。這是為了判斷子網路是私有還是公有。建議您不要依賴此行為，而是明確地新增私有或公有角色標籤。AWS Load Balancer Controller 不會檢查路由表。其還需要存在私有和公有標籤才能成功自動探索。

考量事項

- 每當在具有註釋的叢集上建立Kubernetes輸入 AWS 資源時，AWS Load Balancer 控制器就會建立 ALB 和必要的支援資源。`kubernetes.io/ingress.class: alb`傳入資源設定 ALB 在叢集內透過 HTTP 或 HTTPS 路由到不同的 Pods。為了確保您的傳入物件使用 AWS Load Balancer Controller，請新增以下註釋到您的 Kubernetes 傳入規格。如需詳細資訊，請參閱 GitHub 上的傳入規格。

```
annotations:
    kubernetes.io/ingress.class: alb
```

> **Note**
>
> 如果您正在負載平衡至 IPv6 Pods，請新增以下註釋到您的傳入規格。您只能在 IPv6 上負載平衡至 IP 目標，而不能負載平衡執行個體目標。如果沒有此註釋，負載平衡將在 IPv4 上執行。
>
> ```
> alb.ingress.kubernetes.io/ip-address-type: dualstack
> ```

- AWS Load Balancer Controller 支援下列流量模式：
  - 執行個體 – 將叢集中的節點註冊作為 ALB 的目標。到達 ALB 的流量被路由到服務的 NodePort，然後代理到您的 Pods。這是預設的流量模式。您也可以使用 `alb.ingress.kubernetes.io/target-type: instance` 註釋明確指定它。

    > **Note**
    >
    > 您的Kubernetes服務必須指定NodePort或 "LoadBalancer" 類型才能使用此流量模式。

  - IP：註冊 Pods 作為 ALB 的目標。到達 ALB 的流量會直接路由到服務的 Pods。您必須指定 `alb.ingress.kubernetes.io/target-type: ip` 註釋才能使用此流量模式。當目標 Pods 在 Fargate 上執行時，必須使用 IP 目標類型。
- 要標記由控制器建立的 ALB，請新增以下註釋至控制器：`alb.ingress.kubernetes.io/tags`。如需由 AWS Load Balancer Controller 支援的所有可用注釋，請參閱 GitHub 上的傳入注釋。
- 升級或降級 ALB 控制器版本可能會導致依賴控制器的功能發生重大變化。如需有關每個版本中引入的重大變更的詳細資訊，請參閱 GitHub 上的 ALB 控制器版本備註。

若要跨多個服務資源共用 Application Load Balancer，則請使用 **IngressGroups**

若要將傳入加入群組，請將下列註解新增至 Kubernetes 傳入資源規格。

```
alb.ingress.kubernetes.io/group.name: my-group
```

群組名稱必須：

- 是 63 個字元或以下的長度。
- 由小寫字母、數字、- 和 . 組成
- 開頭和結尾為字母或數字。

控制器會自動合併相同傳入群組中所有傳入的傳入規則。控制器透過單個 ALB 提供支援。在傳入中定義的大多數註解僅適用於該傳入定義的路徑。根據預設，傳入資源不屬於任何傳入群組。

> **Warning**
>
> 可能的安全威脅：僅當所有具有建立或修改傳入資源之 RBAC 許可的 Kubernetes 使用者都在相同的信任界限內時，才為傳入指定傳入群組。如果您使用群組名稱新增註釋，其他 Kubernetes 使用者可以建立或修改其傳入，使其屬於同一傳入群組。這樣做可能會產生不良行為，例如以較高優先順序規則覆寫現有規則。

您可以新增傳入資源的訂單編號。

```
alb.ingress.kubernetes.io/group.order: '10'
```

該編號可以是 1 到 1000 之間的數字。系統會先評估相同傳入群組中所有傳入數量的最低數量。系統會使用 0 值評估沒有此注釋的所有傳入。具有較高數量的重複規則可以覆寫具有較低數量的規則。根據預設，相同傳入群組中傳入之間的規則順序是由命名空間和名稱依詞典編纂順序決定。

> **Important**
>
> 確定相同傳入群組中的每個傳入都有唯一的優先順序編號。您不能在整個傳入中有重複的訂單編號。

(選用) 部署範例應用程式

必要條件

- 叢集 VPC 中必須至少有一個公有或私有子網路。
- 將 AWS Load Balancer Controller 部署在您的叢集上。如需詳細資訊，請參閱 什麼是 AWS Load Balancer Controller？。我們建議使用 2.7.2 版或更新版本。

部署範例應用程式

您可以在具有 Amazon EC2 節點、Fargate Pods 或兩者兼有的叢集上執行範例應用程式。

1. 如果您沒有部署到 Fargate，請略過此步驟。如果您要部署到 Fargate，請建立 Fargate 描述檔。您可以透過執行下列命令來建立描述檔，也可以使用命令中 name 和 namespace 相同的值藉由 AWS Management Console 進行。使用自己的取代 *example values*。

   ```
   eksctl create fargateprofile \
       --cluster my-cluster \
       --region region-code \
       --name alb-sample-app \
       --namespace game-2048
   ```

2. 將遊戲 2048 部署為範例應用程式，以確認是否因為輸入物件而AWS Load Balancer Controller建立 AWS ALB。針對您正在部署的子網路類型，完成相關的步驟。

   a. 如果要部署到使用 IPv6 系列建立的叢集中的 Pods，請跳到下一步。

      - 公有

        ```
        kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
        ```

      - 私有

        1. 下載清單檔案。

           ```
           curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
           ```

        2. 編輯檔案並尋找顯示 alb.ingress.kubernetes.io/scheme: internet-facing 的行。

3. 將 *internet-facing* 變更為 **internal**，並儲存檔案。
4. 將清單檔案套用至叢集。

```
kubectl apply -f 2048_full.yaml
```

b. 如果要部署到使用 IPv6 系列建立的叢集中的 Pods，請完成下列步驟。

1. 下載清單檔案。

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
```

2. 在編輯器中開啟檔案，然後在傳入規格中新增以下一行註釋。

```
alb.ingress.kubernetes.io/ip-address-type: dualstack
```

3. 如果您正在負載平衡至內部 Pods，而不是面向網際網路的 Pods，請變更該行，即將 `alb.ingress.kubernetes.io/scheme: internet-facing` 變更為 `alb.ingress.kubernetes.io/scheme: internal`
4. 儲存檔案。
5. 將清單檔案套用至叢集。

```
kubectl apply -f 2048_full.yaml
```

3. 幾分鐘後，使用下列指令來驗證傳入資源已建立。

```
kubectl get ingress/ingress-2048 -n game-2048
```

範例輸出如下。

```
NAME           CLASS    HOSTS   ADDRESS
                       PORTS   AGE
ingress-2048   <none>   *       k8s-game2048-ingress2-xxxxxxxxxx-yyyyyyyyyy.region-code.elb.amazonaws.com   80      2m32s
```

> **Note**
>
> 若在私有子網路中建立負載平衡器，則先前輸出中位於 ADDRESS 下的值前面會加上 internal-。

如果幾分鐘後尚未成功建立傳入，請執行下列命令以檢視 AWS Load Balancer Controller 日誌。這些日誌可能包含錯誤訊息，您可用於診斷部署的相關問題。

```
kubectl logs -f -n kube-system -l app.kubernetes.io/instance=aws-load-balancer-controller
```

4. 如果您部署公有子網路，請開啟瀏覽器，再導覽至先前指令輸出中的 ADDRESS URL 查看範例應用程式。如果沒有看到任何內容，請重新整理瀏覽器，然後再試一次。如果您部署至私有子網路，則需要從 VPC 內的裝置 (例如堡壘主機) 檢視頁面。如需詳細資訊，請參閱 AWS上的 Linux 堡壘主機。
5. 完成對範例應用程式的實驗後，請透過執行下列命令之一將其刪除。
   - 如果您應用了清單檔案，而不是應用下載的複本，請使用以下命令。

     ```
     kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
     ```

   - 如果您下載並編輯清單檔案，請使用下列命令。

     ```
     kubectl delete -f 2048_full.yaml
     ```

限制可指派給服務的外部 IP 地址

Kubernetes 服務可透過以下方式從叢集內部連線：

- Kubernetes 自動指派的叢集 IP 地址
- 您為服務規則中的 `externalIPs` 屬性指定的 IP 地址。外部 IP 地址不是由 Kubernetes 管理，是叢集管理員的責任。使用 `externalIPs` 指定的外部 IP 地址不同於雲端提供商指派給類型服務 `LoadBalancer` 的外部 IP 地址。

如需進一步了解 Kubernetes 服務，請參閱 Kubernetes 文件中的服務。您可以限制可以在服務規格中為 `externalIPs` 指定的 IP 地址。

限制可以在服務規格中為 **`externalIPs`** 指定的 IP 地址

1. 部署 `cert-manager` 以管理網路掛鉤憑證。如需詳細資訊，請參閱 `cert-manager`文件。

```
kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.5.4/
cert-manager.yaml
```

2. 確認是否正在執行 cert-manager Pods。

```
kubectl get pods -n cert-manager
```

範例輸出如下。

```
NAME                                       READY   STATUS    RESTARTS   AGE
cert-manager-58c8844bb8-nlx7q              1/1     Running   0          15s
cert-manager-cainjector-745768f6ff-696h5   1/1     Running   0          15s
cert-manager-webhook-67cc76975b-4v4nk      1/1     Running   0          14s
```

3. 檢閱您現有的服務，以確保其中沒有指派給這些服務的外部 IP 地址，而這些地址不包含在要限制地址的 CIDR 區塊中。

```
kubectl get services -A
```

範例輸出如下。

```
NAMESPACE                       NAME                                      TYPE
   CLUSTER-IP       EXTERNAL-IP     PORT(S)         AGE
cert-manager                    cert-manager                              ClusterIP
   10.100.102.137   <none>          9402/TCP        20m
cert-manager                    cert-manager-webhook                      ClusterIP
   10.100.6.136     <none>          443/TCP         20m
default                         kubernetes                                ClusterIP
   10.100.0.1       <none>          443/TCP         2d1h
externalip-validation-system    externalip-validation-webhook-service     ClusterIP
   10.100.234.179   <none>          443/TCP         16s
kube-system                     kube-dns                                  ClusterIP
   10.100.0.10      <none>          53/UDP,53/TCP   2d1h
my-namespace                    my-service                                ClusterIP
   10.100.128.10    192.168.1.1     80/TCP          149m
```

如果有任何值是不位於要限制存取之區塊內的 IP 地址，則您需要將地址變更為位於區塊內，然後重新部署服務。例如，上一個輸出中的 my-service 服務具有指派給其的外部 IP 地址，而該地址不位於步驟 5 中的 CIDR 區塊範例內。

4. 下載外部 IP Webhook 清單檔案。您也可以檢視 GitHub 上的 Webhook 來源程式碼。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/externalip-webhook.yaml
```

5. 指定 CIDR 區塊。在編輯器中開啟下載的檔案，並在下列這幾行的開頭移除 #。

```
#args:
#- --allowed-external-ip-cidrs=10.0.0.0/8
```

使用您自己的 CIDR 區塊取代 `10.0.0.0/8`。您可以指定需要的區塊，不限數量。如果指定多個區塊，則請在區塊之間加上逗號。

6. 如果您的叢集不在us-west-2 AWS 區域，則使用以下命令 eplace (取代)檔案中的 us-west-2、602401143452 和 amazonaws.com。在執行命令之前，請用 Amazon 容器映像登錄檔 中來自清單的 AWS 區域 值取代 *region-code* 和 *111122223333*。

```
sed -i.bak -e 's|602401143452|111122223333|' externalip-webhook.yaml
sed -i.bak -e 's|us-west-2|region-code|' externalip-webhook.yaml
sed -i.bak -e 's|amazonaws.com||' externalip-webhook.yaml
```

7. 將清單檔案套用至叢集。

```
kubectl apply -f externalip-webhook.yaml
```

嘗試使用為 `externalIPs` 指定的 IP 地址將服務部署至叢集，如果 IP 地址未包含於 Specify CIDR blocks (指定 CIDR 區塊) 步驟中指定的區塊，則嘗試將會失敗。

將容器映像從一個儲存庫複製到另一個儲存庫

本主題介紹如何從節點無法存取的儲存庫中提取容器映像，並將該映像推送到節點可存取的儲存庫。您可以將映像推送到 Amazon ECR 或節點可存取的替代儲存庫。

必要條件

- 在您的電腦上安裝和設定的 Docker 引擎。如需相關說明，請參閱 Docker 文件中的安裝 Docker 引擎。
- 已在裝置或 AWS CloudShell 上安裝和設定 AWS Command Line Interface (AWS CLI) 版本 `2.12.3` 或更新版本，或是版本 `1.27.160` 或更新版本。若要檢查您目前的版本，請使用 **`aws --version`**

| cut -d / -f2 | cut -d ' ' -f1。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI 落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。AWS CloudShell 中安裝的 AWS CLI 版本也可能比最新版本落後數個版本。若要更新，請參閱《AWS CloudShell 使用者指南》中的安裝 AWS CLI 至您的主目錄。

- 如果您希望節點透過 Amazon 網路推送容器映像到私有 Amazon ECR 儲存庫，或從中提取容器映像，則需使用適用於 Amazon ECR 的介面 VPC 端點。如需詳細資訊，請參閱《Amazon 彈性容器登錄檔使用者指南》中的建立適用於 Amazon ECR 的 VPC 端點。

請完成以下步驟，以從某個儲存庫中提取容器映像，並將其推送到您的儲存庫。在本主題所提供的以下範例中，會提取適用於 Amazon VPC CNI plugin for Kubernetes 指標協助程式的映像。當您依照以下步驟操作時，請務必使用您的值取代 *example values*。

將容器映像從一個儲存庫複製到另一個儲存庫

1. 如果您尚未擁有 Amazon ECR 儲存庫或其他儲存庫，請先建立您的節點可存取的儲存庫。以下命令會建立 Amazon ECR 私有儲存庫。Amazon ECR 私有儲存庫名稱必須以字母開頭。名稱僅能包含小寫字母、數字、連字號 (-)、底線 (_) 和正斜線 (/)。如需詳細資訊，請參閱《Amazon 彈性容器登錄檔使用者指南》中的建立私有儲存庫。

   您可以使用您選擇取代 *cni-metrics-helper*。最佳實務是為每個映像建立個別儲存庫。建議您採取此做法，因為映像標籤在儲存庫中必須是唯一的。使用 Amazon ECR 支援的 AWS 區域 取代 *region-code*。

   ```
   aws ecr create-repository --region region-code --repository-name cni-metrics-helper
   ```

2. 判定節點需要提取的映像的登錄檔、儲存庫和標籤 (選用)。此資訊採用 `registry/repository[:tag]` 格式。

   許多有關於安裝映像的 Amazon EKS 主題都要求您套用清單檔案檔案或使用 Helm Chart 來安裝映像。然而在套用清單檔案檔案或安裝 Helm Chart 之前，請檢視清單檔案的內容或圖表的 `values.yaml` 檔案。如此一來，您可以判定要提取的登錄檔、儲存庫和標籤。

   例如，您可以在適用於 Amazon VPC CNI plugin for Kubernetes 指標協助程式的清單檔案中找到以下這行。登錄檔為 `602401143452.dkr.ecr.us-west-2.amazonaws.com`，這是一個 Amazon ECR 私有登錄檔。儲存庫為 `cni-metrics-helper`。

   ```
   image: "602401143452.dkr.ecr.us-west-2.amazonaws.com/cni-metrics-helper:v1.12.6"
   ```

您可能會看到如以下映像位置的變化：

- 僅有 `repository-name:tag`。在此案例中，`docker.io` 通常是未經指定的登錄檔，因為在沒有指定登錄檔的情況下，依照預設 Kubernetes 會將其置於儲存庫名稱的前面。
- `repository-name/repository-namespace/repository:tag`。儲存庫命名空間為選用，但有時會由儲存庫擁有者指定，以將映像分類。例如，Amazon ECR Public Gallery 中的所有 Amazon EC2 映像都使用 `aws-ec2` 命名空間。

在使用 Helm 安裝映像之前，請檢視 Helm `values.yaml` 檔案，以判定映像位置。例如，適用於 Amazon VPC CNI plugin for Kubernetes 指標協助程式的 `values.yaml` 檔案包含以下各行。

```
image:
  region: us-west-2
  tag: v1.12.6
  account: "602401143452"
  domain: "amazonaws.com"
```

3. 提取清單檔案檔案中指定的容器映像。

   a. 如果您要從公有登錄檔 (例如 Amazon ECR Public Gallery) 進行提取，則可以跳到下一個子步驟，因為不需要身分驗證。在此範例中，您將向包含 CNI 指標協助程式映像的儲存庫的 Amazon ECR 私有登錄檔進行身分驗證。 Amazon EKS 會維護 Amazon 容器映像登錄檔 所列每個登錄檔中的映像。您可以使用不同登錄檔的資訊取代 *602401143452* 和 *region-code*，以對任何登錄檔進行身分驗證。每個支援 Amazon EKS 的 AWS 區域 中都有一個個別的登錄檔。

   ```
   aws ecr get-login-password --region region-code | docker login --username AWS --
   password-stdin 602401143452.dkr.ecr.region-code.amazonaws.com
   ```

   b. 提取映像。在此範例中，您將從上一個子步驟中所驗證的登錄表中提取。將 *602401143452* 和 *region-code* 取代為您在上一個子步驟中提供的資訊。

   ```
   docker pull 602401143452.dkr.ecr.region-code.amazonaws.com/cni-metrics-
   helper:v1.12.6
   ```

4. 標記與登錄檔、儲存庫和標籤一起提取的映像。以下範例假設您從清單檔案檔案中提取映像，並將其推送到您在第一個步驟中建立的 Amazon ECR 私有儲存庫。使用您的帳戶 ID 取代

111122223333。使用您在其中建立 Amazon ECR 私有儲存庫的 AWS 區域 取代 *region-code*。

```
docker tag cni-metrics-helper:v1.12.6 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

5. 對登錄檔進行身分驗證。在此範例中，您需對您在第一個步驟中建立的 Amazon ECR 私有登錄檔進行身分驗證。如需詳細資訊，請參閱《Amazon 彈性容器登錄檔使用者指南》中的登錄檔身分驗證。

```
aws ecr get-login-password --region region-code | docker login --username AWS --password-stdin 111122223333.dkr.ecr.region-code.amazonaws.com
```

6. 將映像推送到您的儲存庫。在此範例中，您需將映像推送到您在第一個步驟中建立的 Amazon ECR 私有儲存庫。如需詳細資訊，請參閱《Amazon 彈性容器登錄檔使用者指南》中的推送 Docker 映像。

```
docker push 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

7. 針對您所推送的映像，使用 `registry/repository:tag` 更新您在先前步驟中用於確定映像的清單檔案檔案。如果您是使用 Helm Chart 進行安裝，則通常會出現一個可供指定 `registry/repository:tag` 的選項。安裝圖表時，請為推送到儲存庫的映像指定 `registry/repository:tag`。

Amazon 容器映像登錄檔

將 AWS Amazon EKS 附加元件部署到叢集時，節點會從附加元件安裝機制中指定的登錄檔中提取所需容器映像，例如，安裝清單檔案或 Helm `values.yaml` 檔案。從 Amazon EKS Amazon ECR 私有儲存庫中提取映像。Amazon EKS 會將映像複寫到每個 Amazon EKS 支援的 AWS 區域 中的儲存庫。您的節點可以透過網際網路，從以下任何登錄檔中提取容器映像。或者，如果您在 VPC 中建立了適用於 Amazon ECR 的介面 VPC 端點 (AWS PrivateLink)，則您的節點便可以透過 Amazon 的網路提取映像。登錄檔要求使用 AWS IAM 帳戶進行身分驗證。您的節點使用 Amazon EKS 節點 IAM 角色進行驗證，該角色具有與其相關聯 AmazonEC2ContainerRegistryReadOnly 受管 IAM 政策的權限。

AWS 區域	登錄檔
af-south-1	877085696533.dkr.ecr.af-south-1.amazonaws.com
ap-east-1	800184023465.dkr.ecr.ap-east-1.amazonaws.com
ap-northeast-1	602401143452.dkr.ecr.ap-northeast-1.amazonaws.com
ap-northeast-2	602401143452.dkr.ecr.ap-northeast-2.amazonaws.com
ap-northeast-3	602401143452.dkr.ecr.ap-northeast-3.amazonaws.com
ap-south-1	602401143452.dkr.ecr.ap-south-1.amazonaws.com
ap-south-2	900889452093.dkr.ecr.ap-south-2.amazonaws.com
ap-southeast-1	602401143452.dkr.ecr.ap-southeast-1.amazonaws.com
ap-southeast-2	602401143452.dkr.ecr.ap-southeast-2.amazonaws.com
ap-southeast-3	296578399912.dkr.ecr.ap-southeast-3.amazonaws.com
ap-southeast-4	491585149902.dkr.ecr.ap-southeast-4.amazonaws.com
ca-central-1	602401143452.dkr.ecr.ca-central-1.amazonaws.com

AWS 區域	登錄檔
ca-west-1	761377655185.dkr.ecr.ca-west-1.amazonaws.com
cn-north-1	918309763551.dkr.ecr.cn-north-1.amazonaws.com.cn
cn-northwest-1	961992271922.dkr.ecr.cn-northwest-1.amazonaws.com.cn
eu-central-1	602401143452.dkr.ecr.eu-central-1.amazonaws.com
eu-central-2	900612956339.dkr.ecr.eu-central-2.amazonaws.com
eu-north-1	602401143452.dkr.ecr.eu-north-1.amazonaws.com
eu-south-1	590381155156.dkr.ecr.eu-south-1.amazonaws.com
eu-south-2	455263428931.dkr.ecr.eu-south-2.amazonaws.com
eu-west-1	602401143452.dkr.ecr.eu-west-1.amazonaws.com
eu-west-2	602401143452.dkr.ecr.eu-west-2.amazonaws.com
eu-west-3	602401143452.dkr.ecr.eu-west-3.amazonaws.com
il-central-1	066635153087.dkr.ecr.il-central-1.amazonaws.com
me-south-1	558608220178.dkr.ecr.me-south-1.amazonaws.com

AWS 區域	登錄檔
me-central-1	759879836304.dkr.ecr.me-central-1.amazonaws.com
sa-east-1	602401143452.dkr.ecr.sa-east-1.amazonaws.com
us-east-1	602401143452.dkr.ecr.us-east-1.amazonaws.com
us-east-2	602401143452.dkr.ecr.us-east-2.amazonaws.com
us-gov-east-1	151742754352.dkr.ecr.us-gov-east-1.amazonaws.com
us-gov-west-1	013241004608.dkr.ecr.us-gov-west-1.amazonaws.com
us-west-1	602401143452.dkr.ecr.us-west-1.amazonaws.com
us-west-2	602401143452.dkr.ecr.us-west-2.amazonaws.com

Amazon EKS 附加元件

附加元件是提供 Kubernetes 應用程式支援操作功能的軟體，但並不專用於應用程式。這包括可觀察性代理程式或 Kubernetes 驅動程式等軟體，可讓叢集與基礎 AWS 資源互動，以用於聯網、運算和儲存。附加軟體通常由Kubernetes社群、雲端供應商或第三方廠商建置和維護。 AWS Amazon EKS 會自動安裝自我管理的附加元件，例如 Amazon VPC CNI plugin for Kubernetes、`kube-proxy`，以及每個叢集的 CoreDNS。您可以變更附加元件的預設設定，並在需要時進行更新。

Amazon EKS 附加元件提供 Amazon EKS 叢集精選附加元件集的安裝和管理。所有 Amazon EKS 附加元件都包含最新的安全修補程式、錯誤修正，並經過 AWS 驗證可與 Amazon EKS 搭配使用。Amazon EKS 附加元件可讓您持續確保 Amazon EKS 叢集安全穩定，並降低安裝、設定和更新附加元件所需的工作量。如果是自我管理的附加元件 (例如 `kube-proxy`) 已在叢集上執行，且可作為

Amazon EKS 附加元件使用，則可以安裝 kube-proxy Amazon EKS 附加元件開始受益於 Amazon EKS 附加元件的功能。

您可以透過 Amazon EKS API 更新 Amazon EKS 附加元件的特定 Amazon EKS 受管組態欄位。您也可以在附加元件啟動後，直接在 Kubernetes 叢集中修改非由 Amazon EKS 管理的組態欄位。這包括定義附加元件適用的特定組態欄位。這些變更一旦進行，就不會被 Amazon EKS 覆寫。您可使用 Kubernetes 伺服器端套用功能加以實現。如需詳細資訊，請參閱 Kubernetes 欄位管理。

您可以使用 Amazon EKS 附加元件搭配任何 Amazon EKS 節點類型。

考量事項

- 若要設定叢集的附加元件，則您的 IAM 主體必須具有 IAM 許可才能使用附加元件。如需詳細資訊，請參閱 Amazon Elastic Kubernetes Service 定義的動作中在其名稱中具有 Addon 的動作。
- Amazon EKS 附加元件會在您為叢集佈建或設定的節點上執行。節點類型包括 Amazon EC2 執行個體和 Fargate。
- 您可以修改不由 Amazon EKS 管理的欄位，以自訂 Amazon EKS 附加元件的安裝。如需詳細資訊，請參閱 Kubernetes 欄位管理。
- 如果您使用建立叢集 AWS Management Console，Amazon EKS 和 CoreDNS Amazon EKS kube-proxy 附加元件會自動新增至您的叢集。Amazon VPC CNI plugin for Kubernetes若您使用 eksctl 來建立具有 config 檔案的叢集，則 eksctl 也可以使用 Amazon EKS 附加元件建立叢集。如果使用不具有 config 檔案的 eksctl 建立叢集或使用任何其他工具建立叢集，則會安裝自我管理 kube-proxy、Amazon VPC CNI plugin for Kubernetes 和 CoreDNS 附加元件，而非 Amazon EKS 附加元件。您可以自行管理這些附加元件，也可以在建立叢集之後手動新增 Amazon EKS 附加元件。
- 該 eks:addon-cluster-admin ClusterRoleBinding 將 cluster-admin ClusterRole 繫結至 eks:addon-manager Kubernetes 身分。此角色具有建立 Kubernetes 命名空間並將附加元件安裝到命名空間的 eks:addon-manager 身分所需的許可。若已移除 eks:addon-cluster-admin ClusterRoleBinding，Amazon EKS 叢集會繼續運作，但 Amazon EKS 將無法再管理任何附加元件。以下平台版本開始的所有叢集皆會使用新的 ClusterRoleBinding。

Kubernete s 平 版本	Amazon EKS 平台版本
1.25	eks.2
1.24	eks.4
1.23	eks.4
1.22	eks.5
1.21	eks.3

您可以使用 Amazon EKS API、、和來新增、更新或刪除 Amazon EKS 附加元件。 AWS Management Console AWS CLI`eksctl`如需詳細資訊，請參閱 管理 Amazon EKS 附加元件。您也可以使用 AWS CloudFormation 建立 Amazon EKS 附加元件。

來自 Amazon EKS 的可用 Amazon EKS 附件元件

您可在叢集上建立下列 Amazon EKS 附加元件。您隨時可以使用`eksctl`、或檢視最新的可用附加元件清單 AWS CLI。 AWS Management Console若要查看所有可用的附加元件或安裝附加元件，請參閱建立附加元件。如果附加元件需要 IAM 許可，則您必須擁有叢集的 IAM OpenID Connect (OIDC) 提供者。若要判斷您是否已經擁有一個，或是需要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。您可以在安裝附加元件之後對其更新或將其刪除。

選擇附加元件以進一步了解它及其安裝需求。

Amazon VPC CNI plugin for Kubernetes

- 名稱 – `vpc-cni`
- 說明 – 為叢集提供原生 VPC 聯網的 Kubernetes 容器網路介面 (CNI) 外掛程式。此附加元件的自我管理類型或受管類型預設會安裝在每個 Amazon EC2 節點上。
- 所需的 IAM 許可 – 此附加元件會利用 Amazon EKS 的服務帳戶的 IAM 角色功能。如果您的叢集使用 `IPv4` 系列，則需要 AmazonEKS_CNI_Policy 中的許可。如果您的叢集使用 `IPv6` 系列，則必須

在 IPv6 模式下建立具有許可的 IAM 政策。您可以建立 IAM 角色，將其中一個政策連接至該角色，並使用下列命令為附加元件使用的 Kubernetes 服務帳戶加上注釋。

將 *my-cluster* 取代為您的叢集名稱，並將 *AmazonEKSVPCCNIRole* 取代為您角色的名稱。如果您的叢集使用 IPv6 系列，則請將 *AmazonEKS_CNI_Policy* 取代為您建立的政策名稱。此命令要求您在裝置上安裝 eksctl。如果您需要使用其他工具來建立角色、為其附加政策並註釋 Kubernetes 服務帳戶，請參閱 設定Kubernetes服務帳戶以擔任 IAM 角色。

```
eksctl create iamserviceaccount --name aws-node --namespace kube-system --cluster my-cluster --role-name AmazonEKSVPCCNIRole \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy --approve
```

- 其他資訊 — 若要進一步了解附加元件的可設定，請參閱上的 aws-v pc-cni-k8s。 GitHub要了解有關該插件的更多信息，請參閱建議：CNI 通過 AWS VPC 進行Kubernetes聯網的插件。如需有關建立附加元件的詳細資訊，請參閱 建立 Amazon EKS 附加元件。
- 更新資訊 — 您一次只能更新一個次要版本。例如，如果目前的版本是 1.28.*x*-eksbuild.*y*，並且您想要更新至 1.30.*x*-eksbuild.*y*，則您必須將目前的版本更新至 1.29.*x*-eksbuild.*y*，然後再更新至 1.30.*x*-eksbuild.*y*。如需更新附加元件的詳細資訊，請參閱 更新 Amazon EKS 附加元件。

CoreDNS

- 名稱 – coredns
- 說明 — 一個靈活、可擴展的 DNS 伺服器，可用來做為 Kubernetes 叢集 DNS。根據預設，在您建立叢集時，會安裝此附加元件的自我管理類型或受管理類型。當您啟動具有至少一個節點的 Amazon EKS 叢集時，依預設會部署兩個 CoreDNS 映像複本，而不論叢集中部署的節點數量為何。CoreDNS Pods 為叢集中的所有 Pods 提供名稱解析。如果您的叢集包含命名空間與 CoreDNS deployment 的命名空間相匹配的 AWS Fargate 設定檔，則可將 CoreDNS Pods 部署至 Fargate 節點。
- 必要的 IAM 許可 – 此附加元件不需要任何許可。
- 其他資訊 – 若要進一步了解 CoreDNS，請參閱 Kubernetes 文件中的使用 CoreDNS 進行服務探索和自訂 DNS 服務。

Kube-proxy

- 名稱 – kube-proxy

- 說明 — 維護每個 Amazon EC2 節點上的網路規則。其可以與您的 Pods 進行網路通訊。此附加元件的自我管理類型或受管類型預設會安裝在叢集的每個 Amazon EC2 節點上。
- 必要的 IAM 許可 – 此附加元件不需要任何許可。
- 其他資訊 – 若要進一步了解 kube-proxy，請參閱 Kubernetes 文件中的 kube-proxy。
- 更新資訊 – 在更新目前版本之前，請考量下列需求：
  - Amazon EKS 叢集上的 Kube-proxy 有與 Kubernetes 相同的相容性和差異政策。
  - Kube-proxy 的次要版本必須與 Amazon EC2 節點上的 kubelet 次要版本相同。
  - Kube-proxy 不能比叢集控制平面的次要版本還要新。
  - Amazon EC2 節點上的 kube-proxy 版本不能為比控制平面舊兩個以上的次要版本。例如，如果您的控制平面執行 Kubernetes 1.30，則kube-proxy次要版本不能早於 1.28。
  - 如果您最近已將叢集更新至新的 Kubernetes 次要版本，請先將 Amazon EC2 節點更新至相同的次要版本，然後再將 kube-proxy 更新至與您的節點相同的次要版本。

Amazon EBS CSI 驅動程式

- 名稱 – aws-ebs-csi-driver
- 說明 – 為您的叢集提供原生 Amazon EBS 儲存的 Kubernetes 容器儲存界面 (CSI) 外掛程式。
- 所需的 IAM 許可 – 此附加元件會利用 Amazon EKS 的服務帳戶的 IAM 角色功能。AmazonEBSCSIDriverPolicy AWS 受管理策略中的權限是必需的。建立 IAM 角色，並按照下列命令連接受管政策。將 *my-cluster* 取代為您的叢集名稱，並將 *AmazonEKS_EBS_CSI_DriverRole* 取代為您角色的名稱。此命令要求您在裝置上安裝 eksctl。如果您需要使用其他工具，或需要使用自訂 KMS 金鑰進行加密，請參閱 建立 Amazon EBS CSI 驅動程式 IAM 角色。

```
eksctl create iamserviceaccount \
    --name ebs-csi-controller-sa \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKS_EBS_CSI_DriverRole \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy
 \
    --approve
```

- 其他資訊 – 若要進一步瞭解附加元件，請參閱 Amazon EBS CSI 驅動程式。

Amazon EFS CSI 驅動程式

- 名稱 – `aws-efs-csi-driver`
- 描述 – 為您的叢集提供原生 Amazon EFS 儲存的 Kubernetes 容器儲存界面 (CSI) 外掛程式。
- 所需的 IAM 許可 – 此附加元件會利用 Amazon EKS 的服務帳戶的 IAM 角色功能。AmazonEFSCSIDriverPolicy AWS 受管理策略中的權限是必需的。您可以建立 IAM 角色，並使用下列命令連接受管政策。將 *my-cluster* 取代為您的叢集名稱，並將 *AmazonEKS_EFS_CSI_DriverRole* 取代為您角色的名稱。這些命令要求您在裝置上安裝 eksctl。如果您需要使用其他工具，請參閱 建立 IAM 角色。

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy
 \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

- 其他資訊 – 若要進一步瞭解附加元件，請參閱 Amazon EFS CSI 驅動程式。

適用於 Amazon S3 CSI 驅動程式的 Mountpoint

- 名稱 – `aws-mountpoint-s3-csi-driver`
- 說明 – 為您的叢集提供 Amazon S3 儲存的 Kubernetes 容器儲存介面 (CSI) 外掛程式。
- 所需的 IAM 許可 – 此附加元件會利用 Amazon EKS 的服務帳戶的 IAM 角色功能。建立的 IAM 角色將需要提供 S3 存取權的政策。建立政策時，請遵循 Mountpoint IAM 許可建議。或者，您可以使用 AWS 受管理的策略 AmazonS3FullAccess，但是此受管理策略授與的權限超過所需的權限 Mountpoint。

您可以建立 IAM 角色，並使用下列命令將政策連接至該角色。*###################### #### ######### AWS ## ### Amazoneks_S3_CS ######### ARN # Amazonks DriverRole _S3_CSI_ ARN#DriverRole*這些命令要求您在裝置上安裝 eksctl。如需使用 IAM 主控台的指示 AWS CLI，或請參閱建立 IAM 角色。

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

- 其他資訊 – 若要進一步瞭解附加元件，請參閱 適用於 Amazon S3 CSI 驅動程式的 Mountpoint。

CSI 快照控制器

- 名稱 – `snapshot-controller`
- 說明 – 容器儲存介面 (CSI) 快照控制器可讓您在相容 CSI 驅動程式 (例如 Amazon EBS CSI 驅動程式) 中使用快照功能。
- 必要的 IAM 許可 – 此附加元件不需要任何許可。
- 其他資訊 – 若要進一步瞭解附加元件，請參閱 CSI 快照控制器。

AWS 發行版 OpenTelemetry

- 名稱 – `adot`
- 說明 — OpenTelemetry（ADOT）的發行AWS 版是項目的安全，可用於生產， AWS 支持的發行版。 OpenTelemetry
- 必要的 IAM 許可 – 只有當您使用其中一個可透過進階配置選擇加入的預先設定自訂資源時，此附加元件才需要 IAM 許可。

- 其他資訊 — 如需詳細資訊，請參閱發行AWS 版中有關OpenTelemetry使用 EKS 附加元件的發行 AWS 版入門以取得說明文件。OpenTelemetry

  ADOT 要求將 `cert-manager` 作為先決條件部署在叢集上，否則如果直接使用 Amazon EKS Terraform `cluster_addons` 屬性部署，此附加元件將無法運作。如需更多需求，請參閱發行版中 EKS 附加元件的開始OpenTelemetry使用 AWS 發行版的要求以取 AWS 得說明文件。OpenTelemetry

Amazon GuardDuty 代理

- 名稱 – `aws-guardduty-agent`
- 說明 — Amazon GuardDuty 是一種安全監控服務，可分析和處理基礎資料來源，包括 AWS CloudTrail 管理事件和 Amazon VPC 流程日誌。Amazon GuardDuty 也會處理Kubernetes稽核日誌和執行階段監控等功能。
- 必要的 IAM 許可 – 此附加元件不需要任何許可。
- 其他資訊 — 如需詳細資訊，請參閱 Amazon EKS 叢集的執行階段監控。 GuardDuty
  - 若要偵測 Amazon EKS 叢集中潛在的安全威脅，請啟用 Amazon GuardDuty 執行階段監控，並將GuardDuty安全代理程式部署到 Amazon EKS 叢集。

Amazon CloudWatch 可觀測代理

- 名稱 – `amazon-cloudwatch-observability`
- 說明 Amazon CloudWatch 代理是 AWS 提供的監控和可觀察性服務。此附加元件會安裝 CloudWatch 代理 CloudWatch 程式，並透過增強 Amazon EKS 的可觀察性啟用應用程式訊號和 CloudWatch 容器洞見。
- 所需的 IAM 許可 – 此附加元件會利用 Amazon EKS 的服務帳戶的 IAM 角色功能。在AWSXrayWriteOnlyAccess和CloudWatchAgentServer策略 AWS 管理的策略中需要權限。您可以建立 IAM 角色，為該角色附加受管政策，並使用下列命令為附加元件使用的 Kubernetes 服務帳戶加上注釋。將 *`my-cluster`* 取代為您的叢集名稱，並將 *`AmazonEKS_Observability_role`* 取代為您角色的名稱。此命令要求您在裝置上安裝 `eksctl`。如果您需要使用其他工具來建立角色、為其附加政策並註釋 Kubernetes 服務帳戶，請參閱 設定Kubernetes服務帳戶以擔任 IAM 角色。

```
eksctl create iamserviceaccount \
    --name cloudwatch-agent \
    --namespace amazon-cloudwatch \
    --cluster my-cluster \
```

```
    --role-name AmazonEKS_Observability_Role \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/AWSXrayWriteOnlyAccess \
    --attach-policy-arn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy \
    --approve
```

- 其他資訊 — 如需詳細資訊，請參閱安裝 CloudWatch 代理程式。

Amazon EKS Pod 身分識別代理程式

- 名稱 – `eks-pod-identity-agent`
- 說明 — Amazon EKS Pod 身分提供管理應用程式登入資料的功能，類似於執行個體設定檔為 EC2 執行個體提供登入資料的方式。 Amazon EC2
- 必要的 IAM 許可 – 此附加元件使用者許可來自 Amazon EKS 節點 IAM 角色。
- 更新資訊 — 您一次只能更新一個次要版本。例如，如果目前的版本是 `1.28.`*`x`*`-eksbuild.`*`y`*，並且您想要更新至 `1.30.`*`x`*`-eksbuild.`*`y`*，則您必須將目前的版本更新至 `1.29.`*`x`*`-eksbuild.`*`y`*，然後再更新至 `1.30.`*`x`*`-eksbuild.`*`y`*。如需更新附加元件的詳細資訊，請參閱 更新 Amazon EKS 附加元件。

來自獨立軟體廠商的其他 Amazon EKS 附加元件

除了先前的 Amazon EKS 附加元件清單之外，您還可以新增來自獨立軟體廠商的各種操作軟體 Amazon EKS 附加元件。選擇附加元件以進一步了解它及其安裝需求。

從 AWS Marketplace 尋找、採購和部署附加元件到 Amazon EKS () YouTube。

Accuknox

- 發布者：Accuknox
- 名稱 – `accuknox_kubearmor`
- 命名空間：`kubearmor`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 — 請參閱 KubeArmor文件 KubeArmor中的入門使用。

Akuity

- 發布者：Akuity
- 名稱 – `akuity_agent`
- 命名空間：`akuity`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 — 請參閱 Akuity 平台說明文件中的 Akuity EKS 附加元件在 Amazon EKS 上安裝 Akuity 代理程式。

Calyptia

- 發布者：Calyptia
- 名稱 – `calyptia_fluent-bit`
- 命名空間：`calytia-fluentbit`
- 服務帳戶名稱：`clyptia-fluentbit`
- AWS 受管的 IAM 政策 — AWSMarketplaceMeteringRegisterUsage.
- 建立所需 IAM 角色的命令：下列命令要求您擁有叢集的 IAM OpenID Connect (OIDC) 提供者。若要判斷您是否已經擁有一個，或是需要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。將 *my-cluster* 取代為您的叢集名稱，並將 *my-calyptia-role* 取代為您角色的名稱。此命令要求您在裝置上安裝 `eksctl`。如果您需要使用其他工具來建立角色並為 Kubernetes 服務帳戶加上註釋，請參閱 設定Kubernetes服務帳戶以擔任 IAM 角色。

```
eksctl create iamserviceaccount --name service-account-name  --namespace calyptia-
fluentbit --cluster my-cluster --role-name my-calyptia-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/
AWSMarketplaceMeteringRegisterUsage --approve
```

- 設定和使用說明 – 請參閱 Calyptia 文件中的適用於 Fluent Bit 的 Calyptia。

Cisco Observability Collector

- 發布者：Cisco
- 名稱 – `cisco_cisco-cloud-observability-collectors`

- 命名空間：appdynamics
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 — 請參閱思科文件中的使用思科雲端可觀測性 AWS Marketplace 附加元 AppDynamics件。

Cisco Observability Operator

- 發布者：Cisco
- 名稱 – cisco_cisco-cloud-observability-operators
- 命名空間：appdynamics
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 — 請參閱思科文件中的使用思科雲端可觀測性 AWS Marketplace 附加元 AppDynamics件。

CLOUDSOFT

- 發布者：CLOUDSOFT
- 名稱 – cloudsoft_cloudsoft-amp
- 命名空間：cloudsoft-amp
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設置和使用說明 — 請參閱 Amazon EKS 插件在 CLOUDSOFT 文檔中。

Cribl

- 發布者：Cribl
- 名稱 – cribl_cribledge

- 命名空間：cribledge
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 Cribl 文件中的安裝適用於 Edge 的 Cribl Amazon EKS 附加元件。

Dynatrace

- 發布者：Dynatrace
- 名稱 – dynatrace_dynatrace-operator
- 命名空間：dynatrace
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定及使用說明：請參閱 dynatrace 文件中的 Kubernetes monitoring (Kubernetes 監控)。

Datree

- 發布者：Datree
- 名稱 – datree_engine-pro
- 命名空間：datree
- 服務帳戶名稱： datree-webhook-server-awsmp
- AWS 受管的 IAM 政策 — AWSLicenseManagerConsumptionPolicy.
- 建立所需 IAM 角色的命令：下列命令要求您擁有叢集的 IAM OpenID Connect (OIDC) 提供者。若要判斷您是否已經擁有一個，或是需要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。將 *my-cluster* 取代為您的叢集名稱，並將 *my-datree-role* 取代為您角色的名稱。此命令要求您在裝置上安裝 eksctl。如果您需要使用其他工具來建立角色並為 Kubernetes 服務帳戶加上註釋，請參閱 設定Kubernetes服務帳戶以擔任 IAM 角色。

```
eksctl create iamserviceaccount --name datree-webhook-server-awsmp --namespace datree
 --cluster my-cluster --role-name my-datree-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定及使用說明：請參閱 Datree 說明文件中的 Amazon EKS-整合。

Datadog

- 發布者：Datadog
- 名稱 – `datadog_operator`
- 命名空間：`datadog-agent`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 Datadog 文件中的使用 Datadog 運算子附加元件在 Amazon EKS 上安裝 Datadog 代理程式。

Groundcover

- 發布者：groundcover
- 名稱 – `groundcover_agent`
- 命名空間：`groundcover`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 groundcover 文件中的安裝 groundcover Amazon EKS 附加元件。

Grafana Labs

- 發布者：Grafana Labs
- 名稱 – `grafana-labs_kubernetes-monitoring`
- 命名空間：`monitoring`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。

- 設定和使用說明 – 請參閱 Grafana Labs 文件中的 Configure Kubernetes Monitoring as an Add-on with Amazon EKS 一節。

HA Proxy

- 發布者：HA Proxy
- 名稱 – `haproxy-technologies_kubernetes-ingress-ee`
- 命名空間：`haproxy-controller`
- 服務帳戶名稱：`customer defined`
- AWS 受管的 IAM 政策 — AWSLicenseManagerConsumptionPolicy.
- 建立所需 IAM 角色的命令：下列命令要求您擁有叢集的 IAM OpenID Connect (OIDC) 提供者。若要判斷您是否已經擁有一個，或是需要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。將 *my-cluster* 取代為您的叢集名稱，並將 *my-haproxy-role* 取代為您角色的名稱。此命令要求您在裝置上安裝 `eksctl`。如果您需要使用其他工具來建立角色並為 Kubernetes 服務帳戶加上註釋，請參閱 設定Kubernetes服務帳戶以擔任 IAM 角色。

```
eksctl create iamserviceaccount --name service-account-name  --namespace haproxy-
controller --cluster my-cluster --role-name my-haproxy-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和用量指示 — 請參閱 HAProxy 文件中的「在 Amazon EKS from AWS上安裝 HAProxy Enterprise Kubernetes Ingress Controller」。

Kpow

- 發布者：Factorhouse
- 名稱 – `factorhouse_kpow`
- 命名空間：`factorhouse`
- 服務帳戶名稱：`kpow`
- AWS 受管理的 IAM 政策 — AWSLicenseManagerConsumptionPolicy
- 建立所需 IAM 角色的命令：下列命令要求您擁有叢集的 IAM OpenID Connect (OIDC) 提供者。若要判斷您是否已經擁有一個，或是需要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。將 *my-cluster* 取代為您的叢集名稱，並將 *my-kpow-role* 取代為您角色的名稱。此命令要求您在裝置

上安裝 eksctl。如果您需要使用其他工具來建立角色並為 Kubernetes 服務帳戶加上註釋，請參閱設定Kubernetes服務帳戶以擔任 IAM 角色。

```
eksctl create iamserviceaccount --name kpow --namespace factorhouse --cluster my-cluster --role-name my-kpow-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/AWSLicenseManagerConsumptionPolicy --approve
```

- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定及使用說明：請參閱 Kpow 文件中的 AWS Marketplace LM。

Kubecost

- 發布者：Kubecost
- 名稱 – kubecost_kubecost
- 命名空間：kubecost
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 — 請參閱Kubecost文件中的AWS 帳單整合。
- 如果您的叢集是 1.23 版或更新版本，則必須在您的叢集上安裝 the section called “Amazon EBS CSI 驅動程式”，否則您會收到錯誤。

Kasten

- 發布者：Kasten by Veeam
- 名稱 – kasten_k10
- 命名空間：kasten-io
- 服務帳戶名稱：k10-k10
- AWS 受管的 IAM 政策 — AWSLicenseManagerConsumptionPolicy.
- 建立所需 IAM 角色的命令：下列命令要求您擁有叢集的 IAM OpenID Connect (OIDC) 提供者。若要判斷您是否已經擁有一個，或是需要建立一個，請參閱 為您的叢集建立 IAM OIDC 提供者。將 *my-cluster* 取代為您的叢集名稱，並將 *my-kasten-role* 取代為您角色的名稱。此命令要求您在裝置上安裝 eksctl。如果您需要使用其他工具來建立角色並為 Kubernetes 服務帳戶加上註釋，請參閱 設定Kubernetes服務帳戶以擔任 IAM 角色。

```
eksctl create iamserviceaccount --name k10-k10 --namespace kasten-io --cluster my-
cluster --role-name my-kasten-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 — 請參閱 Kasten 文件中有關 AWS 使用 Amazon EKS 附加元件的安裝 K10。
- 其他資訊 — 如果您的 Amazon EKS 叢集是版本 Kubernetes `1.23` 或更新版本，您必須在叢集上以預設的 `StorageClass` 來安裝 Amazon EBS CSI 驅動程式。

Kong

- 發布者：Kong
- 名稱 – `kong_konnect-ri`
- 命名空間：`kong`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 Kong 文件中的安裝 Kong Gateway EKS 附加元件。

LeakSignal

- 發布者：LeakSignal
- 名稱 – `leaksignal_leakagent`
- 命名空間：`leakagent`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 — 請參閱文件中的安裝 LeakAgent 附加元 LeakSignal 件。

NetApp

- 發布者：NetApp

- 名稱 – `netapp_trident-operator`
- 命名空間：`trident`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 安裝和使用說明 — 請參閱文件中的設定 Astra 三叉戟 EKS 附加元件。 NetApp

New Relic

- 發布者：New Relic
- 名稱 – `new-relic_kubernetes-operator`
- 命名空間：`newrelic`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 New Relic 文件中的安裝適用於 EKS 的 New Relic 附加元件。

Rafay

- 發布者：Rafay
- 名稱 – `rafay-systems_rafay-operator`
- 命名空間：`rafay-system`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 Rafay 文件中的安裝 Rafay Amazon EKS 附加元件。

Solo.io

- 發布者：Solo.io
- 名稱 – `solo-io_istio-distro`
- 命名空間：`istio-system`

- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 Solo.io 說明文件中的安裝 Istio。

Stormforge

- 發布者：Stormforge
- 名稱 – `stormforge_optimize-Live`
- 命名空間：`stormforge-system`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 安裝與使用說明 — 請參閱說明 StormForge文件中的安裝 StormForge 代理程式。

Splunk

- 發布者：Splunk
- 名稱 – `splunk_splunk-otel-collector-chart`
- 命名空間：`splunk-monitoring`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定和使用說明 – 請參閱 Splunk 文件中的 Install the Splunk add-on for Amazon EKS 一節。

Teleport

- 發布者：Teleport
- 名稱 – `teleport_teleport`
- 命名空間：`teleport`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。

- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定及使用說明：請參閱 Teleport 文件中的 How Teleport Works (瞬間傳送的運作原理)。

Tetrate

- 發布者 – Tetrate Io
- 名稱 – `tetrate-io_istio-distro`
- 命名空間：`istio-system`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定及使用說明：請參閱 Tetrate Istio Distro 網站。

Upbound Universal Crossplane

- 發布者：Upbound
- 名稱 – `upbound_universal-crossplane`
- 命名空間：`upbound-system`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 設定及使用說明：請參閱 Upbound 文件中的 Upbound Universal Crossplane (UXP)。

Upwind

- 發布者：Upwind
- 名稱 – `upwind`
- 命名空間：`upwind`
- 服務帳戶名稱：服務帳戶不適用於此附加元件。
- AWS 受管 IAM 政策 — 受管政策不適用於此附加元件。
- 自訂 IAM 許可：自訂許可不適用於此附加元件。
- 安裝和使用說明 — 請參閱 Upwind 文件中的安裝步驟。

管理 Amazon EKS 附加元件

Amazon EKS 附加元件是 Amazon EKS 叢集彙整的附加元件軟體集。所有 Amazon EKS 附加元件：

- 包括最新的安全性修補程式和錯誤修正。
- 通過 AWS 與 Amazon EKS 合作驗證。
- 減少管理附加元件軟體所需的工作量。

當 Amazon EKS 附加元件有新版本可用時， AWS Management Console 會通知您。您只需啟動更新，Amazon EKS 會為您更新附加元件軟體。

如需可用附加元件的清單，請參閱 來自 Amazon EKS 的可用 Amazon EKS 附件元件。如需有關 Kubernetes 欄位管理的詳細資訊，請參閱 Kubernetes 欄位管理

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。

建立附加元件

您可以使用eksctl、或建立 Amazon EKS 附加元件。 AWS Management Console AWS CLI如果附加元件需要 IAM 角色，則請參閱 來自 Amazon EKS 的可用 Amazon EKS 附件元件 中特定附加元件的詳細資訊，以了解有關建立角色的詳細資訊。

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 `0.183.0` 或更新版本的 `eksctl` 命令列工具。如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的安裝一節。

使用 **`eksctl`** 建立 Amazon EKS 附加元件

1. 檢視叢集版本可用附加元件的名稱。使用您叢集的版本取代 *`1.30`*。

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 | grep AddonName
```

範例輸出如下。

```
"AddonName": "aws-ebs-csi-driver",
```

```
                    "AddonName": "coredns",
                    "AddonName": "kube-proxy",
                    "AddonName": "vpc-cni",
                    "AddonName": "adot",
                    "AddonName": "dynatrace_dynatrace-operator",
                    "AddonName": "upbound_universal-crossplane",
                    "AddonName": "teleport_teleport",
                    "AddonName": "factorhouse_kpow",
                    [...]
```

2. 檢視您想要建立之附加元件的可用版本。使用您叢集的版本取代 *1.30*。請使用您想要檢視其版本之附加元件的名稱取代 *name-of-addon*。名稱必須是之前步驟中傳回的其中一個名稱。

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 --name name-of-addon | grep AddonVersion
```

下列輸出是針對名為 vpc-cni 的附加元件傳回的範例。您可以看到附加元件有數個可用版本。

```
"AddonVersions": [
    "AddonVersion": "v1.12.0-eksbuild.1",
    "AddonVersion": "v1.11.4-eksbuild.1",
    "AddonVersion": "v1.10.4-eksbuild.1",
    "AddonVersion": "v1.9.3-eksbuild.1",
```

3. 判斷您要建立的附加元件是 Amazon EKS 還是 AWS Marketplace 附加元件。 AWS Marketplace 具有第三方附加元件，需要您完成其他步驟才能建立附加元件。

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 --name name-of-addon | grep ProductUrl
```

如果沒有傳回輸出，則該附加元件是 Amazon EKS。如果返回輸出，則該附加組件是一個 AWS Marketplace 附加組件。下列輸出適用於名為 teleport_teleport 的附加元件。

```
"ProductUrl": "https://aws.amazon.com/marketplace/pp?
sku=3bda70bb-566f-4976-806c-f96faef18b26"
```

您可以在返回的 URL 中了解有關附加元件 AWS Marketplace 的更多資訊。如果附加元件需要訂閱，您可以透過 AWS Marketplace訂閱附加元件。如果您要從建立附加元件 AWS Marketplace，則您用來建立附加元件的 IAM 主體必須具有建

立AWSServiceRoleForAWSLicenseManagerRole服務連結角色的權限。如需有關將許可指派給 IAM 實體的詳細資訊，請參閱《IAM 使用者指南》中的新增和移除 IAM 身分許可。

4. 建立 Amazon EKS 附加元件。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令：

   - 使用您叢集的名稱取代 *my-cluster*。
   - 使用您要建立之附加元件的名稱取代 *name-of-addon*。
   - 如果您想要比最新版本更早的附加元件版本，請使用您要使用的上一步輸出中傳回的版本編號取代 *latest*。
   - 如果附加元件使用服務帳戶角色，請使用您的帳戶 ID 取代 *111122223333*，並使用角色名稱取代 *role-name*。如需為您的服務帳戶建立角色的說明，請參閱您所建立之附加元件的文件。若要指定服務帳戶角色，您的叢集需要具有 IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。

     如果附加元件未使用服務帳戶角色，請刪除 ***--service-account-role-arn arn:aws:iam::111122223333:role/role-name***。
   - 此範例命令會覆寫任何現有自我管理附加元件版本的組態 (如果有的話)。如果您不想覆寫現有自我管理附加元件的組態，請移除 *--force* 選項。如果您移除該選項，並且 Amazon EKS 附加元件需要覆蓋現有自我管理附加元件的組態，則建立 Amazon EKS 附加元件的操作會失敗並會出現一條錯誤訊息，以幫助您解決衝突。指定此選項之前，請確定 Amazon EKS 附加元件未管理您需要管理的設定，因為這些設定會被此選項覆寫。

     ```
     eksctl create addon --cluster my-cluster --name name-of-addon --version latest \
        --service-account-role-arn arn:aws:iam::111122223333:role/role-name --force
     ```

您可以查看該命令所有可用選項的清單。

```
eksctl create addon --help
```

如需有關可用選項的詳細資訊，請參閱 `eksctl` 文件中的 Addons (附加元件)。

AWS Management Console

若要建立 Amazon EKS 附加元件 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取 Clusters (叢集)，然後選取您要為其建立附加元件之叢集的名稱。
3. 選擇附加元件索引標籤。
4. 選擇取得更多附加元件。
5. 選擇您要新增至叢集的附加元件。您可以根據需要新增任意多個 Amazon EKS 附加元件和 AWS Marketplace 附加元件。

   對於AWS Marketplace附加元件，您用來建立附加元件的 IAM 主體必須具有從中讀取附加元件的權利的 AWS LicenseManager權限。 AWS LicenseManager 需要AWSServiceRoleForAWSLicenseManagerRole服務連結角色 (SLR)，以允許 AWS 資源代表您管理授權。SLR 是每個帳戶的一次性要求，您不必為每個附加元件或每個叢集建立單獨的 SLR。如需有關將許可指派給 IAM 主體的詳細資訊，請參閱《IAM 使用者指南》中的新增和移除 IAM 身分許可。

   如果未列出您要安裝的 AWS Marketplace 附加元件，則您可以在搜尋方框中輸入文字來搜尋可用的附加元件。您也可以在篩選選項中，依類別、廠商或定價模式搜尋，然後從搜尋結果中選擇附加元件。選取您要安裝的附加元件後，請選擇 Next (下一步)。
6. 在 Configure selected add-ons settings (設定選取的附加元件設定) 頁面中：
   - 選擇檢視訂閱選項即可開啟訂閱選項表單。檢閱定價詳細資料和法律區段，然後選擇訂閱按鈕以繼續。
   - 針對 Version (版本)，選取您想要安裝的版本。除非您建立的個別附加元件建議使用不同的版本，否則我們建議您使用標記為最新的版本。若要判斷附加元件是否有建議的版本，請參閱您所建立之附加元件的文件。
   - 如果您選取的所有附加元件在 Status (狀態) 下都有 Requires subscription (需要訂閱)，請選取 Next (下一步)。在您建立叢集之後訂閱這些附加元件之前，您無法進一步設定這些附加元件。針對 Status (狀態) 下沒有 Requires subscription (需要訂閱) 的附加元件：
     - 針對 Select IAM role (選取 IAM 角色)，請接受預設選項，除非附加元件需要 IAM 許可。如果附加元件需要 AWS 許可，您可以使用節點的 IAM 角色 (未設定) 或建立用於附加元件的現有角色。若沒有可選取的角色，則表示您沒有現有角色。無論您選擇哪個選項，請參閱您所建立之附加元件的文件，以建立 IAM 政策並將其附加到角色。若要選取 IAM 角

色，您的叢集需要具有 IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。

- 選擇 Optional configuration settings (選用組態設定)。
  - 如果附加元件需要組態，請在 Configuration values (組態值) 方塊中輸入。若要判斷附加元件是否需要組態資訊，請參閱您所建立之附加元件的文件。
  - 對於 Conflict resolution method (衝突解決方法)，請選取其中一個可用選項。
- 選擇 Next (下一步)。

7. 在檢閱並新增頁面上，選擇建立。附加元件安裝完成後，您會看到已安裝的附加元件。
8. 如果您安裝的任何附加元件需要訂閱，請完成下列步驟：
   1. 選擇附加元件右下角的 Subscribe (訂閱) 按鈕。系統將帶您前往 AWS Marketplace中附加元件的頁面。請閱讀附加元件的相關資訊，例如其產品概觀和定價資訊。
   2. 選取附加元件頁面右上方的 Continue to Subscribe (繼續訂閱) 按鈕。
   3. 請完整閱讀條款與條件。如果您同意，請選擇 Accept Terms (接受條款)。處理訂閱可能需要幾分鐘的時間。訂閱正在處理時，Return to Amazon EKS Console (返回至 Amazon EKS 主控台) 按鈕會呈現灰色。
   4. 訂閱完成處理後，Return to Amazon EKS Console (返回至 Amazon EKS 主控台) 按鈕將不再呈現灰色。選擇按鈕以返回叢集的 Amazon EKS 主控台 Add-ons (附加元件) 索引標籤。
   5. 針對您訂閱的附加元件，請選擇 Remove and reinstall (移除後重新安裝)，然後選擇 Reinstall add-on (重新安裝附加元件)。安裝附加元件可能需要幾分鐘時間。安裝完成後，您可以設定附加元件。

AWS CLI

先決條件

您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本`1.27.160`或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。`2.12.3`若要檢查您目前的版本，請使用 **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**。如 `yum`、`apt-get` 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

若要建立 Amazon EKS 附加元件 AWS CLI

1. 判斷有哪些附加元件可用。您可以查看所有可用的附加元件、其類型及其發佈者。您還可以透過 AWS Marketplace查看可用附加元件的 URL。使用您叢集的版本取代 *1.30*。

```
aws eks describe-addon-versions --kubernetes-version 1.30 \
    --query 'addons[].{MarketplaceProductUrl: marketplaceInformation.productUrl,
 Name: addonName, Owner: owner Publisher: publisher, Type: type}' --output table
```

範例輸出如下。

```
-------------------------------------------------------------------------------------------
|
 DescribeAddonVersions
       |
+-----------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
|                     MarketplaceProductUrl                      |
 Name              |      Owner       |  Publisher   |        Type         |
+-----------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
|  None                                                          |  aws-ebs-csi-
driver           |  aws             |  eks         |  storage            |
|  None                                                          |  coredns
                |  aws             |  eks         |  networking         |
|  None                                                          |  kube-proxy
                |  aws             |  eks         |  networking         |
|  None                                                          |  vpc-cni
                |  aws             |  eks         |  networking         |
|  None                                                          |  adot
                |  aws             |  eks         |  observability      |
|  https://aws.amazon.com/marketplace/pp/prodview-brb73nceicv7u |
 dynatrace_dynatrace-operator |  aws-marketplace |  dynatrace   |  monitoring
      |
|  https://aws.amazon.com/marketplace/pp/prodview-uhc2iwi5xysoc |
 upbound_universal-crossplane |  aws-marketplace |  upbound     |  infra-
management    |
|  https://aws.amazon.com/marketplace/pp/prodview-hd2ydsrgqy4li |
 teleport_teleport            |  aws-marketplace |  teleport    |  policy-
management  |
```

```
|  https://aws.amazon.com/marketplace/pp/prodview-vgghgqdsplhvc |
 factorhouse_kpow             |  aws-marketplace |  factorhouse |  monitoring
      |
|  [...]                                                        |  [...]
                |  [...]           |  [...]       |  [...]              |
+-----------------------------------------------------------------
+-------------------------------+------------------+--------------
+---------------------+
```

您的輸出可能不同。在此範例輸出中，有三種不同 networking 類型的附加元件可用，以及五個具有 eks 類型發佈者的附加元件。Owner 欄位中具有 aws-marketplace 的附加元件可能需要訂閱才能安裝它們。您可以造訪 URL 以深入了解附加元件並訂閱該附加元件。

2. 您可以查看每個附加元件可用的版本。使用您的叢集版本取代 *1.30*，並使用上一步傳回的附加元件名稱取代 *vpc-cni*。

```
aws eks describe-addon-versions --kubernetes-version 1.30 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

範例輸出如下。

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+----------------------+
| Defaultversion  |       Version        |
+-----------------+----------------------+
|  False          |  v1.12.0-eksbuild.1  |
|  True           |  v1.11.4-eksbuild.1  |
|  False          |  v1.10.4-eksbuild.1  |
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

Defaultversion 欄位中具有 True 的版本在預設情況下是用來建立附加元件的版本。

3. (選用) 執行下列命令，尋找您所選附加元件的組態選項：

```
aws eks describe-addon-configuration --addon-name vpc-cni --addon-
version v1.12.0-eksbuild.1
```

```
{
```

```
    "addonName": "vpc-cni",
    "addonVersion": "v1.12.0-eksbuild.1",
    "configurationSchema": "{\"$ref\":\"#/definitions/VpcCni\",\"$schema
\":\"http://json-schema.org/draft-06/schema#\",\"definitions\":{\"Cri\":
{\"additionalProperties\":false,\"properties\":{\"hostPath\":{\"$ref\":
\"#/definitions/HostPath\"}},\"title\":\"Cri\",\"type\":\"object\"},\"Env
\":{\"additionalProperties\":false,\"properties\":{\"ADDITIONAL_ENI_TAGS
\":{\"type\":\"string\"},\"AWS_VPC_CNI_NODE_PORT_SUPPORT\":{\"format\":
\"boolean\",\"type\":\"string\"},\"AWS_VPC_ENI_MTU\":{\"format\":\"integer
\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CONFIGURE_RPFILTER\":{\"format
\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG\":
{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_EXTERNALSNAT
\":{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOGLEVEL
\":{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOG_FILE\":{\"type
\":\"string\"},\"AWS_VPC_K8S_CNI_RANDOMIZESNAT\":{\"type\":
\"string\"},\"AWS_VPC_K8S_CNI_VETHPREFIX\":{\"type\":\"string
\"},\"AWS_VPC_K8S_PLUGIN_LOG_FILE\":{\"type\":\"string\"},
\"AWS_VPC_K8S_PLUGIN_LOG_LEVEL\":{\"type\":\"string\"},\"DISABLE_INTROSPECTION
\":{\"format\":\"boolean\",\"type\":\"string\"},\"DISABLE_METRICS\":{\"format
\":\"boolean\",\"type\":\"string\"},\"DISABLE_NETWORK_RESOURCE_PROVISIONING
\":{\"format\":\"boolean\",\"type\":\"string\"},\"ENABLE_POD_ENI\":{\"format
\":\"boolean\",\"type\":\"string\"},\"ENABLE_PREFIX_DELEGATION\":{\"format
\":\"boolean\",\"type\":\"string\"},\"WARM_ENI_TARGET\":{\"format\":\"integer
\",\"type\":\"string\"},\"WARM_PREFIX_TARGET\":{\"format\":\"integer\",
\"type\":\"string\"}},\"title\":\"Env\",\"type\":\"object\"},\"HostPath\":
{\"additionalProperties\":false,\"properties\":{\"path\":{\"type\":\"string\"}},
\"title\":\"HostPath\",\"type\":\"object\"},\"Limits\":{\"additionalProperties
\":false,\"properties\":{\"cpu\":{\"type\":\"string\"},\"memory\":{\"type
\":\"string\"}},\"title\":\"Limits\",\"type\":\"object\"},\"Resources\":
{\"additionalProperties\":false,\"properties\":{\"limits\":{\"$ref\":\"#/
definitions/Limits\"},\"requests\":{\"$ref\":\"#/definitions/Limits\"}},
\"title\":\"Resources\",\"type\":\"object\"},\"VpcCni\":{\"additionalProperties
\":false,\"properties\":{\"cri\":{\"$ref\":\"#/definitions/Cri\"},\"env\":
{\"$ref\":\"#/definitions/Env\"},\"resources\":{\"$ref\":\"#/definitions/
Resources\"}},\"title\":\"VpcCni\",\"type\":\"object\"}}}"
}
```


輸出是標準的 JSON 結構描述。

以下是適用於上述結構描述的有效組態值 (以 JSON 格式) 範例。


```
{
  "resources": {
```

```
    "limits": {
      "cpu": "100m"
    }
  }
}
```

以下是適用於上述結構描述的有效組態值 (以 YAML 格式) 範例。

```
resources:
  limits:
    cpu: 100m
```

4. 判斷附加元件是否需要 IAM 許可。如果是這樣，您需要 (1) 判斷是否要針對服務帳戶使用 EKS 網繭身分或 IAM 角色 (IRSA)，(2) 決定要與附加元件搭配使用之 IAM 角色的 ARN，以及 (3) 決定附加元件使用的 Kubernetes 服務帳戶的名稱。您可以在文件中或使用 AWS API 找到此資訊，請參閱擷取附加元件的 IAM 資訊。

   - 如果附加元件支援，Amazon EKS 建議使用 EKS 網繭身分識別。這需要在叢集上安裝網繭身分識別代理程式。如需搭配附加元件使用網繭識別的相關資訊，請參閱使用網繭身分將 IAM 角色附加至 Amazon EKS 附加元件。
   - 如果附加元件或您的叢集未設定 EKS 網繭身分識別，請使用 IRSA。確認您的叢集上已設定 IRSA。
   - 請參閱 Amazon EKS 附加元件文件，以判斷附加元件是否需要 IAM 許可，以及相關聯 Kubernetes 服務帳戶的名稱。

5. 建立 Amazon EKS 附加元件。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令：

   - 使用您叢集的名稱取代 *my-cluster*。
   - 使用您想要建立之上一步輸出中傳回的附加元件名稱取代 *vpc-cni*。
   - 使用您想要使用之上一步輸出中傳回的版本取代 *version-number*。
   - 如果附加元件不需要 IAM 許可，請刪除*<service-account-configuration>*。
   - 如果附加元件 (1) 需要 IAM 許可，且 (2) 您的叢集使用 EKS Pod 身分識別，請以下列網繭身分識別關聯取代*<service-account-configuration>*。替換*<service-account-name>*為附加元件使用的服務帳戶名稱。*<role-arn>*以 IAM 角色的 ARN 取代。角色必須具有 EKS 網繭識別所需的信任原則。

    - `--pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'`

- 如果附加元件 (1) 需要 IAM 許可，且 (2) 您的叢集使用 IRSA，請以下列 IRSA 組態取代*<service-account-configuration>*。以您*111122223333*的帳戶 ID 和*role-name*您已建立的現有 IAM 角色名稱取代。如需建立角色的說明，請參閱您所建立之附加元件的文件。若要指定服務帳戶角色，您的叢集需要具有 IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。

    - `--service-account-role-arn arn:aws:iam::111122223333:role/role-name`

- 這些範例命令會覆寫任何現有自我管理附加元件版本的 `--configuration-values` 選項 (如果有的話)。將其取代為所需的組態值，例如字串或檔案輸入。如果您不想提供組態值，請刪除 `--configuration-values` 選項。如果您不想覆寫現有自我管理附加元件的組態，請移除該*--resolve-conflicts OVERWRITE*選項。 AWS CLI 如果您移除該選項，並且 Amazon EKS 附加元件需要覆蓋現有自我管理附加元件的組態，則建立 Amazon EKS 附加元件的操作會失敗並會出現一條錯誤訊息，以幫助您解決衝突。指定此選項之前，請確定 Amazon EKS 附加元件未管理您需要管理的設定，因為這些設定會被此選項覆寫。

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    <service-account-configuration> --configuration-values '{"resources":
{"limits":{"cpu":"100m"}}}' --resolve-conflicts OVERWRITE
```

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    <service-account-configuration> --configuration-values 'file://example.yaml'
 --resolve-conflicts OVERWRITE
```

如需可用選項的完整清單，請參閱 Amazon EKS Command Line Reference (《Amazon EKS 命令列參考》) 中的 `create-addon`。如果您建立的附加元件有 `aws-marketplace` 列在上一步的 `Owner` 欄位中，則建立操作可能會失敗，而且您可能會收到類似下列錯誤的錯誤訊息。

```
{
    "addon": {
        "addonName": "addon-name",
```

```
        "clusterName": "my-cluster",
        "status": "CREATE_FAILED",
        "addonVersion": "version",
        "health": {
            "issues": [
                {
                    "code": "AddonSubscriptionNeeded",
                    "message": "You are currently not subscribed to this add-
on. To subscribe, visit the AWS Marketplace console, agree to the seller EULA,
 select the pricing type if required, then re-install the add-on"
[...]
```

如果您收到類似上一個輸出中之錯誤的錯誤，請造訪上一步輸出中的 URL，以訂閱附加元件。訂閱後，再次執行 `create-addon` 命令。

更新附加元件

當新版本發佈或您將叢集更新到新的 Kubernetes 次要版本之後，Amazon EKS 不會自動更新附加元件。若要更新現有叢集的附加元件，您必須啟動更新。在您啟動更新後，Amazon EKS 會為您更新附加元件。在更新附加元件之前，請檢閱附加元件目前的文件。如需可用附加元件的清單，請參閱 來自 Amazon EKS 的可用 Amazon EKS 附件元件。如果附加元件需要 IAM 角色，則請參閱 來自 Amazon EKS 的可用 Amazon EKS 附件元件 中特定附加元件的詳細資訊，以了解有關建立角色的詳細資訊。

您可以使用`eksctl`、或更新 Amazon EKS 附加元件。 AWS Management Console AWS CLI

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 `0.183.0` 或更新版本的 `eksctl` 命令列工具。如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的安裝一節。

使用 **`eksctl`** 更新 Amazon EKS 附加元件

1. 判斷叢集上安裝的目前附加元件和附加元件版本。使用您叢集的名稱取代 *`my-cluster`*。

```
eksctl get addon --cluster my-cluster
```

範例輸出如下。

```
NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
```

```
coredns     v1.8.7-eksbuild.2    ACTIVE  0
kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0                 v1.23.8-eksbuild.2
vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0                 v1.12.0-
eksbuild.1,v1.11.4-eksbuild.1,v1.11.3-eksbuild.1,v1.11.2-eksbuild.1,v1.11.0-
eksbuild.1
```

您的輸出可能看起來有所不同，具體取決於您的叢集上有哪些附加元件和版本。您可以看到，在前面的範例輸出中，叢集上的兩個現有附加元件在 UPDATE AVAILABLE 欄位中具有較新的版本。

2. 更新附加元件。

   1. 將隨後的命令複製到您的裝置。視需要對命令進行下列修改：

      - 使用您叢集的名稱取代 *my-cluster*。
      - *region-code*以叢集所 AWS 區域 在的位置取代。
      - 使用您想要更新的上一步輸出中傳回的附加元件名稱取代 *vpc-cni*。
      - 如果您想要更新為比最新可用版本更早的版本，請使用您要使用的上一步輸出中傳回的版本編號取代 *latest*。某些附加元件有建議的版本。如需詳細資訊，請參閱您所更新之附加元件的文件。
      - 如果附加元件使用 Kubernetes 服務帳戶和 IAM 角色，請使用您的帳戶 ID 取代 *111122223333*，並使用您已建立的現有 IAM 角色的名稱取代 *role-name*。如需建立角色的說明，請參閱您所建立之附加元件的文件。若要指定服務帳戶角色，您的叢集需要具有 IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。

        如果附加元件未使用 Kubernetes 服務帳戶和 IAM 角色，請刪除 **serviceAccountRoleARN: arn:aws:iam::*111122223333*:role/*role-name*** 列。
      - *preserve*此選項會保留附加元件的現有值。如果您已為附加元件設定設定了自訂值，但未使用此選項，Amazon EKS 會以其預設值覆寫您的值。如果您使用此選項，建議您在更新生產叢集上的附加元件之前，測試非生產叢集上的任何欄位和值變更。如果您將此值變更為 overwrite，則所有設定都會變更為 Amazon EKS 預設值。如果您已設定任何設定的自訂值，則可能會使用 Amazon EKS 預設值覆寫這些值。如果您將此值變更為 none，Amazon EKS 不會變更任何設定的值，但更新可能會失敗。若更新失敗，您會收到錯誤訊息，以協助您解決衝突。

        ```
        cat >update-addon.yaml <<EOF
        apiVersion: eksctl.io/v1alpha5
        ```

```
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code

addons:
- name: vpc-cni
  version: latest
  serviceAccountRoleARN: arn:aws:iam::111122223333:role/role-name
  resolveConflicts: preserve
EOF
```

2. 執行修改後的命令來建立 `update-addon.yaml` 檔案。
3. 將組態檔案套用至叢集。

```
eksctl update addon -f update-addon.yaml
```

如需有關更新附加元件的詳細資訊，請參閱 `eksctl` 文件中的 Addons (附加元件)。

AWS Management Console

若要使用更新 Amazon EKS 附加元件 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取 Clusters (叢集)，然後選取您要為其設定附加元件之叢集的名稱。
3. 選擇附加元件索引標籤。
4. 選取附加元件方塊右上方的方塊，然後選擇 Edit (編輯)。
5. 在 Configure ***name of addon*** (設定 name of addon) 頁面中：
   - 選取您要使用的 Version (版本)。附加元件可能有建議的版本。如需詳細資訊，請參閱您所更新之附加元件的文件。
   - 對於選取 IAM 角色，您可以使用節點的 IAM 角色 (未設定) 或建立用於附加元件的現有角色。若沒有可選取的角色，則表示您沒有現有角色。無論您選擇哪個選項，請參閱您所建立之附加元件的文件，以建立 IAM 政策並將其附加到角色。若要選取 IAM 角色，您的叢集需要具有 IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。

- 對於 Code editor，輸入任何附加元件特定組態資訊。如需詳細資訊，請參閱您所更新之附加元件的文件。
- 對於 Conflict resolution method (衝突解決方法)，請選取其中一個選項。如果您已設定附加元件設定的自訂值，我們建議您使用 Preserve (保留) 選項。如果您不選擇此選項，Amazon EKS 會以其預設值覆寫您的值。如果您使用此選項，建議您在更新生產叢集上的附加元件之前，測試非生產叢集上的任何欄位和值變更。

6. 選擇 Update (更新)。

AWS CLI

先決條件

您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。

若要使用更新 Amazon EKS 附加元件 AWS CLI

1. 請參閱已安裝附加元件的清單。使用您叢集的名稱取代 *my-cluster*。

```
aws eks list-addons --cluster-name my-cluster
```

範例輸出如下。

```
{
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni"
    ]
}
```

2. 檢視您想要更新之附加元件的目前版本。請使用叢集名稱取代 *my-cluster* 並使用您想要更新之附加元件的名稱取代 *vpc-cni*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 "addon.addonVersion" --output text
```

範例輸出如下。

```
v1.10.4-eksbuild.1
```

3. 您可以查看哪些版本的附加元件可用於您的叢集版本。請使用叢集版本取代 *1.30* 並使用您想要更新之附加元件的名稱取代 *vpc-cni*。

```
aws eks describe-addon-versions --kubernetes-version 1.30 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

範例輸出如下。

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+----------------------+
| Defaultversion  |       Version        |
+-----------------+----------------------+
|  False          |  v1.12.0-eksbuild.1  |
|  True           |  v1.11.4-eksbuild.1  |
|  False          |  v1.10.4-eksbuild.1  |
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

`Defaultversion` 欄位中具有 `True` 的版本在預設情況下是用來建立附加元件的版本。

4. 更新您的附加元件。將隨後的命令複製到您的裝置。視需要對命令進行下列修改，然後執行修改後的命令。

- 使用您叢集的名稱取代 *my-cluster*。
- 使用您想要更新之附加元件的名稱取代 *vpc-cni*，該名稱在上一步輸出中傳回。
- 使用您想要更新之上一步輸出中傳回的版本取代 *version-number*。某些附加元件有建議的版本。如需詳細資訊，請參閱您所更新之附加元件的文件。
- 如果附加元件使用 Kubernetes 服務帳戶和 IAM 角色，請使用您的帳戶 ID 取代 *111122223333*，並使用您已建立的現有 IAM 角色的名稱取代 *role-name*。如需建立角色的說明，請參閱您所建立之附加元件的文件。若要指定服務帳戶角色，您的叢集需要具有

IAM OpenID Connect (OIDC) 提供者。若要判定您的叢集是否已經擁有一個提供者，或是要建立一個提供者，請參閱 為您的叢集建立 IAM OIDC 提供者。

如果附加元件未使用 Kubernetes 服務帳戶和 IAM 角色，請刪除 **serviceAccountRoleARN: arn:aws:iam::*111122223333*:role/*role-name*** 列。

- **--resolve-conflicts** *PRESERVE* 選項會保留附加元件的現有值。如果您已為附加元件設定設定了自訂值，但未使用此選項，Amazon EKS 會以其預設值覆寫您的值。如果您使用此選項，建議您在更新生產叢集上的附加元件之前，測試非生產叢集上的任何欄位和值變更。如果您將此值變更為 overwrite，則所有設定都會變更為 Amazon EKS 預設值。如果您已設定任何設定的自訂值，則可能會使用 Amazon EKS 預設值覆寫這些值。如果您將此值變更為 none，Amazon EKS 不會變更任何設定的值，但更新可能會失敗。若更新失敗，您會收到錯誤訊息，以協助您解決衝突。

- 如果您要移除所有自訂組態，請使用 *--configuration-values '{}'* 選項執行更新。這會將所有自訂組態設定回預設值。如果您不想變更自訂組態，請不要提供 *--configuration-values* 旗標。如果您要調整自訂組態，請使用新參數取代 *{}*。若要查看參數清單，請參閱「建立附加元件」一節中的 viewing configuration schema (檢視組態結構描述) 步驟。

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
configuration-values '{}' --resolve-conflicts PRESERVE
```

5. 查看更新狀態。請使用叢集名稱取代 *my-cluster* 並使用您所更新之附加元件的名稱取代 *vpc-cni*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

範例輸出如下。

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "UPDATING",
[...]
```

當狀態為 ACTIVE 時，即表示更新已完成。

刪除附加元件

當您刪除 Amazon EKS 附加元件時：

- 此附加元件提供的功能沒有任何停機時間。
- 如果您使用服務帳戶的 IAM 角色 (IRSA)，且附加元件具有與其相關聯的 IAM 角色，則不會移除 IAM 角色。
- 如果您使用的是網繭識別，則會刪除附加元件擁有的任何網繭身分識別相關聯程式。如果您將選--preserve項指定給 AWS CLI，則會保留相關聯。
- Amazon EKS 停止管理附加元件的設定。
- 當有新版本可用時，主控台會停止通知您。
- 您無法使用任何 AWS 工具或 API 更新附加元件。
- 您可以選擇將附加元件軟體保留在叢集上，以便您可以自我管理附加元件軟體，或從叢集中移除附加元件軟體。如果您的叢集上的任何資源都不依賴於附加元件提供的功能，您應該僅從叢集中移除附加元件軟體。

您可以使用 eksctl、 AWS Management Console或 AWS CLI從您的叢集中刪除 Amazon EKS 附加元件。

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

使用 **eksctl** 刪除 Amazon EKS 附加元件

1. 判斷叢集上目前安裝的附加元件。使用您叢集的名稱取代 *my-cluster*。

```
eksctl get addon --cluster my-cluster
```

範例輸出如下。

```
NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
```

```
coredns     v1.8.7-eksbuild.2    ACTIVE  0
kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0
vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0
[...]
```

您的輸出可能看起來有所不同，具體取決於您的叢集上有哪些附加元件和版本。

2. 刪除附加元件。使用叢集名稱取代 *my-cluster*，並使用您想要移除之上一步輸出中傳回的附加元件名稱取代 *name-of-add-on*。若移除 ***--preserve*** 選項，除了 Amazon EKS 不再管理附加元件之外，附加元件軟體也會從叢集中移除。

```
eksctl delete addon --cluster my-cluster --name name-of-addon --preserve
```

AWS Management Console

若要使用刪除 Amazon EKS 附加元件 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中，選取 Clusters (叢集)，然後選取您要為其移除 Amazon EKS 附加元件之叢集的名稱。
3. 選擇附加元件索引標籤。
4. 勾選附加元件方塊右上方的核取方塊，然後選擇 Remove (移除)。如果您希望 Amazon EKS 停止管理附加元件的設定，但想要在叢集上保留附加元件軟體，以便您可以自我管理附加元件的所有設定，請選取 Preserve on the cluster (在叢集上保留)。輸入附加元件名稱，然後選取 Remove (移除)。

AWS CLI

先決條件

已在裝置或 AWS CloudShell上安裝版本 `0.183.0` 或更新版本的 `eksctl` 命令列工具。如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的安裝一節。

若要使用刪除 Amazon EKS 附加元件 AWS CLI

1. 請參閱已安裝附加元件的清單。使用您叢集的名稱取代 *my-cluster*。

```
aws eks list-addons --cluster-name my-cluster
```

範例輸出如下。

```
{
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni",
        "name-of-addon"
    ]
}
```

2. 刪除已安裝的附加元件。請使用叢集名稱取代 *my-cluster* 並使用您想要移除之附加元件的名稱取代 *name-of-add-on*。移除 ***--preserve*** 會將附加元件軟體從叢集中移除。

```
aws eks delete-addon --cluster-name my-cluster --addon-name name-of-addon --
preserve
```

縮寫的範例輸出如下所示。

```
{
    "addon": {
        "addonName": "name-of-add-on",
        "clusterName": "my-cluster",
        "status": "DELETING",
[...]
```

3. 檢查刪除狀態。請使用叢集名稱取代 *my-cluster* 並使用您所移除之附加元件的名稱取代 *name-of-addon*。

```
aws eks describe-addon --cluster-name my-cluster --addon-name name-of-addon
```

刪除附加元件後，範例輸出如下所示。

```
An error occurred (ResourceNotFoundException) when calling the DescribeAddon
 operation: No addon: name-of-addon found in cluster: my-cluster
```

檢索插件版本兼容性

使用 describe-addon-verisionsAPI 列出 EKS 外掛程式的可用版本，以及每個附加元件版本支援的 Kubernetes 版本。

檢索插件版本兼容性（ ）AWS CLI

1. 驗證 AWS CLI 是否已安裝並使用aws sts get-caller-identity。如果此命令不起作用，請了解如何開始使用 AWS CLI.
2. 決定您要擷取版本相容性資訊的附加元件名稱，例如amazon-cloudwatch-observability。
3. 判斷叢集的 Kubernetes 版本，例如。1.28
4. 使用擷 AWS CLI 取與叢集的 Kubernetes 版本相容的附加元件版本。

```
aws eks describe-addon-versions --addon-name amazon-cloudwatch-observability --
kubernetes-version 1.29
```

範例輸出如下。

```
{
    "addons": [
        {
            "addonName": "amazon-cloudwatch-observability",
            "type": "observability",
            "addonVersions": [
                {
                    "addonVersion": "v1.5.0-eksbuild.1",
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.28",
                            "platformVersions": [
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
[...]
```

此輸出顯示附加元件版本`v1.5.0-eksbuild.1`與 Kubernetes 叢集版本相容。`1.28`

Kubernetes 欄位管理

使用標準的最佳實務組態將 Amazon EKS 附加元件安裝到您的叢集。如需將 Amazon EKS 附加元件新增至叢集的詳細資訊，請參閱 Amazon EKS 附加元件。

您可能想要自訂 Amazon EKS 附加元件的組態，以啟用進階功能。Amazon EKS 使用 Kubernetes 伺服器端套用功能來啟用 Amazon EKS 管理附加元件，而不會覆寫非由 Amazon EKS 管理的設定組態。如需詳細資訊，請參閱 Kubernetes 文件中的 伺服器端套用。為了達到此目的，Amazon EKS 會為其安裝的每個附加元件管理一組數量最少的欄位。您可以修改所有非由 Amazon EKS 管理的欄位，或另一個 Kubernetes 控制平面程序 (例如 `kube-controller-manager`)，沒有任何問題。

> **Important**
>
> 修改 Amazon EKS 管理的欄位會阻止 Amazon EKS 管理附加元件，並且可能會導致在更新附加元件時覆寫您的變更。

檢視欄位管理狀態

您可以使用 `kubectl`，查看哪些欄位由 Amazon EKS 管理，適用於任何 Amazon EKS 附加元件。

查看欄位的管理狀態

1. 判定要檢查的附加元件。若要查看部署到叢集的所有 `deployments` 和 `DaemonSets`，請參閱 檢視 Kubernetes 資源。
2. 執行下列命令以檢視附加元件的受管欄位：

```
kubectl get type/add-on-name -n add-on-namespace -o yaml
```

例如，您可以使用下列命令查看 CoreDNS 附加元件的受管欄位。

```
kubectl get deployment/coredns -n kube-system -o yaml
```

欄位管理會列在傳回輸出的下一區段中。

```
[...]
```

```
managedFields:
  - apiVersion: apps/v1
    fieldsType: FieldsV1
    fieldsV1:
[...]
```

> **Note**
>
> 如果您在輸出中看不到 `managedFields`，請新增 **`--show-managed-fields`** 至該命令並再次執行。您正在使用的 `kubectl` 版本會決定是否預設傳回受管欄位。

了解 Kubernetes API 中的欄位管理語法

檢視 Kubernetes 物件的詳細資訊時，輸出中會傳回受管和未受管的欄位。受管欄位可以是以下類型之一：

- 全受管：欄位的所有索引鍵均由 Amazon EKS 管理。修改任何值皆會導致衝突。
- 部分受管：欄位的某些索引鍵由 Amazon EKS 管理。只有對明確由 Amazon EKS 管理的索引鍵進行修改才會造成衝突。

這兩種類型的欄位均使用 `manager: eks` 標記。

每個索引鍵可以是 `.` 表示欄位本身，其一律會映射到一個空集，或映射至表示子欄位或項目的字串。欄位管理的輸出包含下列類型的宣告：

- `f:`*`name`*，其中 *`name`* 是清單中欄位的名稱。
- `k:`*`keys`*，其中 *`keys`* 是清單項目欄位的映射。
- `v:`*`value`*，其中 *`value`* 是清單項目的精準 JSON 格式化值。
- `i:`*`index`*，其中 *`index`* 是清單中項目的位置。

CoreDNS 附加元件的下列輸出部分會說明先前的宣告：

- 全受管欄位：如果受管欄位具有指定的 `f:` (欄位)，但沒有 `k:` (索引鍵)，則會管理整個欄位。修改此欄位中的任何值皆會導致衝突。

  在下列輸出中，您可以看到名為 `coredns` 的容器由 `eks` 管理。`args`、`image` 及 `imagePullPolicy` 子欄位也由 `eks` 管理。修改這些欄位中的任何值皆會導致衝突。

```
[...]
f:containers:
  k:{"name":"coredns"}:
  .: {}
  f:args: {}
  f:image: {}
  f:imagePullPolicy: {}
[...]
manager: eks
[...]
```

- 部分受管欄位：如果受管索引鍵具有指定的值，則會針對該欄位管理宣告的索引鍵。修改指定的索引鍵會導致衝突。

  在下列輸出中，您可以看到 eks 管理 config-volume 和使用 name 索引鍵設定的 tmp 磁碟區。

```
[...]
f:volumes:
  k:{"name":"config-volume"}:
    .: {}
    f:configMap:
      f:items: {}
      f:name: {}
    f:name: {}
  k:{"name":"tmp"}:
    .: {}
    f:name: {}
[...]
manager: eks
[...]
```

- 將索引鍵新增至部分受管欄位：如果只管理特定鍵值，您可以安全地將其他索引鍵 (例如引數) 新增至欄位，而不會造成衝突。如果新增其他索引鍵，請確定未先對欄位進行管理。新增或修改受管的任何值會導致衝突。

  在下列輸出中，您可以看到 name 索引鍵和 name 欄位皆為受管項目。新增或修改任何容器名稱會導致與此受管索引鍵發生衝突。

```
[...]
f:containers:
  k:{"name":"coredns"}:
```

```
[...]
    f:name: {}
[...]
manager: eks
[...]
```

使用網繭身分將 IAM 角色附加至 Amazon EKS 附加元件

某些 Amazon EKS 附加元件需要 IAM 角色許可才能呼叫 AWS API。例如，Amazon VPC CNI 附加元件會呼叫特定 AWS API 來設定帳戶中的聯網資源。這些附加元件必須使用 AWS IAM 獲得許可。更具體地說，執行附加元件之網繭的服務帳戶必須與具有足夠 IAM 政策的 IAM 角色相關聯。

向叢集工作負載授與 AWS 許可的建議方式是使用 Amazon EKS 功能網繭身分。您可以使用網繭身分關聯，將附加元件的服務帳戶對應至 IAM 角色。如果 Pod 使用具有關聯的服務帳戶，則 Amazon EKS 會在 Pod 的容器中設定環境變數。環境變數會將 AWS SDK (包括 AWS CLI) 設定為使用 EKS 網繭身分認證。深入了解 EKS 網繭身分識別。

Amazon EKS 附加元件可協助管理與附加元件對應之網繭身分關聯的生命週期。例如，您可以在單一 API 呼叫中建立或更新 Amazon EKS 附加元件和必要的網繭身分關聯。Amazon EKS 也提供用於擷取建議的 IAM 政策的 API。

建議用法:

1. 確認您的叢集上已設定 Amazon EKS 網繭身分代理程式。
2. 確定您要安裝的附加元件是否需要使用該`describe-addon-versions` AWS CLI 操作的 IAM 許可。如果標`requiresIamPermissions`誌是`true`，那麼您應該使用該`describe-addon-configurations`操作來確定插件所需的權限。回應包括建議的受管 IAM 政策清單。
3. 使用 CLI 作業擷取 Kubernetes 服務帳戶的名稱和建議的身分與存取權管理政策。`describe-addon-configuration`根據您的安全需求評估建議策略的範圍。
4. 使用建議的許可政策以及網繭身分所需的信任政策建立 IAM 角色。如需詳細資訊，請參閱 建立 EKS Pod 身分識別關聯。
5. 使用 CLI 建立或更新 Amazon EKS 附加元件。指定至少一個網繭身分識別關聯。網繭身分識別關聯為 (1) Kubernetes 服務帳戶的名稱，以及 (2) IAM 角色的 ARN。

考量：

- 使用附加元件 API 建立的網繭身分識別關聯屬於個別附加元件。如果您刪除附加元件，網繭身分識別關聯也會一併刪除。您可以在使用 AWS CLI 或 API 刪除附加元件時使用`preserve`選項來防止此

重疊刪除。如有必要，您也可以直接更新或刪除網繭身分識別關聯。附加元件無法承擔現有網繭身分識別關聯的擁有 您必須刪除現有的關聯，然後使用附加元件的建立或更新作業重新建立它。

- Amazon EKS 建議您使用網繭身分關聯來管理附加元件的 IAM 許可。仍然支援先前的方法，即服務帳戶 (IRSA) 的 IAM 角色。您可以為附加元件指定 IRSA `serviceAccountRoleArn` 和網繭身分識別關聯。如果叢集上已安裝 EKS 網繭身分識別代理程式，則`serviceAccountRoleArn`會忽略該代理程式，而 EKS 將使用提供的網繭身分識別關聯。如果未啟用網繭身分識別，`serviceAccountRoleArn`將會使用。
- 如果您更新現有附加元件的網繭身分關聯，Amazon EKS 會啟動附加元件網繭的滾動重新啟動。

擷取有關附加元件的 IAM 資訊

您可以使用 AWS CLI 來判斷 (1) 附加元件是否需要 IAM 許可，以及 (2) 該附加元件建議的 IAM 政策。

擷取有關 Amazon EKS 附加元件的 IAM 資訊 ()AWS CLI

1. 決定您要安裝的附加元件名稱，以及叢集的 Kubernetes 版本。進一步了解可用的 Amazon EKS 附加元件。
2. 使用 AWS CLI 來判斷附加元件是否需要 IAM 許可。

```
aws eks describe-addon-versions \
--addon-name <addon-name> \
--kubernetes-version <kubernetes-version>
```

例如：

```
aws eks describe-addon-versions \
--addon-name aws-ebs-csi-driver \
--kubernetes-version 1.30
```

檢閱下列範例輸出。請注意`true`，`requiresIamPermissions`是和默認的附加版本。擷取建議的 IAM 政策時，您需要指定附加元件版本。

```
{
    "addons": [
        {
            "addonName": "aws-ebs-csi-driver",
            "type": "storage",
            "addonVersions": [
```

```
                {
                    "addonVersion": "v1.31.0-eksbuild.1",
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.30",
                            "platformVersions": [
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
                    "requiresConfiguration": false,
                    "requiresIamPermissions": true
                },
[...]
```

3. 如果附加元件需要 IAM 許可，請使 AWS CLI 用擷取建議的 IAM 政策。

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name <addon-name> \
--addon-version <addon-version>
```

例如：

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name aws-ebs-csi-driver \
--addon-version v1.31.0-eksbuild.1
```

檢閱下列輸出。請記下 recommendedManagedPolicies。

```
[
    {
        "serviceAccount": "ebs-csi-controller-sa",
        "recommendedManagedPolicies": [
            "arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy"
        ]
```

```
    }
]
```

4. 建立 IAM 角色並附加建議的受管政策。或者，您也可以檢閱受管理的原則，並視需要縮減權限的範圍。檢閱建立 IAM 角色以搭配 EKS 網繭身分使用的指示。

使用 IAM 角色更新附加元件

更新 Amazon EKS 附加元件以使用網繭身分識別關聯 ()AWS CLI

1. 確定：

   - `cluster-name`— 要安裝附加元件的 EKS 叢集名稱。
   - `addon-name`— 要安裝的 Amazon EKS 附加元件的名稱。
   - `service-account-name`— 附加元件所使用之 Kubernetes 服務帳戶的名稱。
   - `iam-role-arn`— 具有附加元件足夠許可的 IAM 角色的 ARN。IAM 角色必須具有 EKS 網繭身分識別所需的信任政策。

2. 使用 AWS CLI 更新附加元件。您也可以使用相同的`--pod-identity-assocations`語法，在建立附加元件時指定網繭身分識別關聯。請注意，當您在更新附加元件時指定網繭身分識別關聯時，會覆寫所有先前的網繭身分識別關聯。

   ```
   aws eks update-addon --cluster-name <cluster-name> \
   --addon-name <addon-name> \
   --pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'
   ```

   例如：

   ```
   aws eks update-addon --cluster-name mycluster \
   --addon-name aws-ebs-csi-driver \
   --pod-identity-associations 'serviceAccount=ebs-csi-controller-sa,roleArn=arn:aws:iam::123456789012:role/StorageDriver'
   ```

3. 驗證已建立網繭身分識別關聯：

   ```
   aws eks list-pod-identity-associations --cluster-name <cluster-name>
   ```

   如果執行成功，您應該會看到類似下列的輸出。請注意 EKS 附加元件的所有者。

```
{
    "associations": [
        {
            "clusterName": "mycluster",
            "namespace": "kube-system",
            "serviceAccount": "ebs-csi-controller-sa",
            "associationArn": "arn:aws:eks:us-
west-2:123456789012:podidentityassociation/mycluster/a-4wvljrezsukshq1bv",
            "associationId": "a-4wvljrezsukshq1bv",
            "ownerArn": "arn:aws:eks:us-west-2:123456789012:addon/mycluster/aws-
ebs-csi-driver/9cc7ce8c-2e15-b0a7-f311-426691cd8546"
        }
    ]
}
```

從附加元件移除關聯

從 Amazon EKS 附加元件移除所有網繭身分關聯 ()AWS CLI

1. 確定：
   - `cluster-name`— 要安裝附加元件的 EKS 叢集名稱。
   - `addon-name`— 要安裝的 Amazon EKS 附加元件的名稱。
2. 更新附加元件以指定網繭身分識別關聯的空白陣列。

```
aws eks update-addon --cluster-name <cluster-name> \
--addon-name <addon-name> \
--pod-identity-associations "[]"
```

疑難排解 EKS 附加元件的網繭身分

如果您的附加元件在嘗試 AWS API、SDK 或 CLI 作業時遇到錯誤，請確認下列各項：

- 網繭身分識別代理程式已安裝在您的叢集中。
  - 檢閱如何設定網繭身分識別代理程式。
- 附加元件具有有效的網繭身分識別關聯。
  - 使用擷 AWS CLI 取附加元件所使用之服務帳戶名稱的關聯。

```
aws eks list-pod-identity-associations --cluster-name <cluster-name>
```

- 預定的 IAM 角色具有 EKS 網繭身分所需的信任政策。
  - 使用擷 AWS CLI 取附加元件的信任原則。

```
aws iam get-role --role-name <role-name> --query Role.AssumeRolePolicyDocument
```

- 預期的 IAM 角色具有附加元件的必要許可。
  - 用 AWS CloudTrail 於審查`AccessDenied`或`UnauthorizedOperation`事件。
- 網繭身分識別關聯中的服務帳戶名稱與附加元件使用的服務帳戶名稱相符。
  - 檢閱附加元件的說明文件，以判斷服務帳戶名稱。

在部署期間驗證容器映像

如果您使用 AWS Signer 並想在部署時驗證已簽署的容器映像，您可以使用下列解決方案其中一種：

- Gatekeeper 與 Ratify — 使用 Gatekeeper 作為許可控制器，並使用 AWS Signer 外掛程式設定 Ratify 作為驗證簽章的 Webhook。
- Kyverno — 使用 AWS Signer 外掛程式設定以供驗證簽章之用的 Kubernetes 政策引擎。

> **Note**
>
> 在驗證容器映像簽章之前，請依照您選取的許可控制器的要求，設定 Notation 信任存放區和信任政策。

使用 Elastic Fabric Adapter 的機器學習訓練

本主題說明如何將 Elastic Fabric Adapter (EFA) 與部署在 Amazon EKS 叢集中的 Pods 整合。Elastic Fabric Adapter (EFA) 是 Amazon EC2 執行個體的網路介面，可讓您在 AWS上大規模執行需要高層級節點間通訊的應用程式。其自訂的作業系統略過硬體介面可增強執行個體間通訊的效能，這對於擴展這些應用程式至關重要。藉由 EFA，使用訊息傳遞介面 (MPI) 的高效能運算 (HPC) 應用程式和使用 NVIDIA 集體通訊程式庫 (NCCL) 的機器學習 (ML) 應用程式可擴展到數千個 CPU 或 GPU。因此，您可以獲得內部部署 HPC 叢集的應用程式效能，並具備 AWS 雲端的隨需彈性和彈性。將 EFA 與

Amazon EKS 叢集上執行的應用程式整合，可減少完成大規模分散式訓練工作負載的時間，而無需在叢集中新增其他執行個體。如需 EFA 的詳細資訊，請參閱 Elastic Fabric Adapter。

本主題中描述的 EFA 外掛程式完全支援 Amazon EC2 P4d 執行個體，這些執行個體代表雲端中分散式機器學習中目前最先進的技術。每個 p4d.24xlarge 執行個體具有八個 NVIDIA A100 GPU，以及通過 EFA 的 400 Gbps GPUDirectRDMA。GPUDirectRDMA 可讓您透過 CPU 旁路，在節點之間進行 GPU 對 GPU 的直接通訊，提升集體通訊頻寬並降低延遲。具有 P4d 執行個體的 Amazon EKS 和 EFA 整合提供無縫方法，可利用效能最高的 Amazon EC2 運算執行個體進行分散式機器學習訓練。

必要條件

- 現有 Amazon EKS 叢集。若您尚未擁有叢集，請使用 Amazon EKS 入門 指南之一建立一個叢集。您的叢集必須部署在具有至少一個私有子網路的 VPC 中，其中具有足夠的可用 IP 地址，以便部署節點。私有子網路必須具有外部裝置 (例如 NAT 閘道) 所提供的傳出網際網路存取權。

  如果計劃使用 eksctl 來建立您的節點群組，eksctl 也會為您建立叢集。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 已在裝置或 AWS CloudShell上安裝 kubectl 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。
- 在啟動支援多個 Elastic Fabric Adapter 的工作節點 (例如 p4d.24xlarge) 之前，您必須先安裝 Amazon VPC CNI plugin for Kubernetes 版本 1.7.10 或更新版本。如需有關更新 Amazon VPC CNI plugin for Kubernetes 版本的詳細資訊，請參閱使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件。

建立節點群組

下列程序可協助您建立具有 EFA 介面和 GPUDirect RDMA 之 p4d.24xlarge 支援的節點群組，並針對使用 EFA 的多節點 NCCL 效能執行範例 NVIDIA 集體通訊程式庫 (NCCL) 測試。此範例可用於使用 EFA 在 Amazon EKS 上進行分散式深度學習訓練的範本。

1. 判斷您要在其中部署節點的哪些 Amazon EC2 執行個體類型可以使用支援 EFA。 AWS 區域 取代*region-code*為您 AWS 區域 要在其中部署節點群組的。

   ```
   aws ec2 describe-instance-types --region region-code --filters Name=network-
   info.efa-supported,Values=true \
       --query "InstanceTypes[*].[InstanceType]" --output text
   ```

   部署節點時，您要部署的執行個體類型必須在叢集所在 AWS 區域 的位置中可用。

2. 判斷您想要部署的執行個體類型所在的哪一個 Availability Zone (可用區域) 為可用。在此自學課程中，會使用p4d.24xlarge執行個體類型，而且必須在輸出中傳回您在上一 AWS 區域 個步驟中指定的執行個體類型。在生產叢集中部署節點時，請*p4d.24xlarge*使用上一個步驟中傳回的任何執行個體類型取代。

   ```
   aws ec2 describe-instance-type-offerings --region region-code --location-type
    availability-zone --filters Name=instance-type,Values=p4d.24xlarge \
       --query 'InstanceTypeOfferings[*].Location' --output text
   ```

   範例輸出如下。

   ```
   us-west-2a    us-west-2c    us-west-2b
   ```

   請注意傳回的 Availability Zone (可用區域) 以供稍後步驟使用。將節點部署到叢集時，您的 VPC 必須在輸出中傳回的其中一個 Availability Zone (可用區域) 中具有可用 IP 位址的子網路。

3. 使用eksctl或和建立節點群 AWS CLI 組 AWS CloudFormation。

   eksctl

   先決條件

   已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

1. 將下列內容複製到名為 *efa-cluster.yaml* 的檔案。使用自己的取代 *example values*。您可以使用不同的執行個體取代 *p4d.24xlarge*，但是如果您這樣做，請確保 availabilityZones 值是針對步驟 1 中執行個體類型傳回的可用區域。

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-efa-cluster
  region: region-code
  version: "1.XX"

iam:
  withOIDC: true

availabilityZones: ["us-west-2a", "us-west-2c"]

managedNodeGroups:
  - name: my-efa-ng
    instanceType: p4d.24xlarge
    minSize: 1
    desiredCapacity: 2
    maxSize: 3
    availabilityZones: ["us-west-2a"]
    volumeSize: 300
    privateNetworking: true
    efaEnabled: true
```

2. 在現有叢集中建立受管節點群組。

```
eksctl create nodegroup -f efa-cluster.yaml
```

如果您沒有現有的叢集，則可以執行以下命令來建立叢集和節點群組。

```
eksctl create cluster -f efa-cluster.yaml
```

> **Note**
>
> 由於此範例中使用的執行個體類型具有 GPU，`eksctl` 會為您自動在每個執行個體上安裝 NVIDIA Kubernetes 裝置外掛程式。

AWS CLI and AWS CloudFormation

EFA 聯網有幾個要求，包括建立 EFA 特定的安全群組、建立 Amazon EC2 置放群組，並建立指定一或多個 EFA 介面的啟動範本，並包含 EFA 驅動程式安裝作為 Amazon EC2 使用者資料的一部分。若要進一步了解 EFA 需求，請參閱 Amazon EC2 使用者指南中的開始使用 EFA 和 MPI。下列步驟會為您建立所有這些項目。使用您自己的取代所有*###*。

1. 設定稍後步驟中使用的幾個變數。使用您自己的取代所有 *example values*。使用現有叢集的名稱取代 *my-cluster*。的值稍後node_group_resources_name會用來建立 AWS CloudFormation 堆疊。node_group_name 的值稍後用於建立叢集中的節點群組。

   ```
   cluster_name="my-cluster"
   cluster_region="region-code"
   node_group_resources_name="my-efa-nodegroup-resources"
   node_group_name="my-efa-nodegroup"
   ```

2. 識別 VPC 中與您要部署的執行個體類型位於相同可用區域中的私有子網路。

   a. 擷取叢集版本並將其存放在變數中，以供稍後步驟使用。

   ```
   cluster_version=$(aws eks describe-cluster \
       --name $cluster_name \
       --query "cluster.version" \
       --output text)
   ```

   b. 擷取叢集所在的 VPC ID，並將其存放在變數中以供稍後步驟使用。

   ```
   vpc_id=$(aws eks describe-cluster \
       --name $cluster_name \
       --query "cluster.resourcesVpcConfig.vpcId" \
       --output text)
   ```

   c. 擷取叢集的控制平面安全群組 ID，並將其存放在變數中以供稍後步驟使用。

```
control_plane_security_group=$(aws eks describe-cluster \
    --name $cluster_name \
    --query "cluster.resourcesVpcConfig.clusterSecurityGroupId" \
    --output text)
```

d. 取得位於步驟 1 所傳回可用區域之 VPC 的子網路 ID 清單。

```
aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$vpc_id" "Name=availability-
zone,Values=us-west-2a" \
    --query 'Subnets[*].SubnetId' \
    --output text
```

若未傳回輸出，請嘗試在步驟 1 中傳回的其他可用區域。如果沒有任何子網路位於步驟 1 傳回的可用區域中，則您需要在步驟 1 傳回的可用區域中建立子網路。如果您的 VPC 中沒有空間來建立另一個子網路，則可以在 VPC 中新增 CIDR 區塊，並在新的 CIDR 區塊中建立子網路，或在新的 VPC 中建立新的叢集。

e. 透過檢查子網路的路由表，判定子網路是否為私有子網路。

```
aws ec2 describe-route-tables \
    --filter Name=association.subnet-id,Values=subnet-0d403852a65210a29 \
    --query "RouteTables[].Routes[].GatewayId" \
    --output text
```

範例輸出如下。

```
local
```

如果輸出為 `local igw-02adc64c1b72722e2`，則子網路是公有子網路。您必須在步驟 1 中傳回的可用區域中選取私有子網路。辨識私有子網路後，請記下其 ID，以便在稍後步驟中使用。

f. 使用上一個步驟的私有子網路 ID 設定變數，以供稍後步驟使用。

```
subnet_id=your-subnet-id
```

3. 下載 AWS CloudFormation 範本。

```
curl -O https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/
cloudformation/efa-p4d-managed-nodegroup.yaml
```

4. 將下列文字複製到您的電腦。使用步驟 1 的執行個體類型取代 *p4d.24xlarge*。使用您在步驟 2.b.v 中識別的私有子網路 ID 取代 *subnet-0d403852a65210a29*。使用您在上一個步驟中所下載 efa-p4d-managed-nodegroup.yaml 的路徑取代 *path-to-downloaded-cfn-template*。以您的公有金鑰名稱取代 *your-public-key-name*。完成取代後，請執行修改後的命令。

```
aws cloudformation create-stack \
    --stack-name ${node_group_resources_name} \
    --capabilities CAPABILITY_IAM \
    --template-body file://path-to-downloaded-cfn-template \
    --parameters \
       ParameterKey=ClusterName,ParameterValue=${cluster_name} \
       ParameterKey=ClusterControlPlaneSecurityGroup,ParameterValue=
${control_plane_security_group} \
       ParameterKey=VpcId,ParameterValue=${vpc_id} \
       ParameterKey=SubnetId,ParameterValue=${subnet_id} \
       ParameterKey=NodeGroupName,ParameterValue=${node_group_name} \
       ParameterKey=NodeImageIdSSMParam,ParameterValue=/aws/service/eks/
optimized-ami/${cluster_version}/amazon-linux-2-gpu/recommended/image_id \
       ParameterKey=KeyName,ParameterValue=your-public-key-name \
       ParameterKey=NodeInstanceType,ParameterValue=p4d.24xlarge
```

5. 判定您在上一個步驟中部署之堆疊的部署時間。

```
aws cloudformation wait stack-create-complete --stack-name
 $node_group_resources_name
```

沒有來自上一個命令的輸出，但在建立堆疊前， Shell 提示不會傳回。

6. 使用上一個步驟中的 AWS CloudFormation 堆疊所建立的資源來建立您的節點群組。

   a. 從已部署的 AWS CloudFormation 堆疊擷取資訊，並將其儲存在變數中。

```
node_instance_role=$(aws cloudformation describe-stacks \
    --stack-name $node_group_resources_name \
    --query='Stacks[].Outputs[?OutputKey==`NodeInstanceRole`].OutputValue'
 \
    --output text)
launch_template=$(aws cloudformation describe-stacks \
```

```
    --stack-name $node_group_resources_name \
    --query='Stacks[].Outputs[?OutputKey==`LaunchTemplateID`].OutputValue'
 \
    --output text)
```

b. 建立使用啟動範本和在上一個步驟中所建立之節點 IAM 角色的受管節點群組。

```
aws eks create-nodegroup \
    --cluster-name $cluster_name \
    --nodegroup-name $node_group_name \
    --node-role $node_instance_role \
    --subnets $subnet_id \
    --launch-template id=$launch_template,version=1
```

c. 確認已建立節點。

```
aws eks describe-nodegroup \
   --cluster-name ${cluster_name} \
   --nodegroup-name ${node_group_name} | jq -r .nodegroup.status
```

在上一個命令傳回的狀態為 ACTIVE 之前，請勿繼續。節點可能需要幾分鐘才能準備就緒。

7. 如果您選擇 GPU 執行個體類型，則必須部署 Kubernetes 專用 NVIDIA 裝置外掛程式。請先以您想要的 NVIDIA/k8s-device-plugin 版本來取代 *vX.X.X*，再執行下列命令。

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-
plugin/vX.X.X/nvidia-device-plugin.yml
```

4. 部署 EFA Kubernetes 裝置外掛程式。

EFA Kubernetes 裝置外掛程式會進行偵測並宣告 EFA 介面為 Kubernetes 的可配置資源。應用程式可以耗用 Pod 請求規格中擴充的資源類型 vpc.amazonaws.com/efa，如 CPU 和記憶體。如需詳細資訊，請參閱 Kubernetes 文件中的耗用擴充的資源。一旦提出請求，外掛程式會自動指派 EFA 介面給 Pod，並加以掛載。使用裝置外掛程式可簡化 EFA 設定，並且不需要 Pod 以特殊權限模式執行。

```
helm repo add eks https://aws.github.io/eks-chart
helm install aws-efa-k8s-device-plugin --namespace kube-system eks/aws-efa-k8s-
device-plugin
```

(選用) 部署 EFA 相容的應用程式範例

部署 Kubeflow MPI 運算子

對於 NCCL 測試，您可以套用 Kubeflow MPI 運算子。MPI 運算子可以很容易地在 Kubernetes 上執行 Allreduce 式分散式訓練。如需詳細資訊，請參閱 GitHub 上的 MPI 運算子。

```
kubectl apply -f https://raw.githubusercontent.com/kubeflow/mpi-operator/master/deploy/v2beta1/mpi-operator.yaml
```

執行多節點 NCCL 效能測試，以驗證 GPUDirectRDMA/EFA

要透過 EFA 使用 GPUDirectRDMA 驗證 NCCL 效能，請執行標準 NCCL 效能測試。如需詳細資訊，請參閱 GitHub 上的官方 NCCL 測試儲存庫。您可以使用範例 Dockerfile，其中隨附的這項測試已針對 NVIDIA CUDA 11.2 和 EFA 最新版本建置。

或者，您也可以從 Amazon ECR 存放庫下載可用的 AWS Docker映像檔。

> **Important**
>
> 搭配 Kubernetes 採用 EFA 所需的一項重要考量，就是將大型頁面作為叢集中的資源進行設定和管理。如需詳細資訊，請參閱 Kubernetes 文件中的管理大型頁面。已安裝 EFA 驅動程式的 Amazon EC2 執行個體會預先分配 5128 2M 大型頁面，您可以請求作為資源在任務規格中耗用。

請完成以下步驟，以執行雙節點 NCCL 效能測試。在 NCCL 測試任務範例中，每個工作者請求八個 GPU、Hugepages-2Mi 的 5210Mi、四個 EFA 和 8000Mi 的記憶體，這有效地表示每個工作者會耗用 `p4d.24xlarge` 執行個體的所有資源。

1. 建立 NCCL 測試任務。

   ```
   kubectl apply -f https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/examples/simple/nccl-efa-tests.yaml
   ```

   範例輸出如下。

   管理員. nccl-tests-efa 組織/已建立

2. 檢視您執行的 Pods。

```
kubectl get pods
```

範例輸出如下。

```
NAME                                READY   STATUS     RESTARTS   AGE
nccl-tests-efa-launcher-nbql9  0/1     Init:0/1   0          2m49s
nccl-tests-efa-worker-0          1/1     Running    0          2m49s
nccl-tests-efa-worker-1          1/1     Running    0          2m49s
```

MPI 運算子會建立啟動器 Pod 和 2 個工作者 Pods (每個節點上有一個)。

3. 檢視 `efa-launcher` Pod 的日誌。使用輸出的值取代 *wzr8j*。

```
kubectl logs -f nccl-tests-efa-launcher-nbql9
```

如需更多範例，請參閱 GitHub 上的 Amazon EKS EFA 範例儲存庫。

使用 AWS Inferentia 的機器學習推論

本主題描述了如何建立 Amazon EKS 叢集，其中具有執行 Amazon EC2 Inf1 執行個體的節點，以及 (選用) 如何部署範例應用程式。Amazon EC2 Inf1 執行個體由AWS 推論晶片提供支援，這些晶片由 AWS 客製化建置，可在雲端中提供高效能和最低成本的推論。機器學習模型使用 AWS Neuron 部署到容器中，Neuron 是一種專門的軟體開發套件 (SDK)，由編譯器、執行階段和剖析工具組成，可最佳化 Inferentia 晶片的機器學習推論效能。 AWS 神經元支持流行的機器學習框架，例如 TensorFlow PyTorch，和 MXNet。

> **Note**
>
> Neuron 裝置邏輯 ID 必須是連續的。如果請求多個 Neuron 裝置的 Pod 已排程在某個 `inf1.6xlarge` 或 `inf1.24xlarge` 執行個體類型 (具有多個 Neuron 裝置) 上，倘若 Kubernetes 排程器選取非連續的裝置 ID，該 Pod 將無法啟動。如需詳細資訊，請參閱 GitHub 上的 Device logical IDs must be contiguous (裝置邏輯 ID 必須是連續的)。

必要條件

- 將 `eksctl` 安裝在您的電腦上。如果您尚未安裝，請參閱 `eksctl` 文件中的 Installation 一節。

- 將 kubectl 安裝在您的電腦上。如需詳細資訊，請參閱 安裝或更新 kubectl。
- (選用) 將 python3 安裝在您的電腦上。如果您尚未安裝，請參閱 Python 下載以取得安裝指示。

建立叢集

建立具有 Inf1 Amazon EC2 執行個體之節點的叢集

1. 建立具有 Inf1 Amazon EC2 執行個體之節點的叢集。您可以使用任何 Inf1 執行個體類型取代 *inf1.2xlarge*。eksctl 公用程式偵測到您正在啟動具有 Inf1 執行個體類型的節點群組，並將使用 Amazon EKS 最佳化加速 Amazon Linux AMI 之一來啟動您的節點。

   > **Note**
   >
   > 您無法透 TensorFlow 過「服務」將 IAM 角色用於服務帳戶。

   ```
   eksctl create cluster \
       --name inferentia \
       --region region-code \
       --nodegroup-name ng-inf1 \
       --node-type inf1.2xlarge \
       --nodes 2 \
       --nodes-min 1 \
       --nodes-max 4 \
       --ssh-access \
       --ssh-public-key your-key \
       --with-oidc
   ```

   > **Note**
   >
   > 請注意下列輸出行的值。它將用於稍後的 (選用) 步驟。
   >
   > ```
   > [9]  adding identity "arn:aws:iam::111122223333:role/
   > eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09" to auth
   >  ConfigMap
   > ```

使用Inf1執行個體啟動節點群組時，eksctl會自動安裝 AWS Neuron 裝Kubernetes置外掛程式。這個外掛程式會將 Neuron 裝置做為系統資源公告給 Kubernetes 排程器，如此就能由容器請求。除了預設 Amazon EKS 節點 IAM 政策之外，還會新增 Amazon S3 唯讀存取政策，以便稍後步驟介紹的範例應用程式可以從 Amazon S3 載入訓練過的模型。

2. 請確定所有的 Pods 都已正確啟動。

```
kubectl get pods -n kube-system
```

縮寫輸出：

```
NAME                                   READY   STATUS    RESTARTS   AGE
[...]
neuron-device-plugin-daemonset-6djhp   1/1     Running   0          5m
neuron-device-plugin-daemonset-hwjsj   1/1     Running   0          5m
```

(選擇性) 部署 TensorFlow服務應用程式映像

經過訓練的模型必須編譯至 Inferentia 目標，才能部署到 Inferentia 執行個體上。若要繼續，您需要在 Amazon S3 中儲存一個神經元最佳化 TensorFlow模型。如果您還沒有 SavedModel，請按照教程創建 Neuron 兼容 ResNet 50 模型，並 SavedModel 將結果上傳到 S3。 ResNet-50 是一種流行的機器學習模型，用於圖像識別任務。如需有關編譯神經元模型的詳細資訊，請參閱開AWS 發人員指南中的含 DLAMI 的推論晶片。 AWS Deep Learning AMI

範例部署資訊清單會管理 AWS Deep Learning Containers 所 TensorFlow 提供的預先建置推論服務容器。容器內部是 AWS 神經元運行時和 TensorFlow 服務應用程序。針對 Neuron 最佳化的預建置 Deep Learning Containers 完整清單會在 GitHub 上進行維護，位於可用映像下。在啟動時，DLC 將從 Amazon S3 擷取您的模型、使用儲存的模型啟動神經元 TensorFlow 服務，然後等待預測請求。

分配給服務應用程式的 Nesuron 裝置數量可以進行調整，方法是變更部署 yaml 中的 aws.amazon.com/neuron 資源。請注意， TensorFlow 服務與 Neuron 執行階段之間的通訊會透過 GRPC 進行，因此需要將IPC_LOCK功能傳遞至容器。

1. 將 AmazonS3ReadOnlyAccess IAM 政策新增至步驟 1/建立叢集 中建立的節點執行個體角色。這是必要步驟，以便範例應用程式可以從 Amazon S3 載入訓練過的模型。

```
aws iam attach-role-policy \
```

```
    --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --role-name eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09
```

2. 使用下列內容建立名為 rn50_deployment.yaml 的檔案。更新區域碼和模型路徑以符合您想要的設定。當客戶端向 TensorFlow 服務器發出請求時，模型名稱用於識別目的。此範例使用模型名稱來比對範例 ResNet 50 用戶端指令碼，該指令碼將在稍後的步驟中用於傳送預測要求。

```
aws ecr list-images --repository-name neuron-rtd --registry-id 790709498068 --
region us-west-2
```

```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: eks-neuron-test
  labels:
    app: eks-neuron-test
    role: master
spec:
  replicas: 2
  selector:
    matchLabels:
      app: eks-neuron-test
      role: master
  template:
    metadata:
      labels:
        app: eks-neuron-test
        role: master
    spec:
      containers:
        - name: eks-neuron-test
          image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-
neuron:1.15.4-neuron-py37-ubuntu18.04
          command:
            - /usr/local/bin/entrypoint.sh
          args:
            - --port=8500
            - --rest_api_port=9000
            - --model_name=resnet50_neuron
            - --model_base_path=s3://your-bucket-of-models/resnet50_neuron/
          ports:
            - containerPort: 8500
```

```
            - containerPort: 9000
          imagePullPolicy: IfNotPresent
          env:
            - name: AWS_REGION
              value: "us-east-1"
            - name: S3_USE_HTTPS
              value: "1"
            - name: S3_VERIFY_SSL
              value: "0"
            - name: S3_ENDPOINT
              value: s3.us-east-1.amazonaws.com
            - name: AWS_LOG_LEVEL
              value: "3"
          resources:
            limits:
              cpu: 4
              memory: 4Gi
              aws.amazon.com/neuron: 1
            requests:
              cpu: "1"
              memory: 1Gi
          securityContext:
            capabilities:
              add:
                - IPC_LOCK
```

3. 部署模型。

```
kubectl apply -f rn50_deployment.yaml
```

4. 使用下列內容建立名為 `rn50_service.yaml` 的檔案。HTTP 和 gRPC 連接埠會開啟以接受預測請求。

```
kind: Service
apiVersion: v1
metadata:
  name: eks-neuron-test
  labels:
    app: eks-neuron-test
spec:
  type: ClusterIP
  ports:
    - name: http-tf-serving
```

```
      port: 8500
      targetPort: 8500
    - name: grpc-tf-serving
      port: 9000
      targetPort: 9000
  selector:
    app: eks-neuron-test
    role: master
```

5. 為您的 TensorFlow 模型 Kubernetes Service 應用程式建立服務。

```
kubectl apply -f rn50_service.yaml
```

（可選）針對您的服 TensorFlow 務進行預測

1. 若要在本機測試，請將 gRPC 連接埠轉送至 eks-neuron-test 服務。

```
kubectl port-forward service/eks-neuron-test 8500:8500 &
```

2. 建立一個叫做 tensorflow-model-server-infer.py 的 Python 指令碼，具有以下內容。此指令碼透過 gRPC (為一服務框架) 執行推斷。

```
import numpy as np
   import grpc
   import tensorflow as tf
   from tensorflow.keras.preprocessing import image
   from tensorflow.keras.applications.resnet50 import preprocess_input
   from tensorflow_serving.apis import predict_pb2
   from tensorflow_serving.apis import prediction_service_pb2_grpc
   from tensorflow.keras.applications.resnet50 import decode_predictions

   if __name__ == '__main__':
       channel = grpc.insecure_channel('localhost:8500')
       stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
       img_file = tf.keras.utils.get_file(
           "./kitten_small.jpg",
           "https://raw.githubusercontent.com/awslabs/mxnet-model-server/master/
docs/images/kitten_small.jpg")
       img = image.load_img(img_file, target_size=(224, 224))
       img_array = preprocess_input(image.img_to_array(img)[None, ...])
       request = predict_pb2.PredictRequest()
```

```
        request.model_spec.name = 'resnet50_inf1'
        request.inputs['input'].CopyFrom(
            tf.make_tensor_proto(img_array, shape=img_array.shape))
        result = stub.Predict(request)
        prediction = tf.make_ndarray(result.outputs['output'])
        print(decode_predictions(prediction))
```

3. 執行指令碼，將預測提交至您的服務。

```
python3 tensorflow-model-server-infer.py
```

範例輸出如下。

```
[[(u'n02123045', u'tabby', 0.68817204), (u'n02127052', u'lynx', 0.12701613),
 (u'n02123159', u'tiger_cat', 0.08736559), (u'n02124075', u'Egyptian_cat',
 0.063844085), (u'n02128757', u'snow_leopard', 0.009240591)]]
```

叢集管理

本章節包括可協助您管理叢集的以下主題。您也可以與 AWS Management Console 來檢視 Kubernetes 資源相關資訊。

- Kubernetes 儀表板是 Kubernetes 叢集的一般用途、以 Web 為基礎的 UI。它可讓使用者管理叢集中執行的應用程式並對其進行疑難排解，以及管理叢集本身。如需詳細資訊，請參閱 Kubernetes 儀板表 GitHub 儲存庫。
- 安裝 Kubernetes 指標伺服器：Kubernetes 指標伺服器是叢集中資源使用狀況資料的彙總工具。根據預設，它不會部署在您的叢集中，但會由 Kubernetes 附加元件使用，例如 Kubernetes 儀表板和 Horizontal Pod Autoscaler。在本主題中，您將了解如何安裝指標伺服器。
- 搭配 Amazon EKS 使用 Helm：適用於 Kubernetes 的 Helm 套件管理工具可協助您在 Kubernetes 叢集上安裝和管理應用程式。此主題協助您安裝和執行 Helm 二進位檔，讓您可以在本機電腦上使用 Helm CLI 安裝和管理圖表。
- 為您的 Amazon EKS 資源加上標籤 – 為協助您管理 Amazon EKS 資源，您可以用標籤形式將您自己的中繼資料指派給每個資源。本主題說明標籤並示範如何建立它們。
- Amazon EKS 服務配額 – 對於每個 AWS 服務，您的 AWS 帳戶有預設配額，先前稱為限制。了解 Amazon EKS 的配額，以及如何提高配額。

成本監控

成本監控是在 Amazon EKS 上管理Kubernetes叢集的重要方面。透過瞭解叢集成本，您可以最佳化資源使用率、設定預算，並針對部署做出資料驅動的決策。Amazon EKS 提供兩種成本監控解決方案，每個解決方案都有其獨特的優勢，可協助您有效地追蹤和分配成本：

AWS Amazon EKS 的帳單分攤成本分配資料 — 此原生功能與 AWS 帳單主控台無縫整合，讓您可以使用熟悉的介面和用於其他 AWS 服務的工作流程來分析和分配成本。透過分割成本分配，您可以直接與其他 AWS 支出一起深入瞭解Kubernetes成本，讓您更輕鬆地在整個環境中最佳化成本。 AWS 您也可以利用成本類別和成本異常偵測等現有 AWS 帳單功能，進一步增強您的成本管理功能。如需詳細資訊，請參閱 AWS 帳單使用者指南中的了解分割成本分配資料。

Kubecost— Amazon EKS 支持 Kubecost，這是一種成本監控工具。Kubecost 提供功能豐富、Kubernetes 原生的成本監控方法，可透過 Kubernetes 資源、成本最佳化建議，以及儀表板和報告提供精細的成本明細分。 out-of-the-box Kubecost 還透過整合成本和用量報告來擷取準確的定價資料，確保您能精確掌握 Amazon EKS AWS 成本。瞭解如何安裝Kubecost。

AWS 帳單 — 分割成本分配

使用 Amazon EKS 的分 AWS 割成本分配資料進行成本監控

您可以使用 Amazon EKS 的分 AWS 割成本分配資料，取得 Amazon EKS 叢集的精細成本可見性。這可讓您分析、最佳化及退款應用程式的成本與Kubernetes用量。您可以根據應用程式使用的 Amazon EC2 CPU 和記憶體資源，將Kubernetes應用程式成本分配給個別業務單位和團隊。Amazon EKS 的分割成本分配資料可讓您瞭解每個網繭的成本，並可讓您使用命名空間、叢集和其他Kubernetes原語彙總每個網繭的成本資料。以下是您可以用來分析 Amazon EKS 成本分配資料的Kubernetes基元範例。

- 叢集名稱
- 部署
- 命名空間
- 節點
- 工作負載名稱
- 工作負載類型

如需使用分割成本分配資料的詳細資訊，請參閱 AWS 帳單使用者指南中的了解分割成本分配資料。

設定成本與使用量報告

您可以在成本管理主控台或 AWS SDK 中開啟 EKS 的分割成本分配資料。 AWS Command Line Interface

針對「分割成本配置資料」使用下列項目：

1. 選擇加入分割成本配置資料。如需詳細資訊，請參閱 AWS Cost and Usage Report 使用指南中的啟用分割成本配置資料。
2. 將資料包含在新的或現有的報表中。
3. 檢視報告。您可以使用帳單和成本管理主控台，或在 Amazon Simple Storage Service 中檢視報告檔案。

熊本

Amazon EKS 支援 Kubecost，可讓您用於監控按 Kubernetes 資源 (包括 Pods、節點、命名空間和標籤) 細分的成本。身為 Kubernetes 平台管理員和財務負責人，您可以使用 Kubecost 視覺化 Amazon EKS 費用明細、分配成本，以及向應用程式團隊等組織單位收取費用。您可以根據內部團隊和業務單

位的實際 AWS 帳單，為內部團隊和業務單位提供透明且準確的成本資料。此外，您還可以根據他們的基礎設施環境和叢集內的使用模式，獲得客製化的成本最佳化建議。如需 Kubecost 的詳細資訊，請參閱 Kubecost 文件。

Amazon EKS 提供一套 AWS 優化的Kubecost叢集成本可見度。您可以使用現有的 AWS 支援合約來取得支援。

必要條件

- 現有 Amazon EKS 叢集。若要部署叢集，請參閱 Amazon EKS 入門。叢集必須具有 Amazon EC2 節點，因為您無法在 Fargate 節點上執行 Kubecost。
- 已在裝置或 AWS CloudShell上安裝 `kubectl` 命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 `1.29`，則可以搭配使用 `kubectl` `1.28`、`1.29` 或 `1.30` 版。若要安裝或升級 `kubectl`，請參閱 安裝或更新 `kubectl`。
- 已在裝置或 AWS CloudShell上設定 Helm 3.9.0 版或更新版本。若要安裝或更新 Helm，請參閱 the section called “使用 Helm”。
- 如果您的叢集是 `1.23` 版或更新版本，則必須在您的叢集上安裝 the section called “Amazon EBS CSI 驅動程式”。

若要安裝久貝費

1. 決定Kubecost要安裝的版本。您可以在 Amazon ECR Public Gallery 中的 kubecost/cost-analyzer 查看可用版本。如需有關Kubecost版本和 Amazon EKS 相容性的詳細資訊，請參閱 Kubecost 文件中的環境需求。
2. 使用下列命令安裝 Kubecost。*`# ECR ################## 1.108.1#`*

   ```
   helm upgrade -i kubecost oci://public.ecr.aws/kubecost/cost-analyzer --
   version kubecost-version \
       --namespace kubecost --create-namespace \
       -f https://raw.githubusercontent.com/kubecost/cost-analyzer-helm-chart/develop/
   cost-analyzer/values-eks-cost-monitoring.yaml
   ```

   Kubecost 會定期推出新版本。您可以使用 `helm upgrade` 來更新版本。依預設，安裝會包括本機 Prometheus 伺服器與 `kube-state-metrics`。您可以依照 Integrating with Amazon EKS cost monitoring (與 Amazon EKS 成本監控整合) 所述來自訂部署，以使用 Amazon Managed Service for Prometheus。如需您可以設定的所有其他設定清單，請參閱上的範例組態檔案 GitHub。

3. 確定所需的 Pods 正在執行。

```
kubectl get pods -n kubecost
```

範例輸出如下。

```
NAME                                           READY   STATUS    RESTARTS   AGE
kubecost-cost-analyzer-b9788c99f-5vj5b         2/2     Running   0          3h27m
kubecost-kube-state-metrics-99bb8c55b-bn2br    1/1     Running   0          3h27m
kubecost-prometheus-server-7d9967bfc8-9c8p7    2/2     Running   0          3h27m
```

4. 在您的裝置上，啟用連接埠轉送以公開 Kubecost 儀表板。

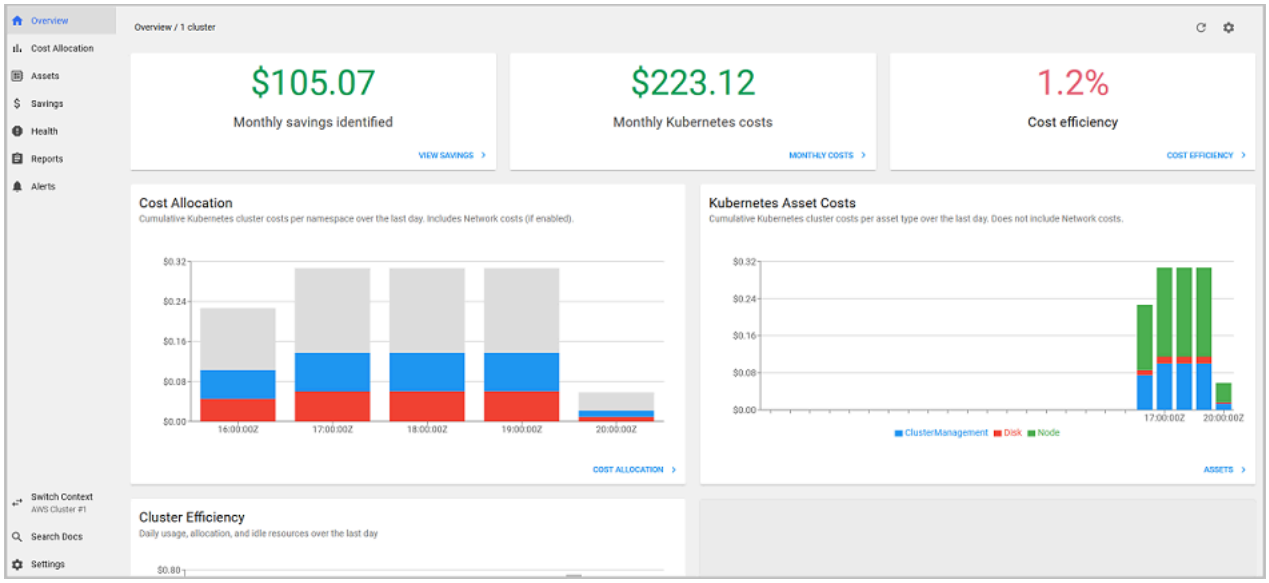
```
kubectl port-forward --namespace kubecost deployment/kubecost-cost-analyzer 9090
```

或者，您可以使用 AWS Load Balancer Controller 來公開 Kubecost ，並使用 Amazon Cognito 進行驗證、授權和使用者管理。如需詳細資訊，請參閱如何使用應用程式負載平衡器和 Amazon Cognito 驗證 Kubernetes Web 應用程式的使用者。

5. 在您完成上一個步驟的同一部裝置上，開啟 Web 瀏覽器並輸入下列位址。

```
http://localhost:9090
```

瀏覽器中會顯示 Kubecost 的 Overview (概觀) 頁面。Kubecost 可能需要 5-10 分鐘的時間來收集指標。您可以查看您的 Amazon EKS 支出，包括累積叢集成本、相關聯的 Kubernetes 資產成本和每月彙總支出。



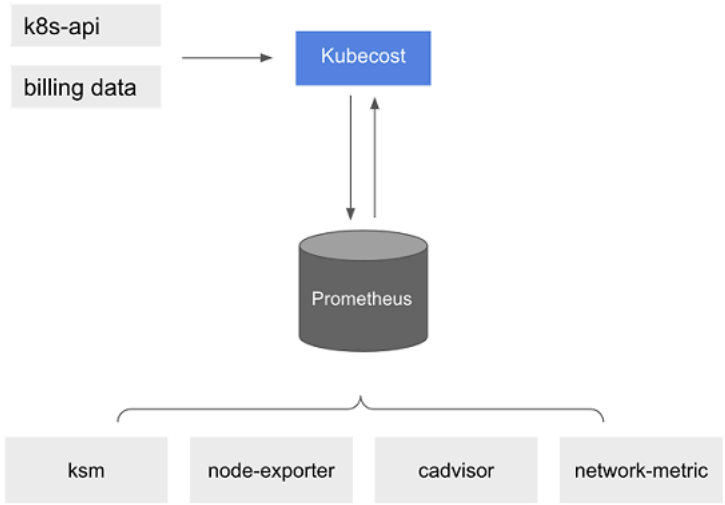
6. 要在叢集層級追蹤成本，請標記您的 Amazon EKS 資源以進行計費。如需詳細資訊，請參閱 標記您的資源以便計費。

您也可以在儀表板的左窗格中選取相應項目來檢視以下資訊：

- Cost allocation (成本分配) – 檢視過去七天內每個命名空間和其他維度的每月 Amazon EKS 成本和累計成本。這有助於了解應用程式的哪些部分產生了 Amazon EKS 支出。
- Assets (資產) – 檢視與 Amazon EKS 資源相關聯的 AWS 基礎設施資產的成本。

額外功能

- Export cost metrics (匯出成本指標) – Amazon EKS 最佳化成本監控是與 Kubecost 和 Prometheus 一起部署，形成開放原始碼監控系統和時間序列資料庫。Kubecost 會從 Prometheus 讀取指標並執行成本分配計算，然後將指標重新寫入 Prometheus。Kubecost 前端會從 Prometheus 讀取指標，並顯示在 Kubecost 使用者介面上。下圖說明此架構。



在預先安裝 Prometheus 之後，您可寫入查詢並擷取 Kubecost 資料到目前的商業智慧系統，以便進一步分析。您也可以將其用作目前 Grafana 儀表板的資料來源，以顯示您的內部團隊熟悉的 Amazon EKS 叢集成本。要了解有關如何編寫Prometheus查詢的更多信息，請參閱上的Prometheus 配置readme文件 GitHub 或使用 KubecostGithub 存儲庫中的示例 Grafana JSON 模型作為參考。

- AWS Cost and Usage Report 整合 — 若要為 Amazon EKS 叢集執行成本分配計算，請從價目表 API Kubecost 擷取的公開定 AWS 價資訊 AWS 服務 和 AWS 資源。您也可AWS Cost and Usage Report以Kubecost與整合，以提高您的 AWS 帳戶. 這類資訊包括企業折扣方案、預留執行個體使

用量、Savings Plans 和 Spot 使用量。若要進一步了解 AWS Cost and Usage Report 整合的運作方AWS 式，請參閱Kubecost文件中的帳單整合。

移除 Kubecost

您可以使用下列命令將 Kubecost 從您的叢集中移除。

```
helm uninstall kubecost --namespace kubecost
kubectl delete ns kubecost
```

常見問答集

請參閱以下有關搭配使用 Kubecost 與 Amazon EKS 的常見問題和解答。

自訂套件的 Kubecost 和免費版本的 Kubecost (也稱為 OpenCost) 有何區別？

AWS 並Kubecost協同合作提供自訂版本的Kubecost. 此版本提供商業功能的子集，不收取額外費用。請見下列表格以瞭解 Kubecost 的自訂套件中隨附的功能。

功能	Kubecost 免費方案	Amazon EKS 已最佳化 Kubecost 自訂套件	Kubecost Enterprise
部署	使用者託管	使用者託管	使用者託管或 Kubecost 託管 (SaaS)
支援的叢集數目	無限制	無限制	無限制
支援的資料庫	區域 Prometheus	本機 Prometheus 或 Amazon Managed Service for Prometheus	Prometheus、Amazon Managed Service for Prometheus、Cortex，或 Thanos
資料庫保留支援	15 天	無限的歷史記錄資料	無限的歷史記錄資料
Kubecost API 保留 (ETL)	15 天	15 天	無限的歷史記錄資料
叢集成本能見度	單一叢集	統一的多重叢集	統一的多重叢集

功能	Kubecost 免費方案	Amazon EKS 已最佳化 Kubecost 自訂套件	Kubecost Enterprise
混合雲端能見度	-	Amazon EKS 和 Amazon EKS Anywhere 叢集	多重雲端和混合雲端支援
警示和週期性報告	-	效率警示、預算警示、支出變更警示，以及更多支援的功能	效率警示、預算警示、支出變更警示，以及更多支援的功能
已儲存的報告	-	使用 15 天資料的報告	使用無限期歷史記錄資料的報告
雲端計費整合	每一個叢集都需要	自訂定價支援 AWS (包括多個叢集和多個帳戶)	自訂定價支援 AWS (包括多個叢集和多個帳戶)
Savings 建議	單一叢集深入解析	單一叢集深入解析	多重叢集深入解析
治理：稽核	-	-	稽核歷史記錄成本事件
單一登入 (SSO) 支援	-	支援 Amazon Cognito	Okta, Auth0, PingID, KeyCloak
角色型存取控制 (RBAC) 與 SAML **2.0**	-	-	Okta, Auth0, PingID, Keycloak
企業培訓和入門	-	-	全方位服務培訓和 FinOps 入門

什麼是 Kubecost API 保留 (ETL) 功能？

Kubecost ETL 功能可彙總並整理指標，藉此顯示各種規模程度的成本能見度 (例如 `namespace-level`、`pod-level`，以及 `deployment-level`)。若為自訂 Kubecost 套件，客戶可以從過去 15 天的指標中取得資料和深入解析。

什麼是警示和週期性報告功能？ 它包含了哪些警示和報告？

Kubecost 警示可讓團隊即時接收 Kubernetes 支出和雲端支出的更新。週期性報告可讓團隊接收歷史記錄 Kubernetes 和雲端支出的自訂檢視。兩者均可使用 Kubecost UI 或 Helm 值來設定。它們支援電子郵件、Slack，以及 Microsoft Teams。

儲存的報告包含哪些內容？

Kubecost 儲存的報告是成本和效率指標的預先定義檢視。它們包含依叢集、命名空間、標籤以及更多類別來區分的成本。

什麼是雲端計費整合？

與 AWS 帳單 API 整合Kubecost可顯示 out-of-cluster 成本 (例如 Amazon S3)。此外，它也能讓 Kubecost 協調 Kubecost 具有實際計費資料的叢集內預測，以說明 Spot 使用量、Savings Plans 和企業折扣。

Savings 建議包含哪些內容？

Kubecost 提供深入解析和自動化功能，協助使用者最佳化 Kubernetes 基礎結構和支出。

此功能是否需要付費？

否。您可以免費使用此版本的 Kubecost。如果您想要此套件中未包含的其他Kubecost功能，可以Kubecost透過或Kubecost直接向購買的企業授權。 AWS Marketplace

是否提供支援服務？

是。您可以在「聯繫」與 AWS Support 團隊打開支持案例 AWS。

我是否需要授權才能使用 Amazon EKS 整合提供的 Kubecost 功能？

否.

我可以整Kubecost合以獲得 AWS Cost and Usage Report 更準確的報告嗎？

是。您可以將 Kubecost 設定為從 AWS Cost and Usage Report 擷取資料以獲得準確的成本情況，包括折扣、Spot 定價、預留執行個體定價等。如需詳細資訊，請參閱AWS Kubecost文件中的帳單整合。

此版本是否支援 Amazon EC2 上自我管理 Kubernetes 叢集的成本管理？

否。此版本僅與 Amazon EKS 叢集相容。

Kubecost 可以在 AWS Fargate上追蹤 Amazon EKS 的成本嗎？

Kubecost 會盡最大努力顯示在 Fargate 上使用 Amazon EKS 時的叢集成本情況，但準確度低於在 Amazon EC2 上使用 Amazon EKS。這主要是因為用量計費方式有所不同。在 Fargate 上使用 Amazon EKS 時，您是按耗用的資源付費。在 Amazon EC2 節點上使用 Amazon EKS 時，您是按佈建的資源付費。Kubecost 會根據節點規格 (包括 CPU、RAM 和暫時性儲存) 計算 Amazon EC2 節點的成本。使用 Fargate 時，系統會根據 Fargate Pods 所請求的資源來計算成本。

如何獲得更新和新版的 Kubecost？

您可以使用標準 Helm 升級程序升級您的 Kubecost 版本。最新版本位於 Amazon ECR Public Gallery 中。

是否支援 **`kubectl-cost`** CLI？ 如何安裝？

是。`Kubectl-cost` 是 Kubecost 的開放原始碼工具 (Apache 2.0 License)，提供對 Kubernetes 成本分配指標的 CLI 存取。若要安裝`kubectl-cost`，請參閱安裝於 GitHub。

是否支援 Kubecost 使用者介面？ 如何存取？

Kubecost 提供 Web 儀表板，您可透過 `kubectl` 連接埠轉送、輸入或負載平衡器來存取。您也可以使用 AWS Load Balancer Controller 來公開 Kubecost，並使用 Amazon Cognito 進行驗證、授權和使用者管理。如需詳細資訊，請參閱如何在 AWS 部落格上使用應用程式負載平衡器和 Amazon Cognito 驗證 Kubernetes Web 應用程式的使用者。

是否支援 Amazon EKS Anywhere？

否.

安裝 Kubernetes 指標伺服器

Kubernetes 指標伺服器是叢集中的資源使用資料的彙總，預設不會部署於 Amazon EKS 叢集。如需詳細資訊，請參閱 GitHub 上的 Kubernetes 指標伺服器。指標伺服器通常會由其他 Kubernetes 附加元件使用，例如 Horizontal Pod Autoscaler 或 Kubernetes 儀表板。如需詳細資訊，請參閱 Kubernetes 文件中的資源指標管道。此主題說明如何在 Amazon EKS 叢集上部署 Kubernetes 指標伺服器。

> **Important**
>
> 這些指標是用於 point-in-time 分析，並不是歷史分析的準確來源。它們不能作為監視解決方案或其他非自動擴展目的之用。如需監控工具的相關資訊，請參閱 Amazon EKS 中的可觀測性。

部署指標伺服器

1. 使用下列命令部署指標伺服器：

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
```

如果您使用的是 Fargate，則需要更改此文件。在預設組態中，公制伺服器會使用連接埠 10250。該端口保留在 Fargate。將元件 .yaml 中連接埠 10250 的參照取代為另一個連接埠，例如 10251。

2. 使用下列命令確認 `metrics-server` 部署正在執行所需數量的 Pods。

```
kubectl get deployment metrics-server -n kube-system
```

範例輸出如下。

```
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
metrics-server   1/1     1            1           6m
```

搭配 Amazon EKS 使用 Helm

適用於 Kubernetes 的 Helm 套件管理工具可協助您在 Kubernetes 叢集上安裝和管理應用程式。如需詳細資訊，請參閱 Helm 文件。此主題協助您安裝和執行 Helm 二進位檔，讓您可以在本機系統上使用 Helm CLI 安裝和管理圖表。

> **Important**
>
> 將 Helm 圖表安裝在 Amazon EKS 叢集上之前，務必設定 `kubectl` 以搭配 Amazon EKS 來運作。若您尚未完成此動作，請參閱建立或更新 Amazon EKS 叢集的 kubeconfig 檔案 之後再繼續。若下列命令在您的叢集上成功執行，就表示您的設定正確。
>
> ```
> kubectl get svc
> ```

在本機系統上安裝 Helm 二進位檔

1. 為您的用戶端作業系統執行適當的命令。

- 若您在 macOS 上使用 Homebrew，請透過下列命令安裝這些二進位檔。

```
brew install helm
```

- 若您在 Windows 上使用 Chocolatey，請透過下列命令安裝這些二進位檔案。

```
choco install kubernetes-helm
```

- 若您使用 Linux，請使用下列命令安裝二進位檔案。

```
curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 >
 get_helm.sh
chmod 700 get_helm.sh
./get_helm.sh
```

> **Note**
>
> 如果您收到必須先安裝 `openssl` 的訊息，您可以使用下列命令進行安裝。
>
> ```
> sudo yum install openssl
> ```

2. 若要在 PATH 中取得新的二進位檔，請關閉目前的終端機視窗並開啟新的視窗。
3. 查看您安裝的 Helm 版本。

```
helm version | cut -d + -f 1
```

範例輸出如下。

```
v3.9.0
```

4. 此時，您可以執行任何 Helm 命令 (例如 `helm install` *chart-name*)，在叢集中安裝、修改、刪除或查詢 Helm 圖表。如果您是 Helm 新手，而且未安裝任何特定圖表，您可以：

- 透過安裝範例圖表進行實驗。請參閱 Helm 快速入門指南中的安裝範例圖表。
- 建立範例圖表並將其推送到 Amazon ECR。如需詳細資訊，請參閱 Amazon Elastic Container Registry 使用者指南中的推送 Helm Chart。
- 從 eks-Cart GitHub 回購或從中安裝 Amazon EKS 圖表。 ArtifactHub

為您的 Amazon EKS 資源加上標籤

您可以使用標籤協助您管理 Amazon EKS 資源。本主題提供標籤功能的概觀，並展示您如何建立標籤。

主題

> **Note**
>
> 標籤是一種與 Kubernetes 標籤和註釋分開的中繼資料類型。如需有關這些其他中繼資料類型的詳細資訊，請參閱 Kubernetes 文件中的下列章節：
>
> - 標籤和選取器
> - 註釋

標籤基本概念

標籤是指派給 AWS 資源的標籤。每個標籤皆包含索引鍵與選用值。

使用標籤，您可以對 AWS 資源進行分類。例如，您可以依用途、擁有者或環境來分類資源。當您有許多相同類型的資源時，您可以依據先前指派給特定資源的標籤來快速識別該資源。例如，您可以為 Amazon EKS 叢集定義一組標籤，協助您追蹤每個叢集的擁有者和堆疊層級。建議您為每個資源類型設計一組一致的標籤金鑰。然後，就可以根據您新增的標籤來搜尋和篩選資源。

新增標籤後，您可以隨時編輯標籤索引鍵和值，或從資源移除標籤。如果您刪除資源，也會刪除任何該資源的標籤。

標籤對 Amazon EKS 來說不具有任何語意意義，並會嚴格解譯為字元字串。您可以將標籤的值設為空白字串。不過，您無法將標籤的值設為 null。若您將與現有標籤具有相同鍵的標籤新增到該資源，則新值會覆寫早前的值。

如果您使用 AWS Identity and Access Management (IAM)，您可以控制 AWS 帳戶中哪些使用者有權管理標籤。

標記您的 資源

以下 Amazon EKS 資源支援標籤：

- 叢集
- 受管節點群組
- Fargate 描述檔

您可以使用以下項目標記這些資源：

- 如果使用 Amazon EKS 主控台，則可以隨時將標籤套用到新資源或現有資源。您可以在相關資源頁面使用 Tags (標籤) 索引標籤進行此操作。如需詳細資訊，請參閱 透過主控台使用標籤。
- 如果使用 `eksctl`，則可以在使用 `--tags` 選項建立時將標籤套用到資源。
- 如果您使用的 AWS CLI是 Amazon EKS API 或 AWS 開發套件，則可以使用相關 API 動作上的`tags`參數將標籤套用至新資源。您也可以使用 `TagResource` API 動作將標籤套用到現有資源。如需詳細資訊，請參閱TagResource。

當您使用某些資源建立動作時，您也可以在建立資源的同時為資源指定標籤。如果在建立資源時無法套用標籤，則無法建立資源。此機制可確保您要標記的資源是以您指定的標籤建立，不然就根本不會建立。如果您在建立資源時標記資源，則不需要在建立資源之後執行自訂標記指令碼。

標籤不會傳播到與您建立的資源相關聯的其他資源。例如，Fargate 設定檔標籤不會傳播到與 Fargate 設定檔相關聯的其他資源，例如與設定檔一起排程的 Pods。

標籤限制

以下限制適用於標籤：

- 資源最多可與 50 個標籤建立關聯。
- 單一資源的標籤索引鍵不能重複。每個標籤索引鍵都必須是唯一的，而且只能有一個值。
- 索引鍵的長度上限是 128 個 UTF-8 字元。
- 索引鍵的長度上限是 256 個 UTF-8 字元。
- 如果多個 AWS 服務 和資源使用您的標記結構描述，請限制您使用的字元類型。某些服務可能對允許的字元設有限制。通常允許的字元為：字母、數字和空格，以及下列字元：**+ - = . _ : / @**。

- 標籤鍵與值皆區分大小寫。
- 請勿使用 `aws:`、`AWS:` 或任何大小寫組合作為索引鍵或值的字首。這些僅保留 AWS 用途。您不可編輯或刪除具此字首的標籤金鑰或值。具有此前綴的標籤不會計入您的 tags-per-resource限制。

標記您的資源以便計費

當您將標籤套用到 Amazon EKS 叢集時，您可以在成本與用量報告中使用標籤來分配成本。成本與用量報告中的計量資料會顯示所有 Amazon ECS 叢集的用量。如需詳細資訊，請參閱《AWS Billing 使用者指南》中的 AWS 成本與用量報告。

特`aws:eks:cluster-name`別是 AWS 產生的成本分配標籤可讓您在 Cost Explorer 中依個別 Amazon EKS 叢集劃分 Amazon EC2 執行個體成本。但是，此標籤不會擷取控制平面費用。標籤會自動新增至參與 Amazon EKS 叢集的 Amazon EC2 執行個體中。無論執行個體是使用 Amazon EKS 受管節點群組、Karpenter 或直接使用 Amazon EC2 佈建，都會發生此行為。此特定標籤不會計入 50 個標籤限制。若要使用標籤，帳戶擁有者必須在 AWS Billing 主控台中啟用該標籤，或使用 API 啟用。當 AWS Organizations 管理帳戶擁有者啟動標籤時，也會為所有組織成員帳戶啟用該標籤。

您也可以根據具有相同標籤索引鍵值的資源來整理您的帳單資訊。例如，您可以使用特定應用程式名稱來標記數個資源，然後整理帳單資訊。這樣一來，您就可以查看該應用程式跨數項服務的總成本。如需有關使用標籤設定成本分配報告的詳細資訊，請參閱《AWS Billing 使用者指南》中的每月成本分配報告。

> **Note**
>
> 如果您剛啟用報告，當月資料會在 24 小時之後提供檢視。

Cost Explorer 是一種報告工具，可作為 AWS 免費方案的一部分使用。您可以使用 Cost Explorer 檢視過去 13 個月的 Amazon EKS 資源圖表。您還可以預測未來三個月可能花費的金額。您可以查看在一段時間內的 AWS 資源支出模式。例如，您可以用它來找出需進一步調查的領域，以及查看您可用來了解成本的趨勢。您也可以指定資料的時間範圍，以及根據天或月檢視時間資料。

透過主控台使用標籤

您可以使用 Amazon EKS 主控台，管理與新的或現有的叢集和受管節點群組相關聯的標籤。

當您在 Amazon EKS 主控台中選取資源限定頁面時，該頁面會顯示那些資源的清單。例如，若您從左側導航窗格選取 Clusters (叢集)，主控台會顯示 Amazon EKS 叢集的清單。當您從支援標籤的其中一個清單選取資源時 (例如，特定的叢集)，您便可以在 Tags (標籤) 索引標籤上檢視和管理其標籤。

您也可以在中使用標籤編輯器 AWS Management Console，它提供了管理標籤的統一方式。若要取得更多AWS 資訊，請參閱〈標籤編輯器使用指南〉中的〈使用AWS 標籤編輯器標記資源

在建立資源時新增標籤

在建立 Amazon EKS 叢集、受管節點群組和 Fargate 描述檔時，您可以將標籤新增至其中。如需詳細資訊，請參閱 建立 Amazon EKS 叢集。

在資源上新增和刪除標籤

您可以直接從資源的頁面新增或刪除與叢集相關聯的標籤。

在個別資源上新增或刪除標籤

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在導覽列上，選取 AWS 區域 要使用的。
3. 在左側導覽窗格中選擇 Clusters (叢集)。
4. 選擇特定叢集。
5. 選擇 Tags (標籤) 索引標籤，然後選擇 Manage tags (管理標籤)。
6. 在 Manage tags (管理標籤) 頁面上，視需要新增或刪除標籤。
   - 若要新增標籤，請選擇 Add tag (新增標籤)。然後指定每個標籤的索引鍵和值。
   - 若要移除標籤，請選擇 Remove tag (移除標籤)。
7. 針對您要新增或刪除的每個標籤重複此程序。
8. 選擇 Update (更新) 以完成操作。

透過 CLI、API 或 `eksctl` 使用標籤

使用下列 AWS CLI 命令或 Amazon EKS API 操作來新增、更新、列出和刪除資源的標籤。您只能使用 `eksctl` 新增標籤，同時使用一個命令建立新資源。

Amazon EKS 資源的標記支援

任務	AWS CLI	AWS Tools for Windows PowerShell	API 動作
新增或覆寫一或多個標籤。	`tag-resource`	`Add-EKSResourceTag`	`TagResource`

任務	AWS CLI	AWS Tools for Windows PowerShell	API 動作
刪除一或多個標籤。	untag-resource	Remove-EKSResourceTag	UntagResource

下列範例示範如何使用 AWS CLI來標記或取消標記資源。

範例 1：標記現有的叢集

以下命令標記現有的叢集。

```
aws eks tag-resource --resource-arn resource_ARN --tags team=devs
```

範例 2：取消標記現有的叢集

以下命令從現有的叢集刪除標籤。

```
aws eks untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

範例 3：列出資源的標籤

以下命令列出與現有資源相關聯的標籤。

```
aws eks list-tags-for-resource --resource-arn resource_ARN
```

當您使用某些資源建立動作時，您可以在建立資源的同時指定標籤。下列動作支援在建立資源時指定標籤。

任務	AWS CLI	AWS Tools for Windows PowerShell	API 動作	**eksctl**
建立叢集	create-cluster	New-EKSCluster	CreateCluster	create cluster

任務	AWS CLI	AWS Tools for Windows PowerShell	API 動作	**eksctl**
建立受管節點群組*	create-nodegroup	New-EKSNodegroup	CreateNodegroup	create nodegroup
建立 Fargate 設定檔	create-fargate-profile	New-EKSFargateProfile	CreateFargateProfile.html	create fargateprofile

* 如果還想在建立受管節點群組時為 Amazon EC2 執行個體加上標籤，請使用啟動範本建立受管節點群組。如需詳細資訊，請參閱 標記 Amazon EC2 執行個體。如果您的執行個體已經存在，則可以手動為執行個體加上標籤。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的標記資源。

Amazon EKS 服務配額

Amazon EKS 已與 Service Quotas 整合，這項 AWS 服務可讓您從中央位置檢視和管理配額。如需詳細資訊，請參閱Service Quotas 使用者指南中的什麼是 Service Quotas？。透過 Service Quotas 整合，您可以使用和快速查詢 Amazon EKS 和 AWS Fargate 服務配額的 AWS Management Console 價值。 AWS CLI

AWS Management Console

若要檢視 Amazon EKS 和 Fargate 服務配額 AWS Management Console

1. 開啟 Service Quotas 主控台，網址為 https://console.aws.amazon.com/servicequotas/。
2. 在左側導覽窗格中，選擇 AWS 服務。
3. 從 AWS 服務 清單中，搜尋並選取 Amazon Elastic Kubernetes Service (Amazon EKS) 或 AWS Fargate。

   在 [服務配額] 清單中，您可以看到服務配額名稱、套用的值 (如果有的話)、 AWS 預設配額，以及配額值是否可調整。
4. 若要檢視服務配額的其他資訊 (例如說明)，請選擇配額名稱。
5. (選用) 若要請求增加配額，請選取您要增加的配額、選取 Request quota increase (請求增加配額)、輸入或選取必要資訊，然後選取 Request (請求)。

若要使用更多使用 Service Quotas AWS Management Console，請參閱服務配額使用者指南。若要請求提升配額，請參閱《Service Quotas 使用者指南》中的請求提升配額。

AWS CLI

若要檢視 Amazon EKS 和 Fargate 服務配額 AWS CLI

執行下列命令，以檢視您的 Amazon EKS 配額。

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code eks \
    --output table
```

執行下列命令，以檢視您的 Fargate 配額。

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

> **Note**
>
> 傳回的配額是可在目前 AWS 區域中，在此帳戶之 Fargate 上同時執行的 Amazon ECS 任務或 Amazon EKS Pods 的數量。

若要使用更多使用服務配額 AWS CLI，請參閱《AWS CLI 命令參考》`service-quotas`中的。若要請求提升配額，請參閱 AWS CLI 命令參考中的 `request-service-quota-increase` 命令。

Service Quotas

名稱	預設	可調整	描述
每個叢集的存取項目	每個受支援的區域：3,000 個	否	每個叢集的存取項目數量上限。
叢集	每個受支援的區域：100	是	在目前區域中，此帳戶的 EKS 叢集的最大數量。
每個叢集的控制平面安全群組	每個受支援的區域：4	否	每個叢集的控制平面安全群組的最大數量 (您在建立叢集時所指定的群組)
EKS Anywhere Enterprise 訂閱	每個受支援的區域：10	是	目前區域中此帳戶的 EKS Anywhere Enterprise 訂閱數量上限。
每個叢集的 Fargate 描述檔	每個受支援的區域：10	是	每個叢集的 Fargate 描述檔的最大數量。
每個 Fargate 描述檔選取器的標籤對	每個受支援的區域：5	是	每個 Fargate 描述檔選取器的標籤對數目上限
每個叢集的受管節點群組	每個受支援的區域：30	是	每個叢集的受管節點群組數目上限
每個受管節點群組的節點	每個受支援的區域：450	是	每個受管節點群組的節點數目上限。
每個叢集的公有端點存取 CIDR 範圍	每個受支援的區域：40	否	每個叢集的公有端點存取 CIDR 範圍數目上限（您在建立或更新叢集時所指定的公有端點）。
已註冊的叢集	每個受支援的區域：10	是	在目前區域中，此帳戶的已註冊的叢集最大數量。

名稱	預設	可調整	描述
每個 Fargate 描述檔的選取器	每個受支援的區域：5	是	每個 Fargate 描述檔選取器的最大數量。

> **Note**
>
> 預設值是由設定的初始配額 AWS。這些預設值與實際套用的配額值和可能的服務配額上限不同。如需詳細資訊，請參閱《Service Quotas 使用者指南》中的 Service Quotas 術語。

在 Service Quotas 主控台中，Amazon Elastic Kubernetes Service (Amazon EKS) 之下會列出這些服務配額。若要針對顯示為可調整的值申請增加配額，請參閱《Service Quotas 使用指南》中的請求提高配額。

AWS Fargate 服務配額

Service Quotas 主控台中的 AWS Fargate 服務列出若干服務配額。下表僅描述適用於 Amazon EKS 的配額。您可以設定警示，在您的用量接近服務配額時發出警示。如需詳細資訊，請參閱 建立 CloudWatch 警示來監控 Fargate 資源用量指標。

新的可 AWS 帳戶 能具有較低的初始配額，隨著時間的推移而增加。Fargate 不斷監控每個帳戶中的帳戶使用情況 AWS 區域，然後根據使用情況自動增加配額。您也可以針對顯示為可調整的值申請增加配額。如需詳細資訊，請參閱《Service Quotas 使用者指南》中的請求提高配額。

名稱	預設	可調整	描述
Fargate 隨需 vCPU 資源計數	6	是	在當前區域中，此帳戶中能以 Fargate 隨需同時執行的 Fargate vCPUs 數量。

> **Note**
>
> 預設值是由設定的初始配額 AWS。這些預設值與實際套用的配額值和可能的服務配額上限不同。如需詳細資訊，請參閱《Service Quotas 使用者指南》中的 Service Quotas 術語。

> **Note**
>
> Fargate 亦會強制執行 Amazon ECS 任務和 Amazon EKS Pods 啟動速率配額。如需詳細資訊，請參閱 Amazon ECS 指南中的AWS Fargate 節流配額。

Amazon EKS 中的安全

雲安全 AWS 是最高的優先級。身為 AWS 客戶，您可以從資料中心和網路架構中獲益，該架構專為滿足對安全性最敏感的組織的需求而打造。

安全是 AWS 與您之間共同的責任。共同責任模型 將此描述為雲端的安全和雲端內的安全：

- 雲端的安全性 — AWS 負責保護在 AWS 雲端中執行 AWS 服務的基礎架構。對於 Amazon EKS，AWS 負責Kubernetes控制平面，其中包括控制平面節點和etcd資料庫。在 AWS 合規計畫中，第三方稽核員會定期測試並驗證我們的安全功效。若要了解適用於 Amazon EKS 的合規計劃，請參閱合規計劃範圍內的AWS 服務。
- 雲端內部的安全 – 您的責任包含下列領域：
  - 資料平面的安全組態，包括允許流量從 Amazon EKS 控制平面傳遞至客戶 VPC 的安全群組各項組態
  - 節點和容器本身的組態
  - 節點的作業系統 (包括更新和安全修補程式)
  - 其他相關的應用程式軟體：
    - 設定及管理網路控制，例如防火牆規則
    - 搭配 IAM 或另以其他方式管理平台層級身分與存取管理
  - 資料的機密性、公司的要求以及適用法律和法規

本文件有助於您了解如何在使用 Amazon EKS 時套用共同責任模型。下列主題說明如何將 Amazon EKS 設定為符合您的安全與合規目標。您也會學到如何使用其他可 AWS 協助您監控和保護 Amazon EKS 資源的服務。

> **Note**
>
> Linux 容器由控制群組 (cgroup) 和命名空間組成，這些命名空間有助於限制容器可存取的內容，但是所有容器皆會與主機 Amazon EC2 執行個體共用相同的 Linux 核心。非常不建議您以根使用者身分執行容器 (UID 0)，或授予容器存取主機資源或命名空間 (例如主機網路或主機 PID 命名空間)，因為這樣做會降低容器所提供之隔離的有效性。

主題

- Amazon EKS 的 Identity and Access Management
- Amazon Elastic Kubernetes Service 的合規驗證
- Amazon EKS 的恢復能力
- Amazon EKS 中的基礎設施安全
- Amazon EKS 中的組態與漏洞分析
- Amazon EKS 的安全最佳實務
- Pod 安全政策
- Pod 安全政策 (PSP) 移除常見問答集
- 搭配使用 AWS Secrets Manager 秘密和 Kubernetes
- Amazon EKS 連接器考量事項

憑證簽署

Kubernetes 憑證 API 可實現 X.509 憑證佈建的自動化程序。API 具有一個命令列介面，可供 Kubernetes API 用戶端從憑證授權機構 (CA) 請求和獲取 X.509 憑證。您可以使用 `CertificateSigningRequest` (CSR) 資源來請求受指示的簽署者簽署憑證。您的請求會在簽署之前獲核准或拒絕。Kubernetes 支援具有明確定義行為的內建簽署者與自訂簽署者。透過這種方式，客戶可以預測其 CSR 會發生什麼情況。如需憑證簽署的相關資訊，請參閱 Certificate Signing Requests (憑證簽署請求)。

其中一個內建簽署者是 `kubernetes.io/legacy-unknown`。CSR 資源的 `v1beta1` API 接受此傳統未知的簽署者。然而，穩定的 CSR `v1` API 不允許將 `signerName` 設定為 `kubernetes.io/legacy-unknown`。

Amazon EKS 版本 `1.21` 和較早版本允許 `legacy-unknown` 值在 `v1beta1` CSR API 中作為 `signerName`。此 API 可讓 Amazon EKS 憑證授權機構 (CA) 能夠產生憑證。然而，在 Kubernetes 版本 `1.22` 中，`v1beta1` CSR API 會由 `v1` CSR API 所取代。此 API 不支援「傳統未知」的 signerName。如果您想使用 Amazon EKS CA 在叢集上產生憑證，則必須使用自訂簽署者。其是在 Amazon EKS 第 `1.22` 版中引入。若要使用 CSR `v1` API 版本並產生新憑證，您必須遷移任何現有清單檔案和 API 用戶端。使用現有 `v1beta1` API 建立的現有憑證有效且可正常運作，直到憑證過期為止。這包含下列項目：

- 信任分佈：無。在 Kubernetes 叢集中，此簽署者沒有標準的信任或分佈。
- 允許的主體：任何
- 允許的 x509 擴充功能：榮譽 subjectAltName 和金鑰使用擴充功能，並捨棄其他擴充功能

- 允許的金鑰使用方式：不得包括 [「金鑰加密」、「數位簽章」、「伺服器身分驗證」] 以外的用法

> **Note**
>
> 不支援用戶端憑證簽署。

- 到期/憑證生命週期：1 年 (預設和最長)
- 允許/不允許 CA 位元：不允許

使用 signerName 產生 CSR 的範例

以下步驟將展示如何使用 `signerName: beta.eks.amazonaws.com/app-serving` 為 DNS 名稱 `myserver.default.svc` 產生服務憑證。將此作為您自己環境的指南。

1. 執行 `openssl genrsa -out myserver.key 2048` 命令以產生 RSA 私有金鑰。

```
openssl genrsa -out myserver.key 2048
```

2. 執行下列命令來產生憑證請求。

```
openssl req -new -key myserver.key -out myserver.csr -subj "/
CN=myserver.default.svc"
```

3. 產生 CSR 請求的 base64 值，並將其存放於變數中以供稍後步驟中使用。

```
base_64=$(cat myserver.csr | base64 -w 0 | tr -d "\n")
```

4. 執行下列命令以建立名為 `mycsr.yaml` 的檔案。在下列範例中，`beta.eks.amazonaws.com/app-serving` 是 `signerName`。

```
cat >mycsr.yaml <<EOF
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: myserver
spec:
  request: $base_64
  signerName: beta.eks.amazonaws.com/app-serving
  usages:
    - digital signature
```

```
    - key encipherment
    - server auth
EOF
```

5. 提交 CSR。

```
kubectl apply -f mycsr.yaml
```

6. 核准服務憑證。

```
kubectl certificate approve myserver
```

7. 驗證憑證已核發。

```
kubectl get csr myserver
```

範例輸出如下。

```
NAME       AGE     SIGNERNAME                            REQUESTOR
 CONDITION
myserver   3m20s   beta.eks.amazonaws.com/app-serving   kubernetes-admin
 Approved,Issued
```

8. 匯出已核發的憑證。

```
kubectl get csr myserver -o jsonpath='{.status.certificate}'| base64 -d
 > myserver.crt
```

將叢集升級至 Kubernetes 1.24 之前的憑證簽署考量

在 Kubernetes `1.23` 和較早版本中，具有未驗證 IP 和 DNS 主體別名 (SAN) 的 `kubelet` 服務憑證會自動與未驗證的 SAN 一起發行。已佈建的憑證會省略 SAN。在 `1.24` 和更新版本的叢集中，如果無法驗證 SAN，則不會發行 `kubelet` 服務憑證。如此可防止 `kubectl exec` 和 `kubectl logs` 命令運作。

將叢集升級到 `1.24` 之前，請先完成下列步驟，判斷叢集是否擁有尚未核准的憑證簽署請求 (CSR)：

1. 執行下列命令。

```
kubectl get csr -A
```

範例輸出如下。

```
NAME        AGE   SIGNERNAME                        REQUESTOR
                        REQUESTEDDURATION   CONDITION
csr-7znmf   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-42-149.region.compute.internal      <none>
 Approved
csr-9xx5q   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-65-38.region.compute.internal      <none>
 Approved, Issued
```

如果傳回的輸出顯示具有 kubernetes.io/kubelet-serving 簽署者的 CSR 已 Approved 但尚未 Issued，以用於您的節點，則您需要核准該請求。

2. 手動核准 CSR。使用您自己的值取代 csr-*7znmf*。

```
kubectl certificate approve csr-7znmf
```

若要在未來自動核准 CSR，建議您編寫核准控制器，該控制器可自動驗證並核准包含 Amazon EKS 無法驗證之 IP 或 DNS SAN 的 CSR。

Amazon EKS 的 Identity and Access Management

AWS Identity and Access Management (IAM) 可協助系統管理員安全地控制 AWS 資源存取權。 AWS 服務 IAM 管理員可以控制驗證 (已登入) 和授權 (具有許可) 來使用 Amazon EKS 資源。IAM 是您可以使用的 AWS 服務 ，無需額外付費。

物件

您如何使用 AWS Identity and Access Management (IAM) 會有所不同，具體取決於您在 Amazon EKS 中所做的工作。

服務使用者 – 如果您使用 Amazon EKS 執行任務，您的管理員會為您提供您需要的憑證和許可。隨著您為了執行作業而使用的 Amazon EKS 功能數量變多，您可能會需要額外的許可。了解存取許可的管理方式可協助您向管理員請求正確的許可。如果您無法存取 Amazon EKS 中的功能，請參閱 疑難排解 IAM。

服務管理員 – 若您在公司負責管理 Amazon EKS 資源，您應該具備服務使用的完整存取權限。您的任務是判斷服務使用者應存取的 Amazon EKS 功能和資源。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要進一步了解貴公司可搭配 Amazon EKS 使用 IAM 的方式，請參閱 Amazon EKS 如何搭配 IAM 運作。

IAM 管理員 – 如果您是 IAM 管理員，建議您掌握如何撰寫政策以管理 Amazon EKS 存取權的詳細資訊。若要檢視您可以在 IAM 中使用 Amazon EKS 身分型政策範例，請參閱 Amazon EKS 身分型政策範例。

使用身分驗證

驗證是您 AWS 使用身分認證登入的方式。您必須以 IAM 使用者身分或假設 IAM 角色進行驗證 (登入 AWS)。 AWS 帳戶根使用者

您可以使用透過 AWS 身分識別來源提供的認證，以聯合身分識別身分登入。 AWS IAM Identity Center (IAM 身分中心) 使用者、貴公司的單一登入身分驗證，以及您的 Google 或 Facebook 登入資料都是聯合身分識別的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。當您使 AWS 用同盟存取時，您會間接擔任角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需有關登入的詳細資訊 AWS，請參閱《AWS 登入 使用指南》 AWS 帳戶中的如何登入您的。

如果您 AWS 以程式設計方式存取，請 AWS 提供軟體開發套件 (SDK) 和命令列介面 (CLI)，以使用您的認證以加密方式簽署您的要求。如果您不使用 AWS 工具，則必須自行簽署要求。如需使用建議的方法自行簽署請求的詳細資訊，請參閱 IAM 使用者指南中的簽署 AWS API 請求。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如， AWS 建議您使用多重要素驗證 (MFA) 來增加帳戶的安全性。如需更多資訊，請參閱 AWS IAM Identity Center 使用者指南中的多重要素驗證和 IAM 使用者指南中的在 AWS中使用多重要素驗證 (MFA)。

AWS 帳戶 根使用者

當您建立時 AWS 帳戶，您會從一個登入身分開始，該身分可完整存取該帳戶中的所有資源 AWS 服務和資源。此身分稱為 AWS 帳戶 root 使用者，可透過使用您用來建立帳戶的電子郵件地址和密碼登入來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱《IAM 使用者指南》中的需要根使用者憑證的任務。

IAM 使用者和群組

IAM 使用者是您內部的身分，具 AWS 帳戶 有單一人員或應用程式的特定許可。建議您盡可能依賴暫時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需更多資訊，請參閱 IAM 使用者指南中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供暫時憑證。如需進一步了解，請參閱 IAM 使用者指南中的建立 IAM 使用者 (而非角色) 的時機。

IAM 角色

IAM 角色是您 AWS 帳戶 內部具有特定許可的身分。它類似 IAM 使用者，但不與特定的人員相關聯。您可以切換角色，在中暫時擔任 IAM 角色。 AWS Management Console 您可以透過呼叫 AWS CLI 或 AWS API 作業或使用自訂 URL 來擔任角色。如需使用角色的方法更多相關資訊，請參閱 IAM 使用者指南中的使用 IAM 角色。

使用暫時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 – 若要向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需有關聯合角色的相關資訊，請參閱 IAM 使用者指南中的為第三方身分提供者建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱 AWS IAM Identity Center 使用者指南中的許可集。
- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權 – 您可以使用 IAM 角色，允許不同帳戶中的某人 (信任的委託人) 存取您帳戶中的資源。角色是授予跨帳戶存取權的主要方式。但是，對於某些策略 AWS 服務，您可以將策略直接附加到資源（而不是使用角色作為代理）。若要了解跨帳戶存取權角色和資源型政策間的差異，請參閱 IAM 使用者指南中的 IAM 角色與資源類型政策的差異。
- 跨服務訪問 — 有些 AWS 服務 使用其他 AWS 服務功能。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。

- 轉寄存取工作階段 (FAS) — 當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。
- 服務角色 – 服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需更多資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務。
- 服務連結角色 — 服務連結角色是連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

- 在 Amazon EC2 上執行的應用程式 — 您可以使用 IAM 角色來管理在 EC2 執行個體上執行的應用程式以及發出 AWS CLI 或 AWS API 請求的臨時登入資料。這是在 EC2 執行個體內儲存存取金鑰的較好方式。若要將 AWS 角色指派給 EC2 執行個體並提供給其所有應用程式，請建立連接至執行個體的執行個體設定檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得暫時憑證。如需更多資訊，請參閱 IAM 使用者指南中的利用 IAM 角色來授予許可給 Amazon EC2 執行個體上執行的應用程式。

若要了解是否要使用 IAM 角色或 IAM 使用者，請參閱 IAM 使用者指南中的建立 IAM 角色 (而非使用者) 的時機。

使用政策管理存取權

您可以透 AWS 過建立原則並將其附加至 AWS 身分識別或資源來控制中的存取。原則是一個物件 AWS ，當與身分識別或資源相關聯時，會定義其權限。 AWS 當主參與者 (使用者、root 使用者或角色工作階段) 提出要求時，評估這些原則。政策中的許可決定是否允許或拒絕請求。大多數原則會 AWS 以 JSON 文件的形式儲存在中。如需 JSON 政策文件結構和內容的更多相關資訊，請參閱 IAM 使用者指南中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該原則的使用者可以從 AWS Management Console AWS CLI、或 AWS API 取得角色資訊。

身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱 IAM 使用者指南中的建立 IAM 政策。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管理的策略是獨立策略，您可以將其附加到您的 AWS 帳戶. 受管政策包括 AWS 受管政策和客戶管理的策略。若要了解如何在受管政策及內嵌政策間選擇，請參閱 IAM 使用者指南中的在受管政策和內嵌政策間選擇。

資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主參與者可以包括帳戶、使用者、角色、同盟使用者或。 AWS 服務

資源型政策是位於該服務中的內嵌政策。您無法在以資源為基礎的政策中使用 IAM 的 AWS 受管政策。

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些委託人 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon S3 和 Amazon VPC 是支援 ACL 的服務範例。 AWS WAF若要進一步了解 ACL，請參閱 Amazon Simple Storage Service 開發人員指南中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較不常見的原則類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體 (IAM 使用者或角色) 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交

集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可範圍的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 實體許可範圍。

- 服務控制策略（SCP）— SCP 是 JSON 策略，用於指定中組織或組織單位（OU）的最大權限。AWS Organizations AWS Organizations 是一種用於分組和集中管理您企業擁 AWS 帳戶 有的多個服務。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 限制成員帳戶中實體的權限，包括每個 AWS 帳戶根使用者帳戶。如需組織和 SCP 的更多相關資訊，請參閱 AWS Organizations 使用者指南中的 SCP 運作方式。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合使用者的暫時工作階段時，作為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需更多資訊，請參閱 IAM 使用者指南中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。要了解如何在涉及多個政策類型時 AWS 確定是否允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

Amazon EKS 如何搭配 IAM 運作

在您使用 IAM 管理對 Amazon EKS 的存取權之前，您應該了解哪些 IAM 功能可以與 Amazon EKS 搭配使用。若要取得 Amazon EKS 和其他 AWS 服務如何與 IAM 搭配運作的高階檢視，請參閱 IAM 使用者指南中的與 IAM 搭配使用的AWS 服務。

主題

- Amazon EKS 身分型政策
- Amazon EKS 資源型政策
- 以 Amazon EKS 標籤為基礎的授權
- Amazon EKS IAM 角色

Amazon EKS 身分型政策

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。Amazon EKS 支援特定動作、資源和條件索引鍵。若要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

動作

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼条件下可以對什麼資源執行哪些動作。

JSON 政策的 `Action` 元素描述您可以用來允許或拒絕政策中存取的動作。原則動作通常與關聯的 AWS API 作業具有相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

Amazon EKS 中的政策動作會在動作之前使用下列字首：`eks:`。例如，若要授予某人取得 Amazon EKS 叢集描述性資訊的許可，請在其政策中加入 `DescribeCluster` 動作。政策陳述式必須包含 `Action` 或 `NotAction` 元素。

若要在單一陳述式中指定多個動作，請用逗號分隔，如下所示：

```
"Action": ["eks:action1", "eks:action2"]
```

您也可以使用萬用字元 (*) 來指定多個動作。例如，若要指定開頭是 `Describe` 文字的所有動作，請包含以下動作：

```
"Action": "eks:Describe*"
```

若要查看 Amazon EKS 動作的清單，請參閱服務授權參考中的 Amazon Elastic Kubernetes Service 定義的動作。

資源

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

`Resource` JSON 政策元素可指定要套用動作的物件。陳述式必須包含 `Resource` 或 `NotResource` 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

```
"Resource": "*"
```

Amazon EKS 叢集資源有以下 ARN。

```
arn:aws:eks:region-code:account-id:cluster/cluster-name
```

如需 ARN 格式的詳細資訊，請參閱 Amazon 資源名稱 (ARN) 和 AWS 服務命名空間。

例如，若要在陳述式*my-cluster*中使用名稱指定叢集，請使用下列 ARN：

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/my-cluster"
```

若要指定屬於特定帳戶的所有叢集 AWS 區域，請使用萬用字元 (*)：

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/*"
```

有些 Amazon EKS 動作無法對特定資源執行，例如用來建立資源的動作。在這些情況下，您必須使用萬用字元 (*)。

```
"Resource": "*"
```

若要查看 Amazon EKS 資源類型及其 ARN 的詳細資訊，請參閱服務授權參考中的 Amazon Elastic Kubernetes Service 定義的資源。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 Amazon Elastic Kubernetes Service 定義的動作。

條件索引鍵

Amazon EKS 會定義自己的一組條件索引鍵，也支援使用一些全域條件金鑰。若要查看所有 AWS 全域條件金鑰，請參閱 IAM 使用者指南中的AWS 全域條件內容金鑰。

當 OpenID Connect 提供商關聯到您的叢集時，可以設定條件索引鍵。如需詳細資訊，請參閱 IAM 政策範例。

所有 Amazon EC2 操作都支援 `aws:RequestedRegion` 和 `ec2:Region` 條件索引鍵。如需詳細資訊，請參閱範例：限制特定 AWS 區域的存取權。

如需 Amazon EKS 條件金鑰的清單，請參閱服務授權參考中的 Amazon Elastic Kubernetes Service 定義的條件。若要了解您可以搭配哪些動作和資源使用條件索引鍵，請參閱 Amazon Elastic Kubernetes Service 定義的動作。

範例

若要檢視 Amazon EKS 身分型政策的範例，請參閱 Amazon EKS 身分型政策範例。

當您建立 Amazon EKS 叢集時，建立該叢集的 IAM 主體會由 Amazon EKS 控制平面中叢集的角色型存取控制 (RBAC) 組態來自動授予 `system:masters` 許可。此主體不會出現在任何可見的組態中，因此請務必記下最初建立叢集的主體。若要賦予其他 IAM 主體與叢集互動的能力，您必須編輯 Kubernetes 中的 `aws-auth ConfigMap` 並用 `group` 的名稱建立一個 Kubernetes `rolebinding` 或 `clusterrolebinding`，其名稱是您在 `aws-auth ConfigMap` 中指定的名稱。

若要取得有關使用的更多資訊 ConfigMap，請參閱授予 Kubernetes API 的存取權 。

Amazon EKS 資源型政策

Amazon EKS 不支援資源型政策。

以 Amazon EKS 標籤為基礎的授權

您可以將標籤連接至 Amazon EKS 資源，或是在請求中將標籤傳遞至 Amazon EKS。若要根據標籤控制存取，請使用 `aws:ResourceTag/`*`key-name`*、`aws:RequestTag/`*`key-name`* 或 `aws:TagKeys` 條件索引鍵，在政策的條件元素中，提供標籤資訊。如需標記 Amazon EKS 資源的詳細資訊，請參閱為您的 Amazon EKS 資源加上標籤。如需您在條件索引鍵下可使用哪些標籤動作的詳細資訊，請參閱《服務授權參考》中的 Amazon EKS 定義的動作。

Amazon EKS IAM 角色

IAM 角色是您 AWS 帳戶中具有特定許可的實體。

將暫時憑證與 Amazon EKS 搭配使用

您可以搭配聯合使用暫時憑證、擔任 IAM 角色，或是擔任跨帳戶角色。您可以透過呼叫AssumeRole或等 AWS STS API 作業來取得臨時安全登入資料GetFederationToken。

Amazon EKS 支援使用臨時憑證。

服務連結角色

服務連結角色可讓 AWS 服務存取其他服務中的資源，以代表您完成動作。服務連結角色會顯示在您的 IAM 帳戶中，並由該服務所擁有。 管理員可以檢視，但不能編輯服務連結角色的許可。

Amazon EKS 支援服務連結角色。如需建立或管理 Amazon EKS 服務連結角色的詳細資訊，請參閱 使用 Amazon EKS 的服務連結角色。

服務角色

此功能可讓服務代表您擔任服務角色。此角色可讓服務存取其他服務中的資源，以代表您完成動作。服務角色會出現在您的 IAM 帳戶中，且由該帳戶所擁有。這表示 IAM 管理員可以變更此角色的許可。不過，這樣可能會破壞此服務的功能。

Amazon EKS 支援服務角色。如需詳細資訊，請參閱 Amazon EKS 叢集 IAM 角色 及 Amazon EKS 節點 IAM 角色。

在 Amazon EKS 中選擇 IAM 角色

在 Amazon EKS 中建立叢集資源時，您必須選擇一個角色，以允許 Amazon EKS 代表您存取其他數個 AWS 資源。如果您之前已建立服務角色，則 Amazon EKS 會提供一個角色清單供您選擇。請務必選擇連接 Amazon EKS 受管政策的角色。如需更多詳細資訊，請參閱 檢查現有的叢集角色 及 檢查現有的節點角色。

Amazon EKS 身分型政策範例

根據預設，IAM 使用者和角色不具備建立或修改 Amazon EKS 資源的許可。他們也無法使用 AWS Management Console AWS CLI、或 AWS API 執行工作。IAM 管理員必須建立 IAM 政策，授予使用者和角色在指定資源上執行特定 API 作業的所需許可。管理員接著必須將這些政策連接至需要這些許可的 IAM 使用者或群組。

若要了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的在 JSON 標籤上建立政策。

當您建立 Amazon EKS 叢集時，建立該叢集的 IAM 主體會由 Amazon EKS 控制平面中叢集的角色型存取控制 (RBAC) 組態來自動授予 `system:masters` 許可。此主體不會出現在任何可見的組態中，因此請務必記下最初建立叢集的主體。若要賦予其他 IAM 主體與叢集互動的能力，您必須編輯 Kubernetes 中的 `aws-auth ConfigMap` 並用 `group` 的名稱建立一個 Kubernetes `rolebinding` 或 `clusterrolebinding`，其名稱是您在 `aws-auth ConfigMap` 中指定的名稱。

若要取得有關使用的更多資訊 ConfigMap，請參閱授予 Kubernetes API 的存取權 。

主題

- 政策最佳實務
- 使用 Amazon EKS 主控台
- 允許 IAM 使用者檢視他們自己的許可

- 在 AWS 雲端上建立 Kubernetes 叢集
- 在 Outpost 上建立本機 Kubernetes 叢集
- 更新 Kubernetes 叢集
- 所有叢集的清單或描述

政策最佳實務

身分型政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 Amazon EKS 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管原則並邁向最低權限權限 — 若要開始將權限授與使用者和工作負載，請使用可授與許多常見使用案例權限的AWS 受管理原則。它們可用在您的 AWS 帳戶. 建議您透過定義特定於您使用案例的 AWS 客戶管理政策，進一步降低使用權限。如需更多資訊，請參閱 IAM 使用者指南中的 AWS 受管政策或任務職能的AWS 受管政策。
- 套用最低許可許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的權限。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 中的政策和許可。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。您也可以使用條件來授與對服務動作的存取權 (如透過特定) 使用這些動作 AWS 服務，例如 AWS CloudFormation。如需更多資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您編寫安全且實用的政策。如需更多資訊，請參閱 IAM 使用者指南中的 IAM Access Analyzer 政策驗證。
- 需要多因素身份驗證 (MFA) — 如果您的案例需要 IAM 使用者或根使用者 AWS 帳戶，請開啟 MFA 以獲得額外的安全性。若要在呼叫 API 作業時請求 MFA，請將 MFA 條件新增至您的政策。如需更多資訊，請參閱 IAM 使用者指南中的設定 MFA 保護的 API 存取。

如需 IAM 中最佳實務的相關資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。

使用 Amazon EKS 主控台

若要存取 Amazon EKS 主控台，IAM 主體必須擁有最基本的一組許可。這些許可允許主體列出和檢視您 AWS 帳戶中 Amazon EKS 資源的詳細資訊。如果您建立比最低必要許可更嚴格的身分型政策，則對於具有該政策的主體而言，主控台將無法如預期運作。

為了確保您的 IAM 主體仍然可以使用 Amazon EKS 主控台，建立擁有唯一名稱的政策，例如 AmazonEKSAdminPolicy。將政策連接至主體。如需詳細資訊，請參閱 IAM 使用者指南中的新增和移除 IAM 身分許可。

> **Important**
>
> 下列範例政策可讓主體檢視主控台內的組態標籤上的資訊。若要檢視 AWS Management Console中概觀和資源標籤的資訊，則該主體還需要 Kubernetes 許可。如需詳細資訊，請參閱所需的許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "eks.amazonaws.com"
                }
            }
        }
    ]
}
```

您不需要為只對 AWS CLI 或 AWS API 進行呼叫的主體允許最低主控台權限。反之，只需允許存取符合您嘗試執行之 API 作業的動作就可以了。

允許 IAM 使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視連接到他們使用者身分的內嵌及受管政策。此原則包含在主控台上或以程式設計方式使用 AWS CLI 或 AWS API 完成此動作的權限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

在 AWS 雲端上建立 Kubernetes 叢集

AWS 區域您可以 AWS 區域 使用您要在 AWS 區域 中建立叢集的取代。如果您在 AWS Management Console中看到警告顯示 The actions in your policy do not support resource-level permissions and require you to choose **`All resources`** (您政策中的動作不支援資源層級許可，且需要您選擇)，您可以放心忽略此警告。如果您的帳戶已具有該*AWSServiceRoleForAmazonEKS*角色，則可以從策略中移除該`iam:CreateServiceLinkedRole`動作。如果您曾在帳戶中建立 Amazon EKS 叢集，則此角色已存在，除非您已將其刪除。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
eks.amazonaws.com/AWSServiceRoleForAmazonEKS",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "iam:AWSServiceName": "eks"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        }
    ]
}
```

在 Outpost 上建立本機 Kubernetes 叢集

AWS 區域您可以 AWS 區域 使用您要在 AWS 區域 中建立叢集的取代。如果您在 AWS Management Console中看到警告顯示 The actions in your policy do not support resource-level permissions and require you to choose **`All resources`** (您政策中的動作不支援資源層級許可，且需要您選擇)，您可以放心忽略此警告。如果您的帳戶已具有 `AWSServiceRoleForAmazonEKSLocalOutpost` 角色，您可以移除來自政策的 `iam:CreateServiceLinkedRole` 動作。如果您曾在帳戶中建立位於 Outpost 上的 Amazon EKS 本機叢集，則此角色已存在，除非您已將其刪除。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
```

```
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "iam:GetRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/outposts.eks-
local.amazonaws.com/AWSServiceRoleForAmazonEKSLocalOutpost"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "iam:ListAttachedRolePolicies"
            ]
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:TagInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/eks-local-*",
            "Effect": "Allow"
        },
    ]
}
```

更新 Kubernetes 叢集

AWS 區域

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:UpdateClusterVersion",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        }
    ]
}
```

所有叢集的清單或描述

此範例政策包含列出及說明帳戶中所有叢集所需的最低許可。IAM 主體必須能夠列出和描述叢集才能使用update-kubeconfig AWS CLI 命令。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

使用 Amazon EKS 的服務連結角色

Amazon Elastic Kubernetes Service 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Amazon EKS 的一種特殊 IAM 角色類型。Amazon EKS 預先定義服務連結角色，包含服務代表您呼叫其他 AWS 服務所需的所有許可。

主題

- 使用 Amazon EKS 叢集的角色
- 使用 Amazon EKS 節點群組的角色
- 使用 Amazon EKS Fargate 描述檔的角色

- 使用角色將 Kubernetes 叢集連線至 Amazon EKS
- 使用位於 Outpost 上的 Amazon EKS 本機叢集的角色

使用 Amazon EKS 叢集的角色

Amazon Elastic Kubernetes Service 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Amazon EKS 的一種特殊 IAM 角色類型。Amazon EKS 預先定義服務連結角色，並包含服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓設定 Amazon EKS 更為簡單，因為您不必手動新增必要的許可。Amazon EKS 定義其服務連結角色的許可，除非另有定義，否則僅有 Amazon EKS 可以擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您的 Amazon EKS 資源，避免您不小心移除資源的存取許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，尋找 Service-Linked Role (服務連結角色) 欄中顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

Amazon EKS 的服務連結角色許可

Amazon EKS 使用名為 `AWSServiceRoleForAmazonEKS` 的服務連結角色：此角色可讓 Amazon EKS 管理您帳戶中的叢集。連接的政策允許角色管理下列資源：網路介面、安全群組、記錄和 VPC。

> **Note**
>
> `AWSServiceRoleForAmazonEKS` 服務連結角色並非叢集建立所需的角色。如需詳細資訊，請參閱 Amazon EKS 叢集 IAM 角色。

`AWSServiceRoleForAmazonEKS` 服務連結角色信任下列服務擔任角色：

- `eks.amazonaws.com`

此角色許可政策允許 Amazon EKS 對指定資源完成下列動作：

- `AmazonEKSServiceRolePolicy`

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立 Amazon EKS 的服務連結角色

您不需要手動建立一個服務連結角色。當您在 AWS Management Console、或 AWS API 中建立叢集時 AWS CLI，Amazon EKS 會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您建立叢集時，Amazon EKS 會再次為您建立服務連結角色。

編輯 Amazon EKS 的服務連結角色

Amazon EKS 不允許您編輯 AWSServiceRoleForAmazonEKS 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

刪除 Amazon EKS 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除服務連結角色

在您使用 IAM 刪除服務連結角色之前，您必須先刪除該角色所使用的任何資源。

> **Note**
>
> 若 Amazon EKS 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要刪除 **AWSServiceRoleForAmazonEKS** 角色使用的 Amazon EKS 資源。

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 如果您的叢集具有任何節點群組或 Fargate 描述檔，則必須先將其刪除，才能刪除叢集。如需詳細資訊，請參閱 刪除受管節點群組 及 刪除 Fargate 描述檔。
4. 在 Clusters (叢集) 頁面上，選擇您要刪除的叢集，然後選擇 Delete (刪除)。
5. 在刪除確認視窗中輸入叢集的名稱，然後選擇 Delete (刪除)。

6. 重複對您帳戶中的任何其他叢集執行此程序。等候所有刪除作業完成。

手動刪除服務連結角色

使用 IAM 主控台或 AWS API 刪除AWSServiceRoleForAmazonEKS服務連結角色。 AWS CLI如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

Amazon EKS 服務連結角色的支援區域

Amazon EKS 在所有提供服務的區域中支援使用服務連結角色。如需詳細資訊，請參閱 Amazon EKS endpoints and quotas (Amazon EKS 端點和配額)。

使用 Amazon EKS 節點群組的角色

Amazon EKS 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Amazon EKS 的一種特殊 IAM 角色類型。Amazon EKS 預先定義服務連結角色，包含服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓設定 Amazon EKS 更為簡單，因為您不必手動新增必要的許可。Amazon EKS 定義其服務連結角色的許可，除非另有定義，否則僅有 Amazon EKS 可以擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您的 Amazon EKS 資源，避免您不小心移除資源的存取許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，尋找 Service-Linked Role (服務連結角色) 欄中顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

Amazon EKS 的服務連結角色許可

Amazon EKS 使用名為 AWSServiceRoleForAmazonEKSNodegroup 的服務連結角色：此角色可讓 Amazon EKS 管理您帳戶中的節點群組。連接的政策允許角色管理下列資源：Auto Scaling 群組、安全群組、啟動範本和 IAM 執行個體描述檔。

AWSServiceRoleForAmazonEKSNodegroup 服務連結角色信任下列服務以擔任角色：

- eks-nodegroup.amazonaws.com

此角色許可政策允許 Amazon EKS 對指定資源完成下列動作：

- AWSServiceRoleForAmazonEKSNodegroup

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立 Amazon EKS 的服務連結角色

您不需要手動建立一個服務連結角色。當您 CreateNodegroup 使用 AWS Management Console、或 AWS API 時 AWS CLI，Amazon EKS 會為您建立服務連結角色。

> **Important**
>
> 此服務連結角色可以顯示在您的帳戶，如果您於其他服務中完成一項動作時，可以使用支援此角色的功能。如果您在 2017 年 1 月 1 日之前使用 Amazon EKS 服務，那麼該服務開始支援服務連結角色時，Amazon EKS 就會在您的帳戶中建立該 AWSServiceRoleForAmazonEKSNodegroup 角色。若要進一步了解，請參閱顯示在我的 IAM 帳戶中的新角色。

在 Amazon EKS（AWS API）中創建服務鏈接角色

您不需要手動建立一個服務連結角色。當您在 AWS Management Console、或 AWS API 中建立受管節點群組時 AWS CLI，Amazon EKS 會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您建立另一個受管節點群組時，Amazon EKS 會再次為您建立服務連結角色。

編輯 Amazon EKS 的服務連結角色

Amazon EKS 不允許您編輯 AWSServiceRoleForAmazonEKSNodegroup 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

刪除 Amazon EKS 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除服務連結角色

在您使用 IAM 刪除服務連結角色之前，您必須先刪除該角色所使用的任何資源。

> **Note**
>
> 若 Amazon EKS 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要刪除 **AWSServiceRoleForAmazonEKSNodegroup** 角色使用的 Amazon EKS 資源。

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 選取 Compute (運算) 標籤。
4. 在 Node Groups (節點群組) 區段中，選擇要刪除的節點群組。
5. 在刪除確認視窗中，輸入節點群組的名稱，然後選擇 Delete (刪除)。
6. 重複對叢集中的任何其他節點群組執行此程序。等候所有刪除作業完成。

手動刪除服務連結角色

使用 IAM 主控台或 AWS API 刪除`AWSServiceRoleForAmazonEKSNodegroup`服務連結角色。AWS CLI如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

Amazon EKS 服務連結角色的支援區域

Amazon EKS 在所有提供服務的區域中支援使用服務連結角色。如需詳細資訊，請參閱 Amazon EKS endpoints and quotas (Amazon EKS 端點和配額)。

使用 Amazon EKS Fargate 描述檔的角色

Amazon EKS 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Amazon EKS 的一種特殊 IAM 角色類型。Amazon EKS 預先定義服務連結角色，並包含服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓設定 Amazon EKS 更為簡單，因為您不必手動新增必要的許可。Amazon EKS 定義其服務連結角色的許可，除非另有定義，否則僅有 Amazon EKS 可以擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您的 Amazon EKS 資源，避免您不小心移除資源的存取許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，尋找 Service-Linked Role (服務連結角色) 欄中顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

Amazon EKS 的服務連結角色許可

Amazon EKS 使用名為 AWSServiceRoleForAmazonEKSForFargate 的服務連結角色：該角色允許 Amazon EKS Fargate 設定 Fargate Pods 所需的 VPC 聯網。已連接的政策允許角色建立和刪除彈性網路介面，並描述彈性網路介面和資源。

AWSServiceRoleForAmazonEKSForFargate 服務連結角色信任下列服務以擔任角色：

- eks-fargate.amazonaws.com

此角色許可政策允許 Amazon EKS 對指定資源完成下列動作：

- AmazonEKSForFargateServiceRolePolicy

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立 Amazon EKS 的服務連結角色

您不需要手動建立一個服務連結角色。當您在 AWS Management Console、或 AWS API 中建立 Fargate 設定檔時 AWS CLI，Amazon EKS 會為您建立服務連結角色。

> **Important**
>
> 此服務連結角色可以顯示在您的帳戶，如果您於其他服務中完成一項動作時，可以使用支援此角色的功能。如果您在 2019 年 12 月 13 日之前使用 Amazon EKS 服務，那麼該服務開始支援服務連結角色時，Amazon EKS 就會在您的帳戶中建立該 AWSServiceRoleForAmazonEKSForFargate 角色。若要進一步了解，請參閱顯示在我的 IAM 帳戶中的新角色。

在 Amazon EKS（AWS API）中創建服務鏈接角色

您不需要手動建立一個服務連結角色。當您在 AWS Management Console、或 AWS API 中建立 Fargate 設定檔時 AWS CLI，Amazon EKS 會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您建立另一個受管節點群組時，Amazon EKS 會再次為您建立服務連結角色。

編輯 Amazon EKS 的服務連結角色

Amazon EKS 不允許您編輯 AWSServiceRoleForAmazonEKSForFargate 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

刪除 Amazon EKS 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除服務連結角色

在您使用 IAM 刪除服務連結角色之前，您必須先刪除該角色所使用的任何資源。

> **Note**
>
> 若 Amazon EKS 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要刪除 **AWSServiceRoleForAmazonEKSForFargate** 角色使用的 Amazon EKS 資源。

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 在 Clusters (叢集) 頁面上，選取您的叢集。
4. 選取 Compute (運算) 標籤。
5. 如果 Fargate Profiles (Fargate 描述檔) 區段中有任何 Fargate 描述檔，請個別選取每個描述檔，然後選擇 Delete (刪除)。
6. 在刪除確認視窗中輸入描述檔的名稱，然後選擇 Delete (刪除)。
7. 針對叢集中的任何其他 Fargate 描述檔，以及您帳戶中的任何其他叢集，重複此程序。

手動刪除服務連結角色

使用 IAM 主控台或 AWS API 刪除 AWSServiceRoleForAmazonEKSForFargate服務連結角色。 AWS CLI如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

Amazon EKS 服務連結角色的支援區域

Amazon EKS 在所有提供服務的區域中支援使用服務連結角色。如需詳細資訊，請參閱 Amazon EKS endpoints and quotas (Amazon EKS 端點和配額)。

使用角色將 Kubernetes 叢集連線至 Amazon EKS

Amazon EKS 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Amazon EKS 的一種特殊 IAM 角色類型。Amazon EKS 預先定義服務連結角色，包含服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓設定 Amazon EKS 更為簡單，因為您不必手動新增必要的許可。Amazon EKS 定義其服務連結角色的許可，除非另有定義，否則僅有 Amazon EKS 可以擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您的 Amazon EKS 資源，避免您不小心移除資源的存取許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，尋找 Service-Linked Role (服務連結角色) 欄中顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

Amazon EKS 的服務連結角色許可

Amazon EKS 使用名為 `AWSServiceRoleForAmazonEKSConnector` 的服務連結角色：此角色可讓 Amazon EKS 連接 Kubernetes 叢集。連接的政策可讓角色管理必要的資源，以連線至已註冊的 Kubernetes 叢集。

`AWSServiceRoleForAmazonEKSConnector` 服務連結角色信任下列服務以擔任角色：

- `eks-connector.amazonaws.com`

此角色許可政策允許 Amazon EKS 對指定資源完成下列動作：

- `AmazonEKSConnectorServiceRolePolicy`

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立 Amazon EKS 的服務連結角色

您不需要手動建立一個服務連結角色來連線叢集。當您在 AWS Management Console、、或 AWS API 中連接叢集時 AWS CLIeksctl，Amazon EKS 會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。在連線叢集時，Amazon EKS 會再次為您建立服務連結角色。

編輯 Amazon EKS 的服務連結角色

Amazon EKS 不允許您編輯 AWSServiceRoleForAmazonEKSConnector 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

刪除 Amazon EKS 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除服務連結角色

在您使用 IAM 刪除服務連結角色之前，您必須先刪除該角色所使用的任何資源。

> **Note**
>
> 若 Amazon EKS 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要刪除 **AWSServiceRoleForAmazonEKSConnector** 角色使用的 Amazon EKS 資源。

1. 在 https://console.aws.amazon.com/eks/home#/clusters 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中選擇 Clusters (叢集)。
3. 在 Clusters (叢集) 頁面上，選取您的叢集。
4. 選取 Deregister (取消註冊) 索引標籤，然後選取 OK (確定) 索引標籤。

手動刪除服務連結角色

使用 IAM 主控台或 AWS API 刪除 AWSServiceRoleForAmazonEKSConnector服務連結角色。 AWS CLI如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

使用位於 Outpost 上的 Amazon EKS 本機叢集的角色

Amazon Elastic Kubernetes Service 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Amazon EKS 的一種特殊 IAM 角色類型。Amazon EKS 預先定義服務連結角色，並包含服務代表您呼叫其他 AWS 服務所需的所有許可。

服務連結角色可讓設定 Amazon EKS 更為簡單，因為您不必手動新增必要的許可。Amazon EKS 定義其服務連結角色的許可，除非另有定義，否則僅有 Amazon EKS 可以擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您的 Amazon EKS 資源，避免您不小心移除資源的存取許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，尋找 Service-Linked Role (服務連結角色) 欄中顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

Amazon EKS 的服務連結角色許可

Amazon EKS 使用名為 `AWSServiceRoleForAmazonEKSLocalOutpost` 的服務連結角色：此角色可讓 Amazon EKS 管理您帳戶中的本機叢集。已連結的政策允許角色管理下列資源：網路介面、安全群組、日誌和 Amazon EC2 執行個體。

> **Note**
>
> `AWSServiceRoleForAmazonEKSLocalOutpost` 服務連結角色並非叢集建立所需的角色。如需詳細資訊，請參閱 Amazon EKS 叢集 IAM 角色。

`AWSServiceRoleForAmazonEKSLocalOutpost` 服務連結角色信任下列服務擔任角色：

- `outposts.eks-local.amazonaws.com`

此角色許可政策允許 Amazon EKS 對指定資源完成下列動作：

- `AmazonEKSServiceRolePolicy`

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立 Amazon EKS 的服務連結角色

您不需要手動建立一個服務連結角色。當您在 AWS Management Console、或 AWS API 中建立叢集時 AWS CLI，Amazon EKS 會為您建立服務連結角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您建立叢集時，Amazon EKS 會再次為您建立服務連結角色。

編輯 Amazon EKS 的服務連結角色

Amazon EKS 不允許您編輯 `AWSServiceRoleForAmazonEKSLocalOutpost` 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色。

刪除 Amazon EKS 的服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除服務連結角色

在您使用 IAM 刪除服務連結角色之前，您必須先刪除該角色所使用的任何資源。

> **Note**
>
> 若 Amazon EKS 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要刪除 **`AWSServiceRoleForAmazonEKSLocalOutpost`** 角色使用的 Amazon EKS 資源。

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導覽窗格中選擇 Amazon EKS Clusters (叢集)。
3. 如果您的叢集具有任何節點群組或 Fargate 描述檔，則必須先將其刪除，才能刪除叢集。如需詳細資訊，請參閱 刪除受管節點群組 及 刪除 Fargate 描述檔。
4. 在 Clusters (叢集) 頁面上，選擇您要刪除的叢集，然後選擇 Delete (刪除)。
5. 在刪除確認視窗中輸入叢集的名稱，然後選擇 Delete (刪除)。
6. 重複對您帳戶中的任何其他叢集執行此程序。等候所有刪除作業完成。

手動刪除服務連結角色

使用 IAM 主控台或 AWS API 刪除AWSServiceRoleForAmazonEKSLocalOutpost服務連結角色。AWS CLI如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

Amazon EKS 服務連結角色的支援區域

Amazon EKS 在所有提供服務的區域中支援使用服務連結角色。如需詳細資訊，請參閱 Amazon EKS endpoints and quotas (Amazon EKS 端點和配額)。

Amazon EKS 叢集 IAM 角色

每個叢集都必須具備 Amazon EKS 叢集 IAM 角色。受 Amazon EKS 管理的 Kubernetes 叢集使用此角色來管理節點，而舊式雲端供應商則使用此角色為服務建立具有 Elastic Load Balancing 的負載平衡器。

建立 Amazon EKS 叢集之前，您必須先建立具有以下任一 IAM 政策的 IAM 角色：

- AmazonEKSClusterPolicy
- 自訂 IAM 政策。以下是 Kubernetes 叢集可管理節點的最低許可，但無法讓舊式雲端供應商建立具有 Elastic Load Balancing 的負載平衡器。您的自訂 IAM 政策必須至少擁有下列許可：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:*:*:instance/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:TagKeys": "kubernetes.io/cluster/*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeNetworkInterfaces",
```

```
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeAvailabilityZones",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> 在 2023 年 10 月 3 日之前，每個叢集的 IAM 角色ClusterPolicy都需要 AmazonEks。在 2020 年 4 月 16 日之前，需要 AmazonEKS 馬遜ServicePolicy和亞馬遜ClusterPolicy公司，並且建議的角色名稱為。eksServiceRole使用AWSServiceRoleForAmazonEKS服務連結角色時，在 2020 年 4 月 16 日或之後建立的叢集不再需要 AmazonEks ServicePolicy 政策。

檢查現有的叢集角色

您可使用以下程序，檢查您的帳戶是否已有 Amazon EKS 叢集角色。

在 IAM 主控台中檢查 **eksClusterRole**

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 搜尋 eksClusterRole 的角色清單。如果包含 eksClusterRole 的角色不存在，請參閱 建立 Amazon EKS 叢集角色 以建立角色。如果包含 eksClusterRole 的角色存在，請選取角色以檢視連接的政策。
4. 選擇許可。
5. 確保將 AmazonEks ClusterPolicy 管理政策附加到該角色上。如果已連接政策，則您的 Amazon EKS 叢集角色應已設定妥當。
6. 選擇 Trust Relationships (信任關係)，然後選擇 Edit trust policy (編輯信任政策)。
7. 確認信任關係包含下列政策。如果信任關係符合下列政策，請選擇 Cancel (取消)。如果信任關係不符合，請將政策複製到 Edit trust policy (編輯信任政策) 視窗中，然後選擇 Update policy (更新政策)。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

建立 Amazon EKS 叢集角色

您可以使用 AWS Management Console 或 AWS CLI 來建立叢集角色。

AWS Management Console

在 IAM 主控台中建立 Amazon EKS 叢集角色

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 在信任的實體類型)下，選取 AWS 服務。
4. 從其他 AWS 服務的使用案例下拉式清單中，選擇 EKS。
5. 針對您的使用案例選擇 EKS - Cluster (EKS - 叢集)，然後選擇 Next (下一步)。
6. 在 Add permissions (新增許可) 標籤上，選擇 Next (下一步)。
7. 針對 Role name (角色名稱)，為您的角色輸入唯一名稱 (例如 **eksClusterRole**)。
8. 針對 Description (描述)，輸入描述性文字，如 **Amazon EKS - Cluster role**。
9. 選擇建立角色。

AWS CLI

1. 將下列內容複製到名為 *cluster-trust-policy.json* 的檔案。

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. 建立角色。您可以將 ***eksClusterRole*** 取代為您選擇的任何名稱。

```
aws iam create-role \
  --role-name eksClusterRole \
  --assume-role-policy-document file://"cluster-trust-policy.json"
```

3. 將所需的 IAM 政策連接至角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name eksClusterRole
```

Amazon EKS 節點 IAM 角色

Amazon EKS 節點kubelet常駐程式會代表您呼叫 AWS API。節點透過 IAM 執行個體描述檔和關聯的政策，取得這些 API 呼叫的許可。啟動節點並在叢集中註冊之前，您必須先為那些節點建立啟動時要使用的 IAM 角色。這項要求會套用到運用 Amazon 提供的 Amazon EKS 最佳化 AMI，或是任何其他您打算使用的節點 AMI 來啟動的節點。此外，此要求皆適用於受管節點群組和自我管理節點。

> **Note**
>
> 您無法使用用於建立任何叢集的相同角色。

建立節點之前，您必須先建立具有以下許可的 IAM 角色：

- kubelet 描述 VPC 中之 Amazon EC2 資源的許可，例如 AmazonEKSWorkerNodePolicy 政策提供的許可。此政策也為 Amazon EKS Pod 身分識別代理程式提供許可。

- kubelet 使用 Amazon Elastic Container Registry (Amazon ECR) 中之容器映像的許可，例如 AmazonEC2ContainerRegistryReadOnly 政策提供的許可。由於用於聯網的內建附加元件會執行使用 Amazon ECR 容器映像的 Pod，因此需要 Amazon Elastic Container Registry (Amazon ECR) 容器映像的使用許可。
- (選用) Amazon EKS Pod 身分識別代理程式使用 eks-auth:AssumeRoleForPodIdentity 動作擷取 Pod 憑證的許可。如果您不使用 AmazonEks WorkerNodePolicy，則除了 EC2 許可之外，還必須提供此許可才能使用 EKS 網繭身分。
- (選用) 如果您不使用 IRSA 或 EKS Pod 身分識別向 VPC CNI Pod 授予許可，則必須透過執行個體角色為 VPC CNI 提供許可。您可以使用 AmazonEKS_CNI_Policy 受管政策 (如果您使用 IPv4 系列建立叢集) 或您建立的 IPv6 政策 (如果您使用 IPv6 系列建立叢集)。但是，我們建議您將政策連接至專門用於 Amazon VPC CNI 附加元件的單獨角色，而不是將政策連接至此角色。如需為 Amazon VPC CNI 附加元件建立單獨角色的更多資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

> **Note**
>
> 在 2023 年 10 月 3 日前，每個受管節點群組的 IAM 角色都必須具有 AmazonEKSWorkerNodePolicy 和 AmazonEC2ContainerRegistryReadOnly。Amazon EC2 節點群組的 IAM 角色必須與 Fargate 設定檔不同。如需詳細資訊，請參閱 Amazon EKS Pod 執行 IAM 角色。

檢查現有的節點角色

您可使用以下程序，檢查您的帳戶是否已有 Amazon EKS 節點角色。

在 IAM 主控台中檢查 **eksNodeRole**

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 搜尋 eksNodeRole、AmazonEKSNodeRole 或 NodeInstanceRole 的角色清單。如果具有這些名稱之一的角色不存在，請參閱 建立 Amazon EKS 節點 IAM 角色 以建立角色。如果包含 eksNodeRole、AmazonEKSNodeRole 或 NodeInstanceRole 的角色存在，請選取角色以檢視連接的政策。
4. 選擇許可。

5. 請確定 AmazonEks WorkerNodePolicy 和 AmazonEC2 ContainerRegistryReadOnly 受管政策已附加至角色，或使用最低許可附加自訂政策。

> **Note**
>
> 若 AmazonEKS_CNI_Policy 政策已連接至角色，我們建議將其移除，並轉而將其連接到映射至 aws-node Kubernetes 服務帳戶的 IAM 角色。如需詳細資訊，請參閱 設定為 Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

6. 選擇 Trust Relationships (信任關係)，然後選擇 Edit trust policy (編輯信任政策)。
7. 確認信任關係包含下列政策。如果信任關係符合下列政策，請選擇 Cancel (取消)。如果信任關係不符合，請將政策複製到 Edit trust policy (編輯信任政策) 視窗中，然後選擇 Update policy (更新政策)。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

建立 Amazon EKS 節點 IAM 角色

您可以使用 AWS Management Console 或建立節點 IAM 角色 AWS CLI。

AWS Management Console

在 IAM 主控台中建立 Amazon EKS 節點角色

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 在 Roles (角色) 頁面上，選擇 Create role (建立角色)。
4. 在 Select trusted entity (選取信任的實體) 頁面上，執行以下作業：

a. 在 Trusted entity type (信任的實體類型) 區段中，選擇 AWS service (服務)。

b. 在 Use case (使用案例) 下，選擇 EC2。

c. 選擇下一步。

5. 在新增許可頁面上，連接自訂政策或執行下列動作：

a. 在 Filter policies (篩選政策) 方塊中，輸入 **AmazonEKSWorkerNodePolicy**。

b. WorkerNodePolicy在搜尋結果中選取 AmazonEks 左側的核取方塊。

c. 選擇 Clear filters (清除篩選條件)。

d. 在 Filter policies (篩選政策) 方塊中，輸入 **AmazonEC2ContainerRegistryReadOnly**。

e. 選取搜尋結果ContainerRegistryReadOnly中亞馬遜 EC2 左側的核取方塊。

AmazonEKS_CNI_Policy 受管政策或您建立的 IPv6 政策，也必須連接至此角色或映射至 aws-node Kubernetes 服務帳戶的其他角色。建議您將政策指派給與 Kubernetes 服務帳戶相關聯的角色，而不要將政策指派給此角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

f. 選擇下一步。

6. 在 Name, review, and create (命名、檢閱和建立) 頁面上，執行以下作業：

a. 針對 Role name (角色名稱)，為您的角色輸入唯一名稱 (例如 **AmazonEKSNodeRole**)。

b. 針對 Description (描述)，請以描述性文字 (例如 **Amazon EKS - Node role**) 取代目前的文字。

c. 藉由連接標籤做為鍵值對，在新增標籤 (選用) 下將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的標記 IAM 資源。

d. 選擇建立角色。

AWS CLI

1. 執行下列命令以建立 node-role-trust-relationship.json 檔案。

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

2. 建立 IAM 角色。

```
aws iam create-role \
  --role-name AmazonEKSNodeRole \
  --assume-role-policy-document file://"node-role-trust-relationship.json"
```

3. 將兩個所需的受管 IAM 政策連接到角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name AmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name AmazonEKSNodeRole
```

4. 根據您建立叢集所使用的 IP 系列，將以下 IAM 政策之一連接至 IAM 角色。政策必須連接至此角色或與 Kubernetes `aws-node` 服務帳戶相關聯的角色，且該帳戶係用於 Amazon VPC CNI plugin for Kubernetes。建議您將政策指派給與 Kubernetes 服務帳戶相關聯的角色。如要將政策指派給與 Kubernetes 服務帳戶相關聯的角色，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

   - IPv4

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
       --role-name AmazonEKSNodeRole
     ```

   - IPv6

     1. 複製下列文字並將它儲存至名為 *vpc-cni-ipv6-policy*.json 的檔案。

        ```
        {
            "Version": "2012-10-17",
        ```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. 建立 IAM 政策。

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

3. 將 IAM 政策連接至 IAM 角色。使用您的帳戶 ID 取代 *111122223333*。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSNodeRole
```

Amazon EKS Pod 執行 IAM 角色

需要 Amazon EKS Pod 執行角色才能Pods在 AWS Fargate 基礎設施上執行。

當您的叢集Pods在 AWS Fargate 基礎結構上建立時，在 Fargate 基礎架構上執行的元件必須代表您呼叫 AWS API。這樣他們就可以執行動作，例如從 Amazon ECR 提取容器映像或將日誌路由到其他 AWS 服務。Amazon EKS Pod 執行角色提供進行此類工作的 IAM 許可。

建立 Fargate 設定檔時，您必須使用設定檔為在 Fargate 基礎設施上執行的 Amazon EKS 元件指定 Pod 執行角色。此角色會新增至叢集的 Kubernetes 角色型存取控制 (RBAC)，以進行授權。這可讓在 Fargate 基礎設施上執行的 `kubelet` 向您的 Amazon EKS 叢集註冊，以便它可以在該叢集中作為節點出現。

> **Note**
>
> Fargate 設定檔的 IAM 角色必須與 Amazon EC2 節點群組不同。

> **Important**
>
> 在 Fargate Pod 中執行的容器，無法採用與 Pod 執行角色相關聯的 IAM 許可。若要授予 Fargate 中的容器存取其他 AWS 服務的Pod權限，您必須使用服務帳戶的 IAM 角色。

建立 Fargate 描述檔之前，您必須建立具有 `AmazonEKSFargatePodExecutionRolePolicy` 的 IAM 角色。

檢查是否有正確設定的現有 Pod 執行角色

您可使用以下程序，檢查您的帳戶是否已有正確設定的 Amazon EKS Pod 執行角色。為了避免出現混淆代理人安全問題，根據 `SourceArn` 的角色限制存取是非常重要的一環。您可以視需要修改執行角色，以納入對其他叢集上的 Fargate 描述檔的支援。

若要在 IAM 主控台中檢查 Amazon EKS Pod 執行角色

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 在「角色」頁面上，搜尋 AmazonE FargatePodExecutionRole ks 的角色清單。如果該角色不存在，請參閱「建立 Amazon EKS Pod 執行角色」以建立角色。如果該角色存在，請選取角色。
4. 在 AmazonEKS FargatePodExecutionRole 頁上，執行以下操作：
   a. 選擇許可。
   b. 確保將 AmazonEks FargatePodExecutionRolePolicy Amazon 託管政策附加到該角色上。
   c. 選擇 Trust relationships (信任關係)。
   d. 選擇 Edit trust policy (編輯信任政策)。

5. 在 Edit trust policy (編輯信任政策)頁面上，確認信任關係包含以下政策，並且在叢集上具有 Fargate 描述檔的行。若是如此，請選擇 Cancel (取消)。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

如果政策相符，但沒有指定叢集上的 Fargate 描述檔的行，您可以在 `ArnLike` 物件頂端新增以下行。使用叢集所在的 AWS 區域 取代 *region-code*，用帳戶 ID 取代 *111122223333*，再以您的叢集名稱取代 *my-cluster*。

```
"aws:SourceArn": "arn:aws:eks:region-code:111122223333:fargateprofile/my-cluster/
*",
```

如果政策不相符，請將完整前一個政策複製到表單中，然後選擇 Update policy (更新政策)。使用叢集所在的 AWS 區域 取代 *region-code*。如果您想在帳戶中使用相同的 AWS 區域 角色，請將*#### us-iso-east us-isob-east* 取代為. * 使用您的帳戶 ID 取代 *111122223333*，再以您的叢集名稱取代 *my-cluster*。若要在您的帳戶中的所有叢集使用相同角色，請使用 * 取代 *my-cluster*。

建立 Amazon EKS Pod 執行角色

如果您還沒有叢集的 Amazon EKS Pod 執行角色，可以使用 AWS Management Console 或 AWS CLI 來建立它。

AWS Management Console

使用 AWS Management Console建立 AWS FargatePod 執行角色

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 在 Roles (角色) 頁面上，選擇 Create role (建立角色)。
4. 在 Select trusted entity (選取信任的實體) 頁面上，執行以下作業：
   a. 在 Trusted entity type (信任的實體類型) 區段中，選擇 AWS service (服務)。
   b. 從其他 AWS 服務的使用案例下拉式清單中，選擇 EKS。
   c. 選擇 EKS – Fargate Pod。
   d. 選擇下一步。
5. 在 Add permissions (新增許可) 頁面上，選擇 Next (下一步)。
6. 在 Name, review, and create (命名、檢閱和建立) 頁面上，執行以下作業：
   a. 針對 Role name (角色名稱)，為您的角色輸入唯一名稱 (例如 **`AmazonEKSFargatePodExecutionRole`**)。
   b. 藉由連接標籤做為鍵值對，在新增標籤 (選用) 下將中繼資料新增至角色。如需有關在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的標記 IAM 資源。
   c. 選擇建立角色。
7. 在「角色」頁面上，搜尋 AmazonE FargatePodExecutionRole ks 的角色清單。選擇角色。
8. 在 AmazonEKS FargatePodExecutionRole 頁上，執行以下操作：
   a. 選擇 Trust relationships (信任關係)。
   b. 選擇 Edit trust policy (編輯信任政策)。
9. 在 Edit trust policy (編輯信任政策) 頁面上，執行下列動作：
   a. 請複製以下內容，並在 Edit trust policy (編輯信任政策) 表單貼上。將*#### us-iso-east us-isob-east* 替換為叢集所在 AWS 區域 的位置。如果您想在帳戶中使用相同的 AWS 區域 角色，請將*#### us-iso-east us-isob-east* 取代為. * 使用您的帳戶 ID 取代 *111122223333*，再以您的叢集名稱取代 *my-cluster*。若要在您的帳戶中的所有叢集使用相同角色，請使用 * 取代 *my-cluster*。

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. 選擇更新政策。

AWS CLI

若要建立 AWS FargatePod執行角色 AWS CLI

1. 複製以下內容並在名為 *pod-execution-role-trust-policy*.json 的檔案貼上。將*### # us-iso-east us-isob-east* 替換為叢集所在 AWS 區域 的位置。如果您想在帳戶中使用相同的 AWS 區域 角色，請將*#### us-iso-east us-isob-east* 取代為. * 使用您的帳戶 ID 取代 *111122223333*，再以您的叢集名稱取代 *my-cluster*。若要在您的帳戶中的所有叢集使用相同角色，請使用 * 取代 *my-cluster*。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. 建立 Pod 執行 IAM 角色。

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-policy.json"
```

3. 將必要的 Amazon EKS 受管 IAM 政策連接到角色。

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSFargatePodExecutionRolePolicy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Amazon EKS 連接器 IAM 角色

您可以連接Kubernetes叢集，以便在 AWS Management Console. 若要連接至 Kubernetes 叢集，請建立 IAM 角色。

檢查是否有現有的 EKS 連接器角色

您可使用以下程序，檢查帳戶是否已有 Amazon EKS 連接器角色。

在 IAM 主控台中檢查 **AmazonEKSConnectorAgentRole**

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側導覽窗格中，選擇 Roles (角色)。
3. 搜尋 AmazonEKSConnectorAgentRole 的角色清單。如果包含 AmazonEKSConnectorAgentRole 的角色不存在，請參閱 建立 Amazon EKS 連接器代理程式角色 以建立角色。如果包含 AmazonEKSConnectorAgentRole 的角色存在，請選取角色以檢視連接的政策。
4. 選擇許可。
5. 請確定已將 AmazonEks ConnectorAgentPolicy 管理政策附加至該角色。如果已連接政策，則您的 Amazon EKS 連接器角色應已設定妥當。

6. 選擇 Trust Relationships (信任關係)，然後選擇 Edit trust policy (編輯信任政策)。
7. 確認信任關係包含下列政策。如果信任關係符合下列政策，請選擇 Cancel (取消)。如果信任關係不符合，請將政策複製到 Edit trust policy (編輯信任政策) 視窗中，然後選擇 Update policy (更新政策)。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

建立 Amazon EKS 連接器代理程式角色

您可以使用 AWS Management Console 或 AWS CloudFormation 來建立連接器代理程式角色。

AWS CLI

1. 建立名為 `eks-connector-agent-trust-policy.json` 的檔案，其中包含用於 IAM 角色的下列 JSON。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
```

```
    ]
}
```

2. 建立名為 eks-connector-agent-policy.json 的檔案，其中包含用於 IAM 角色的下列 JSON。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SsmControlChannel",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateControlChannel"
            ],
            "Resource": "arn:aws:eks:*:*:cluster/*"
        },
        {
            "Sid": "ssmDataplaneOperations",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenDataChannel",
                "ssmmessages:OpenControlChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

3. 使用信任政策和您在先前清單項目中建立的政策，建立 Amazon EKS 連接器代理程式角色。

```
aws iam create-role \
     --role-name AmazonEKSConnectorAgentRole \
     --assume-role-policy-document file://eks-connector-agent-trust-policy.json
```

4. 將政策連接到 Amazon EKS 連接器代理程式角色。

```
aws iam put-role-policy \
     --role-name AmazonEKSConnectorAgentRole \
     --policy-name AmazonEKSConnectorAgentPolicy \
     --policy-document file://eks-connector-agent-policy.json
```

AWS CloudFormation

若要使用建立您的 Amazon EKS 連接器代理程式角色 AWS CloudFormation。

1. 將以下 AWS CloudFormation 範本儲存到本機系統上的文字檔案中。

> **Note**
>
> 此範本也會建立服務連結角色，否則會在呼叫 `registerCluster` API 時建立。如需詳細資訊，請參閱 使用角色將 Kubernetes 叢集連線至 Amazon EKS。

```
---
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Provisions necessary resources needed to register clusters in EKS'
Parameters: {}
Resources:
  EKSConnectorSLR:
    Type: AWS::IAM::ServiceLinkedRole
    Properties:
      AWSServiceName: eks-connector.amazonaws.com

  EKSConnectorAgentRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action: [ 'sts:AssumeRole' ]
            Principal:
              Service: 'ssm.amazonaws.com'

  EKSConnectorAgentPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: EKSConnectorAgentPolicy
      Roles:
        - {Ref: 'EKSConnectorAgentRole'}
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
```

```
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateControlChannel' ]
            Resource:
            - Fn::Sub: 'arn:${AWS::Partition}:eks:*:*:cluster/*'
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateDataChannel',
 'ssmmessages:OpenDataChannel', 'ssmmessages:OpenControlChannel' ]
            Resource: "*"
Outputs:
  EKSConnectorAgentRoleArn:
    Description: The agent role that EKS connector uses to communicate with AWS #
#.
    Value: !GetAtt EKSConnectorAgentRole.Arn
```

2. 開啟主 AWS CloudFormation 控台，網址為 https://console.aws.amazon.com/cloudformation。
3. 選擇 Create stack (建立堆疊) (使用新資源或現有資源)。
4. 針對 Specify template (指定範本)，選擇 Upload a template file (上傳範本檔案)，然後選擇 Choose file (選擇檔案)。
5. 選擇您剛建立的檔案，然後選擇 Next (下一步)。
6. 針對 Stack name (堆疊名稱)，輸入您角色的名稱 (例如 `eksConnectorAgentRole`)，然後選擇 Next (下一步)。
7. 在 Configure stack options (設定堆疊選項) 頁面，選擇 Next (下一步)。
8. 在 Review (檢閱) 頁面上，檢閱您的資訊，確認該堆疊可能會建立 IAM 資源，並選擇 Create stack (建立堆疊)。

AWS Amazon Elastic Kubernetes Service 的受管政策

受 AWS 管理的策略是由建立和管理的獨立策略 AWS。 AWS 受管理的策略旨在為許多常見使用案例提供權限，以便您可以開始將權限指派給使用者、群組和角色。

請記住， AWS 受管理的政策可能不會為您的特定使用案例授與最低權限權限，因為這些權限可供所有 AWS 客戶使用。我們建議您定義使用案例專屬的客戶管理政策，以便進一步減少許可。

您無法變更受 AWS 管理策略中定義的權限。如果 AWS 更新 AWS 受管理原則中定義的權限，則此更新會影響附加原則的所有主體識別 (使用者、群組和角色)。 AWS 當新的啟動或新 AWS 服務 的 API 操作可用於現有服務時，最有可能更新 AWS 受管理策略。

如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

AWS 受管政策 AmazonEKS_CNI_Policy

您可以將 AmazonEKS_CNI_Policy 連接到 IAM 實體。在建立 Amazon EC2 節點群組之前，必須將此政策連接到節點 IAM 角色，或連接到由 Amazon VPC CNI plugin for Kubernetes 專門使用的 IAM 角色。這樣一來，可以代表您執行動作。我們建議您將政策連接至僅供外掛程式使用的角色。如需更多詳細資訊，請參閱 使用 Amazon VPC CNI plugin for Kubernetes Amazon EKS 附加元件 及 設定為 Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

許可詳細資訊

此政策包含下列許可，這些許可能讓 Amazon EKS 完成下列任務：

- **ec2:*NetworkInterface**和 **ec2:*PrivateIpAddresses** — 允許 Amazon VPC CNI 外掛程式執行動作，例如佈建彈性網路界面和 IP 地址，以Pods便為在 Amazon EKS 中執行的應用程式提供聯網。
- **ec2**讀取動作 — 允許 Amazon VPC CNI 外掛程式執行諸如描述執行個體和子網路之類的動作，以查看 Amazon VPC 子網路中的可用 IP 地址數量。VPC CNI 可以使用每個子網路中的可用 IP 位址來挑選具有最多可用 IP 位址的子網路，以便在建立 elastic network interface 時使用。

若要檢視 JSON 政策文件的最新版本，請參閱《受管策略參考指南》中的《政策》。 AWS

AWS 管理策略：AmazonEKS ClusterPolicy

您可以將 AmazonEKSClusterPolicy 連接到 IAM 實體。在建立叢集之前，您必須擁有已附加此政策的叢集 IAM 角色。 KubernetesAmazon EKS 管理的叢集會代表您撥打其他 AWS 服務的電話。他們會執行此作業，以管理您搭配本服務使用的資源。

此政策包含下列許可，這些許可能讓 Amazon EKS 完成下列任務：

- **autoscaling**：讀取和更新 Auto Scaling 群組的組態。Amazon EKS 不會使用這些許可，但會保留在政策中以確保回溯相容性。
- **ec2**：使用與 Amazon EC2 節點相關聯的磁碟區和網路資源。這是必要項目，以便 Kubernetes 控制平面可以將執行個體加入叢集，並動態佈建和管理 Kubernetes 持久性磁碟區請求的 Amazon EBS 磁碟區。

- **elasticloadbalancing**：使用 Elastic Load Balancer，並將節點作為目標新增至 Elastic Load Balancers。這是必要項目，以便 Kubernetes 控制平台可以動態佈建 Kubernetes 服務要求的 Elastic Load Balancer。
- **iam** – 建立服務連結角色。這是必要項目，以便 Kubernetes 控制平台可以動態佈建 Kubernetes 服務要求的 Elastic Load Balancer。
- **kms** – 從 AWS KMS中讀取金鑰。這是 Kubernetes 控制平面支援在 `etcd` 中存放的 Kubernetes 秘密之秘密加密的必要條件。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南ClusterPolicy中的 AmazonEks。

AWS 管理策略：AmazonEKS FargatePodExecutionRolePolicy

您可以將 `AmazonEKSFargatePodExecutionRolePolicy` 連接到 IAM 實體。在建立 Fargate 設定檔之前，您必須先建立 Fargate Pod 執行角色並將此政策連接至該角色。如需詳細資訊，請參閱 建立 Fargate Pod 執行角色 及 AWS Fargate 設定檔。

此政策授予角色許可，這些許可提供對Pods在 Fargate 上執行 Amazon EKS 所需的其他 AWS 服務資源的存取權。

許可詳細資訊

此政策包含下列許可，這些許可能讓 Amazon EKS 完成下列任務：

- **ecr**：允許在 Fargate 上執行的 Pod 提取存放在 Amazon ECR 中的容器映像。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南 FargatePodExecutionRolePolicy中的 AmazonEks。

AWS 管理策略：AmazonEKS ForFargateServiceRolePolicy

您不得將 `AmazonEKSForFargateServiceRolePolicy` 連接到 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 Amazon EKS 代表您執行動作。如需詳細資訊，請參閱 AWSServiceRoleforAmazonEKSForFargate。

此政策授予 Amazon EKS 執行 Fargate 任務的必要許可。只有在您擁有 Fargate 節點時才會使用此政策。

許可詳細資訊

此政策包含下列允許 Amazon EKS 完成下列任務的許可。

- **ec2**：建立和刪除彈性網路介面，並描述彈性網路介面和資源。這是必要的，因此 Amazon EKS Fargate 服務可以設定 Fargate Pod 所需的 VPC 聯網。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南ForFargateServiceRolePolicy 中的 AmazonEks。

AWS 管理策略：AmazonEKS ServicePolicy

您可以將 AmazonEKSServicePolicy 連接到 IAM 實體。在 2020 年 4 月 16 日之前建立的叢集，需要您建立 IAM 角色並將此政策連接到該角色。在 2020 年 4 月 16 日或之後建立的叢集不需要您建立角色，也不需要您指派此政策。當您使用具有iam:CreateServiceLinkedRole權限的 IAM 主體建立叢集時，會自動為您建立 AWS ServiceRoleforAmazonEKS 服務連結角色。服務連結角色具有可連接至其中的 AWS 管理策略：AmazonEKS ServiceRolePolicy。

此政策允許 Amazon EKS 建立和管理操作 Amazon EKS 叢集所需的資源。

許可詳細資訊

此政策包含下列允許 Amazon EKS 完成下列任務的許可。

- **eks** – 在您開始更新之後，更新叢集的 Kubernetes 版本。Amazon EKS 不會使用這些許可，但會保留在政策中以確保回溯相容性。
- **ec2**：使用彈性網路介面及其他網路資源和標籤。Amazon EKS 要求這樣做，以設定可促進節點與 Kubernetes 控制平面之間的通訊的網路。
- **route53**：將 VPC 與託管區域建立關聯。Amazon EKS 必須執行這項操作，才能為您的 Kubernetes 叢集 API 伺服器啟用私有端點聯網。
- **logs**：日誌事件。這是必要的，以便 Amazon EKS 可以將Kubernetes控制平面日誌運送到 CloudWatch。
- **iam** – 建立服務連結角色。這是必需的，以便 Amazon EKScan 代表您建立 AWSServiceRoleForAmazonEKS 服務連結角色。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南ServicePolicy中的 AmazonEks。

AWS 管理策略：AmazonEKS ServiceRolePolicy

您不得將 AmazonEKSServiceRolePolicy 連接到 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 Amazon EKS 代表您執行動作。如需詳細資訊，請參閱 Amazon EKS 的服務連結角色許可。當您使用具有iam:CreateServiceLinkedRole權限的 IAM 主體建立叢集時，會自動為您建立 AWS ServiceRoleforAmazonEKS 服務連結角色，並將此政策附加至該叢集。

此原則允許服務連結角色代表您呼叫 AWS 服務。

許可詳細資訊

此政策包含下列允許 Amazon EKS 完成下列任務的許可。

- **ec2** – 建立並描述彈性網路介面和 Amazon EC2 執行個體、叢集安全群組以及建立叢集所需的 VPC。
- **iam**：列出連接至 IAM 角色的所有受管政策。這是必要項目，以便 Amazon EKS 可以列出和驗證建立叢集所需的所有受管政策和許可。
- 將 VPC 與託管區域建立關聯 – Amazon EKS 必須執行這項操作，才能為您的 Kubernetes 叢集 API 伺服器啟用私有端點聯網。
- 日誌事件 — 這是必要的，這樣 Amazon EKS 才能將Kubernetes控制平面日誌運送到 CloudWatch。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南ServiceRolePolicy中的 AmazonEks。

AWS 管理策略：亞馬遜 VPC ResourceController

您可將 AmazonEKSVPCResourceController 政策連接到 IAM 身分。如果您使用的是 Pods 的安全群組，您必須將此政策連接至 Amazon EKS 叢集 IAM 角色，以代表您執行動作。

此政策會授予叢集角色許可，以管理節點的彈性網路介面和 IP 地址。

許可詳細資訊

此政策包含下列許可，這些許可能讓 Amazon EKS 完成下列任務：

- **ec2** – 管理彈性網路介面和 IP 地址，以支援 Pod 安全群組和 Windows 節點。

若要檢視 JSON 政策文件的最新版本，請參閱《 AWS 受管策略參考指南》ResourceController中的《AmazonKsVPC》。

AWS 管理策略：AmazonEKS WorkerNodePolicy

您可以將 AmazonEKSWorkerNodePolicy 連接到 IAM 實體。您必須將此政策連接至您建立 Amazon EC2 節點時指定的 IAM 角色，從而可讓 Amazon EKS 代表您執行動作。如果您使用 eksctl 建立節點群組，則其會建立節點 IAM 角色並自動將此政策連接至角色。

此政策授予 Amazon EKS Amazon EC2 節點許可，以連接到 Amazon EKS 叢集。

許可詳細資訊

此政策包含下列許可，這些許可能讓 Amazon EKS 完成下列任務：

- **ec2**：讀取執行個體磁碟區和網路資訊。如此一來，Kubernetes 節點才能描述節點加入 Amazon EKS 叢集所需的 Amazon EC2 資源相關資訊，這是必要的。
- **eks**：選擇性地將叢集描述為節點引導的一部分。
- **eks-auth:AssumeRoleForPodIdentity** – 允許擷取節點上 EKS 工作負載的憑證。必須要有這項，EKS Pod 身分識別才能正常運作。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南WorkerNodePolicy中的 AmazonEks。

AWS 受管理策略： AWSServiceRoleForAmazonEKSNodegroup

您不得將 AWS ServiceRoleForAmazonEKSNodegroup 連接到 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 Amazon EKS 代表您執行動作。如需詳細資訊，請參閱 Amazon EKS 的服務連結角色許可。

此政策可授予 AWS ServiceRoleForAmazonEKSNodegroup 角色許可，允許它在您的帳戶中建立和管理 Amazon EC2 節點群組。

許可詳細資訊

此政策包含下列許可，這些許可能讓 Amazon EKS 完成下列任務：

- **ec2**：使用安全群組、標籤和啟動範本。這對於 Amazon EKS 受管節點群組來說是必要的，如此才能啟用遠端存取組態。此外，Amazon EKS 受管節點群組代表您建立啟動範本。這是為了設定支援每個受管節點群組的 Amazon EC2 Auto Scaling 群組。
- **iam**：建立服務連結角色並傳遞角色。Amazon EKS 受管節點群組必須執行這項操作，才能管理建立受管節點群組時所傳遞之角色的執行個體描述檔。此執行個體描述檔適用於作為受管節點群組一部

分的 Amazon EC2 執行個體。Amazon EKS 需要為其他服務 (例如 Amazon EC2 Auto Scaling 群組) 建立服務連結角色。這些許可會用於建立受管節點群組。

- **autoscaling**：使用安全的 Auto Scaling 群組。Amazon EKS 受管節點群組必須執行這項操作，才能管理支援每個受管節點群組的 Amazon EC2 Auto Scaling 群組。它也可用來支援功能，例如在節點群組更新期間終止或回收節點時移出 Pods。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管理策略參考指南AWSServiceRoleForAmazonEKSNodegroup中的。

AWS 管理政策:亞馬遜 DriverPolicy

AmazonEBSCSIDriverPolicy 政策允許 Amazon EBS 容器儲存介面 (CSI) 驅動程式代表您建立、修改、連接、分離和刪除磁碟區。其還授予 EBS CSI 驅動程式建立和刪除快照以及列出您的執行個體、磁碟區和快照的許可。

若要檢視 JSON 政策文件的最新版本，請參閱《 AWS 管理策略參考指南》DriverServiceRolePolicy中的 AmazonBSCSI。

AWS 管理策略:亞馬遜 DriverPolicy

AmazonEFSCSIDriverPolicy 政策可讓 Amazon EFS 容器儲存介面 (CSI) 代表您建立和刪除存取點。它還授予 Amazon EFS CSI 驅動程式許可，列出您的存取點檔案系統、掛載目標，以及 Amazon EC2 可用區域。

若要檢視 JSON 政策文件的最新版本，請參閱《 AWS 管理策略參考指南》DriverServiceRolePolicy中的《AmazonEFSSI》。

AWS 管理策略：AmazonEKS LocalOutpostClusterPolicy

您可將此政策連接至 IAM 實體。建立本機叢集之前，您必須將此原則附加至叢集角色。KubernetesAmazon EKS 管理的叢集會代表您撥打其他 AWS 服務的電話。他們會執行此作業，以管理您搭配本服務使用的資源。

AmazonEKSLocalOutpostClusterPolicy 包含以下許可：

- **ec2**：Amazon EC2 執行個體作為控制平面執行個體成功加入叢集所需的許可。
- **ssm**：允許 Amazon EC2 Systems Manager 連線至控制平面執行個體，Amazon EKS 會使用該連線進行通訊並管理帳戶中的本機叢集。
- **logs**— 允許執行個體將日誌推送到 Amazon CloudWatch。

- **secretsmanager**— 允許執行個體從中安全地取得和刪除控制平面執行個體的啟動程序資料 AWS Secrets Manager。
- **ecr**：允許在控制平面執行個體上執行的 Pods 和容器提取存放在 Amazon Elastic Container Registry 中的容器映像。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南LocalOutpostClusterPolicy中的AmazonEks。

AWS 管理策略：AmazonEKS LocalOutpostServiceRolePolicy

您無法將此政策連接至 IAM 實體。如果您使用具有 `iam:CreateServiceLinkedRole` 許可的 IAM 主體建立叢集，Amazon EKS 會為您自動建立 `AWSServiceRoleforAmazonEKSLocalOutpost` 服務連結角色，並將此政策與其連接。此原則允許服務連結角色代表您為本機叢集呼叫 AWS 服務。

`AmazonEKSLocalOutpostServiceRolePolicy` 包含以下許可：

- **ec2**：允許 Amazon EKS 使用安全功能、網路和其他資源，成功啟動並管理帳戶中的控制平面執行個體。
- **ssm**：允許 Amazon EC2 Systems Manager 連線至控制平面執行個體，Amazon EKS 會使用該連線進行通訊並管理帳戶中的本機叢集。
- **iam**：允許 Amazon EKS 管理與控制平面執行個體相關聯的執行個體設定檔。
- **secretsmanager**— 允許 Amazon EKS 將控制平面執行個體的啟動程序資料放入其中，以 AWS Secrets Manager 便在執行個體啟動期間安全地參考。
- **outposts**：允許 Amazon EKS 從您的帳戶取得 Outpost 資訊，以在 Outpost 中成功啟動本機叢集。

若要檢視 JSON 政策文件的最新版本，請參閱 AWS 受管政策參考指南LocalOutpostServiceRolePolicy中的 AmazonEks。

Amazon EKS 更新 AWS 受管政策

檢視 Amazon EKS AWS 受管政策更新的詳細資訊，因為此服務開始追蹤這些變更。如需有關此頁面變更的自動提醒，請訂閱 Amazon EKS 文件歷程記錄頁面上的 RSS 摘要。

變更	描述	日期
AmazonEKS_CNI_Policy — 更新至現有政策	Amazon EKS 添加了新的`ec2:DescribeSubnets` 許可，以允許Amazon VPC CNI plugin for Kubernetes查看您的 Amazon VPC 子網絡中的免費 IP 地址數量。 VPC CNI 可以使用每個子網路中的可用 IP 位址來挑選具有最多可用 IP 位址的子網路，以便在建立 elastic network interface 時使用。	2024年3月4日
AmazonEKS WorkerNodePolicy-更新到現有策略	Amazon EKS 新增了新的許可來允許 EKS Pod 身分識別。 Amazon EKS Pod 身分識別代理程式使用節點角色。	2023 年 11 月 26 日
介紹了亞馬遜食品安DriverPolicy全指數。	AWS 介紹了`AmazonEFSCSIDriverPolicy` .	2023 年 7 月 26 日
添加了權限到 AmazonEKS。ClusterPolicy	在建立負載平衡器時，新增允許 Amazon EKS 在子網路自動探索期間取得 AZ 詳細資訊的 `ec2:DescribeAvailabilityZones` 許可。	2023 年 2 月 7 日
更新了亞馬遜 BSCSI 中的政策條件。DriverPolicy	已移除在 `StringLike` 金鑰欄位中使用萬用字元的無效政策條件。也已將新條件 `ec2:ResourceTag/kubernetes.io/created-for/pvc/name: "*"` 新增至 `ec2:DeleteVolume` ，允許 EBS CSI 驅動程式刪除由樹狀內外掛程式建立的磁碟區。	2022 年 11 月 17 日
添加了權限到 AmazonEKS。LocalOutpostServiceRolePolicy	已新增 `ec2:DescribeVPCAttribute` 、`ec2:GetConsoleOutput` 和 `ec2:DescribeSecret` ，以	2022 年 10 月 24 日

變更	描述	日期
	允許更好的先決條件驗證和受管生命週期控制。還將 ec2:DescribePlacementGroups 和 "arn:aws:ec2:*:*:placement-group/*" 新增至 ec2:RunInstances ，以支援 Outposts 上的控制平面 Amazon EC2 執行個體的置放控制。	
在 Amazon 更新亞馬遜彈性容器註冊表許可。LocalOutpostClusterPolicy	已將動作 ecr:GetDownloadUrlForLayer 從所有資源區段移至限定範圍區段。已新增資源 arn:aws:ecr:*:*:repository/eks/* 。已移除資源 arn:aws:ecr:*:*:repository/eks/eks-certificates-controller-public 。新增的 arn:aws:ecr:*:*:repository/eks/* 資源涵蓋此資源。	2022 年 10 月 20 日
添加了權限到 AmazonEKS。LocalOutpostClusterPolicy	已新增 arn:aws:ecr:*:*:repository/kubelet-config-updater Amazon Elastic Container Registry 儲存庫，以便叢集控制平面執行個體可以更新部分 kubelet 引數。	2022 年 8 月 31 日
介紹 AmazonEKS。LocalOutpostClusterPolicy	AWS 介紹了AmazonEKSLocalOutpostClusterPolicy .	2022 年 8 月 24 日
介紹 AmazonEKS。LocalOutpostServiceRolePolicy	AWS 介紹了AmazonEKSLocalOutpostServiceRolePolicy .	2022 年 8 月 23 日
介紹亞馬遜 BSCSI。DriverPolicy	AWS 介紹了AmazonEBSCSIDriverPolicy .	2022 年 4 月 4 日

變更	描述	日期
添加了權限到 AmazonEKS。WorkerNodePolicy	已新增 `ec2:DescribeInstanceTypes` 以啟用可自動發現發現執行個體層級屬性的 Amazon EKS 最佳化 AMI。	2022 年 3 月 21 日
已將權限新增至 AWSServiceRoleForAmazonEKSNodegroup.	已新增 `autoscaling:EnableMetricsCollection` 許可以允許 Amazon EKS 啟用指標收集。	2021 年 12 月 13 日
添加了權限到 AmazonEKS。ClusterPolicy	已新增 `ec2:DescribeAccountAttributes` 、`ec2:DescribeAddresses` 以及 `ec2:DescribeInternetGateways` 許可，以允許 Amazon EKS 為 Network Load Balancer 建立服務連結角色。	2021 年 6 月 17 日
Amazon EKS 已開始追蹤變更。	Amazon EKS 開始追蹤其 AWS 受管政策的變更。	2021 年 6 月 17 日

疑難排解 IAM

本主題涵蓋一些搭配 Amazon EKS 使用 IAM 時可能遇到的常見錯誤，並提供解決方法。

AccessDeniedException

如果您在呼叫 AWS API 作業`AccessDeniedException`時收到，則您使用的 IAM 主體登入資料沒有進行該呼叫的必要權限。

```
An error occurred (AccessDeniedException) when calling the DescribeCluster operation:
User: arn:aws:iam::111122223333:user/user_name is not authorized to perform:
eks:DescribeCluster on resource: arn:aws:eks:region:111122223333:cluster/my-cluster
```

在先前的範例訊息中，使用者不具備呼叫 Amazon EKS `DescribeCluster` API 操作的許可。若要將 Amazon EKS 管理員許可提供給 IAM 主體，請參閱 Amazon EKS 身分型政策範例。

如需關於 IAM 的一般資訊，請參閱《IAM 使用者指南》的使用政策控制存取。

您無法在 Compute (運算) 標籤上看見 Nodes (節點),或是在 Resources (資源) 標籤上看見任何內容，並收到 AWS Management Console錯誤。

您可能會看到內容為 `Your current user or role does not have access to Kubernetes objects on this EKS cluster` 的主控台錯誤訊息。請確定與您搭配使用的 IAM 主體使用者 AWS Management Console 具有必要的權限。如需詳細資訊，請參閱 所需的許可。

aws-auth **`ConfigMap`** 不會授予對叢集的存取權限

AWS IAM Authenticator 不允許在 `ConfigMap` 中使用角色 ARN 的路徑。因此，在您指定 `rolearn` 之前，請移除該路徑。例如，請將 `arn:aws:iam::111122223333:role/team/developers/eks-admin` 變更為 `arn:aws:iam::111122223333:role/eks-admin`。

我沒有授權執行 iam：PassRole

如果您收到錯誤，告知您無權執行 `iam:PassRole` 動作，您的政策必須更新，允許您將角色傳遞給 Amazon EKS。

有些 AWS 服務 允許您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

當名為 `marymajor` 的 IAM 使用者嘗試使用主控台在 Amazon EKS 中執行動作時，發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 `iam:PassRole` 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想要允許我 AWS 帳戶以外的人員存取我的 Amazon EKS 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

若要進一步了解，請參閱以下內容：

- 若要了解 Amazon EKS 是否支援這些功能，請參閱 Amazon EKS 如何搭配 IAM 運作。
- 若要了解如何提供您所擁有資源 AWS 帳戶 的存取權，請參閱《IAM 使用者指南》中的另一個您擁有 AWS 帳戶 的 IAM 使用者提供存取權限。
- 若要了解如何將資源存取權提供給第三方 AWS 帳戶，請參閱 IAM 使用者指南中的提供第三方 AWS 帳戶 擁有的存取權。
- 若要了解如何透過聯合身分提供存取權，請參閱 IAM 使用者指南中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 若要了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱 IAM 使用者指南中的 IAM 角色與資源型政策的差異。

Pod 容器會接收到下列錯誤：**An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region**

如果您的應用程式明確向全域端點 (`https://sts.amazonaws`) 發出要求，且您的Kubernetes服務帳戶設定為使用區 AWS STS 域端點，則您的容器會收到此錯誤。您可以使用下列其中一種選項來解決此問題：

- 更新您的應用程式程式碼，以移除對 AWS STS 全域端點的明確呼叫。
- 更新應用程式的程式碼以明確呼叫區域端點，例如 `https://sts.us-west-2.amazonaws.com`。應用程式應具有內建備援，以便在該 AWS 區域的服務發生故障時選擇不同的 AWS 區域 。如需詳細資訊，請參閱《IAM 使用者指南》中的管理 AWS 區域中的 AWS STS。
- 將服務帳戶設定為使用全域端點。`1.22` 版以前的所有版本依預設使用全域端點，但版本 `1.22` 和更新版本的叢集依預設使用區域端點。如需更多詳細資訊，請參閱 設定服務帳戶的 AWS Security Token Service 端點。

預設 Amazon EKS 所建立的 Kubernetes 角色及使用者

當您建立 Kubernetes 叢集時，會在該叢集上建立數個預設 Kubernetes 身分，以便 Kubernetes 能正常運作。Amazon EKS 會為其每個預設元件建立 Kubernetes 身分。身分為叢集元件提供以 Kubernetes 角色為基礎的授權控制 (RBAC)。如需詳細資訊，請參閱 Kubernetes 文件中的使用 RBAC 授權。

當您將選用的附加元件安裝到叢集時，可能會將其他 Kubernetes 身分新增至叢集。如需有關本主題未討論之身分的詳細資訊，請參閱附加元件的文件。

您可以使用 AWS Management Console 或 `kubectl` 命令列工具檢視叢集上 Amazon EKS 所建立之 Kubernetes 身分的清單。所有使用者身分都會出現在透過 Amazon 提供給您的kube稽核日誌中 CloudWatch。

AWS Management Console

先決條件

您使用的 IAM 實體必須具有 所需的許可 中所述的許可。

若要檢視 Amazon EKS 建立的身分，請使用 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在 Clusters (叢集) 清單中，選擇包含您要檢視之身分的叢集。
3. 選擇 Resources (資源) 標籤。
4. 在 Resource types (資源類型) 下選擇 Authorization (授權)。
5. 選擇ClusterRoles、ClusterRoleBindings、、角色或RoleBindings。所有前面帶有 eks 的資源皆是由 Amazon EKS 所建立。其他由 Amazon EKS 建立的身分資源包含：
   - ClusterRole和命ClusterRoleBinding名的 AWS 節點。aws-node 資源支援 Amazon VPC CNI plugin for Kubernetes，Amazon EKS 會在所有叢集上安裝它。
   - 一個ClusterRole具名vpc-resource-controller-role和一個ClusterRoleBinding命名vpc-resource-controller-rolebinding。這些資源支援 Amazon VPC 資源控制器，Amazon EKS 會在所有叢集上安裝該控制器。

   除了您在主控台中看到的資源之外，您的叢集上還存在下列特殊使用者身分，但在叢集的組態中不會顯示出來：

   - **`eks:cluster-bootstrap`**：用於叢集引導期間的 `kubectl` 操作。
   - **`eks:support-engineer`**：用於叢集管理操作。
6. 選擇特定資源以檢視其相關詳細資訊。依預設，您可以在 Structured view (結構化檢視) 中檢視資訊。您可以在詳細資訊頁面的右上角，選擇 Raw view (原始檢視) 以查看資源的所有資訊。

Kubectl

先決條件

您使用 (IAM) 或AWS Identity and Access Management OpenID Connect (OIDC)) 列出叢集上Kubernetes資源的實體，必須經過 IAM 或您的OIDC身分提供者驗證。必須授予實體在叢集上使用 Role、ClusterRole、RoleBinding 的 Kubernetes get 和 list 動詞以及 ClusterRoleBinding 資源的許可，以便與實體互相搭配使用。如需有關授予 IAM 實體叢集存取權的詳細資訊，請參閱the section called “授與庫伯內特 API 的存取權”。如需有關授予經由您的 OIDC 提供者驗證之實體叢集存取權的詳細資訊，請參閱從OpenID Connect身分識別提供者驗證叢集的使用者。

使用 **kubectl** 檢視 Amazon EKS 所建立的身分

執行您要查看的資源類型的命令。所有前面帶有 eks 的傳回資源皆是由 Amazon EKS 所建立。除了命令輸出中傳回的資源之外，您的叢集上還存在下列特殊使用者身分，但在叢集的組態中不會顯示出來：

- **eks:cluster-bootstrap**：用於叢集引導期間的 kubectl 操作。
- **eks:support-engineer**：用於叢集管理操作。

ClusterRoles— 範圍限定於您的叢集，因此授予角色的任何權限ClusterRoles都會套用至叢集上任何Kubernetes命名空間中的資源。

下列命令會傳回叢集上由 Amazon EKS 建立的所有 Kubernetes ClusterRoles。

```
kubectl get clusterroles | grep eks
```

除了前面帶有 的輸出中傳回的 ClusterRoles 之外，還存在以下 ClusterRoles。

- **aws-node**：此 ClusterRole 支援 Amazon VPC CNI plugin for Kubernetes，Amazon EKS 會在所有叢集上安裝它。
- **vpc-resource-controller-role**：此 ClusterRole 支援 Amazon VPC 資源控制器，Amazon EKS 會在所有叢集上安裝該控制器。

若要查看 ClusterRole 的規格，請將以下命令中的 *ek: k8s-metrics* 取代為先前命令輸出中傳回的 ClusterRole。下列範例會傳回 *eks:k8s-metrics* ClusterRole 的規格。

```
kubectl describe clusterrole eks:k8s-metrics
```

範例輸出如下。

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names  Verbs
  ---------         -----------------  --------------  -----
                    [/metrics]         []              [get]
  endpoints         []                 []              [list]
  nodes             []                 []              [list]
  pods              []                 []              [list]
  deployments.apps  []                 []              [list]
```

ClusterRoleBindings— 範圍ClusterRoleBindings為您的叢集。

下列命令會傳回叢集上由 Amazon EKS 建立的所有 Kubernetes ClusterRoleBindings。

```
kubectl get clusterrolebindings | grep eks
```

除了輸出中傳回的 ClusterRoleBindings 之外，還存在以下 ClusterRoleBindings。

- **aws-node**：此 ClusterRoleBinding 支援 Amazon VPC CNI plugin for Kubernetes，Amazon EKS 會在所有叢集上安裝它。
- **vpc-resource-controller-rolebinding**：此 ClusterRoleBinding 支援 Amazon VPC 資源控制器，Amazon EKS 會在所有叢集上安裝該控制器。

若要查看 ClusterRoleBinding 的規格，請將以下命令中的 *ek: k8s-metrics* 取代為先前命令輸出中傳回的 ClusterRoleBinding。下列範例會傳回 *eks:k8s-metrics* ClusterRoleBinding 的規格。

```
kubectl describe clusterrolebinding eks:k8s-metrics
```

範例輸出如下。

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  eks:k8s-metrics
Subjects:
```

```
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

角色：Roles 設定範圍為 Kubernetes 命名空間內。所有 Amazon EKS 所建立的 Roles 設定範圍皆為 kube-system 命名空間內。

下列命令會傳回叢集上由 Amazon EKS 建立的所有 Kubernetes Roles。

```
kubectl get roles -n kube-system | grep eks
```

若要查看 Role 的規格，請將以下命令中的 *ek: k8s-metrics* 取代為先前命令輸出中傳回之 Role 的名稱。下列範例會傳回 *eks:k8s-metrics* Role 的規格。

```
kubectl describe role eks:k8s-metrics -n kube-system
```

範例輸出如下。

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names              Verbs
  ---------         -----------------  --------------              -----
  daemonsets.apps   []                 [aws-node]                  [get]
  deployments.apps  []                 [vpc-resource-controller]  [get]
```

RoleBindings— 範圍RoleBindings為命名空間Kubernetes。所有 Amazon EKS 所建立的 RoleBindings 設定範圍皆為 kube-system 命名空間內。

下列命令會傳回叢集上由 Amazon EKS 建立的所有 Kubernetes RoleBindings。

```
kubectl get rolebindings -n kube-system | grep eks
```

若要查看 RoleBinding 的規格，請將以下命令中的 *ek: k8s-metrics* 取代為先前命令輸出中傳回的 RoleBinding。下列範例會傳回 *eks:k8s-metrics* RoleBinding 的規格。

```
kubectl describe rolebinding eks:k8s-metrics -n kube-system
```

範例輸出如下。

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  Role
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Amazon Elastic Kubernetes Service 的合規驗證

若要瞭解 AWS 服務 是否屬於特定規範遵循方案的範圍內，請參閱AWS 服務 遵循規範計劃方案中的，並選擇您感興趣的合規方案。如需一般資訊，請參閱AWS 規範計劃AWS。

您可以使用下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱下載中的報告中的 AWS Artifact。

您在使用時的合規責任取決 AWS 服務 於您資料的敏感性、公司的合規目標以及適用的法律和法規。AWS 提供下列資源以協助遵循法規：

- 安全性與合規性快速入門指南 — 這些部署指南討論架構考量，並提供部署以安全性和合規性 AWS 為重點的基準環境的步驟。
- 在 Amazon Web Services 上架構 HIPAA 安全性與合規性 — 本白皮書說明公司如何使用建立符合 HIPAA 資格的應 AWS 用程式。

> **Note**
>
> 並非所有人 AWS 服務 都符合 HIPAA 資格。如需詳細資訊，請參閱 HIPAA 資格服務參照。

- AWS 合規資源AWS — 此工作簿和指南集合可能適用於您的產業和所在地。
- AWS 客戶合規指南 — 透過合規的角度瞭解共同的責任模式。這份指南總結了在多個架構 (包括美國國家標準技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準化組織 (ISO)) 中，保 AWS 服務 護指引並對應至安全控制的最佳實務。
- 使用AWS Config 開發人員指南中的規則評估資源 — 此 AWS Config 服務會評估您的資源組態符合內部實務、產業準則和法規的程度。

- AWS Security Hub— 這 AWS 服務 提供了內部安全狀態的全面視圖 AWS。Security Hub 使用安全控制，可評估您的 AWS 資源並檢查您的法規遵循是否符合安全業界標準和最佳實務。如需支援的服務和控制清單，請參閱 Security Hub controls reference。
- Amazon GuardDuty — 透過監控環境中的 AWS 帳戶可疑和惡意活動，藉此 AWS 服務 偵測您的工作負載、容器和資料的潛在威脅。 GuardDuty 可協助您滿足特定合規性架構所要求的入侵偵測需求，例如 PCI DSS 等各種合規性需求。
- AWS Audit Manager— 這 AWS 服務 有助於您持續稽核您的 AWS 使用情況，以簡化您管理風險的方式，以及遵守法規和業界標準的方式。

Amazon EKS 的恢復能力

AWS 全球基礎架構是以 AWS 區域 與可用區域為中心建置的。AWS 區域 提供多個分開且隔離的實際可用區域，並以具備低延遲、高輸送量和高度備援特性的聯網相互連結。透過可用區域，您所設計與操作的應用程式和資料庫，就能夠在可用區域之間自動容錯移轉，而不會發生中斷。可用區域的可用性、容錯能力和擴充能力，均較單一或多個資料中心的傳統基礎設施還高。

Amazon EKS 可在多個 AWS 可用區域內執行和擴展 Kubernetes 控制平面，以確保高可用性。Amazon EKS 會根據負載來自動擴展控制平面執行個體，且會偵測及取代狀態不佳的控制平面執行個體，還會對控制平面進行自動化程式修補。啟動版本更新後，Amazon EKS 會更新控制平面，並在更新期間維持控制平面的高可用性。

此控制平面包含至少兩個 API 伺服器執行個體和三個 `etcd` 執行個體，這些執行個體在 AWS 區域 內的三個可用區域中執行。Amazon EKS：

- 主動監控控制平面執行個體上的負載，並自動進行擴展以確保高效能。
- 可以自動偵測並取代有問題的控制平面執行個體，而且視需要在 AWS 區域 內的可用區域中重新啟動。
- 可以利用 AWS 區域 的架構維持高可用性。因此，Amazon EKS 能夠提供 API 伺服器端點可用性的 SLA。

如需 AWS 區域 與可用區域的詳細資訊，請參閱 AWS全球基礎設施。

Amazon EKS 中的基礎設施安全

作為一項受管服務，Amazon Elastic Kubernetes Service 受到全球網路安 AWS 全的保護。有關 AWS 安全服務以及如何 AWS 保護基礎結構的詳細資訊，請參閱AWS 雲端安全 若要使用基礎架構安全性的最佳做法來設計您的 AWS 環境，請參閱安全性支柱架構良 AWS 好的架構中的基礎結構保護。

您可以使用 AWS 已發佈的 API 呼叫透過網路存取 Amazon EKS。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

此外，請求必須使用存取索引鍵 ID 和與 IAM 主體相關聯的私密存取索引鍵來簽署。或者，您可以使用 AWS Security Token Service (AWS STS) 來產生暫時安全憑證來簽署請求。

建立 Amazon EKS 叢集時，為要使用的叢集指定 VPC 子網路。Amazon EKS 需要至少兩個處於可用區域的子網路。我們建議使用具有公有和私有子網的 VPC，以便 Kubernetes 在公有子網中建立公有負載平衡器，以平衡負載到私有子網中的節點上執行的 Pods 流量。

如需 VPC 考量的詳細資訊，請參閱 Amazon EKS VPC 與子網要求和注意事項。

如果您使用Amazon EKS 入門逐步解說中提供的 AWS CloudFormation 範本建立 VPC 和節點群組，則您的控制平面和節點安全群組會使用我們建議的設定進行設定。

如需安全群組考量的詳細資訊，請參閱 Amazon EKS 安全群組與考量。

建立新的叢集時，Amazon EKS 會為您用來與叢集通訊的受管 Kubernetes API 伺服器建立端點 (使用 Kubernetes 管理工具，例如 `kubectl`)。預設情況下，此 API 伺服器端點對網際網路公開，而 API 伺服器的存取是使用 AWS Identity and Access Management (IAM) 和原生Kubernetes角色型存取控制 (RBAC) 的組合來保護對 API 伺服器的存取。

您可啟用 Kubernetes API 伺服器的私有存取，讓節點和 API 伺服器間的所有通訊都不會離開 VPC。您可以限制可以從網際網路存取 API 伺服器的 IP 地址，或完全停用對 API 伺服器的網際網路存取。

如需修改叢集端點存取的詳細資訊，請參閱 修改叢集端點存取。

您可以搭配 Amazon VPC CNI 或第三方工具 (例如 Project Calico) 來實作 Kubernetes 網路政策。如需有關使用適用於網路政策的 Amazon VPC CNI 的詳細資訊，請參閱 為 Kubernetes 網路政策設定您的叢集。Project Calico 是第三方開放原始碼專案。如需詳細資訊，請參閱 Project Calico 文件。

Amazon EKS 中的組態與漏洞分析

安全性是設定和維護 Kubernetes 叢集和應用程式的重要考量因素。以下列出資源供您分析 EKS 叢集的安全性設定、檢查弱點的資源，以及與可為您執行該分析的 AWS 服務整合。

Amazon EKS 的互聯網安全中心（CIS）基準

網際網路安全中心 (CIS) Kubernetes 效能標竿提供 Amazon EKS 安全組態的指引。基準化分析：

- 適用於您負責 Kubernetes 元件的安全組態的 Amazon EC2 節點 (受管和自我管理)。
- 提供經社群核准的標準方式，以確保您在使用 Amazon EKS 時能安全地設定 Kubernetes 叢集和節點。
- 由四個部分組成：控制平面記錄設定、節點安全設定、政策和受管服務。
- 支援 Amazon EKS 目前可用的所有 Kubernetes 版本，並且可以使用 kube-bench 予以執行，這是一個標準開源工具，可用於在 Kubernetes 叢集上使用 CIS 基準檢查組態。

若要進一步了解，請參閱 CIS Amazon EKS 基準簡介。

Amazon EKS 平台版本

Amazon EKS 平台版本代表叢集控制平面的功能，包括啟用了哪些 Kubernetes API 伺服器旗標和目前的Kubernetes修補程式版本。新叢集是使用最新的平台版本部署。如需詳細資訊，請參閱 Amazon EKS 平台版本。

您可以將 Amazon EKS 叢集更新為較新的 Kubernetes 版本。由於 Amazon EKS 會提供新的 Kubernetes 版本，所以我們建議您主動更新您的叢集，以便使用最新的可用版本。如需 EKS 中 Kubernetes 版本的詳細資訊，請參閱 Amazon EKS Kubernetes 版本。

作業系統弱點清單

AL2023 漏洞清單

在 Amazon 安全中心追蹤 Amazon Linux 2023 的Linux安全或隱私權事件，或訂閱相關聯的 RSS 摘要。安全與隱私權事件包括影響問題的概觀和套件，並說明如何更新執行個體以修正問題。

Amazon 2 漏洞列表

在 Amazon 安全中心追蹤 Amazon Linux 2 的Linux安全或隱私權事件，或訂閱相關聯的 RSS 摘要。安全與隱私權事件包括影響問題的概觀和套件，並說明如何更新執行個體以修正問題。

使用亞馬遜檢測節點檢測

您可以使用 Amazon Inspector 檢查節點是否有意外的網路存取問題，以及 Amazon EC2 執行個體是否有漏洞。

使用 Amazon 進行叢集和節點偵 GuardDuty

Amazon GuardDuty 威脅偵測服務可協助保護您的帳戶、容器、工作負載和 AWS 環境中的資料。除其他功能外，還 GuardDuty提供下列兩項功能，可偵測 EKS 叢集的潛在威脅：EKS 防護和執行階段監控。

如需更多詳細資訊，請參閱 使用偵測威脅 Amazon GuardDuty。

Amazon EKS 的安全最佳實務

Amazon EKS 的安全最佳實務會在 Github 上進行維護：https://aws.github.io/aws-eks-best-practices/security/docs/

Pod 安全政策

Kubernetes Pod 安全政策許可控制器，會根據一套規則驗證建立和更新 Pod 的請求。根據預設，Amazon EKS 叢集隨附於一套完全寬鬆的安全政策，不受任何限制。如需詳細資訊，請參閱 Kubernetes 文件中的 Pod 安全政策。

> **Note**
>
> `PodSecurityPolicy` (PSP) 已於 Kubernetes 版本 `1.21` 中棄用並在 Kubernetes `1.25` 中移除。PSPs 正為 Pod 安全許可 (PSA) 所取代。PSA 是內建的許可控制器，會實作在 Pod 安全標準 (PSS) 中概述的安全控制項。PSA 和 PSS 都已進入測試版功能狀態，並且在 Amazon EKS 中預設啟用。為了處理 PSP 會在 `1.25` 中移除的問題，我們建議您在 Amazon EKS 中實作 PSS。如需詳細資訊，請參閱 AWS 部落格上的 Implementing Pod Security Standards in Amazon EKS (在 Amazon EKS 中實作 Pod 安全標準)。

Amazon EKS 預設 Pod 安全政策

使用 Kubernetes 版本 `1.13` 或更新版本的 Amazon EKS 叢集有一個名為 Pod 的預設 `eks.privileged` 安全政策。此政策針對系統可接受哪些類型的 Pod 不設限制，等同於在停用 Kubernetes 控制器的情況下執行 `PodSecurityPolicy`。

> **Note**
>
> 此政策的建立，是為了讓未啟用 PodSecurityPolicy 控制器的叢集維持回溯相容性。您可以為叢集、個別命名空間和服務帳戶建立較嚴格的政策，然後刪除預設政策，以啟用較嚴格的政策。

您可以使用下列命令來檢視預設政策。

```
kubectl get psp eks.privileged
```

範例輸出如下。

```
NAME             PRIV   CAPS   SELINUX    RUNASUSER   FSGROUP    SUPGROUP
 READONLYROOTFS   VOLUMES
eks.privileged   true   *      RunAsAny   RunAsAny    RunAsAny   RunAsAny   false
      *
```

如需詳細資訊，您可以使用下列命令來描述政策。

```
kubectl describe psp eks.privileged
```

範例輸出如下。

```
Name:  eks.privileged

Settings:
  Allow Privileged:                       true
  Allow Privilege Escalation:             0xc0004ce5f8
  Default Add Capabilities:               <none>
  Required Drop Capabilities:             <none>
  Allowed Capabilities:                   *
  Allowed Volume Types:                   *
  Allow Host Network:                     true
  Allow Host Ports:                       0-65535
  Allow Host PID:                         true
  Allow Host IPC:                         true
  Read Only Root Filesystem:              false
  SELinux Context Strategy: RunAsAny
    User:                                 <none>
    Role:                                 <none>
```

```
    Type:                                 <none>
    Level:                                <none>
  Run As User Strategy: RunAsAny
    Ranges:                               <none>
  FSGroup Strategy: RunAsAny
    Ranges:                               <none>
  Supplemental Groups Strategy: RunAsAny
    Ranges:                               <none>
```

您可以在 eks.privileged 中檢視 Pod 安裝或還原預設 Pod 安全政策 安全政策、叢集角色和叢集角色連結的完整 YAML 檔案。

刪除預設 Amazon EKS Pod 安全政策

在為 Pods 建立更多限制性政策後，您可以刪除預設的 Amazon EKS eks.privileged Pod 安全政策，以便啟用您的自訂政策。

> **Important**
>
> 如果您使用 CNI 外掛程式版本 1.7.0 或更新版本，並將自訂 Pod 安全政策指派給用於 Daemonset 部署的 aws-node Pods 的 aws-node Kubernetes 服務帳戶，則該政策必須在其 allowedCapabilities 區段中包含 NET_ADMIN，以及在政策的 spec 中包含 hostNetwork: true 和 privileged: true。

刪除預設 Pod 安全政策

1. 建立名為 *privileged-podsecuritypolicy.yaml* 的檔案，其中具有 安裝或還原預設 Pod 安全政策 範例檔案中的內容。
2. 使用下列命令來刪除 YAML。這會刪除預設的 Pod 安全政策、ClusterRole，以及與之相關聯的 ClusterRoleBinding。

```
kubectl delete -f privileged-podsecuritypolicy.yaml
```

安裝或還原預設 Pod 安全政策

如果您是從舊版 Kubernetes 升級，或已修改或刪除預設的 Amazon EKS eks.privileged Pod 安全政策，您可以透過下列步驟將其還原。

安裝或還原預設 Pod 安全政策

1. 建立稱為 *privileged-podsecuritypolicy.yaml* 的檔案，其中具有以下內容。

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: eks.privileged
  annotations:
    kubernetes.io/description: 'privileged allows full unrestricted access to
      Pod features, as if the PodSecurityPolicy controller was not enabled.'
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
spec:
  privileged: true
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  volumes:
  - '*'
  hostNetwork: true
  hostPorts:
  - min: 0
    max: 65535
  hostIPC: true
  hostPID: true
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
  readOnlyRootFilesystem: false

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:podsecuritypolicy:privileged
```

```
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
rules:
- apiGroups:
  - policy
  resourceNames:
  - eks.privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:podsecuritypolicy:authenticated
  annotations:
    kubernetes.io/description: 'Allow all authenticated users to create privileged
 Pods.'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:podsecuritypolicy:privileged
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: system:authenticated
```

2. 使用下列命令來套用 YAML。

```
kubectl apply -f privileged-podsecuritypolicy.yaml
```

Pod 安全政策 (PSP) 移除常見問答集

PodSecurityPolicy 已在 Kubernetes1.21 中被棄用，且已在 Kubernetes 1.25 中移除。如果您 PodSecurityPolicy 在叢集中使用，則必須先移轉至內建 Kubernetes Pod 安全性標準(PSS)或 policy-

as-code 解決方案，才能將叢集升級**1.25**至版本，以避免工作負載中斷。選擇任何常見問答集以了解詳細資訊。

什麼是 PSP？

PodSecurityPolicy是內建的許可控制器，可讓叢集系統管理員控制規格的安全性敏感層面Pod。如果 Pod 符合其 PSP 的要求，則 Pod 會照常得到叢集的許可。如果 Pod 不符合 PSP 要求，則 Pod 會被拒絕且無法執行。

PSP 移除是特定於 Amazon EKS，還是在上游 Kubernetes 中被移除？

這是 Kubernetes 專案中的上游變更，而不是在 Amazon EKS 中執行的變更。PSP 已在 Kubernetes `1.21` 中被棄用，並在 Kubernetes `1.25` 中移除。Kubernetes 社群發現 PSP 存在嚴重的可用性問題。其中包括意外授予比預期更廣泛的許可，以及難以檢查在特定情況下適用的 PSPs。如果沒有進行重大變更，就無法解決這些問題。這就是 Kubernetes 社群 決定移除 PSP 的主要原因。

如何檢查我是否在 Amazon EKS 叢集中使用 PSPs？

若要檢查您是否在叢集使用 PSPs，您可以執行下列命令：

```
kubectl get psp
```

若要查看叢集中 PSPs 正在影響的 Pods，請執行下列命令。此命令會輸出 Pod 名稱、命名空間和 PSPs：

```
kubectl get pod -A -o jsonpath='{range.items[?(@.metadata.annotations.kubernetes
\.io/psp)]}{.metadata.name}{"\t"}{.metadata.namespace}{"\t"}
{.metadata.annotations.kubernetes\.io/psp}{"\n"}'
```

如果我在 Amazon EKS 叢集中使用 PSPs，該怎麼辦？

將叢集升級到 `1.25` 之前，您必須將自己的 PSPs 遷移至下列其中一個替代選項：

- Kubernetes PSS.
- 來自Kubernetes環境的 P olicy-as-code 解決方案。

為了回應 PSP 棄用和從一開始就控制 Pod 安全性的持續需求，Kubernetes 社群建立了具有 (PSS) 和 Pod 安全許可 (PSA) 的內建解決方案。PSA Webhook 會實作 PSS 中定義的控制項。

您可以在 EKS 最佳實務指南中檢閱將 PSPs 遷移至內建 PSS 的最佳實務。我們也建議您檢閱有關在 Amazon EKS 中實作 Pod 安全標準的部落格。其他參考資料包括從 PodSecurityPolicy 內建 PodSecurity 許可控制器遷移，以及對應 PodSecurityPolicies 至網繭安全性標準。

P olicy-as-code 解決方案提供護欄以引導叢集使用者，並透過規定的自動化控制來防止不必要的行為。P olicy-as-code 解決方案通常會使用 Kubernetes 動態許可控制器，透過 Webhook 呼叫攔截 Kubernetes API 伺服器要求流程。P olicy-as-code 解決方案會根據以程式碼撰寫和儲存的原則來變更和驗證要求承載。

有幾種開源 policy-as-code 解決方案可用於Kubernetes. 若要檢閱移轉PSPs至 policy-as-code 解決方案的最佳做法，請參閱上的網繭安全性頁面的 P olicy-as-code 部分 GitHub。

我在叢集中看到名為 **eks.privileged** 的 PSP。這是什麼，我能做些什麼？

具有 Kubernetes 版本 `1.13` 或更高版本的 Amazon EKS 叢集具有名為 `eks.privileged` 的預設值 PSP。此政策在 `1.24` 和更早版本的叢集中建立。它不會在 `1.25` 和更新版本的叢集中使用。Amazon EKS 會自動將此 PSP 遷移到以 PSS 為基礎的強制執行。您不需要執行任何動作。

當我將叢集更新為版本 **1.25** 時，Amazon EKS 是否會對存在於現有叢集中的 PSPs 進行任何變更？

不會。除了 `eks.privileged` 之外 (這是由 Amazon EKS 建立的 PSP)，升級至 `1.25` 時不會對叢集中的其他 PSPs 執行任何變更。

如果我沒有遷移 PSP，Amazon EKS 是否會阻止叢集更新為版本 **1.25**？

不會。如果您尚未遷移 PSP，Amazon EKS 將不會阻止叢集更新至版本 `1.25`。

如果在將叢集更新為版本之前忘記PSPs將我的 policy-as-code 解決方案移轉到PSS/PSA或移轉至解決方案，該怎麼辦**1.25**？ 我可以在更新叢集後進行遷移嗎？

當包含 PSP 的叢集升級到 Kubernetes 版本 `1.25` 時，API 伺服器將無法識別 `1.25` 的 PSP 資源。這可能會導致 Pods 取得不正確的安全範圍。如需詳盡的隱含清單，請參閱從內建許可控制器移轉 PodSecurityPolicy 至內建 PodSecurity 許可控制器。

此變更如何影響 Windows 工作負載的 Pod 安全性？

我們預期不會對 Windows 工作負載產生任何特定影響。 PodSecurityContext 有一個在適用於視`windowsOptions`窗的 `PodSpec` `v1` API 中調用的字段Pods。這會使用 Kubernetes `1.25` 中的

PSS。如需有關針對 Windows 工作負載強制執行 PSS 的詳細資訊和最佳實務，請參閱 EKS 最佳實務指南和 Kubernetes 文件。

搭配使用 AWS Secrets Manager 秘密和 Kubernetes

若要顯示來自 Secrets Manager 的秘密與參數存放區的參數，以做為掛載於 Amazon EKS Pods 的檔案，您可以將 AWS Secrets and Configuration Provider (ASCP) 用於 Kubernetes Secrets Store CSI Driver。

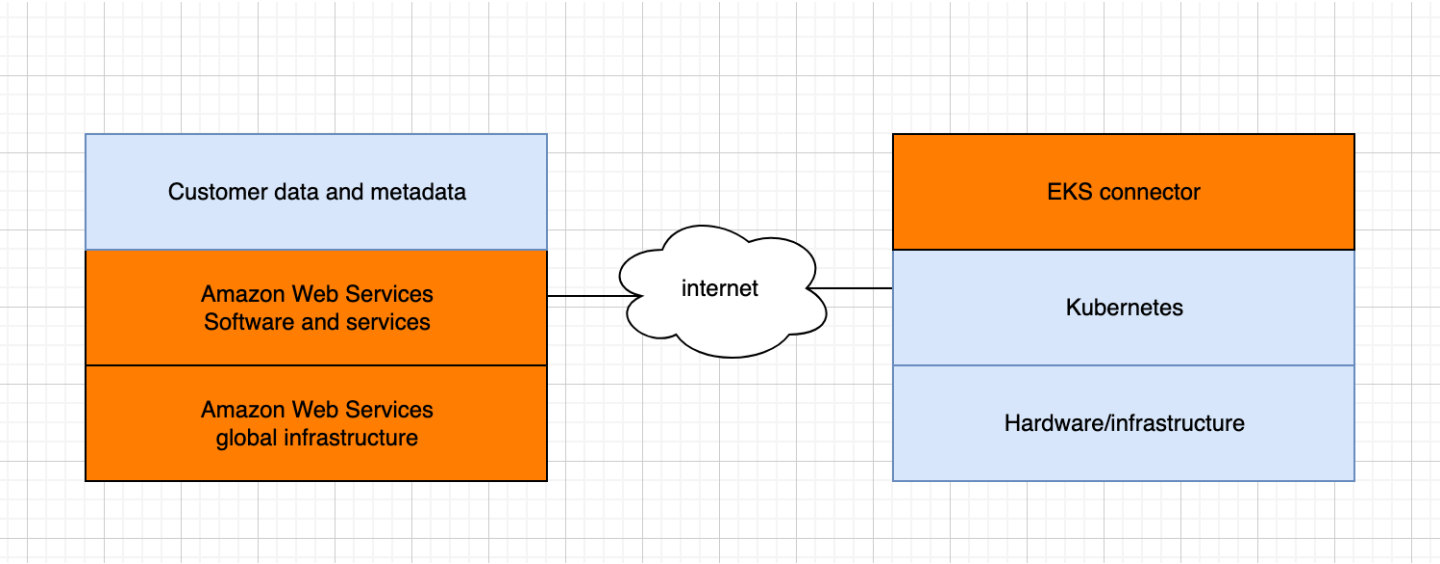
透過 ASCP，您可以在 Secrets Manager 中存放和管理您的秘密，然後透過在 Amazon EKS 上執行的工作負載擷取這些秘密。您可以使用 IAM 角色和政策，限制對叢集中特定 Kubernetes Pods 秘密的存取權。ASCP 會擷取 Pod 身分識別並交換 IAM 角色的身分識別。ASCP 會假設 Pod 的 IAM 角色，然後可以從已授權該角色的 Secrets Manager 擷取秘密。

如果您使用 Secrets Manager 自動輪換您的秘密，您也可以使用 Secrets Store CSI Driver 輪換調解器功能，以確保您從 Secrets Manager 擷取最新的秘密。

如需詳細資訊，請參閱《AWS Secrets Manager 使用者指南》中的使用 Amazon EKS 中的 Secrets Manager 秘密。

Amazon EKS 連接器考量事項

Amazon EKS 連接器是一個在您的 Kubernetes 叢集上執行的開放原始碼元件。此叢集可以位於 AWS 環境之外。這對安全責任產生了其他考量。可透過下圖說明此設定。橘色代表 AWS 責任，藍色則代表客戶責任：



本主題說明了如果連接的叢集位於 AWS 之外的責任模型內的差異。

AWS 責任

- 維護、建置和交付 Amazon EKS 連接器，該連接器是在客戶的 Kubernetes 叢集上執行的開放原始碼元件，並與 AWS 通訊。
- 維護連接的 Kubernetes 叢集和 AWS 服務之間的傳輸與應用程式層級通訊安全。

客戶責任

- Kubernetes 叢集的特定安全，尤其是沿著以下幾行：
  - Kubernetes 秘密必須得到適當的加密和保護。
  - 封鎖對 `eks-connector` 命名空間的存取。
- 設定角色型存取控制 (RBAC) 許可，以管理從 AWS 的 IAM 主體存取。如需說明，請參閱 授予 IAM 主體存取權以檢視叢集上的 Kubernetes 資源。
- 安裝和升級 Amazon EKS 連接器。
- 維護支援連接的 Kubernetes 叢集的硬體、軟體和基礎架構。
- 保護他們的 AWS 帳戶 (例如，透過保護您的根使用者憑證)。

檢視 Kubernetes 資源

您可以使用 AWS Management Console來檢視部署至叢集的 Kubernetes 資源。您無法使用 AWS CLI 或檢視Kubernetes資源`eksctl`。若要檢視使用命令列工具的 Kubernetes 資源，請使用 `kubectl`。

先決條件

若要檢視中 [運算] 索引標籤上的 [資源] 索引標籤和 [節點] 區段 AWS Management Console，您使用的 IAM 主體必須具有特定的 Kubernetes IAM 和許可。如需詳細資訊，請參閱 所需的許可。

若要檢視Kubernetes資源 AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 在 Clusters (叢集) 清單中，選取包含您要檢視的 Kubernetes 資源的叢集。
3. 選取 Resources (資源) 標籤。
4. 選取您要檢視資源的 Resource Type (資源類型) 群組，例如 Workloads (工作負載)。您可以看到該群組中的資源類型清單。
5. 選取資源類型，例如 Workloads (工作負載) 群組中的 Deployments (部署)。您可以看到資源類型的說明、一個可供獲取有關資源類型詳細資訊的 Kubernetes 文件連結，以及部署在叢集上的該類型資源列表。如果清單是空的，則不會將該類型的資源部署至您的叢集。
6. 請選取資源以檢視其詳細資訊。請試試看下列範例：

   - 選取 Workloads (工作負載) 群組，選取 Deployments (部署) 資源類型，然後選取 coredns 資源。當您選取資源時，依預設，您位於 Structured view (結構式檢視)。針對某些資源類型，您會在 Structured view (結構式檢視) 看到 Pods 區段。本區段列出由工作負載管理的 Pods。您可以選取任何列出的 Pod 來檢視關於 Pod 的資訊。並非所有資源類型都顯示於 Structured View (結構式檢視)。若選取資源頁面右上角的 Raw view (原始檢視)，您可以看到來自 Kubernetes API 對於資源的完整 JSON 回應。
   - 選取 Cluster (叢集) 群組，然後選取 Nodes (節點) 資源類型。您將看到叢集中所有節點的清單。節點可以是任何 Amazon EKS 節點類型。這個列表與您選取叢集的 Compute (運算) 標籤時，在 Nodes (節點)看到的列表為同一列表。從清單中選取節點資源。在 Structured view (結構式檢視) 中，您還會看到 Pods 區段。本區段顯示在節點上運行的所有 Pods。

所需的許可

若要檢視中 [運算] 索引標籤上的 [資源] 索引標籤和節點區段 AWS Management Console，您使用的 IAM 主體必須具有特定的最低 Kubernetes IAM 和許可。完成以下步驟，將所需的許可指派給您的 IAM 主體。

1. 請確保將 `eks:AccessKubernetesApi` 和檢視 Kubernetes 資源所需的其他必要 IAM 許可指派給您正在使用的 IAM 主體。如需如何編輯 IAM 主體許可的詳細資訊，請參閱《IAM 使用者指南》中的控制對主體的存取。如需如何編輯角色許可的詳細資訊，請參閱《IAM 使用者指南》中的變更角色許可 (主控台)。

   以下範例政策包含主體欲檢視帳戶中所有叢集的 Kubernetes 資源所需必要許可。將 *111122223333* 取代為您的 AWS 帳戶 ID。

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "eks:ListFargateProfiles",
                   "eks:DescribeNodegroup",
                   "eks:ListNodegroups",
                   "eks:ListUpdates",
                   "eks:AccessKubernetesApi",
                   "eks:ListAddons",
                   "eks:DescribeCluster",
                   "eks:DescribeAddonVersions",
                   "eks:ListClusters",
                   "eks:ListIdentityProviderConfigs",
                   "iam:ListRoles"
               ],
               "Resource": "*"
           },
           {
               "Effect": "Allow",
               "Action": "ssm:GetParameter",
               "Resource": "arn:aws:ssm:*:111122223333:parameter/*"
           }
       ]
   }
   ```

若要檢視已連線叢集中的節點，Amazon EKS connector IAM 角色應能模擬叢集中的主體。這可讓 Amazon EKS 連接器 將主體對應至 Kubernetes 使用者。

2. 建立繫結至 Kubernetes `role` 或 `clusterrole` 且具有檢視 Kubernetes 資源所需必要許可的 Kubernetes `rolebinding` 或 `clusterrolebinding`。若要進一步了解 Kubernetes 角色和角色繫結，請參閱 Kubernetes 文件中的 Using RBAC Authorization (使用 RBAC 授權)。您可以將以下清單檔案之一套用至建立具備 Kubernetes 所需必要許可的 `role` 和 `rolebinding` 或 `clusterrole` 和 `clusterrolebinding` 的叢集：

   檢視所有命名空間中的 Kubernetes 資源

   檔案中的群組名稱為 `eks-console-dashboard-full-access-group`。使用下列命令套用清單檔案至您的叢集：

   ```
   kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
   ```

   檢視特定命名空間中的 Kubernetes 資源

   此檔案中的命名空間為 `default`。檔案中的群組名稱為 `eks-console-dashboard-restricted-access-group`。使用下列命令套用清單檔案至您的叢集：

   ```
   kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
   ```

   若您需要變更 Kubernetes 群組名稱、命名空間、許可或檔案中的任何其他組態，請先下載該檔案並進行編輯，然後再將其套用至叢集：

   1. 執行下列其中一個命令，下載檔案：

      ```
      curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
      ```

      ```
      curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
      ```

   2. 視必要編輯檔案。

   3. 使用下列其中一個命令套用清單檔案至您的叢集：

```
kubectl apply -f eks-console-full-access.yaml
```

```
kubectl apply -f eks-console-restricted-access.yaml
```

3. 將 IAM 主體對應至 aws-auth ConfigMap 中的 Kubernetes 使用者或群組。您可以使用例如 eksctl 的工具來更新 ConfigMap，或者您可以透過編輯來手動更新它。

> **Important**
>
> 我們建議您使用 eksctl 或其他工具來編輯 ConfigMap。如需有關您可使用的其他工具的資訊，請參閱《Amazon EKS 最佳實務指南》中的使用工具對 aws-authConfigMap 進行變更。格式錯誤的 aws-auth ConfigMap可能會導致您失去叢集存取權。

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

1. 在 ConfigMap 檢視目前的映射項目。使用您叢集的名稱取代 *my-cluster*。*region-code*以叢集所 AWS 區域 在的位置取代。

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

範例輸出如下。

```
ARN
                    USERNAME                                GROUPS
              ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA     system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. 為角色新增映射項目。此範例假設您在第一個步驟中將 IAM 許可連接至名為 *my-console-viewer-role* 的角色。使用您的帳戶 ID 取代 *111122223333*。

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws:iam::111122223333:role/my-console-viewer-role \
    --group eks-console-dashboard-full-access-group \
    --no-duplicate-arns
```

> **Important**
>
> 角色 ARN 不能包含例如 `role/my-team/developers/my-role` 的路徑。ARN 的格式必須是 `arn:aws:iam::111122223333:role/my-role`。在此範例中，需移除 `my-team/developers/`。

範例輸出如下。

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-console-viewer-role" to auth ConfigMap
```

3. 為使用者新增映射項目。IAM 最佳實務建議您將許可授予角色而非使用者。此範例假設您在第一個步驟中將 IAM 許可連接至名為 *my-user* 的使用者。使用您的帳戶 ID 取代 *111122223333*。

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user \
    --group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

範例輸出如下。

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-user" to auth ConfigMap
```

4. 再次檢視 ConfigMap 中的映射項目。

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

範例輸出如下。

```
ARN
                  USERNAME                                GROUPS
                    ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:role/my-console-viewer-role
                                                          eks-console-
dashboard-full-access-group
arn:aws:iam::111122223333:user/my-user
                                                          eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

如需新增使用者或角色到 aws-auth ConfigMap 的詳細資訊，請參閱 將 IAM 主體新增至 Amazon EKS 叢集。

1. 開啟 aws-auth ConfigMap 進行編輯。

```
kubectl edit -n kube-system configmap/aws-auth
```

2. 新增映射至 aws-auth ConfigMap，但不要取代任何現有映射。以下範例新增了在第一個步驟新增許可的 IAM 主體之間的映射，並新增了在上一個步驟建立的 Kubernetes 群組：

   - *my-console-viewer-role* 角色和 eks-console-dashboard-full-access-group。
   - *my-user* 使用者和 eks-console-dashboard-restricted-access-group。

   這些範例假設您在第一個步驟中將 IAM 許可連接至名為 *my-console-viewer-role* 的角色和名為 *my-user* 的使用者。請*111122223333*以您的 AWS 帳號 ID 取代。

```
apiVersion: v1
data:
mapRoles: |
```

```
  - groups:
    - eks-console-dashboard-full-access-group
    rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
    username: my-console-viewer-role
mapUsers: |
  - groups:
    - eks-console-dashboard-restricted-access-group
    userarn: arn:aws:iam::111122223333:user/my-user
    username: my-user
```

> **Important**
>
> 角色 ARN 不能包含例如 role/my-team/developers/my-console-viewer-role 的路徑。ARN 的格式必須是 arn:aws:iam::*111122223333*:role/*my-console-viewer-role*。在此範例中，需移除 my-team/developers/。

3. 儲存檔案並結束您的文字編輯器。

Amazon EKS 中的可觀測性

您可以使用許多可用的監控或日誌記錄工具在 Amazon EKS 中觀察您的資料。您可以將 Amazon EKS 日誌資料串流至合作夥伴工具， AWS 服務 以進行資料分析。中有許多可用的服務可提供 AWS Management Console 用於疑難排解 Amazon EKS 問題的資料。您也可以使用 AWS支援的開放原始碼解決方案來監控 Amazon EKS 基礎設施。

在 Amazon EKS 主控台左側導覽窗格中選取叢集後，您就可以選取叢集名稱來檢視叢集執行狀況和詳細資訊。若要檢視有關部署到叢集的任何現有 Kubernetes 資源詳細資訊，請參閱 檢視 Kubernetes 資源。

監控是維護 Amazon EKS 和 AWS 解決方案的可靠性、可用性和效能的重要組成部分。我們建議您從 AWS 解決方案的所有部分收集監視資料。這樣，如果出現多點故障，您可以更輕鬆地進行偵錯。開始監控 Amazon EKS 前，請確保您的監控計畫可以解決下列問題。

- 您的目標是什麼？ 如果叢集大幅擴展，您是否需要即時通知？
- 需要觀察哪些資源？
- 您需要多長時間觀察這些資源？ 貴公司是否希望快速應對風險？
- 您要使用哪些工具？ 如果您已經在啟動過程中執行 AWS Fargate ，則可以使用內建的記錄路由器。
- 您要由誰來執行監控任務？
- 當出現問題時，您希望向誰傳送通知？

在 Amazon EKS 上記錄和監控

Amazon EKS 提供了用於日誌記錄和監控的內建工具。控制平面日誌記錄工具記錄對叢集的所有 API 呼叫、稽核資訊 (擷取哪些使用者對叢集執行哪些操作)，以及以角色為基礎的資訊。如需詳細資訊，請參閱《AWS 規範指引》中的「在 Amazon EKS 上的記錄和監控」。

Amazon EKS 控制平面日誌記錄可將稽核和診斷日誌直接從 Amazon EKS 控制平面提供到您帳戶中的 CloudWatch 日誌。這些日誌可讓您輕鬆執行叢集並確保叢集的安全。您可以選擇所需的確切日誌類型，並將日誌作為日誌串流傳送到中每個 Amazon EKS 叢集的 CloudWatch群組。如需詳細資訊，請參閱 Amazon EKS 控制平面記錄。

> **Note**
>
> 當您在 Amazon 中檢查 Amazon EKS 身份驗證器日誌時 CloudWatch，會顯示包含類似下列範例文字的文字的項目。

```
level=info msg="mapping IAM role" groups="[]"
 role="arn:aws:iam::111122223333:role/XXXXXXXXXXXXXXXXXX-
NodeManagerRole-XXXXXXXX" username="eks:node-manager"
```

預期應包含此文字的項目。username 是 Amazon EKS 內部服務角色，可對受管節點群組和 Fargate 執行特定操作。
對於低層級、可自訂的日誌記錄，可以使用 Kubernetes 日誌記錄。

Amazon EKS 與這項服務整合在一起 AWS CloudTrail，該服務可提供 Amazon EKS 中使用者、角色或 AWS 服務所採取的動作記錄。 CloudTrail 以事件形式擷取 Amazon EKS 的所有 API 呼叫。擷取的呼叫包括從 Amazon EKS 主控台執行的呼叫，以及對 Amazon EKS API 作業發出的程式碼呼叫。如需詳細資訊，請參閱 使用 AWS CloudTrail 日誌記錄 Amazon EKS API 呼叫。

Kubernetes API 伺服器公開多個可用於監控和分析的指標。如需詳細資訊，請參閱 Prometheus 指標。

若要針Fluent Bit對自訂 Amazon CloudWatch 日誌進行設定，請參閱 Amazon CloudWatch 使用者指南Fluent Bit中的設定。

在 Amazon EKS 中記錄和監控工具

Amazon Web Services 提供各種工具讓您可用於監控 Amazon EKS。您可以設定某些工具來設定自動監控，但有些工具則需要手動呼叫。建議您在您的環境和現有工具集允許的範圍內自動執行監控任務。

日誌記錄工具

區域	工具	描述	設定
應用程式	Amazon CloudWatch 容器洞察	容器洞見會從您的容器化應用程式和微型服務收集、彙總及總結指標和日誌。	設定程序
控制平台	AWS CloudTrail	它記錄由使用者、角色或服務	設定程序

區域	工具	描述	設定
		所進行的 API 呼叫。	
執行個體的多 AWS Fargate 個區域	AWS Fargate 路由器日誌	對於 AWS Fargate 執行個體，它會將記錄串流至 AWS 服務或合作夥伴工具。使用 AWS for Fluent Bit。日誌可以流式傳輸到其他 AWS 服務 或合作夥伴工具。	設定程序

監控工具

區域	工具	描述	設定
應用程式	CloudWatch Container Insights	CloudWatch Container Insights 會從您的容器化應用程式和微服務收集、彙總和摘要指標和記錄。	設定程序
應用程式	AWS Distro for OpenTelemetry (ADOT)	它會收集相關的指標、追蹤資料和中繼資料，並將其傳送給 AWS 監控服務或合作夥伴。您可以透過 CloudWatch	設定程序

區域	工具	描述	設定
		容器深入解析進行設定。	
應用程式	Amazon DevOps 大師	可偵測節點級的執行性能和可用性。	設定程序
應用程式	AWS X-Ray	接收有關您的應用程式的追蹤資料。此追蹤資料包括傳入和傳出請求以及有關請求的中繼資料。對於 Amazon EKS，實作需要 OpenTelemetry 附加元件。	設定程序
應用程式	Amazon CloudWatch 可觀測運算符	Amazon CloudWatch 可觀測性操作員會收集指標、日誌和追蹤資料。它將它們發送到 Amazon CloudWatch 和 AWS X-Ray.	設定程序
控制平台	Prometheus	CloudWatch 記錄擷取、封存儲存和資料掃描速率適用於已啟用的控制平面記錄檔。	設定程序

Prometheus 指標

Prometheus 是湊集端點的監控和時間序列資料庫。它提供查詢、彙總和儲存收集之資料的能力。您也可以將其用於警示和警示彙總。本主題說明如何將 Prometheus 設定為受管或開放原始碼選項。監控 Amazon EKS 控制平面指標是常見的使用案例。

Amazon Managed Service for Prometheus 是與 Prometheus 相容的監控和警示服務，可讓您輕鬆地大規模監控容器化應用程式和基礎設施。這是一項全受管服務，既可自動擴展指標的擷取、儲存、查詢和提醒，它還與 AWS 安全服務集成，以便快速安全地訪問您的數據。您可以使用開放原始碼 PromQL 查詢語言來查詢指標並根據指標發出提醒。

如需如何在開啟 Prometheus 指標後使用該指標的詳細資訊，請參閱《Amazon Managed Service for Prometheus 使用者指南》。

建立叢集時開啟 Prometheus 指標

> **Important**
>
> Prometheus 資源的 Amazon 受管服務不在叢集生命週期之外，需要獨立於叢集進行維護。刪除叢集時，請務必同時刪除任何適用的抓取工具，以停止適用的成本。如需詳細資訊，請參閱 Amazon Prometheus 受管服務使用者指南中的尋找和刪除抓取工具。

建立新叢集時，您可以開啟將指標傳送至 Prometheus 的選項。在中 AWS Management Console，此選項位於建立新叢集的 [設定可觀測性] 步驟中。如需詳細資訊，請參閱 建立 Amazon EKS 叢集。

Prometheus 透過稱為湊集的提取型模式，從您的叢集中探索並收集指標。湊集器的設定是為了從您的叢集基礎設施和容器化應用程式收集資料。

當您開啟傳送 Prometheus 指標的選項時，Amazon Managed Service for Prometheus 會提供完全受管的無代理程式湊集器。使用以下進階組態選項以根據需要自訂預設湊集器。

湊集器別名

(選用) 輸入湊集器的唯一別名。

目的地

選擇 Amazon Managed Service for Prometheus 工作區。工作區是專用於儲存和查詢 Prometheus 指標的邏輯空間。使用此工作區，您能夠跨具有該工作區存取權的帳戶檢視 Prometheus 指標。建

立新工作區選項會讓 Amazon EKS 知道可以使用您提供的工作區別名來代表您建立工作區。使用選取現有工作區選項，您可以從下拉式清單中選取現有的工作區。如需工作區的詳細資訊，請參閱《Amazon Managed Service for Prometheus 使用者指南》中的管理工作區。

服務存取

本節總結您在傳送 Prometheus 指標時授予的許可：

- 允許 Amazon Managed Service for Prometheus 描述湊集的 Amazon EKS 叢集
- 允許遠端寫入 Amazon 受管 Prometheus 工作區

如果 `AmazonManagedScraperRole` 已經存在，則湊集器會使用它。選擇 `AmazonManagedScraperRole` 連結以查看許可詳細資訊。如果 `AmazonManagedScraperRole` 尚未存在，請選擇檢視許可詳細資訊連結，以查看您透過傳送 Prometheus 指標來授予的特定許可。

子網

檢視湊集器會繼承的子網路。如果您需要變更它們，請返回建立叢集指定聯網步驟。

安全群組

檢視湊集器會繼承的安全群組。如果您需要變更它們，請返回建立叢集指定聯網步驟。

湊集器組態

視需要修改 YAML 格式的湊集器組態。若要執行這項操作，請使用表單或上傳取代 YAML 檔案。如需詳細資訊，請參閱《Amazon Managed Service for Prometheus 使用者指南》中的湊集器組態。

Amazon Managed Service for Prometheus 是指作為 AWS 受管收集器與叢集一起建立的無代理程式湊集器。如需有關 AWS 受管收集器的詳細資訊，請參閱AWS Amazon Prometheus 受管服務使用者指南中的受管收集器。

> **Important**
>
> 您必須設定 `aws-auth ConfigMap` 來授予湊集器叢集內許可。如需詳細資訊，請參閱《Amazon Managed Service for Prometheus 使用者指南》中的設定 Amazon EKS 叢集。

檢視 Prometheus 湊集器詳細資訊

建立叢集並啟用 Prometheus 指標選項後，您可以檢視 Prometheus 湊集器詳細資訊。在中檢視叢集時 AWS Management Console，請選擇 [可觀測性] 索引標籤。表格會顯示叢集的湊集器清單，包含湊集器 ID、別名、狀態和建立日期等資訊。

若要檢視湊集器的更多詳細資訊，請選擇湊集器 ID 連結。例如，您可以檢視湊集器組態、Amazon Resource Name (ARN)、遠端寫入 URL 和聯網資訊。您可以使用湊集器 ID 作為輸入到 Amazon Managed Service for Prometheus API 操作 (例如 `DescribeScraper` 和 `DeleteScraper`)。您也可以使用 API 來建立更多湊集器。

如需使用 Prometheus API 的詳細資訊，請參閱 Amazon Managed Service for Prometheus API 參考。

使用 Helm 部署 Prometheus

或者，您可以使用 Helm V3 部署 Prometheus 到您的叢集中。如果您已經安裝了 Helm，可以使用 `helm version` 命令檢查您的版本。Helm 是適用於 Kubernetes 叢集的套件管理工具。如需有關 Helm 及其安裝方式的詳細資訊，請參閱 搭配 Amazon EKS 使用 Helm。

在您為 Amazon EKS 叢集設定 Helm 之後，即可用它來依照下列步驟部署 Prometheus。

使用 Helm 部署 Prometheus

1. 建立 Prometheus 命名空間。

```
kubectl create namespace prometheus
```

2. 新增 `prometheus-community` 圖表儲存庫。

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
```

3. 部署 Prometheus。

```
helm upgrade -i prometheus prometheus-community/prometheus \
    --namespace prometheus \
    --set alertmanager.persistence.storageClass="gp2" \
    --set server.persistentVolume.storageClass="gp2"
```

> **Note**
>
> 如果執行此命令時收到錯誤 `Error: failed to download "stable/prometheus" (hint: running `helm repo update` may help)`，請執行 `helm repo update prometheus-community`，然後嘗試再次執行步驟 2 命令。如果收到錯誤 `Error: rendered manifests contain a resource that already exists`，請執行 `helm uninstall` *`your-release-name`* `-n` *`namespace`*，然後嘗試再次執行步驟 3 命令。

4. 確認 `prometheus` 命名空間中的所有 Pods 皆處於 `READY` 狀態。

```
kubectl get pods -n prometheus
```

範例輸出如下。

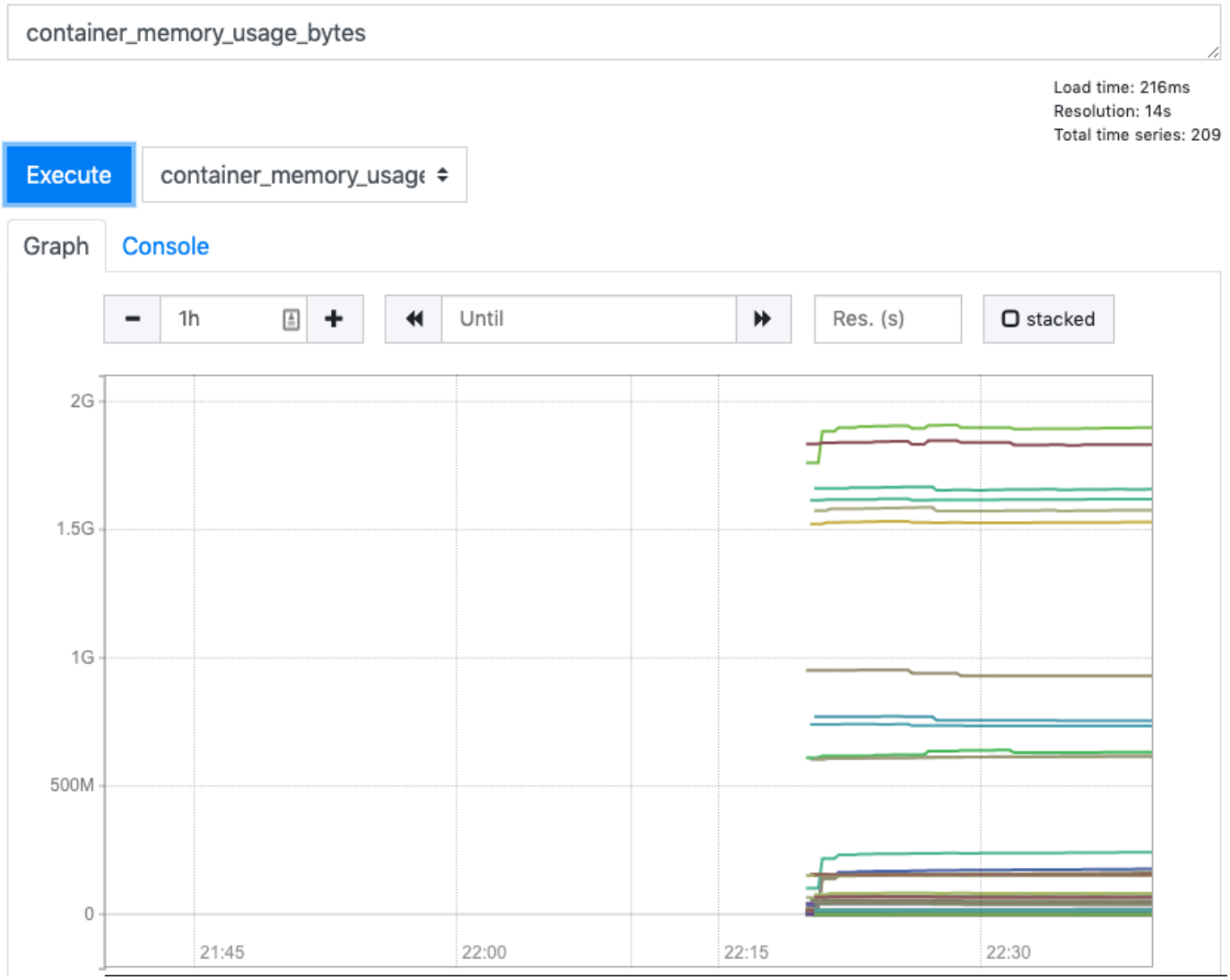
```
NAME                                             READY   STATUS    RESTARTS   AGE
prometheus-alertmanager-59b4c8c744-r7bgp         1/2     Running   0          48s
prometheus-kube-state-metrics-7cfd87cf99-jkz2f   1/1     Running   0          48s
prometheus-node-exporter-jcjqz                   1/1     Running   0          48s
prometheus-node-exporter-jxv2h                   1/1     Running   0          48s
prometheus-node-exporter-vbdks                   1/1     Running   0          48s
prometheus-pushgateway-76c444b68c-82tnw          1/1     Running   0          48s
prometheus-server-775957f748-mmht9               1/2     Running   0          48s
```

5. 使用 `kubectl` 將 Prometheus 主控台的連接埠轉送到本機機器。

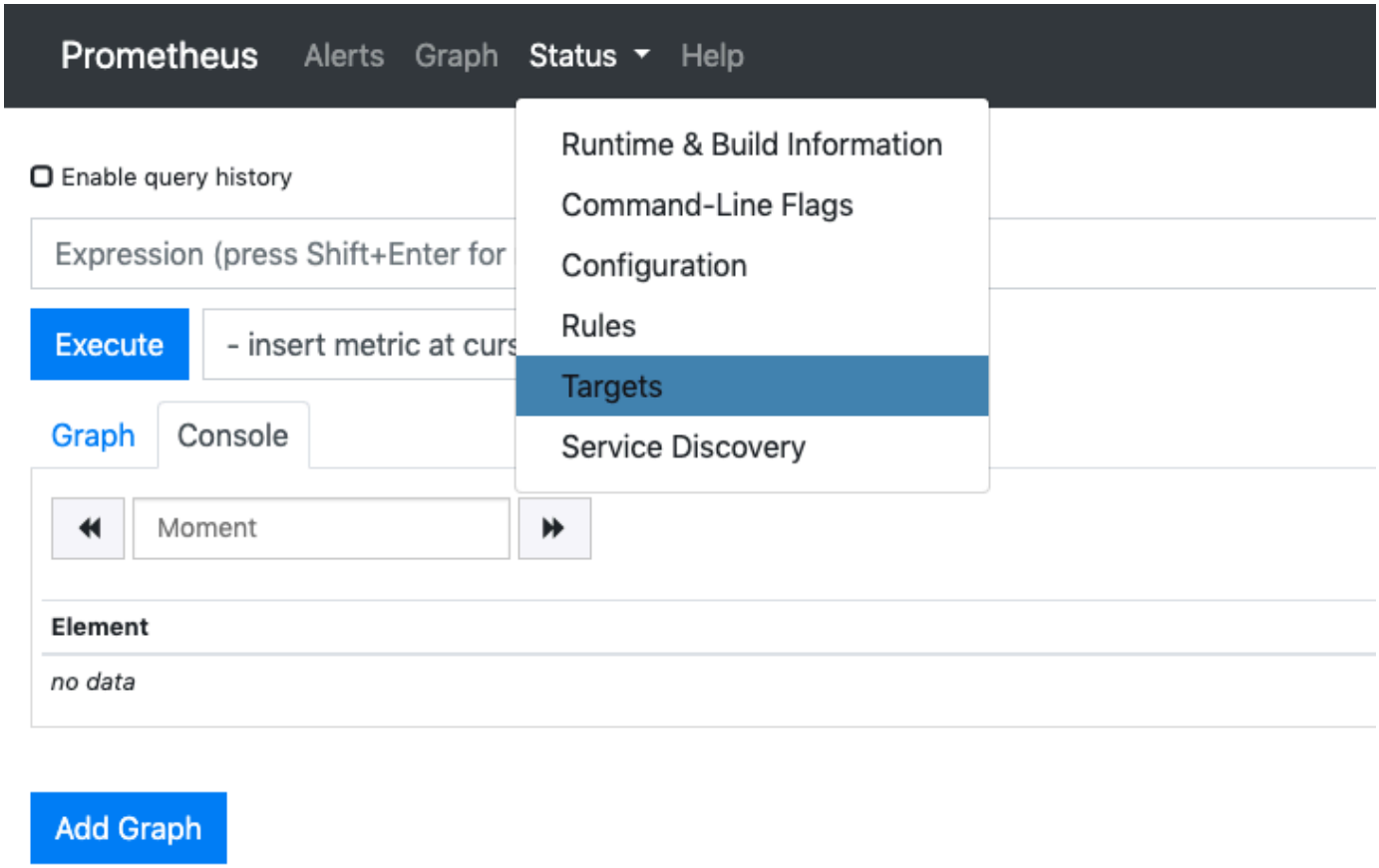
```
kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
```

6. 將 Web 瀏覽器連接至 `http://localhost:9090` 以檢視 Prometheus 主控台。

7. 從 – insert metric at cursor (- 在游標處插入指標) 功能表中選擇一個指標，然後選擇 Execute (執行)。選擇 Graph (圖表) 標籤以顯示一段時間內的指標。下圖顯示一段時間內的 `container_memory_usage_bytes`。



8. 選擇頂部導覽列中的 Status (狀態)，然後選擇 Targets (目標)。



將會顯示使用服務探索連接至 Prometheus 的所有 Kubernetes 端點。

檢視控制平面原始指標

作為部署 Prometheus 的替代方案，Kubernetes API 伺服器會公開一些以 Prometheus 格式顯示的指標。這些指標對於監視和分析非常有用。它們會透過參照 `/metrics` HTTP API 的指標端點於內部公開。類似其他端點，此端點在 Amazon EKS 控制平面公開。此端點主要對於查看特定指標有用。若要隨時間分析指標，建議您部署 Prometheus。

若要檢視原始指標輸出，請使用 `kubectl` 與 `--raw` 旗標。此命令可讓您傳遞任何 HTTP 路徑並傳回原始回應。

```
kubectl get --raw /metrics
```

範例輸出如下。

```
[...]
```

```
# HELP rest_client_requests_total Number of HTTP requests, partitioned by status code,
 method, and host.
# TYPE rest_client_requests_total counter
rest_client_requests_total{code="200",host="127.0.0.1:21362",method="POST"} 4994
rest_client_requests_total{code="200",host="127.0.0.1:443",method="DELETE"} 1
rest_client_requests_total{code="200",host="127.0.0.1:443",method="GET"} 1.326086e+06
rest_client_requests_total{code="200",host="127.0.0.1:443",method="PUT"} 862173
rest_client_requests_total{code="404",host="127.0.0.1:443",method="GET"} 2
rest_client_requests_total{code="409",host="127.0.0.1:443",method="POST"} 3
rest_client_requests_total{code="409",host="127.0.0.1:443",method="PUT"} 8
# HELP ssh_tunnel_open_count Counter of ssh tunnel total open attempts
# TYPE ssh_tunnel_open_count counter
ssh_tunnel_open_count 0
# HELP ssh_tunnel_open_fail_count Counter of ssh tunnel failed open attempts
# TYPE ssh_tunnel_open_fail_count counter
ssh_tunnel_open_fail_count 0
```

此原始輸出會逐字傳回 API 伺服器公開的內容。不同的指標會分行列出，每一行都包含指標名稱、標籤和值。

```
metric_name{"tag"="value"[,...]}
            value
```

Amazon EKS 附加支持 Amazon CloudWatch

Amazon CloudWatch Observability 會收集即時日誌、指標和追蹤資料。它將它們發送到 Amazon CloudWatch 和 AWS X-Ray. 您可以安裝此附加元件以同時啟用 CloudWatch 應用程式訊號，並 CloudWatchContainer Insights增強 Amazon EKS 的可觀察性。這有助於監控基礎設施和容器化應用程式的運作狀態與效能。Amazon CloudWatch Observability Operator 旨在安裝和設定必要的元件。

Amazon EKS 支援 Amazon CloudWatch Observability Operator 作為 Amazon EKS 附加元件。附加元件允許Container Insights叢集中Linux的Windows工作節點和工作節點。若要Container Insights啟用 Windows，Amazon EKS 附加元件版本必須等於`1.5.0`或更高版本。目前，Amazon EKS Windows 上不支持 CloudWatch 應用程序信號。

以下主題描述如何開始使用 Amazon EKS 叢集的 Amazon CloudWatch Observability Operator。

- 如需安裝此附加元件的指示，請參閱 Amazon 使用 CloudWatch 者指南中的使用可 CloudWatch 觀察性 Amazon EKS 附加元件安裝 CloudWatch 代理程式。
- 如需 CloudWatch 應用程式訊號的詳細資訊，請參閱應用程式訊號

- 如需詳細資訊Container Insights，請參閱 Amazon 使用 CloudWatch 者指南Container Insights中的使用。

Amazon EKS 控制平面記錄

Amazon EKS 控制平面日誌記錄可將稽核和診斷日誌直接從 Amazon EKS 控制平面提供到您帳戶中的 CloudWatch 日誌。這些日誌可讓您輕鬆執行叢集並確保叢集的安全。您可以選擇所需的確切日誌類型，並將日誌作為日誌串流傳送到中每個 Amazon EKS 叢集的 CloudWatch群組。如需詳細資訊，請參閱 Amazon CloudWatch 記錄。

您可以選擇要為每個新的或現有的 Amazon EKS 叢集啟用哪些日誌類型，以開始使用 Amazon EKS 控制平面記錄。您可以使用 AWS Management Console、AWS CLI (1.16.139 版本或更新版本) 或透過 Amazon EKS API，啟用或停用個別叢集的日誌類型。啟用後，日誌會自動從 Amazon EKS 叢集傳送到相同帳戶中的 CloudWatch 日誌。

當您使用 Amazon EKS 控制平面記錄時，必須為您執行的每個叢集支付標準 Amazon EKS 定價。對於從叢集傳送至 CloudWatch CloudWatch 記錄的任何記錄，需支付標準記錄資料擷取和儲存費用。您也將支付您在叢集中佈建的任何 AWS 資源的費用，例如 Amazon EC2 執行個體或 Amazon EBS 磁碟區。

以下叢集控制平面日誌類型可供使用。每個日誌類型皆對應 Kubernetes 控制平面的某個元件。如需進一步了解這些元件的詳細資訊，請參閱 Kubernetes 文件中的 Kubernetes 元件。

API 伺服器 (**api**)

叢集的 API 伺服器是公開 Kubernetes API 的控制平面元件。如果您在啟動叢集時或之後隨即啟用 API 伺服器日誌，則日誌中會包含用於啟動 API 伺服器的 API 伺服器旗標。如需詳細資訊，請參閱 Kubernetes 文件中的 `kube-apiserver` 以及稽核政策。

稽核 (**audit**)

Kubernetes 稽核日誌可提供對叢集產生影響的個別使用者、系統管理員或系統元件的記錄。如需詳細資訊，請參閱 Kubernetes 文件中的稽核。

驗證器 (**authenticator**)

驗證器日誌是 Amazon EKS 獨有的。這些日誌代表 Amazon EKS 使用 IAM 憑證進行 Kubernetes 角色型存取控制 (RBAC) 身分驗證的控制平面元件。如需詳細資訊，請參閱叢集管理。

控制器管理員 (**controllerManager**)

控制器管理員負責管理 Kubernetes 隨附的核心控制迴圈。如需詳細資訊，請參閱 Kubernetes 文件中的 kube-controller-manager。

排程器 (**scheduler**)

排程器元件會管理在叢集中執行 Pods 的時間和位置。如需詳細資訊，請參閱 Kubernetes 文件中的 kube-scheduler。

啟用和停用控制平面日誌

根據預設，叢集控制平面記錄檔不會傳送至 CloudWatch 記錄檔。您必須個別啟用每種記錄類型，才能傳送叢集的記錄。 CloudWatch 記錄檔擷取、封存儲存和資料掃描速率適用於已啟用的控制平面記錄檔。如需詳細資訊，請參閱CloudWatch 定價。

若要更新控制平面日誌記錄組態，Amazon EKS 需要每個子網路中最多五個可用的 IP 地址。當您啟用日誌類型時，將會以日誌詳細資訊等級 2 傳送日誌。

AWS Management Console

使用 AWS Management Console 啟用或停用控制平面日誌

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇叢集名稱以顯示您叢集的資訊。
3. 選擇可觀測性索引標籤。
4. 在控制平面日誌記錄區段中，選擇管理日誌記錄。
5. 對於每個日誌類型，選擇其日誌類型為開啟或關閉。根據預設，系統會關閉每個日誌類型。
6. 選擇 Save changes (儲存變更) 以完成操作。

AWS CLI

使用 AWS CLI 啟用或停用控制平面日誌

1. 使用以下命令檢查您的 AWS CLI 版本。

```
aws --version
```

如果您的 AWS CLI 版本低於 1.16.139，您必須先更新到最新版本。若要安裝或升級 AWS CLI，請參閱《AWS Command Line Interface 使用者指南》中的安裝 AWS Command Line Interface。

2. 使用下列 AWS CLI 命令更新叢集的控制平面日誌匯出組態。使用叢集名稱取代 *my-cluster* 並指定所需的端點存取值。

> **Note**
>
> 下列命令會將所有可用的記錄檔類型傳送至 CloudWatch 記錄檔。

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

範例輸出如下。

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "InProgress",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

3. 使用以下命令以及上個命令傳回的叢集名稱和更新 ID，監控日誌組態更新的狀態。當狀態出現 Successful 時，您的更新就完成了。

```
aws eks describe-update \
    --region region-code\
    --name my-cluster \
    --update-id 883405c8-65c6-4758-8cee-2a7c1340a6d9
```

範例輸出如下。

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "Successful",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

檢視叢集控制平面日誌

為 Amazon EKS 叢集啟用任何控制平面日誌類型後，您可以在主控 CloudWatch 台上檢視它們。

若要進一步了解如何檢視、分析和管理日誌 CloudWatch，請參閱 Amazon CloudWatch 日誌使用者指南。

在 CloudWatch 主控台上檢視叢集控制平面記錄

1. 開啟 CloudWatch 主控台。此連結會開啟主控台及顯示目前可用的日誌群組，並以 /aws/eks 字首進行篩選。
2. 選擇您要檢視日誌的叢集。日誌群組的名稱格式是 /aws/eks/*my-cluster*/cluster。
3. 選擇要檢視的日誌串流。以下清單說明每個日誌類型的日誌串流名稱格式。

> **Note**
>
> 隨著日誌串流資料增加，日誌串流名稱將會輪換。當特定日誌類型有多個日誌串流時，您可以使用最新的 Last Event Time (上次事件時間) 來尋找日誌串流名稱，以檢視最新的日誌串流。

- Kubernetes API 伺服器元件記錄 (**api**) – `kube-apiserver-`*`1234567890abcdef01234567890abcde`*
- 稽核 (**audit**) – `kube-apiserver-audit-`*`1234567890abcdef01234567890abcde`*
- 驗證器 (**authenticator**) – `authenticator-`*`1234567890abcdef01234567890abcde`*
- 控制器管理員 (**controllerManager**) – `kube-controller-manager-`*`1234567890abcdef01234567890abcde`*
- 排程器 (**scheduler**) – `kube-scheduler-`*`1234567890abcdef01234567890abcde`*

4. 查看日誌串流事件。

   例如，在檢視 `kube-apiserver-`*`1234567890abcdef01234567890abcde`* 的頂端時，您應該會看到叢集的初始 API 伺服器旗標。

> **Note**
>
> 如果您未在日誌串流的開頭看到 API 伺服器日誌，則 API 伺服器日誌檔案可能已在伺服器上啟用 API 伺服器記錄之前在伺服器上輪換。在啟用 API 伺服器記錄之前旋轉的任何記錄檔都無法匯出至 CloudWatch。
> 不過，您可以使用相同的 Kubernetes 版本建立新的叢集，並在建立叢集時啟用 API 伺服器記錄。使用相同平台版本的叢集已啟用相同的旗標，因此您的旗標應符合新叢集的旗標。在中檢視完新叢集的旗標後 CloudWatch，您可以刪除新叢集。

使用 AWS CloudTrail 日誌記錄 Amazon EKS API 呼叫

Amazon EKS 已與 AWS CloudTrail 整合。CloudTrail 服務會提供由 Amazon EKS 中的使用者、角色或 AWS 服務所執行之動作的記錄。CloudTrail 會將 Amazon EKS 的所有 API 呼叫擷取為事件。事件包括從 Amazon EKS 主控台的呼叫，以及對 Amazon EKS API 操作的程式碼呼叫。

若您建立追蹤，便可將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 的事件。其中包括 Amazon EKS 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。使用 CloudTrail 收集的資訊，您可以判斷關於請求的數個詳細資訊。例如，您可以判斷向 Amazon EKS 提出請求的時間、提出請求的 IP 地址、以及提出請求的人員。

若要進一步了解 CloudTrail，請參閱《AWS CloudTrail 使用者指南》。

主題

- CloudTrail 中的 Amazon EKS 資訊
- 了解 Amazon EKS 日誌檔案項目
- 啟用 Auto Scaling 群組指標集合

CloudTrail 中的 Amazon EKS 資訊

當您建立 AWS 帳戶時，也會在 AWS 帳戶中啟用 CloudTrail。在 Amazon EKS 中發生活動時，該活動將與 Event history (事件歷史紀錄) 中的其他 AWS 服務事件一起記錄在 CloudTrail 事件中。您可以檢視、搜尋和下載 AWS 帳戶的最新事件。如需詳細資訊，請參閱使用 CloudTrail 事件歷史記錄檢視事件。

若要持續記錄您 AWS 帳戶中正在進行的事件 (包括 Amazon EKS 的事件)，請建立追蹤。追蹤能讓 CloudTrail 將日誌檔交付至 Amazon S3 儲存貯體。依預設，當您在主控台中建立追蹤時，該追蹤會套用至所有的 AWS 區域。此追蹤會記錄來自 AWS 分割區中所有 AWS 區域 的事件，並將日誌檔案交付至您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析和處理 CloudTrail 日誌中所收集的事件資料。如需詳細資訊，請參閱下列資源。

- 建立追蹤的概觀
- CloudTrail 支援的服務和整合
- 設定 CloudTrail 的 Amazon SNS 通知
- 接收多個區域的 CloudTrail 日誌檔案和接收多個帳戶的 CloudTrail 日誌檔案

CloudTrail 會記錄所有 Amazon EKS 動作，並記錄在《Amazon EKS API 參考》中。例如，對 `CreateCluster`、`ListClusters` 和 `DeleteCluster` 區段的呼叫，會在 CloudTrail 日誌檔案中產生項目。

每個事件或日誌項目包含提出請求之 IAM 身分類型及所使用之憑證的資訊。如果使用暫時性憑證，此項目會說明憑證的取得方式。

如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

了解 Amazon EKS 日誌檔案項目

追蹤是一種組態，能讓事件以日誌檔案的形式交付到您指定的 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含一或多個日誌項目。事件代表來自任何來源的單一請求，其中包含請求動作的相關資訊。其中包含了動作的日期和時間，以及所使用的請求參數等資訊。CloudTrail 日誌檔並非依公有 API 呼叫的堆疊追蹤排序，因此不會以任何特定順序出現。

以下範例顯示的是展示 `CreateCluster` 動作的 CloudTrail 日誌項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::111122223333:user/username",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "username"
  },
  "eventTime": "2018-05-28T19:16:43Z",
  "eventSource": "eks.amazonaws.com",
  "eventName": "CreateCluster",
  "awsRegion": "region-code",
  "sourceIPAddress": "205.251.233.178",
  "userAgent": "PostmanRuntime/6.4.0",
  "requestParameters": {
    "resourcesVpcConfig": {
      "subnetIds": [
        "subnet-a670c2df",
        "subnet-4f8c5004"
      ]
    },
    "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
    "clusterName": "test"
  },
  "responseElements": {
    "cluster": {
      "clusterName": "test",
      "status": "CREATING",
      "createdAt": 1527535003.208,
```

```
      "certificateAuthority": {},
      "arn": "arn:aws:eks:region-code:111122223333:cluster/test",
      "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
      "version": "1.10",
      "resourcesVpcConfig": {
        "securityGroupIds": [],
        "vpcId": "vpc-21277358",
        "subnetIds": [
          "subnet-a670c2df",
          "subnet-4f8c5004"
        ]
      }
    }
  },
  "requestID": "a7a0735d-62ab-11e8-9f79-81ce5b2b7d37",
  "eventID": "eab22523-174a-499c-9dd6-91e7be3ff8e3",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Amazon EKS 服務連結角色的日誌項目

Amazon EKS 服務連結角色會對 AWS 資源進行 API 呼叫。呼叫如果是由 Amazon EKS 服務連結角色進行，會顯示搭配 username: AWSServiceRoleForAmazonEKS 和 username: AWSServiceRoleForAmazonEKSNodegroup 的 CloudTrail 日誌項目。如需 Amazon EKS 和服務連結角色的詳細資訊，請參閱 使用 Amazon EKS 的服務連結角色。

下列範例顯示 CloudTrail 日誌項目，其示範 AWSServiceRoleForAmazonEKSNodegroup 服務連結角色所做的 DeleteInstanceProfile 動作，如 sessionContext 中所示。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3WHGPEZ7SJ2CW55C5:EKS",
        "arn": "arn:aws:sts::111122223333:assumed-role/
AWSServiceRoleForAmazonEKSNodegroup/EKS",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
```

```
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA3WHGPEZ7SJ2CW55C5",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/eks-
nodegroup.amazonaws.com/AWSServiceRoleForAmazonEKSNodegroup",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAmazonEKSNodegroup"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-02-26T00:56:33Z"
            }
        },
        "invokedBy": "eks-nodegroup.amazonaws.com"
    },
    "eventTime": "2020-02-26T00:56:34Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "DeleteInstanceProfile",
    "awsRegion": "region-code",
    "sourceIPAddress": "eks-nodegroup.amazonaws.com",
    "userAgent": "eks-nodegroup.amazonaws.com",
    "requestParameters": {
        "instanceProfileName": "eks-11111111-2222-3333-4444-abcdef123456"
    },
    "responseElements": null,
    "requestID": "11111111-2222-3333-4444-abcdef123456",
    "eventID": "11111111-2222-3333-4444-abcdef123456",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

啟用 Auto Scaling 群組指標集合

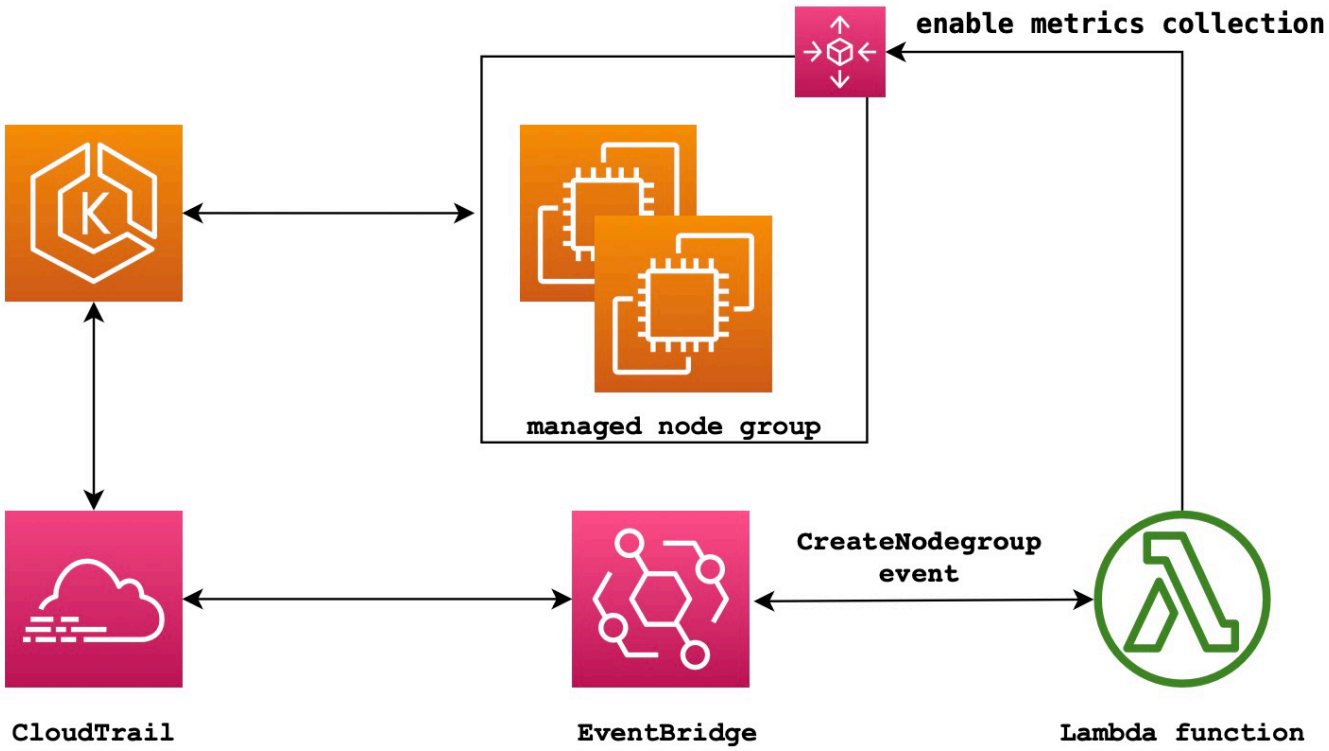
本主題說明如何使用 AWS Lambda 和 AWS CloudTrail 啟用 Auto Scaling 群組指標集合。Amazon EKS 不會自動針對為受管節點建立的 Auto Scaling 群組啟用群組指標集合。

您可以使用 Auto Scaling 群組指標來追蹤 Auto Scaling 群組的變更，並針對閾值設定警示。Auto Scaling 群組指標可在 Auto Scaling 主控台或 Amazon 主 CloudWatch控台中使用。啟用後，Auto Scaling 群組會 CloudWatch 每分鐘將取樣資料傳送到 Amazon。啟用這些指標不會產生額外費用。

透過啟用 Auto Scaling 群組指標集合，您將能夠監控受管節點群組的擴縮。Auto Scaling 群組指標報告 Auto Scaling 群組最小、最大和所需規模。如果節點群組中的節點數量低於最小規模，這代表節點

群組運作狀態不佳，您可以建立警示。追蹤節點群組規模在調整最大計數時也很有用，讓資料平面不會耗盡容量。

建立受管節點群組時，AWS CloudTrail會將`CreateNodegroup`事件傳送至 Amazon EventBridge。透過建立符合`CreateNodegroup`事件的 Amazon EventBridge 規則，您可以觸發 Lambda 函數，為與受管節點群組關聯的 Auto Scaling 群組啟用群組指標收集。



啟用 Auto Scaling 群組指標集合

1. 建立適用於 Lambda 的 IAM 角色。

```
LAMBDA_ROLE=$(aws iam create-role \
  --role-name lambda-asg-enable-metrics \
  --assume-role-policy-document '{"Version": "2012-10-17","Statement":
 [{ "Effect": "Allow", "Principal": {"Service": "lambda.amazonaws.com"}, "Action":
 "sts:AssumeRole"}]}' \
  --output text \
  --query 'Role.Arn')
echo $LAMBDA_ROLE
```

2. 建立允許描述 Amazon EKS 節點群組並啟用 Auto Scaling 群組指標集合的政策。

```
cat > /tmp/lambda-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
            "eks:DescribeNodegroup",
            "autoscaling:EnableMetricsCollection"
            ],
        "Resource": [
            "*"
        ]
      }
    ]
}
EOF
LAMBDA_POLICY_ARN=$(aws iam create-policy \
  --policy-name lambda-asg-enable-metrics-policy \
  --policy-document file:///tmp/lambda-policy.json \
  --output text \
  --query 'Policy.Arn')
echo $LAMBDA_POLICY_ARN
```

3. 將政策連接至 Lambda 的 IAM 角色。

```
aws iam attach-role-policy \
  --policy-arn $LAMBDA_POLICY_ARN \
  --role-name lambda-asg-enable-metrics
```

4. 新增受AWSLambdaBasicExecutionRole管理的策略，該策略具有將日誌寫入日誌所需的 CloudWatch 權限。

```
aws iam attach-role-policy \
  --role-name lambda-asg-enable-metrics \
  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
```

5. 建立 Lambda 代碼。

```
cat > /tmp/lambda-handler.py <<EOF
import json
import boto3
```

```
import time
import logging

eks = boto3.client('eks')
autoscaling = boto3.client('autoscaling')

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    ASG_METRICS_COLLLECTION_TAG_NAME = "ASG_METRICS_COLLLECTION_ENABLED"
    initial_retry_delay = 10
    attempts = 0

    #print(event)

    if not event["detail"]["eventName"] == "CreateNodegroup":
        print("invalid event.")
        return -1

    clusterName = event["detail"]["requestParameters"]["name"]
    nodegroupName = event["detail"]["requestParameters"]["nodegroupName"]
    try:
        metricsCollectionEnabled = event["detail"]["requestParameters"]["tags"]
[ASG_METRICS_COLLLECTION_TAG_NAME]
    except KeyError:
        print(ASG_METRICS_COLLLECTION_TAG_NAME, "tag not found.")
        return

    # Check if metrics collection is enabled in tags
    if metricsCollectionEnabled.lower() != "true":
        print("Metrics collection is not enabled in nodegroup tags.")
        return

    # Get the name of the associated autoscaling group
    print("Getting the autoscaling group name for nodegroup=", nodegroupName, ",
 cluster=", clusterName )
    for i in range(0,10):
        try:
            autoScalingGroup =
 eks.describe_nodegroup(clusterName=clusterName,nodegroupName=nodegroupName)
["nodegroup"]["resources"]["autoScalingGroups"][0]["name"]
        except:
            attempts += 1
```

```
            print("Failed to obtain the associated autoscaling group for
 nodegroup", nodegroupName, "Retrying in", initial_retry_delay*attempts,
 "seconds.")
            time.sleep(initial_retry_delay*attempts)
        else:
            break

    print("Enabling metrics collection on autoscaling group ", autoScalingGroup)

    # Enable metrics collection in the autoscaling group
    try:
        enableMetricsCollection =
 autoscaling.enable_metrics_collection(AutoScalingGroupName=autoScalingGroup,Granularity="1
    except:
        print("Unable to enable metrics collection on nodegroup=",nodegroup)
    print("Enabled metrics collection on nodegroup", nodegroupName)
EOF
```

6. 建立部署套件。

```
cd /tmp
zip function.zip lambda-handler.py
```

7. 建立 Lambda 函數。

```
LAMBDA_ARN=$(aws lambda create-function --function-name asg-enable-metrics-
collection \
  --zip-file fileb://function.zip --handler lambda-handler.lambda_handler \
  --runtime python3.9 \
  --timeout 600 \
  --role $LAMBDA_ROLE \
  --output text \
  --query 'FunctionArn')
echo $LAMBDA_ARN
```

8. 建立 EventBridge 規則。

```
RULE_ARN=$(aws events put-rule --name CreateNodegroupRuleToLambda \
  --event-pattern "{\"source\":[\"aws.eks\"],\"detail-type\":[\"AWS API Call via
 CloudTrail\"],\"detail\":{\"eventName\":[\"CreateNodegroup\"],\"eventSource\":
[\"eks.amazonaws.com\"]}}" \
  --output text \
  --query 'RuleArn')
```

```
echo $RULE_ARN
```

9. 新增 Lambda 函數作為目標。

```
aws events put-targets --rule CreateNodegroupRuleToLambda \
 --targets "Id"="1","Arn"="$LAMBDA_ARN"
```

10. 新增允許 EventBridge 叫用 Lambda 函數的政策。

```
aws lambda add-permission \
  --function-name asg-enable-metrics-collection \
  --statement-id CreateNodegroupRuleToLambda \
  --action 'lambda:InvokeFunction' \
  --principal events.amazonaws.com \
  --source-arn $RULE_ARN
```

Lambda 函數會為您透過將 ASG_METRICS_COLLLECTION_ENABLED 設定 TRUE 來標記之任何受管節點群組啟用 Auto Scaling 群組指標集合。若要確認 Auto Scaling 群組指標集合已啟用，請導航至 Amazon EC2 主控台中相關聯的 Auto Scaling 群組。在 Monitoring (監控) 索引標籤中，您應該會看到 Enable (啟用) 核取方塊已啟用。

ADOT 運算子的 Amazon EKS 附加元件支援

Amazon EKS 支援使用 AWS Management Console、AWS CLI 和 Amazon EKS API 來安裝和管理 AWS Distro for OpenTelemetry (ADOT) 運算子。這可以讓您更輕鬆地啟用在 Amazon EKS 上執行的應用程式，以便將指標和追蹤資料傳送到多個監控服務選項，例如 Amazon CloudWatch、Prometheus 和 X-Ray。

如需詳細資訊，請參閱 AWS Distro for OpenTelemetry 文件中的使用 EKS 附加元件的 AWS Distro for OpenTelemetry 入門。

更多與 Amazon EKS 整合的 AWS 服務

除了其他章節涵蓋的服務外，Amazon EKS 還與更多 AWS 服務合作以提供其他解決方案。本主題識別使用 Amazon EKS 來新增功能的其他服務，或 Amazon EKS 用來執行任務的服務。

主題

- 使用 AWS CloudFormation 建立 Amazon EKS 資源
- Amazon EKS 和 AWS 本機區域
- Deep Learning 容器
- Amazon VPC Lattice
- AWS Resilience Hub
- 使用偵測威脅 Amazon GuardDuty
- 使用 Amazon 安全湖與 Amazon EKS
- Amazon Detective

使用 AWS CloudFormation 建立 Amazon EKS 資源

Amazon EKS 已整合 AWS CloudFormation，這項服務可協助您建立 AWS 資源的模型和設定，以減少建立和管理資源和基礎設施的時間。您建立一個描述所有所需之 AWS 資源的範本 (如 Amazon EKS 叢集)，而 AWS CloudFormation 負責為您佈建與設定這些資源。

當您使用 AWS CloudFormation 時，您可以重複使用您的範本，重複、一致的設定您的 Amazon EKS 資源。您只需要描述您的資源一次，然後在多個 AWS 帳戶與區域內重複佈建相同資源即可。

Amazon EKS 和 AWS CloudFormation 範本

若要佈建和配置 Amazon EKS 資源和相關服務的資源，則必須了解 AWS CloudFormation 範本。範本是以 JSON 或 YAML 格式化的文本檔案。而您亦可以透過這些範本的說明，了解欲在 AWS CloudFormation 堆疊中佈建的資源。如果您不熟悉 JSON 或 YAML，您可以使用 AWS CloudFormation Designer 協助您開始使用 AWS CloudFormation 範本。如需詳細資訊，請參閱 AWS CloudFormation 使用者指南中的什麼是 AWS CloudFormation Designer？。

Amazon EKS 支援在 AWS CloudFormation 中建立叢集和節點群組。如需詳細資訊 (包括 Amazon EKS 資源的 JSON 和 YAML 範本範例)，請參閱 AWS CloudFormation 使用者指南中的 Amazon EKS 資源類型參考。

進一步了解 AWS CloudFormation

若要進一步了解 AWS CloudFormation，請參閱下列資源：

- AWS CloudFormation
- 《AWS CloudFormation 使用者指南》
- 《AWS CloudFormation 命令列介面使用者指南》

Amazon EKS 和 AWS 本機區域

AWS Local Zone 是 AWS 區域 的延伸，與使用者地理位置相近。Local Zones 有自己的網際網路連線，並支援 AWS Direct Connect。在本機區域中建立的資源可以為本機使用者提供低延遲的通訊服務。如需詳細資訊，請參閱 Local Zones。

Amazon EKS 支援本機區域中的特定資源。其中包括自我管理的 Amazon EC2 節點、Amazon EBS 磁碟區和 Aplication Load Balancers (ALB)。在使用本機區域作為 Amazon EKS 叢集的一部分時，建議您考慮以下事項。

節點

您無法在本機區域中使用 Amazon EKS 建立受管節點群組或 Fargate 節點。但是您可以使用 Amazon EC2 API、AWS CloudFormation 或 `eksctl`，在本機區域中建立自我管理的 Amazon EC2 節點。如需詳細資訊，請參閱 自我管理的節點。

網路架構

- Amazon EKS 受管 Kubernetes 控制平面一律在 AWS 區域 中執行。Amazon EKS 受管 Kubernetes 控制平面無法在本機區域中執行。由於本機區域在 VPC 內顯示為子網路，因此 Kubernetes 會將您的本機區域資源做為該子網路的一部分。
- Amazon EKS Kubernetes 叢集會與您在 AWS 區域 或本機區域使用 Amazon EKS 受管彈性網路介面執行的 Amazon EC2 執行個體進行通訊。若要進一步了解 Amazon EKS 聯網架構，請參閱 Amazon EKS 聯網。
- 與區域子網路不同，Amazon EKS 無法將網路介面放置到您的本機區域子網路中。這表示建立叢集時，您不得指定本機區域子網路。

Deep Learning 容器

AWS Deep Learning Containers 是一組 Docker 影像，用於在 Amazon EKS 和 Amazon Elastic Container Service (Amazon ECS) 上訓練和服務 TensorFlow 中的模型。Deep Learning Containers 提供最佳化環境與 TensorFlow、NVIDIA CUDA (適用於 GPU 執行個體) 和 Intel MKL (適用於 CPU 執行個體) 程式庫，並可在 Amazon ECR 中使用。

若要開始使用 Amazon EKS 上的 AWS Deep Learning Containers，請參閱 AWS Deep Learning Containers 開發人員指南中的 Amazon EKS 設定。

Amazon VPC Lattice

Amazon VPC Lattice 是直接內建到 AWS 聯網基礎設施的全受管應用程式聯網服務，可用於跨多個帳戶和虛擬私有雲端 (VPC) 連接、保護和監控服務。有了 Amazon EKS，您可以透過使用 AWS 閘道 API 控制器 (Kubernetes 閘道 API 的實作) 來利用 Amazon VPC Lattice。您可以使用 Amazon VPC Lattice，以簡單且一致的方式設定具有標準 Kubernetes 語意的跨叢集連線。若要開始將 Amazon VPC Lattice 與 Amazon EKS 搭配使用，請參閱 AWS 閘道 API 控制器使用者指南。

AWS Resilience Hub

AWS Resilience Hub 會透過分析 Amazon EKS 叢集的基礎設施以評估其復原能力。AWS Resilience Hub 會使用 Kubernetes 角色型存取控制 (RBAC) 組態，以評估部署到叢集的 Kubernetes 工作負載。如需詳細資訊，請參閱《AWS Resilience Hub 使用者指南》中的啟用對 Amazon EKS 叢集的 AWS Resilience Hub 存取權。

使用偵測威脅 Amazon GuardDuty

Amazon GuardDuty 是一種威脅偵測服務，可協助您保護 AWS 環境中的帳戶、容器、工作負載和資料。使用機器學習 (ML) 模型，以及異常和威脅偵測功能， GuardDuty 持續監控不同的記錄檔來源和執行階段活動，以識別環境中潛在的安全風險和惡意活動並排定優先順序。

除其他功能外，還 GuardDuty 提供下列兩項功能，可偵測 EKS 叢集的潛在威脅：EKS 防護和執行階段監控。

EKS 防護

此功能提供威脅偵測涵蓋範圍，藉由監控相關聯的Kubernetes稽核日誌，協助您保護 Amazon EKS 叢集。 Kubernetes稽核記錄會擷取叢集內的連續動作，包括來自使用者的活動、使用 Kubernetes

API 的應用程式和控制平面。例如， GuardDuty 可以識別呼叫可能竄改Kubernetes叢集中資源的 API 是由未經驗證的使用者叫用。

啟用 EKS 防護後，只 GuardDuty 能存取 Amazon EKS 稽核日誌以進行持續威脅偵測。如果 GuardDuty 識別出叢集的潛在威脅，則會產生特定類型的關聯Kubernetes稽核記錄發現項目。如需Kubernetes稽核日誌中可用發現項目類型的詳細資訊，請參閱 Amazon GuardDuty 使用者指南中的Kubernetes稽核日誌尋找類型。

如需詳細資訊，請參閱 Amazon GuardDuty 使用者指南中的 EKS 防護。

執行期監控

此功能可監控並分析作業系統層級、網路和檔案事件，協助您偵測環境中特定 AWS 工作負載中的潛在威脅。

啟用執行階段監控並在 Amazon EKS 叢集中安裝 GuardDuty 代理程式時，會 GuardDuty 開始監控與此叢集關聯的執行時間事件。如果 GuardDuty 識別出叢集的潛在威脅，就會產生關聯的執行階段監控發現項目。例如，威脅可能從破壞運行易受攻擊 Web 應用程序的單個容器開始。此 Web 應用程式可能具有基礎容器和工作負載的存取權限。在這個案例中，設定不正確的認證可能會導致對帳戶及其中儲存的資料有更廣泛的存取權。

若要設定執行階段監控，請將 GuardDuty 代理程式作為 Amazon EKS 附加元件安裝到叢集。如需附加元件的詳細資訊，請參閱來自 Amazon EKS 的可用 Amazon EKS 附件元件。

如需詳細資訊，請參閱 Amazon GuardDuty 使用者指南中的執行階段監控。

使用 Amazon 安全湖與 Amazon EKS

Amazon Security Lake 是全受管的安全資料湖服務，可讓您集中管理來自各種來源的安全資料，包括 Amazon EKS。透過將 Amazon EKS 與安全湖整合，您可以深入了解在Kubernetes資源上執行的活動，並增強 Amazon EKS 叢集的安全狀態。

> **Note**
>
> 如需將安全湖與 Amazon EKS 搭配使用以及設定資料來源的詳細資訊，請參閱 Amazon 安全湖文件。

使用安全湖與 Amazon Amazon EKS 的好處

集中式安全資料 — Security Lake 會自動從 Amazon EKS 叢集收集並集中安全資料，以及來自其他 AWS 服務、SaaS 提供者、現場部署來源和第三方來源的資料。這可讓您全面檢視整個組織的安全性狀態。

標準化資料格式 — Security Lake 將收集到的資料轉換為開放網路安全架構架構 (OCSF) 格式，這是一種標準的開放原始碼結構描述。這種規範化可以更輕鬆地進行分析，並與其他安全工具和服務整合。

改善威脅偵測 — 透過分析包括 Amazon EKS 控制平面日誌在內的集中式安全資料，您可以更有效地偵測 Amazon EKS 叢集中的潛在可疑活動。這有助於及時識別和響應安全事件。

簡化的資料管理 — Security Lake 透過可自訂的保留和複寫設定來管理安全性資料的生命週期。這樣可簡化資料管理工作，並確保您保留必要的資料，以供合規和稽核之用。

為 Amazon EKS 啟用安全湖

若要開始搭配 Amazon EKS 使用安全湖，請依照下列步驟執行：

1. 為您的 EKS 叢集啟用 Amazon EKS 控制平面記錄功能。如需詳細指示，請參閱啟用和停用控制平面記錄檔。
2. 將 Amazon EKS 稽核日誌新增為安全湖中的來源。然後，安全湖將開始收集有關在 EKS 叢集中執行的 Kubernetes 資源上執行之活動的深入資訊。
3. 根據您的需求，在 Security Lake 中為您的安全性資料設定保留和複寫設定。
4. 使用儲存在 Security Lake 中的標準化 OCSF 資料，進行事件回應、安全性分析，以及與其他 AWS 服務或協力廠商工具整合。例如，您可以使用 Amazon OpenSearch 擷取從 Amazon 安全湖資料產生安全洞見。

分析安全湖中的 EKS 記錄

Security Lake 會將 EKS 記錄事件標準化為 OCSF 格式，讓您更容易分析資料並與其他安全性事件建立關聯。您可以使用各種工具和服務，例如 Amazon Athena QuickSight、Amazon 或第三方安全分析工具，查詢和視覺化標準化資料。

如需 EKS 記錄檔事件之 OCSF 對應的詳細資訊，請參閱 OCS GitHub F 存放庫中的對應參照。

Amazon Detective

Amazon Detective 會協助您分析、調查並快速識別安全調查結果或可疑活動的根本原因。Detective 會自動從您的 AWS 資源收集日誌資料。Detective 接著會使用機器學習、統計分析和圖論來產生視覺化內容，協助您更快地進行有效率的安全調查。Detective 提供預先建置的資料彙總、摘要和內容，可協助您快速分析並判斷潛在安全問題的本質和範圍。如需詳細資訊，請參閱 Amazon Detective 使用者指南。

Detective 組織Kubernetes並將 AWS 數據轉換為調查結果，例如：

- Amazon EKS 叢集詳細資訊，包括建立叢集的 IAM 身分和叢集的服務角色。您可以使用 Detective 調查這些 IAM 身分的 AWS 和 Kubernetes API 活動。
- 容器詳細資料，例如映像和安全性內容。您也可檢閱已終止 Pods 的詳細資訊。
- Kubernetes API 活動，包括 API 活動的整體趨勢以及特定 API 呼叫的詳細資料。例如，您可以顯示在指定時間範圍內成功和失敗的 Kubernetes API 呼叫資料。此外，最新觀察到的 API 呼叫部分可能有助於識別可疑活動。

Amazon EKS 稽核日誌是選用的資料來源套件，可新增至您的 Detective 行為圖表。您可以在帳戶中檢視可用的選用來源套件及其狀態。如需詳細資訊，請參閱《Amazon Detective 使用者指南》中的 Detective 的 Amazon EKS 稽核日誌。

將 Amazon Detective 與 Amazon EKS 搭配使用

檢閱 Amazon EKS 叢集的調查結果

在您檢閱發現項目之前，Detective 必須在叢集所在位於相同 AWS 區域 的情況下啟用至少 48 小時。如需詳細資訊，請參閱《Amazon Detective 使用者指南》中的 設定 Amazon Detective。

1. 打開 Detective 主控台，網址為 https://console.aws.amazon.com/detective/。
2. 從左側導覽窗格選取搜尋。
3. 選取選擇類型，然後選取 EKS 叢集。
4. 輸入叢集名稱或 ARN，然後選擇搜尋。
5. 在搜尋結果中，選取要檢視其活動的叢集名稱。如需有關可檢視內容的詳細資訊，請參閱《Amazon Detective 使用者指南》中涉及 Amazon EKS 叢集的整體 Kubernetes API 活動。

Amazon EKS 故障診斷

此章節涵蓋一些您在使用 Amazon EKS 時可能遇到的常見錯誤，並提供解決方法。如果您需要針對特定 Amazon EKS 區域進行疑難排解，請參閱個別 疑難排解 IAM、對 Amazon EKS 連接器中的問題進行疑難排解 和針對使用 EKS 附加元件的 ADOT 進行疑難排解主題。

如需其他疑難排解資訊，請參閱 AWS re:Post 上有關 Amazon Elastic Kubernetes Service 的知識中心內容。

容量不足

如果您在嘗試建立 Amazon EKS 叢集時收到下列錯誤，則代表其中一個您指定的可用區域沒有足夠的容量來支援叢集。

```
Cannot create cluster 'example-cluster' because region-1d, the targeted Availability Zone, does not currently have sufficient capacity to support the cluster. Retry and choose from these Availability Zones: region-1a, region-1b, region-1c
```

重新嘗試使用叢集 VPC 中的子網路來建立您的叢集，叢集 VPC 託管於此錯誤訊息傳回之可用區域之中。

有些可用區域是叢集無法駐留的。將子網路所在的可用區域與 子網需求和注意事項 中的可用區域清單進行比較。

節點無法加入叢集

有幾個常見的原因會阻擋節點加入叢集：

- 如果節點是受管節點，則 Amazon EKS 會在您建立節點群組時會將項目新增至 `aws-auth ConfigMap`。如果該項目被移除或修改，則您需要重新新增項目。如需詳細資訊，請在您的終端機中輸入 **`eksctl create iamidentitymapping --help`**。您可使用您的叢集名稱取代以下命令中的 *`my-cluster`* 部分，然後執行修改後的命令來檢視目前的 `aws-auth ConfigMap` 項目：**`eksctl get iamidentitymapping --cluster my-cluster`**。您指定之角色的 ARN 不能包含 `/` 以外的路徑。例如，如果您的角色名稱是 `development/apps/my-role`，則您需要在指定角色的 ARN 時將其變更為 `my-role`。請確認您指定的是節點 IAM 角色 ARN (而非執行個體設定檔 ARN)。

如果節點是自我管理的，且您尚未為節點的 IAM 角色的 ARN 建立存取項目，則執行所列出的針對受管節點的命令。如果您已為節點 IAM 角色 ARN 建立存取項目，則可能無法在存取項目中正確進行設定。請確認將節點 IAM 角色 ARN (而非執行個體設定檔 ARN) 指定為 `aws-auth ConfigMap` 項目或存取項目中的主體 ARN。如需存取項目的詳細資訊，請參閱 管理存取項目。

- 節點 AWS CloudFormation 範本ClusterName中的與您希望節點加入的叢集名稱不完全相符。傳遞不正確的值到節點的 `/var/lib/kubelet/kubeconfig` 檔案之不正確的組態內的此欄位中，而節點不會加入叢集。
- 節點未標記為叢集所擁有。您的節點必須套用下列標籤，其中 *`my-cluster`* 會以叢集的名稱取代。

索引鍵	值
`kubernetes.io/cluster/` *`my-cluster`*	owned

- 節點可能無法使用公有 IP 地址存取叢集。確定已將公有子網路中部署的節點指派給公有 IP 地址。如果沒有，您可以在啟動節點後將彈性 IP 地址關聯至節點。如需詳細資訊，請參閱將彈性 IP 地址與執行中的執行個體或網路介面建立關聯。如果公有子網路未設定為自動將公有 IP 地址指派給部署到該子網路的執行個體，則建議您啟用該設定。如需詳細資訊，請參閱修改子網路的公有 IPv4 定址屬性。如果節點部署到私有子網路，則子網路必須具有指派給它的公有 IP 地址的 NAT 閘道路由。
- 您要部署節點 AWS 區域 的 AWS STS 端點未啟用您的帳戶。若要啟用區域，請參閱AWS STS 在 AWS 區域.
- 節點沒有私有 DNS 項目，導致 `kubelet` 日誌包含 `node "" not found` 錯誤。確保建立節點所在的 VPC 具有為 `domain-name` 設定的值以及在 `DHCP options set` 中將 `domain-name-servers` 設定為 `Options`。預設值為 `domain-name:<region>.compute.internal` 和 `domain-name-servers:AmazonProvidedDNS`。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 DHCP 選項集。
- 如果受管理節點群組中的節點未在 15 分鐘內連線到叢集，則會發出健全狀況問題「NodeCreation失敗」，並將主控台狀態設定為`Create failed`。對於啟動時間緩慢的 Windows AMI，可以使用快速啟動來解決此問題。

若要識別和疑難排解導致 Worker 節點無法加入叢集的常見原因，您可以使用 `AWSSupport-TroubleshootEKSWorkerNode` Runbook。如需詳細資訊，請參閱 AWS Systems Manager Automation Runbook 參考 中的 `AWSSupport-TroubleshootEKSWorkerNode`。

未經授權或存取遭拒 (kubectl)

如果您在執行 kubectl 命令時，收到以下其中一個錯誤，則表示未為 Amazon EKS 正確設定 kubectl，或者表示您使用的 IAM 主體 (角色或使用者) 憑證並未映射到對 Amazon EKS 叢集上的 Kubernetes 物件具有足夠許可的 Kubernetes 使用者。

- `could not get token: AccessDenied: Access denied`
- `error: You must be logged in to the server (Unauthorized)`
- `error: the server doesn't have a resource type "svc"`

這種情況可能由下列原因之一造成：

- 該叢集使用一個 IAM 主體的憑證建立，而 kubectl 設定為使用另一個 IAM 主體的憑證。若要解決此問題，請更新 kube config 檔案以使用建立叢集時使用的憑證。如需詳細資訊，請參閱 建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。
- 如果叢集符合 管理存取項目「先決條件」部分的最低平台要求，則 IAM 主體不存在存取項目。如果存在，則沒有為其定義的必要的 Kubernetes 群組名稱，或沒有與其關聯的正確存取政策。如需詳細資訊，請參閱 管理存取項目。
- 如果叢集不符合 管理存取項目 中的最低平台要求，則 aws-auth ConfigMap 中不存在包含 IAM 主體的項目。如果存在，則不會映射到繫結到具有必要許可的 Kubernetes Role 或 ClusterRole 的 Kubernetes 群組名稱。如需有關 Kubernetes 角色型授權 (RBAC) 物件的詳細資訊，請參閱 Kubernetes 文件中的 Using RBAC authorization 一節。您可使用您的叢集名稱取代以下命令中的 *my-cluster* 部分，然後執行修改後的命令來檢視目前的 aws-auth ConfigMap 項目：**eksctl get iamidentitymapping --cluster *my-cluster***。如果您的 IAM 主體 ARN 的項目不在 ConfigMap 中，則請在終端機中輸入 **eksctl create iamidentitymapping --help** 以了解如何建立一個。

如果您安裝並設定 AWS CLI，則可以設定您使用的 IAM 登入資料。如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的設定 AWS CLI。如果您透過擔任 IAM 角色來存取叢集上的 Kubernetes 物件，則也可以將 kubectl 設定為使用 IAM 角色。如需詳細資訊，請參閱 建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

hostname doesn't match

您的系統 Python 版本必須為 2.7.9 或更新版本。否則，您會在 AWS CLI 撥打 Amazon EKS 時收到hostname doesn't match錯誤訊息。如需詳細資訊，請參閱 Python Requests Frequently Asked Questions 中的 What are "hostname doesn't match" errors 部分。

getsockopt: no route to host

Docker 會在 Amazon EKS 叢集的 172.17.0.0/16 CIDR 範圍中執行。我們建議不要將叢集的 VPC 子網路與這個範圍重疊。否則，您會收到以下錯誤：

```
Error: : error upgrading connection: error dialing backend: dial tcp
 172.17.<nn>.<nn>:10250: getsockopt: no route to host
```

Instances failed to join the Kubernetes cluster

如果您Instances failed to join the Kubernetes cluster在中收到錯誤 AWS Management Console，請確定已啟用叢集的私人端點存取，或已正確設定 CIDR 區塊以進行公用端點存取。如需詳細資訊，請參閱 Amazon EKS 叢集端點存取控制。

受管節點群組錯誤代碼

如果受管節點群組遇到硬體運作狀態問題，Amazon EKS 會傳回錯誤代碼，以協助您診斷問題。這些運作狀態檢查不會偵測軟體問題，因為檢查是以 Amazon EC2 運作狀態檢查為基礎。以下清單描述了這些錯誤代碼。

AccessDenied

Amazon EKS 或一或多個受管節點無法透過 Kubernetes 叢集 API 伺服器進行身分驗證或授權。如需有關解決常見原因的詳細資訊，請參閱 修正受管節點群組的 AccessDenied 錯誤的常見原因。私有 Windows AMI 也可能導致此錯誤代碼隨 Not authorized for images 錯誤訊息同時出現。如需詳細資訊，請參閱 Not authorized for images。

AmiIdNotFound

我們找不到與您的啟動範本相關聯的 AMI ID。請確認 AMI 已存在且已與您的帳戶共用。

AutoScalingGroupNot找到

我們找不到與受管節點群組相關聯的 Auto Scaling 群組。您可以使用相同設定來重新建立 Auto Scaling 群組以復原。

ClusterUnreachable

Amazon EKS 或一或多個受管節點無法與您的 Kubernetes 叢集 API 伺服器通訊。如果發生網路中斷或 API 伺服器處理請求逾時，就會發生這種情況。

EC2 SecurityGroup NotFound

我們找不到叢集的叢集安全群組。您必須重新建立叢集。

EC2 SecurityGroup DeletionFailure

無法刪除受管節點群組的遠端存取安全群組。從安全群組移除任何相依項目。

EC2 LaunchTemplate NotFound

我們找不到受管節點群組的 Amazon EC2 啟動範本。您必須重新建立節點群組以復原。

EC2 LaunchTemplate VersionMismatch

受管節點群組的 Amazon EC2 啟動範本版本與 Amazon EKS 建立的版本不相符。您可以還原為 Amazon EKS 所建立的版本以復原。

IamInstanceProfileNot找到

我們找不到受管節點群組的 IAM 執行個體設定檔。您可以使用相同設定來重新建立執行個體描述檔以復原。

IamNodeRoleNot找到

我們找不到受管節點群組的 IAM 角色。您可以使用相同設定來重新建立 IAM 角色以復原。

AsgInstanceLaunchFailures

您的 Auto Scaling 群組嘗試啟動執行個體時失敗。

NodeCreation失敗

您啟動的執行個體無法向 Amazon EKS 叢集註冊。這項失敗的常見原因是節點 IAM 角色許可不足或節點缺少對外網際網路存取。您的節點必須符合下列任何一項要求：

- 能夠使用公有 IP 地址存取網際網路。與節點所在的子網路相關聯的安全群組必須允許通訊。如需更多詳細資訊，請參閱 子網需求和注意事項 及 Amazon EKS 安全群組與考量。
- 您的節點和 VPC 必須符合 私有叢集要求 的要求。

InstanceLimit已超過

您的 AWS 帳戶無法啟動指定執行個體類型的任何其他執行個體。您可以請求提升 Amazon EC2 執行個體限制來復原。

InsufficientFree地址

與受管節點群組相關聯的一個或多個子網沒有足夠的可用 IP 地址供新節點使用。

InternalFailure

這些錯誤通常是由 Amazon EKS 伺服器端問題所造成。

修正受管節點群組的 **AccessDenied** 錯誤的常見原因

在受管節點群組上執行操作時發生 `AccessDenied` 錯誤的最常見原因，是缺少 `eks:node-manager` `ClusterRole` 或 `ClusterRoleBinding`。Amazon EKS 會在您的叢集中設定這些資源，作為與受管節點群組上線的一部分，這些資源是管理節點群組所必需的。

`ClusterRole` 可能隨著時間而改變，但看起來應與以下範例相似：

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
  - list
  - watch
  - patch
```

```
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
```

ClusterRoleBinding 可能隨著時間而改變，但看起來應與以下範例相似：

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

驗證 eks:node-manager ClusterRole 是否存在。

```
kubectl describe clusterrole eks:node-manager
```

如果存在，請將輸出與前一個 ClusterRole 範例進行比較。

驗證 eks:node-manager ClusterRoleBinding 是否存在。

```
kubectl describe clusterrolebinding eks:node-manager
```

如果存在，請將輸出與前一個 ClusterRoleBinding 範例進行比較。

如果在請求受管節點群組操作時您發現缺少或損壞 ClusterRole 或 ClusterRoleBinding 是 AcessDenied 錯誤的原因，您可以將其還原。將下列內容儲存到名為 *eks-node-manager-role.yaml* 的檔案中。

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
```

```
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

套用檔案。

```
kubectl apply -f eks-node-manager-role.yaml
```

重試節點群組操作，看看是否可以解決您的問題。

Not authorized for images

導致 Not authorized for images 錯誤訊息的一個潛在原因是使用私有 Amazon EKS Windows AMI 啟動 Windows 受管節點群組。發布新的 Windows AMI 後， AWS 將超過 4 個月的 AMI 設為私有，這使得它們無法再訪問。如果您的受管節點群組使用私有 Windows AMI，請考慮更新Windows受管節點群組。雖然我們無法保證我們可以提供對已設為私密 AMI 的存取權，但您可以向 Sup AWS port 人員提交票證來要求存取權限。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的修補程式、安全性更新和 AMI ID。

節點處於NotReady狀態

如果您的節點進入NotReady狀態，這可能表示節點運作狀況不佳且無法排程新Pods節點。這可能是由於各種原因而發生的，例如節點缺乏足夠的資源來容納 CPU、記憶體或可用磁碟空間。

對於 Amazon EKS 最佳化的 Windows AMI，組態中預設不會針對預設指定的運算資源保留。kubelet若要協助避免資源問題，您可以透過提供kube-reserved和/或的組態值，為系統程序保留運算資源system-reserved。kubelet您可以使用啟動程序指-KubeletExtraArgs令碼中的命令列參數來執行此作業 如需詳細資訊，請參閱Kubernetes文件中的為系統精靈保留計算資源，以及本使用指南中的 Bootstrap 指令碼組態參數。

CNI 日誌收集工具

Amazon VPC CNI plugin for Kubernetes 有自己的故障診斷指令碼，在 /opt/cni/bin/aws-cni-support.sh 節點上可用。您可以使用指令碼來收集支援案例和一般故障診斷的診斷日誌。

在您的節點使用以下命令即可執行指令碼：

```
sudo bash /opt/cni/bin/aws-cni-support.sh
```

> **Note**
>
> 如果指令碼不在上述位置，CNI 容器將無法執行。您可使用以下命令手動下載並執行指令碼：
>
> ```
> curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/log-collector-script/linux/eks-log-collector.sh
> ```

```
sudo bash eks-log-collector.sh
```

該指令碼收集的診斷資訊如下。您已部署的 CNI 版本可能比指令碼版本更早。

```
      This is version 0.6.1. New versions can be found at https://github.com/awslabs/
amazon-eks-ami

Trying to collect common operating system logs...
Trying to collect kernel logs...
Trying to collect mount points and volume information...
Trying to collect SELinux status...
Trying to collect iptables information...
Trying to collect installed packages...
Trying to collect active system services...
Trying to collect Docker daemon information...
Trying to collect kubelet information...
Trying to collect L-IPAMD information...
Trying to collect sysctls information...
Trying to collect networking information...
Trying to collect CNI configuration information...
Trying to collect running Docker containers and gather container data...
Trying to collect Docker daemon logs...
Trying to archive gathered information...

 Done... your bundled logs are located in /var/
log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

診斷資訊收集後將存放於：

```
/var/log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

容器執行階段網路尚未就緒

您可能會收到類似下列的 `Container runtime network not ready` 錯誤和授權錯誤：

```
4191 kubelet.go:2130] Container runtime network not ready: NetworkReady=false
 reason:NetworkPluginNotReady message:docker: network plugin is not ready: cni config
 uninitialized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
```

```
4191 kubelet_node_status.go:106] Unable to register node
 "ip-10-40-175-122.ec2.internal" with API server: Unauthorized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
```

這種情況可能由下列原因之一造成：

1. 叢集上沒有 aws-auth ConfigMap，或者它不包含針對您設定節點時所使用 IAM 角色的項目。

   如果節點符合以下條件之一，則此 ConfigMap 項目為必需：

   - 具有任何 Kubernetes 或平台版本之叢集中的受管節點。
   - 版本低於 管理存取項目 主題「先決條件」部分中列出的平台版本之一的叢集中的自我管理節點。

   若要解決此問題，您可使用您的叢集名稱取代以下命令中的 *my-cluster* 部分，然後執行修改後的命令來檢視目前 ConfigMap 中的項目：**eksctl get iamidentitymapping --cluster *my-cluster***。如果執行該命令時遇到錯誤訊息，則可能是因為叢集沒有 aws-auth ConfigMap。以下命令會將項目新增至 ConfigMap。如果 ConfigMap 不存在，則該命令會建立它。將 *111122223333* 取代為 IAM 角色的 AWS 帳戶 識別碼，並以節點角色的名稱取代 *MyAmazoneks NodeRole*。

   ```
   eksctl create iamidentitymapping --cluster my-cluster \
       --arn arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --group
    system:bootstrappers,system:nodes \
       --username system:node:{{EC2PrivateDNSName}}
   ```

   您指定之角色的 ARN 不能包含 / 以外的路徑。例如，如果您的角色名稱是 development/apps/my-role，則您需要在指定角色的 ARN 時將其變更為 my-role。請確認您指定的是節點 IAM 角色 ARN (而非執行個體設定檔 ARN)。

2. 您自我管理節點所在的叢集的平台版本為 管理存取項目 主題「先決條件」中列出的最低版本，但 aws-auth ConfigMap (請參閱上一項) 中沒有針對該節點之 IAM 角色的項目或不存在針對該角色的存取項目。若要解決此問題，您可使用您的叢集名稱取代以下命令中的 *my-cluster* 部分，然後執行修改後的命令來檢視目前的存取項目：**aws eks list-access-entries --cluster-name *my-cluster***。以下命令會為相應節點的 IAM 角色新增一個存取項目。將 *111122223333* 取代為 IAM 角色的 AWS 帳戶 識別碼，並以節點角色的名稱取代 *MyAmazoneks NodeRole*。如果節點為 Windows 節點，請用 **EC2_Windows** 取代 *EC2_Linux*。請確認您指定的是節點 IAM 角色 ARN (而非執行個體設定檔 ARN)。

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --type EC2_Linux
```

TLS 交握逾時

當節點無法建立與公有 API 伺服器端點的連接時，您可能會收到類似下列的錯誤。

```
server.go:233] failed to run Kubelet: could not init cloud provider "aws": error
 finding instance i-1111f2222f333e44c: "error listing AWS instances: \"RequestError:
 send request failed\\ncaused by: Post  net/http: TLS handshake timeout\""
```

kubelet 程序將持續重新產生並測試 API 伺服器端點。在控制平面中執行叢集滾動更新 (例如組態變更或版本更新) 的任何程序期間，也可能會暫時發生錯誤。

若要解決此問題，請檢查路由表和安全群組，以確保來自節點的流量可以到達公有端點。

InvalidClientTokenId

如果您針對中國叢集Pod或DaemonSet部署到中國叢集的服務帳戶使用 IAM 角色 AWS 區域，且尚未在規格中設定AWS_DEFAULT_REGION環境變數，則Pod或DaemonSet可能會收到下列錯誤：

```
An error occurred (InvalidClientTokenId) when calling the GetCallerIdentity operation:
 The security token included in the request is invalid
```

若要解決此問題，您需要將 AWS_DEFAULT_REGION 環境變數新增至您的 Pod 或 DaemonSet 規格，如下列範例 Pod 規格所示。

```
apiVersion: v1
kind: Pod
metadata:
  name: envar-demo
  labels:
    purpose: demonstrate-envars
spec:
  containers:
  - name: envar-demo-container
    image: gcr.io/google-samples/node-hello:1.0
    env:
```

```
    - name: AWS_DEFAULT_REGION
      value: "region-code"
```

VPC 許可 Webhook 憑證過期

如果用來簽署 VPC 許可 Webhook 的憑證過期，則新 Windows Pod 部署的狀態會保持在 `ContainerCreating`。

若要解決此問題，如果資料平面支援舊版 Windows，請參閱 續約 VPC 許可 Webhook 憑證。如果叢集和平台版本晚於 Windows 支援先決條件列出的版本，則建議您移除資料平面上的舊版 Windows 支援，並為您的控制平面啟用它。如此，您不再需要管理 Webhook 憑證。如需詳細資訊，請參閱 為您的 Amazon EKS 叢集啟用 Windows 支援。

升級控制平面之前，節點群組必須符合 Kubernetes 版本

在將控制平面升級為新的 Kubernetes 版本之前，您叢集中的次要版本的受管和 Fargate 節點必須要與控制平面目前版本的版本相同。在將所有 Amazon EKS 受管節點升級為目前的叢集版本前，Amazon EKS `update-cluster-version` API 都會拒絕請求。Amazon EKS 提供 API 來升級受管節點。如需升級受管節點群組 Kubernetes 版本的資訊，請參閱 更新受管節點群組。若要升級 Fargate 節點的版本，請刪除節點所代表的 pod，並在升級控制平面後重新部署 pod。如需詳細資訊，請參閱 更新 Amazon EKS 叢集 Kubernetes 版本。

啟動許多節點時，會出現 `Too Many Requests` 錯誤

如果同時啟動許多節點，您可能會 Amazon EC2 使用者資料執行日誌看見錯誤訊息，其顯示 `Too Many Requests`。這可能是因為控制平面因 `describeCluster` 呼叫超載。超載會導致調節、節點無法執行引導指令碼，以及節點無法完全加入叢集。

確認將 `--apiserver-endpoint`、`--b64-cluster-ca` 和 `--dns-cluster-ip` 引數傳遞至節點的引導指令碼。包含這些引數時，不需要引導指令碼進行 `describeCluster` 呼叫，這有助於防止控制平面超載。如需詳細資訊，請參閱 提供使用者資料以將引數傳遞給 `bootstrap.sh` 檔案，其中包含經 Amazon EKS 優化的 Linux/Bottlerocket AMI。

針對 Kubernetes API 伺服器請求回應的 HTTP 401 未經授權的錯誤

如果叢集上的 Pod 服務帳戶字符已過期，您會看到這些錯誤。

您的 Amazon EKS 叢集的 Kubernetes API 伺服器拒絕超過 90 天的字符的請求。在 Kubernetes 舊版本中，字符沒有過期。這意味著倚賴這些字符的客戶端必須在一小時內進行重新整理。為了防止 Kubernetes API 服務器因無效字符而拒絕您的請求，您的工作負載使用的 Kubernetes 用戶端開發套件版本必須與以下版本相同或為更新版本：

- Go 版本 `0.15.7` 和更新版本
- Python 版本 `12.0.0` 和更新版本
- Java 版本 `9.0.0` 和更新版本
- JavaScript 版本`0.10.3`及更新版本
- Ruby `master` 分支
- Haskell 版本 `0.3.0.0`
- C# 版本 `7.0.5` 和更新版本

您可以識別叢集中所有使用過時字符的現有 Pods。如需詳細資訊，請參閱 Kubernetes 服務帳戶。

Amazon EKS 平台版本比目前平台版本落後兩個版本以上

當 Amazon EKS 無法自動更新您的叢集時，平台版本可能會發生這種情況。儘管造成這種情況的原因很多，然而一些常見的原因如下。如果這些問題中有任何一個適用於您的叢集，其可能仍然可以運作，但 Amazon EKS 不會更新其平台版本。

問題

該叢集 IAM 角色已刪除，此角色是在建立叢集時指定。您可以使用以下命令，查看指定了哪個角色。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-cluster --name my-cluster --query cluster.roleArn --output text | cut
 -d / -f 2
```

範例輸出如下。

```
eksClusterRole
```

解決方案

建立具有相同名稱的新叢集 IAM 角色。

問題

已刪除叢集建立期間指定的子網路 – 即叢集建立期間指定要與叢集搭配使用的子網路。您可以使用以下命令，查看指定了哪些子網路。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-cluster --name my-cluster --query cluster.resourcesVpcConfig.subnetIds
```

範例輸出如下。

```
[
"subnet-EXAMPLE1",
"subnet-EXAMPLE2"
]
```

解決方案

確認您的帳戶中是否存在子網路 ID。

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" --query
 "Subnets[*].SubnetId"
```

範例輸出如下。

```
[
"subnet-EXAMPLE3",
"subnet-EXAMPLE4"
]
```

如果輸出中傳回的子網路 ID 與建立叢集時指定的子網路 ID 不符，假如您希望 Amazon EKS 更新叢集，則需要變更叢集使用的子網路。這是因為如果您在建立叢集時指定了兩個以上的子網路，Amazon EKS 會隨機選取您指定在其中建立新彈性網路介面的子網路。這些網路介面可讓控制平面與節點進行通訊。如果叢集選取的子網路不存在，Amazon EKS 將不會更新叢集。您無法控制 Amazon EKS 會從您於建立叢集時指定的子網路中選擇哪些在其中建立新網路介面。

當您啟動叢集的 Kubernetes 版本更新，該更新可能會因相同的原因而失敗。

問題

叢集建立期間指定的安全群組已刪除 – 如果您在叢集建立期間指定了安全群組，您可以使用下列命令查看其 ID。使用您叢集的名稱取代 *my-cluster*。

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.securityGroupIds
```

範例輸出如下。

```
[
    "sg-EXAMPLE1"
]
```

如果傳回 []，若在建立叢集時未指定安全群組，而缺少安全群組並不是問題所在。如果傳回安全群組，則請確認安全群組位於您的帳戶中。

解決方案

請確認這些安全性群組是否位於您的帳戶中。

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=$vpc_id" --query
 "SecurityGroups[*].GroupId"
```

範例輸出如下。

```
[
"sg-EXAMPLE2"
]
```

如果輸出中傳回的安全群組 ID 與建立叢集時指定的安全群組 ID 不符，若您希望 Amazon EKS 更新該叢集，則需要變更叢集使用的安全群組。如果在建立叢集時指定的安全群組 ID 不存在，Amazon EKS 將不會更新叢集。

當您啟動叢集的 Kubernetes 版本更新，該更新可能會因相同的原因而失敗。

Amazon EKS 無法更新叢集平台版本的其他原因

- 您在建立叢集時指定的每個子網路中，沒有至少六個 (雖然我們建議使用 16 個) 可用的 IP 地址。如果子網路中可用的 IP 地址不足，則需要釋出子網路中的 IP 地址，或是需要變更叢集使用的子網路，使用具有足夠 IP 地址的子網路。
- 您在建立叢集時啟用了密碼加密，且您指定的 AWS KMS 金鑰已被刪除。假如您希望 Amazon EKS 更新叢集，您必須建立新的叢集

含解析路徑的叢集健康狀態常見問題與錯誤

Amazon EKS 會偵測 Amazon EKS 叢集以及叢集基礎設施的問題，並將問題存放在叢集運作狀態中。藉助叢集運作狀態資訊，您可以更快速地偵測、診斷並解決叢集問題，這可讓您建立更安全且更安全的應用程式環境 up-to-date。此外，由於必要的基礎設施或叢集組態出現問題，您可能無法升級至較新版本的 Kubernetes，也可能無法讓 Amazon EKS 在降級的叢集上安裝安全更新。Amazon EKS 可能需要 3 小時才能偵測到問題或偵測某個問題是否已解決。

維護 Amazon EKS 叢集的運作狀態是 Amazon EKS 及其使用者共同的責任。使用者負責 IAM 角色和 Amazon VPC 子網路的必備基礎設施，以及其他必須事先提供的必要基礎設施。Amazon EKS 偵測此基礎設施和叢集的組態變更。

若要在 Amazon EKS 主控台中查看叢集的運作狀態，請在 Amazon EKS 叢集詳細資訊頁面的概觀索引標籤中，找到名為運作狀態問題的區段。相關資料也可以透過呼叫 EKS API 中的 `DescribeCluster` 動作來取得，例如從 AWS Command Line Interface中進行呼叫。

這個功能有什麼用？

> 您可以提高 Amazon EKS 叢集運作狀態的可見度、快速診斷並修復任何問題，而無需花費時間偵錯或開啟 AWS 支援案例。例如：您不小心刪除了 Amazon EKS 叢集的子網路，Amazon EKS 將無法建立跨帳戶網路界面和Kubernetes AWS CLI 命令 (例如 `kubectl` exec 或日誌)。`kubectl`這些操作將會失敗並顯示錯誤：`"Error from server: error dialing backend: remote error: tls: internal error."`。這時會出現內容為此的 Amazon EKS 運作狀態問題：`subnet-da60e280 was deleted: could not create network interface`。

此功能如何與其他 AWS 服務相關聯或運作？

> IAM 角色和 Amazon VPC 子網路是叢集運作狀態對其進行問題偵測的其中兩項必備基礎設施。如果這些資源設定不正確，則此功能將傳回詳細錯誤資訊。

有運作狀態問題的叢集是否收費？

> 是的。每個 Amazon EKS 叢集均以標準 Amazon EKS 定價計費。但叢集運作狀態功能是免費提供的。

此功能可否用於 AWS Outposts上的 Amazon EKS 叢集？

> 是，在 AWS 雲端中偵測到 EKS 叢集的叢集問題，包括上的延伸叢集 AWS Outposts 和本機叢集。AWS Outposts叢集運作狀態功能不會偵測 Amazon EKS Anywhere 或 Amazon EKS Distro (EKS-D) 的問題。

當偵測到新問題時我會收到通知嗎？

不會。您需要查看 Amazon EKS 主控台或呼叫 EKS `DescribeCluster` API。

主控台是否會向我發出有關運作狀態問題的警告？

是。任何存在運作狀態問題的叢集都會在主控台頂部包含一則橫幅。

前兩欄是 API 回應值所需的內容。「Health」 ClusterIssue 對象的第三個字段是resourceIds，其返回取決于問題類型。

代碼	訊息	ResourceIds	叢集可復原？	
SUBNET_NOT_FOUND	我們找不到目前與叢集關聯的一個或多個子網路。請呼叫 Amazon EKS update-cluster-config API 更新子網路。	子網路 ID	是	
SECURITY_GROUP_NOT_FOUND	我們找不到目前與叢集關聯的一個或多個安全群組。呼叫 Amazon EKS update-cluster-config API 以更新安全群組	安全群組 ID	是	
IP_NOT_AVAILABLE	與叢集關聯的一個或多個子網路沒有足夠的可用 IP 地址供 Amazon EKS 執行叢集管理操作。釋放子網路中的位址，或使用 Amazon EKS update-cluster-config API 將不同的子網路與叢集建立關聯。	子網路 ID	是	
VPC_NOT_FOUND	我們找不到與叢集關聯的 VPC。您必須刪除並重新建立叢集。	VPC ID	否	
ASSUME_ROLE_ACCESS_DENIED	您的叢集並未使用 Amazon EKS service-linked-role。我們無法擔任與叢集關聯的角色，從而無法執行所需的 Amazon EKS 管理操作。請	叢集 IAM 角色	是	

代碼	訊息	ResourceIds	叢集可復原？
	檢查相應角色是否存在並具有所需的信任政策。		
PERMISSION_ACCESS_DENIED	您的叢集並未使用 Amazon EKS service-linked-role。與叢集關聯的角色未授予 Amazon EKS 執行所需管理操作需要的足夠許可。請檢查連接到叢集角色的政策以及是否套用了任何單獨的拒絕政策。	叢集 IAM 角色	是
ASSUME_ROLE_ACCESS_DENIED_USING_SLR	我們無法假設 Amazon EKS 叢集管理 service-linked-role。請檢查相應角色是否存在並具有所需的信任政策。	Amazon EKS service-linked-role	是
PERMISSION_ACCESS_DENIED_USING_SLR	Amazon EKS 叢集管理 service-linked-role 未授與足夠的許可讓 Amazon EKS 執行所需的管理操作。請檢查連接到叢集角色的政策以及是否套用了任何單獨的拒絕政策。	Amazon EKS service-linked-role	是
OPT_IN_REQUIRED	您的帳戶沒有 Amazon EC2 服務訂閱。請在帳戶設定頁面中更新帳戶訂閱。	N/A	是
STS_REGIONAL_ENDPOINT_DISABLED	相應 STS 區域端點已停用。請啟用相應端點，以讓 Amazon EKS 能夠執行所需的叢集管理操作。	N/A	是
KMS_KEY_DISABLED	與叢集關聯的 AWS KMS 金鑰已停用。請重新啟用該金鑰以復原叢集。	KMS Key Arn	是

代碼	訊息	ResourceIds	叢集可復原？
KMS_KEY_NOT_FOUND	我們找不到與您的叢集相關聯的 AWS KMS 金鑰。您必須刪除並重新建立叢集。	KMS Key ARN	否
KMS_GRANT_REVOKED	與叢集相關聯之 AWS KMS 金鑰的授權會遭到撤銷。您必須刪除並重新建立叢集。	KMS Key Arn	否

Amazon EKS 連接器

您可以利用 Amazon EKS 連接器來註冊任何符合標準的 Kubernetes 叢集，並將其連線至 AWS，再到 Amazon EKS 主控台中將其視覺化。將叢集連線後，您可以在 Amazon EKS 主控台中查看叢集的狀態、組態和工作負載。您可以使用此功能來檢視 Amazon EKS 主控台中的已連線叢集，但無法管理它們。Amazon EKS 連接器需要屬於 Github 上的開放原始碼專案的代理程式。有關其他技術內容，包括常見問題集和故障排除，請參閱 對 Amazon EKS 連接器中的問題進行疑難排解

Amazon EKS 連接器會將下列類型的 Kubernetes 叢集連線至 Amazon EKS。

- 內部部署 Kubernetes 叢集
- 在 Amazon EC2 上執行的自我管理叢集
- 其他雲端供應商的受管叢集

Amazon EKS 連接器考量事項

使用 Amazon EKS 連接器之前，請先詳讀下列注意事項：

- 您必須擁有 Kubernetes 叢集的管理權限，才能將其連線至 Amazon EKS。
- 在連線之前，Kubernetes 叢集必須已具備 Linux 64 位元 (x86) 的工作節點。不支援 ARM 工作節點。
- 您的 Kubernetes 叢集中必須具有工作節點，這些節點可向外存取 `ssm.` 和 `ssmmessages.` Systems Manager 端點。如需詳細資訊，請參閱《AWS 一般參考》中的 Systems Manager 端點。
- 根據預設，一個區域中最多可連線 10 個叢集。您可以透過 service quota 主控台請求增加配額。如需詳細資訊，請參閱請求增加配額。
- 外部 Kubernetes 叢集僅支援 Amazon EKS `RegisterCluster`、`ListClusters`、`DescribeCluster` 以及 `DeregisterCluster` API。
- 您必須擁有以下許可才能註冊叢集：
  - eks:RegisterCluster
  - ssm:CreateActivation
  - ssm:DeleteActivation
  - iam:PassRole
- 您必須擁有以下許可才能取消註冊叢集：

- eks:DeregisterCluster
- ssm:DeleteActivation
- ssm:DeregisterManagedInstance

Amazon EKS 連接器的必要 IAM 角色

使用 Amazon EKS 連接器需要以下兩種 IAM 角色：

- Amazon EKS 連接器服務連結角色是在第一次註冊叢集時建立。
- 您必須建立 Amazon EKS 連接器代理程式 IAM 角色。如需詳細資訊，請參閱 Amazon EKS 連接器 IAM 角色。

若要為 IAM 主體啟用叢集和工作負載檢視許可，您必須將 `eks-connector` 和 Amazon EKS 連接器叢集角色套用至叢集。請遵循 授予 IAM 主體存取權以檢視叢集上的 Kubernetes 資源 中的步驟。

連接外部叢集

您可以使用下列程序中的多種方法，將外部 Kubernetes 叢集連接到 Amazon EKS。此程序包含兩個步驟：向 Amazon EKS 註冊叢集，以及在叢集中安裝 `eks-connector` 代理程式。

> **Important**
>
> 您必須在完成第一個步驟後的 3 天內完成第二個步驟，以免註冊過期。

連接器方法

並非所有安裝代理程式的方法都可以在每一種註冊叢集的方法之後使用。下列表格列出了每一種註冊方法，以及可以使用哪些方法來安裝代理程式。

步驟	方法		
註冊叢集	AWS Management Console	AWS Command Line Interface	`eksctl`

步驟	方法		
安裝 代理程式	Helm，YAML 清單檔案	Helm，YAML 清單檔案	YAML 清單檔案

必要條件

- 確認已建立 Amazon EKS 連接器代理程式角色。請遵循 建立 Amazon EKS 連接器代理程式角色 中的步驟。
- 您必須擁有以下許可才能註冊叢集：
  - `eks:RegisterCluster`
  - `ssm:CreateActivation`
  - `ssm:DeleteActivation`
  - `iam:PassRole`

步驟 1：註冊叢集

AWS CLI

必要條件

- 必須安裝 AWS CLI。若要安裝或升級，請參閱安裝 AWS CLI。

向 AWS CLI 註冊叢集

- 對於連接器組態，請指定 Amazon EKS 連接器代理程式 IAM 角色。如需詳細資訊，請參閱Amazon EKS 連接器的必要 IAM 角色。

```
aws eks register-cluster \
     --name my-first-registered-cluster \
     --connector-config roleArn=arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole,provider="OTHER" \
     --region aws-region
```

範例輸出如下。

```
{
    "cluster": {
        "name": "my-first-registered-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-first-registered-
cluster",
        "createdAt": 1627669203.531,
        "ConnectorConfig": {
            "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
            "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
            "activationExpiry": 1627672543.0,
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole"
        },
        "status": "CREATING"
    }
}
```

您會在接下來的步驟中使用 `aws-region`、`activationId` 和 `activationCode` 值。

AWS Management Console

向主控台註冊 Kubernetes 叢集。

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇 Add cluster (新增叢集)，然後選取 Register (註冊) 以顯示組態頁面。
3. 在 Configure cluster (設定叢集) 區段上，填寫下列欄位：
   - Name (名稱) – 叢集的唯一名稱。
   - Provider (供應商) – 選擇以顯示 Kubernetes 叢集供應商的下拉式清單。若沒有相符的特定供應商，請選取其他。
   - EKS Connector role (EKS 連接器角色)：選取用於連接叢集的角色。
4. 選取 Register cluster (註冊叢集)。
5. 系統隨即會顯示叢集概觀頁面。如果您想使用 Helm Chart，請複製 `helm install` 命令並繼續下一個步驟。如果您想要使用 YAML 清單檔案，請選擇下載 YAML 檔案，將清單檔案檔案下載至本機磁碟機。

> **Important**
>
> - 這是您複製 `helm install` 命令或下載此檔案的唯一機會。請勿離開此頁面，以免連結無法存取；若已離開此頁，則必須取消註冊叢集並從頭開始執行步驟。
> - 僅能針對已註冊叢集使用命令或清單檔案檔案一次。如果從 Kubernetes 叢集刪除資源，您必須重新註冊叢集並取得新的清單檔案檔案。

繼續執行下一步，將清單檔案檔案套用到 Kubernetes 叢集。

eksctl

必要條件

- 必須安裝 `eksctl` 版本 `0.68` 或更新版本。若要將其安裝或升級，請參閱 Amazon EKS 入門：`eksctl`。

向 **`eksctl`** 註冊叢集

1. 透過提供名稱、供應商和區域來註冊叢集。

```
eksctl register cluster --name my-cluster --provider my-provider --
region region-code
```

輸出範例：

```
2021-08-19 13:47:26 [#]  creating IAM role "eksctl-20210819194112186040"
2021-08-19 13:47:26 [#]  registered cluster "<name>" successfully
2021-08-19 13:47:26 [#]  wrote file eks-connector.yaml to <current directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-clusterrole.yaml to <current
 directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-console-dashboard-full-access-
group.yaml to <current directory>
2021-08-19 13:47:26 [!]  note: "eks-connector-clusterrole.yaml" and "eks-
connector-console-dashboard-full-access-group.yaml" give full EKS Console access
 to IAM identity "<aws-arn>", edit if required; read https://eksctl.io/usage/
eks-connector for more info
```

```
2021-08-19 13:47:26 [#]  run `kubectl apply -f eks-connector.yaml,eks-connector-
clusterrole.yaml,eks-connector-console-dashboard-full-access-group.yaml` before
 expiry> to connect the cluster
```

這會在您的本機電腦上建立檔案。這些檔案必須在 3 天內套用至外部叢集，否則註冊將會過期。

2. 在可存取叢集的終端機中，套用 `eks-connector-binding.yaml` 檔案：

```
kubectl apply -f eks-connector-binding.yaml
```

步驟 2：安裝 **eks-connector** 代理程式

Helm chart

1. 如果您在上一個步驟中使用 AWS CLI，請分別以 `activationId` 和 `activationCode` 值來取代下列命令中的 `ACTIVATION_CODE` 和 `ACTIVATION_ID`。將 `aws-region` 替換為您在上一個步驟中使用的 AWS 區域。接著執行命令，在註冊的叢集上安裝 `eks-connector` 代理程式：

```
$ helm install eks-connector \
  --namespace eks-connector \
  oci://public.ecr.aws/eks-connector/eks-connector-chart \
  --set eks.activationCode=ACTIVATION_CODE \
  --set eks.activationId=ACTIVATION_ID \
  --set eks.agentRegion=aws-region
```

如果您在上一個步驟中使用 AWS Management Console，請使用您從上一個步驟複製的命令 (已填入這些值)。

2. 檢查已安裝 `eks-connector` 部署的健康狀態，並等待 Amazon EKS 中已註冊叢集的狀態成為 `ACTIVE`。

YAML manifest

將 Amazon EKS 連接器清單檔案檔案套用至 Kubernetes 叢集，以便完成連線。若要執行此動作，您必須使用先前描述的方法。如果未於三天內套用清單檔案，Amazon EKS 連接器註冊將到期。如果叢集連線過期，則必須先取消註冊叢集，才能再次連接叢集。

1. 下載 Amazon EKS 連接器 YAML 檔案。

```
curl -O https://amazon-eks.s3.us-west-2.amazonaws.com/eks-connector/manifests/
eks-connector/latest/eks-connector.yaml
```

2. 編輯 Amazon EKS 連接器 YAML 檔案，以來自上一個步驟輸出的 `aws-region`、`activationId` 和 `activationCode` 來取代 `%AWS_REGION%`、`%EKS_ACTIVATION_ID%`、`%EKS_ACTIVATION_CODE%` 的所有參考。

   下列範例命令可以取代這些值。

```
sed -i "s~%AWS_REGION%~$aws-region~g; s~%EKS_ACTIVATION_ID
%~$EKS_ACTIVATION_ID~g; s~%EKS_ACTIVATION_CODE%~$(echo -n $EKS_ACTIVATION_CODE |
 base64)~g" eks-connector.yaml
```

   > **Important**
   >
   > 確認您的啟用代碼格式為 Base64。

3. 在可存取叢集的終端機中，您可以執行以下命令來套用已更新的清單檔案檔案：

```
kubectl apply -f eks-connector.yaml
```

4. 將 Amazon EKS 連接器清單檔案，以及角色繫結 YAML 檔案套用至 Kubernetes 叢集後，請確認叢集已成功連線。

```
aws eks describe-cluster \
     --name "my-first-registered-cluster" \
     --region AWS_REGION
```

   輸出應該包含 `status=ACTIVE`。

5. (選用) 將標籤新增至您的叢集。如需詳細資訊，請參閱為您的 Amazon EKS 資源加上標籤。

後續步驟

如果您對這些步驟有任何問題，請參閱 對 Amazon EKS 連接器中的問題進行疑難排解。

若要授予其他 IAM 主體對 Amazon EKS 主控台的存取權，以檢視已連接叢集中的 Kubernetes 資源，請參閱 授予 IAM 主體存取權以檢視叢集上的 Kubernetes 資源。

授予 IAM 主體存取權以檢視叢集上的 Kubernetes 資源

授予 IAM 主體存取 Amazon EKS 主控台的許可，以便他們檢視已連線叢集上所執行的 Kubernetes 資源等相關資訊。

必要條件

您用來存取 AWS Management Console 的 IAM 主體必須符合以下要求：

- 它必須具有 `eks:AccessKubernetesApi` IAM 許可。
- Amazon EKS 連接器服務帳戶可以模擬叢集中的 IAM 主體。這可讓 Amazon EKS Connector 將 IAM 主體映射至 Kubernetes 使用者。

建立和套用 Amazon EKS 叢集角色

1. 下載 `eks-connector` 叢集角色範本。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-clusterrole.yaml
```

2. 編輯叢集角色範本 YAML 檔案。將對 `%IAM_ARN%` 的參考取代為 IAM 主體的 Amazon Resource Name (ARN)。
3. 將 Amazon EKS 連接器叢集角色 YAML 套用至 Kubernetes 叢集。

```
kubectl apply -f eks-connector-clusterrole.yaml
```

若要讓 IAM 主體在 Amazon EKS 主控台中檢視 Kubernetes 資源，該主體必須將這些工作負載與 Kubernetes `role` 或 `clusterrole` 相關聯，且須擁有這些資源的必要讀取許可。如需詳細資訊，請參閱 Kubernetes 文件中的使用 RBAC 授權。

設定存取連接叢集的 IAM 主體

1. 您可以下載照下範例清單檔案來分別建立 `clusterrole` 和 `clusterrolebinding` 或 `role` 和 `rolebinding`：

檢視所有命名空間中的 Kubernetes 資源

`eks-connector-console-dashboard-full-access-clusterrole` 叢集角色可讓您存取所有可在主控台中視覺化的命名空間和資源。您可以變更 `role`、`clusterrole` 及其相應繫結的名稱，然後再將其套用至叢集。使用以下命令下載範例檔案。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-console-dashboard-full-access-group.yaml
```

檢視特定命名空間中的 Kubernetes 資源

此檔案中的命名空間為 `default`，因此如果想指定不同的命名空間，請在將其套用至叢集之前編輯該檔案。使用以下命令來下載範例檔案。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-console-dashboard-restricted-access-group.yaml
```

2. 編輯 YAML 檔案的完整存取權或受限存取權，會將 `%IAM_ARN%` 的參考取代為 IAM 主體的 Amazon Resource Name (ARN)。
3. 將 YAML 檔案的完整存取權或受限存取權套用到 Kubernetes 叢集。以您自己的值取代 YAML 檔案值。

```
kubectl apply -f eks-connector-console-dashboard-full-access-group.yaml
```

若要檢視已連接叢集中的 Kubernetes 資源，請參閱 檢視 Kubernetes 資源。Resources (資源) 索引標籤上的某些資源類型的資料不適用於已連接的叢集。

取消註冊叢集

如果連接叢集已使用完畢，即可將其取消註冊。取消註冊後，Amazon EKS 主控台中便不再顯示該叢集。

您必須擁有以下許可，才能呼叫 deregisterCluster API：

- `eks:DeregisterCluster`
- `ssm:DeleteActivation`

- `ssm:DeregisterManagedInstance`

此程序包含兩個步驟：向 Amazon EKS 取消註冊叢集，以及在叢集中取消安裝 eks-connector 代理程式。

若要取消註冊 Kubernetes 叢集

AWS CLI

必要條件

- 必須安裝 AWS CLI。若要安裝或升級，請參閱安裝 AWS CLI。
- 確認已建立 Amazon EKS 連接器代理程式角色。

取消註冊已連接叢集。

```
aws eks deregister-cluster \
    --name my-cluster \
    --region region-code
```

AWS Management Console

1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
2. 選擇 Clusters (叢集)。
3. 在 Clusters (叢集) 頁面上，選取已連接叢集，然後選取 Deregister (取消註冊)。
4. 確認您要取消註冊叢集。

eksctl

必要條件

- 必須安裝 `eksctl` 版本 `0.68` 或更新版本。若要將其安裝或升級，請參閱 Amazon EKS 入門：`eksctl`。
- 確認已建立 Amazon EKS 連接器代理程式角色。

使用 **eksctl** 將叢集取消註冊

- 對於連接器組態，請指定 Amazon EKS 連接器代理程式 IAM 角色。如需詳細資訊，請參閱Amazon EKS 連接器的必要 IAM 角色。

```
eksctl deregister cluster --name my-cluster
```

清除 Kubernetes 叢集中的資源

Helm

- 執行下列命令以解除安裝代理程式。

```
helm -n eks-connector uninstall eks-connector
```

YAML manifest

1. 從 Kubernetes 叢集中刪除 Amazon EKS 連接器 YAML 檔案。

```
kubectl delete -f eks-connector.yaml
```

2. 若您已建立 `clusterrole` 或 `clusterrolebindings` 來為其他 IAM 主體提供叢集存取權，請將其從 Kubernetes 叢集中刪除。

對 Amazon EKS 連接器中的問題進行疑難排解

本主題涵蓋您在使用 Amazon EKS 連接器時可能遇到的一些常見錯誤，包括有關如何解決錯誤和因應措施的說明。

基本疑難排解

本節說明不清楚問題時進行診斷的步驟。

檢查 Amazon EKS 連接器狀態

檢查 Amazon EKS 連接器狀態。

```
kubectl get pods -n eks-connector
```

檢查 Amazon EKS 連接器日誌

Amazon EKS 連接器 Pod 由三個容器組成。若要擷取所有這些容器的完整日誌，以便您可以檢查它們，請執行以下命令：

- `connector-init`

```
kubectl logs eks-connector-0 --container connector-init -n eks-connector
kubectl logs eks-connector-1 --container connector-init -n eks-connector
```

- `connector-proxy`

```
kubectl logs eks-connector-0 --container connector-proxy -n eks-connector
kubectl logs eks-connector-1 --container connector-proxy -n eks-connector
```

- `connector-agent`

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
kubectl exec eks-connector-1 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
```

取得有效的叢集名稱

Amazon EKS 叢集由單個 AWS 帳戶和 AWS 區域內的 `clusterName` 唯一識別。如果您在 Amazon EKS 中有多個連線的叢集，您可以確認目前 Kubernetes 叢集註冊了哪個 Amazon EKS 叢集。為此，請輸入下列內容來找出目前叢集的 `clusterName`。

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
kubectl exec eks-connector-1 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
```

其他命令

擷取所需資訊以進行問題疑難排解時，以下命令很有用。

- 使用以下命令收集 Amazon EKS 連接器中 Pods 使用的映像。

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.containers[*].image}"
 | tr -s '[[:space:]]' '\n'
```

- 使用以下命令判斷正在執行 Amazon EKS 連接器的節點名稱。

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.nodeName}" | tr -s
 '[[:space:]]' '\n'
```

- 執行以下命令，以取得 Kubernetes 用戶端和伺服器版本。

```
kubectl version
```

- 執行以下命令，以取得有關節點的資訊。

```
kubectl get nodes -o wide --show-labels
```

Helm 問題：403 禁止

如果您在執行 helm 安裝命令時收到下列錯誤：

```
Error: INSTALLATION FAILED: unexpected status from HEAD request to https://
public.ecr.aws/v2/eks-connector/eks-connector-chart/manifests/0.0.6: 403 Forbidden
```

您可以執行下列行來修復錯誤：

```
docker logout public.ecr.aws
```

主控台錯誤：叢集停留在待定狀態

如果叢集在您註冊後卡在 Amazon EKS 主控台上的Pending狀態，可能是因為 Amazon EKS 連接器尚未成功將叢集連接到 AWS 叢集。對於已註冊的叢集，Pending 狀態表示連接尚未成功建立。若要解決此問題，請確定您已將清單檔案應用於目標 Kubernetes 叢集。如果您將其套用於叢集，但叢集仍處於 Pending 狀態，則 eks-connector statefulset 很可能無法正常運作。若要排除此問題，請參閱本主題中的 Amazon EKS 連接器 Pods 正處於損毀循環 。

主控台錯誤：User “system:serviceaccount:eks-connector:eks-connector” can't impersonate resource “users” in API group “” 在叢集範圍

Amazon EKS 連接器使用 Kubernetes 使用者模擬以代表來自 AWS Management Console的 IAM 主體採取動作。從 AWS `eks-connector`服務帳戶存取 Kubernetes API 的每個主體都必須獲得權限，才能以 IAM ARN 作為其KubernetesKubernetes使用者名稱來模擬對應使用者。在以下範例中，IAM ARN 映射到 Kubernetes 使用者。

- AWS 帳*111122223333*戶*john*中的 IAM 使用者會對應至使Kubernetes用者。IAM 最佳實務建議您將許可授予角色而非使用者。

  ```
  arn:aws:iam::111122223333:user/john
  ```

- AWS 帳戶*admin*中的 IAM 角色*111122223333*對應至Kubernetes使用者：

  ```
  arn:aws:iam::111122223333:role/admin
  ```

  結果是 IAM 角色 ARN，而不是 AWS STS 工作階段 ARN。

如需如何設定 `ClusterRole` 和 `ClusterRoleBinding` 以授予 `eks-connector` 服務帳戶權限來模擬映射使用者的說明，請參閱 授予 IAM 主體存取權以檢視叢集上的 Kubernetes 資源。請確保在範本中將 `%IAM_ARN%` 取代為 AWS Management Console IAM 主體的 IAM ARN。

主控台錯誤：[...] is forbidden: User [...] cannot list resource “[...] in API group” 在叢集範圍

請考量下列問題。Amazon EKS 連接器已成功模擬目標叢集中的請求 AWS Management Console IAM 主體。Kubernetes但是，模擬主體沒有針對 Kubernetes API 操作的 RBAC 許可。

若要解決此問題，有兩種方法可將許可授予其他使用者。如果您先前透過 Helm Chart 安裝了 eks-connector，您可以透過執行下列命令輕鬆授予使用者存取權。以 IAM 角色的 ARN 清單取代 `userARN1` 和 `userARN2` 以提供存取權來檢視 Kubernetes 資源：

```
helm upgrade eks-connector oci://public.ecr.aws/eks-connector/eks-connector-chart \
    --reuse-values \
    --set 'authentication.allowedUserARNs={userARN1,userARN2}'
```

或者，身為叢集管理員，向個別 Kubernetes 使用者授予適當層級的 RBAC 權限。如需詳細資訊和範例，請參閱 授予 IAM 主體存取權以檢視叢集上的 Kubernetes 資源。

主控台錯誤：Amazon EKS 無法與您的 Kubernetes 叢集 API 伺服器通訊。叢集必須處於作用中狀態，才能成功連接。請過幾分鐘後再試。

如果 Amazon EKS 服務無法與目標叢集中的 Amazon EKS 連接器通訊，則可能是由於以下原因之一：

- 目標叢集中的 Amazon EKS 連接器無法正常運作。
- 目標叢集與 AWS 區域之間連接不良或連接中斷。

若要解決此問題，請查看 Amazon EKS 連接器日誌。如果未看到 Amazon EKS 連接器的錯誤，請在幾分鐘後重試連線。如果您經常遇到目標叢集的高延遲或間歇性連線，請考慮將叢集重新註冊到離您較近的叢集。 AWS 區域

Amazon EKS 連接器 Pods 正處於損毀循環

導致 Amazon EKS 連接器 Pod 進入 `CrashLoopBackOff` 狀態的原因有很多。此問題可能涉及 `connector-init` 容器。檢查 Amazon EKS 連接器 Pod 的狀態。

```
kubectl get pods -n eks-connector
```

範例輸出如下。

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:CrashLoopBackOff   1          7s
```

如果您的輸出與之前的輸出相似，請參閱 檢查 Amazon EKS 連接器日誌 以排除問題。

Failed to initiate eks-connector: InvalidActivation

當您第一次啟動 Amazon EKS 連接器時，它會使用 Amazon Web Services 註冊 `activationId` 和 `activationCode`。註冊可能會失敗，這可能會導致 `connector-init` 容器損毀，產生與以下錯誤相似的錯誤。

```
F1116 20:30:47.261469       1 init.go:43] failed to initiate eks-connector:
 InvalidActivation:
```

若要對此問題進行疑難排解，請考慮以下原因和建議的修正：

- 註冊可能會失敗，因為 activationId 和 activationCode 不在清單檔案檔案中。如果是這種情況，請確保它們是從 RegisterCluster API 操作傳回的正確值，並且 activationCode 位於清單檔案檔案中。activationCode 新增至 Kubernetes 秘密中，因此它必須採用 base64 編碼。如需詳細資訊，請參閱 步驟 1：註冊叢集。
- 註冊可能因啟用過期而失敗。因為出於安全原因，您必須在註冊叢集後的 3 天內啟用 Amazon EKS 連接器。若要解決此問題，請確保在過期日期和時間之前將 Amazon EKS 連接器清單檔案套用至目標 Kubernetes 叢集。若要確認您的啟用過期日期，請呼叫 DescribeCluster API 操作。

```
aws eks describe-cluster --name my-cluster
```

在以下範例回應中，過期日期和時間記錄為 2021-11-12T22:28:51.101000-08:00。

```
{
    "cluster": {
        "name": "my-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-cluster",
        "createdAt": "2021-11-09T22:28:51.449000-08:00",
        "status": "FAILED",
        "tags": {
        },
        "connectorConfig": {
            "activationId": "00000000-0000-0000-0000-000000000000",
            "activationExpiry": "2021-11-12T22:28:51.101000-08:00",
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/my-connector-role"
        }
    }
}
```

如果超過 activationExpiry，請取消註冊叢集，並將其重新註冊。這樣做會產生一個新的啟用。

叢集節點遺漏對外連線

若要正常運作，Amazon EKS 連接器需要多個 AWS 端點的對外連線。如果沒有目標 AWS 區域的對外連線，則無法連接私有叢集。若要解決此問題，您必須新增必要的對外連線。如需連接器要求的相關資訊，請參閱 Amazon EKS 連接器考量事項。

Amazon EKS 連接器 Pods 處於 **ImagePullBackOff** 狀態

如果您執行 `get pods` 命令，並且 Pods 處於 `ImagePullBackOff` 狀態，則它們無法正常運作。如果 Amazon EKS 連接器 Pods 處於 `ImagePullBackOff` 狀態，則它們無法正常運作。檢查 Amazon EKS 連接器 Pods 的狀態。

```
kubectl get pods -n eks-connector
```

範例輸出如下。

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:ImagePullBackOff   0          4s
```

預設的 Amazon EKS 連接器清單檔案檔案引用來自 Amazon ECR Public Gallery 的映像。目標 Kubernetes 叢集可能無法從 Amazon ECR Public Gallery 提取映像。解決 Amazon ECR Public Gallery 映像提取問題，或者考慮選擇在私有容器登錄檔中鏡像映像。

常見問答集

問：Amazon EKS 連接器背後的基礎技術如何運作？

答：Amazon EKS 連接器是基於 AWS Systems Manager (Systems Manager) 代理程式。Amazon EKS 連接器作為您的 Kubernetes 叢集上的 `StatefulSet` 執行。它建立叢集的 API 伺服器與 Amazon Web Services 之間的連線並代理通訊。這樣做是為了在 Amazon EKS 主控台中顯示叢集資料，直到您將叢集從 AWS 中斷連線。Systems Manager 代理程式是開放原始碼專案。如需有關此專案的詳細資訊，請參閱 GitHub 專案頁面。

問：我有一個想要連接的內部部署 Kubernetes 叢集。我是否需要開啟防火牆連接埠才能連接此叢集？

答：您不需要開啟任何防火牆連接埠。Kubernetes 叢集只需要 AWS 區域 的對外連線。AWS 服務永遠不會存取您的內部部署網路中的資源。Amazon EKS 連接器會在您的叢集上執行，並初始化與 AWS 的連線。叢集註冊完成後，僅在您從 Amazon EKS 主控台啟動需要叢集上 Kubernetes API 伺服器提供資訊的動作後，AWS 才會向 Amazon EKS 連接器發出命令。

問：Amazon EKS 連接器將哪些資料從我的叢集傳送到 AWS？

答：Amazon EKS 連接器會傳送您的叢集在 AWS 上註冊所需的技術資訊。它還會針對客戶請求的 Amazon EKS 主控台功能傳送叢集和工作負載中繼資料。僅在您從必須將資料傳送至 AWS 的 Amazon

EKS 主控台或 Amazon EKS API 啟動動作時，Amazon EKS 連接器才會收集或傳送此資料。除了 Kubernetes 版本編號之外，AWS 在預設情況下不會儲存任何資料。它只有在您授權時才會存放資料。

問：我是否可以連接 AWS 區域 外部的叢集?

答：您可以將來自任何位置的叢集連接到 Amazon EKS。此外，您的 Amazon EKS 服務可以位於任何 AWS 公共商業 AWS 區域 中。這適用於從您的叢集到目標 AWS 區域 的有效網路連接。建議您挑選離叢集位置最近的 AWS 區域，以實現 UI 效能最佳化。例如，如果您有在東京執行的叢集，請將叢集連接至東京的 AWS 區域 (也就是 `ap-northeast-1` AWS 區域)，以實現低延遲。您可以將任何位置的叢集連接到任何公共商業 AWS 區域 (中國或 GovCloud AWS 區域 除外) 中的 Amazon EKS。

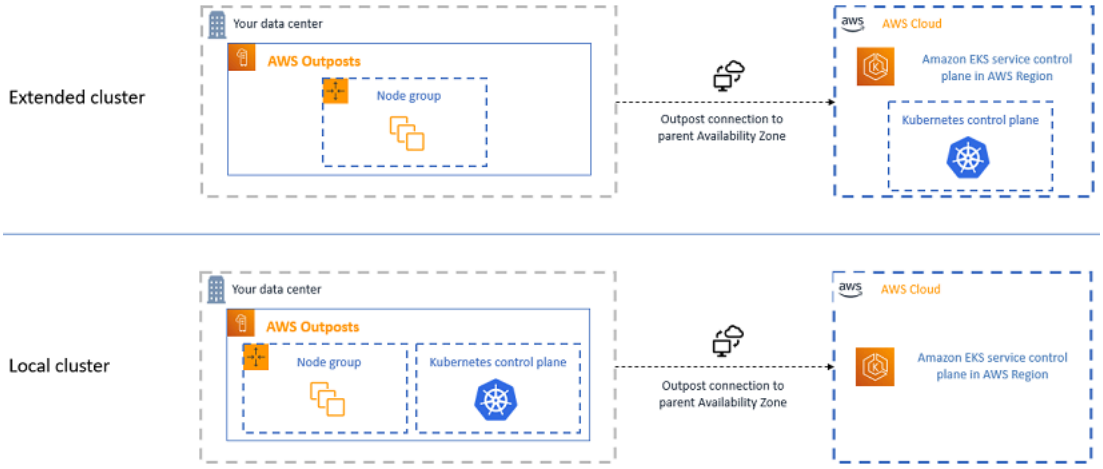
AWS Outposts 上的 Amazon EKS

您可以使用 Amazon EKS 在 AWS Outposts 上執行內部部署 Kubernetes 應用程式。您可使用以下方式在 Outpost 上部署 Amazon EKS：

- 擴充叢集：執行 AWS 區域 中的 Kubernetes 控制平面，以及 Outpost 上的節點。
- 本機叢集：執行 Outpost 上的 Kubernetes 控制平面和節點。

針對兩個部署選項，Kubernetes 控制平面會受 AWS 完全管理。您可以使用在雲端中使用的相同 Amazon EKS API、工具和主控台，在 Outpost 上建立和執行 Amazon EKS。

下圖顯示這些部署選項。



使用每個部署選項的時機

本機叢集和擴充叢集皆為一般用途的部署選項，可用於多種應用程式。

使用本機叢集，您可以在 Outposts 上本機執行整個 Amazon EKS 叢集。此選項可降低因網路暫時與雲端中斷連線而可能導致的應用程式停機風險。這些網路連線中斷可能是因光纖切斷或天氣事件引起的。由於整個 Amazon EKS 叢集會在 Outposts 本機執行，因此仍可使用應用程式。您可在網路與雲端中斷連線期間執行叢集操作。如需更多詳細資訊，請參閱 為網路連線中斷做好準備。如果您擔心從 Outposts 至父項 AWS 區域 的網路連線品質，且需要在網路連線中斷期間實現高可用性，請使用本機叢集部署選項。

由於 Kubernetes 控制平面會在父 AWS 區域 中執行，因此您可以使用擴充的叢集節省 Outpost 的容量。如果您可以投資於從 Outpost 至 AWS 區域 的可靠備援網路連線，則此選項非常適合。此選項的網路連線品質十分重要。Kubernetes 處理 Kubernetes 控制平面和節點之間的網路連線中斷的方

式可能會導致應用程式停機。如需有關 Kubernetes 行為的詳細資訊，請參閱 Kubernetes 文件中的 Scheduling, Preemption, and Eviction (排程、先佔和移出)。

比較部署選項

下表會比較這兩個選項之間的差異。

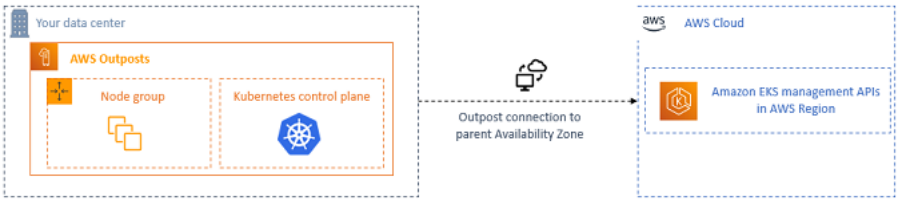
功能	擴充叢集	本機叢集
Kubernetes 控制平面位置	AWS 區域	Outpost
Kubernetes 控制平面帳戶	AWS 帳戶	您的帳戶
區域可用性	請參閱 Service endpoints (服務端點)。	美國東部 (俄亥俄)、美國東部 (維吉尼亞北部)、美國西部 (加利佛尼亞北部)、美國西部 (奧勒岡)、亞太區域 (首爾)、亞太區域 (新加坡)、亞太區域 (雪梨)、亞太區域 (東京)、加拿大 (中部)、歐洲 (法蘭克福)、歐洲 (愛爾蘭)、歐洲 (倫敦)、中東 (巴林) 和南美洲 (聖保羅)
Kubernetes 次要版本	支援的 Amazon EKS 版本。	支援的 Amazon EKS 版本。
平台版本	請參閱 Amazon EKS 平台版本	請參閱 Amazon EKS 本機叢集平台版本
Outpost 外形規格	Outpost 機架	Outpost 機架
使用者介面	AWS Management Console、AWS CLI、Amazon EKS API、`eksctl`、AWS CloudFormation 和 Terraform	AWS Management Console、AWS CLI、Amazon EKS API、`eksctl`、AWS CloudFormation 和 Terraform
受管政策	AmazonEKSClusterPolicy 和 AmazonEKSServiceRolePolicy	AmazonEKSLocalOutpostClusterPolicy 和 AmazonEKSLocalOutpostServiceRolePolicy

功能	擴充叢集	本機叢集
叢集 VPC 和子網路	請參閱 Amazon EKS VPC 與子網要求和注意事項	請參閱 Amazon EKS 本機叢集 VPC 與子網路要求和考量事項
叢集端點存取	公有或私有，或兩者兼具	僅限私有
Kubernetes API 伺服器身分驗證	AWS Identity and Access Management (IAM) 和 OIDC	IAM 和 x.509 憑證
節點類型	僅限自我管理	僅限自我管理
節點運算類型	Amazon EC2 隨需	Amazon EC2 隨需
節點儲存類型	Amazon EBS gp2 和本機 NVMe SSD	Amazon EBS gp2 和本機 NVMe SSD
Amazon EKS 最佳化 AMI	Amazon Linux、Windows 和 Bottlerocket	僅限 Amazon Linux
IP 版本	僅限 IPv4	僅限 IPv4
附加元件	Amazon EKS 附加元件，或自我管理附加元件	僅限自我管理附加元件
預設容器網路介面	Amazon VPC CNI plugin for Kubernetes	Amazon VPC CNI plugin for Kubernetes
Kubernetes 控制平面日誌	Amazon CloudWatch Logs	Amazon CloudWatch Logs
負載平衡	僅限使用 AWS Load Balancer Controller 佈建 Application Load Balancer (無 Network Load Balancer)	僅限使用 AWS Load Balancer Controller 佈建 Application Load Balancer (無 Network Load Balancer)
秘密封套加密	請參閱 在現有叢集上啟用密碼加密	不支援
服務帳戶的 IAM 角色	請參閱 服務帳戶的 IAM 角色	不支援

功能	擴充叢集	本機叢集
故障診斷	請參閱 Amazon EKS 故障診斷	請參閱 AWS Outposts 上 Amazon EKS 本機叢集疑難排解

AWS Outposts 上的 Amazon EKS 本機叢集

您可以使用本機叢集在 AWS Outposts 上本機執行整個 Amazon EKS 叢集。這有助於降低因網路暫時與雲端中斷連線而可能導致的應用程式停機風險。這些連線中斷可能是因光纖切斷或天氣事件引起的。由於整個 Kubernetes 叢集會在 Outpost 本機執行，因此仍可使用應用程式。您可在網路與雲端中斷連線期間執行叢集操作。如需更多詳細資訊，請參閱 為網路連線中斷做好準備。下圖顯示本機叢集部署。



本機叢集通常可與 Outpost 機架搭配使用。

支援 AWS 區域

您可在以下 AWS 區域 中建立本機叢集：美國東部 (俄亥俄)、美國東部 (維吉尼亞北部)、美國西部 (加利佛尼亞北部)、美國西部 (奧勒岡)、亞太區域 (首爾)、亞太區域 (新加坡)、亞太區域 (雪梨)、亞太區域 (東京)、加拿大 (中部)、歐洲 (法蘭克福)、歐洲 (愛爾蘭)、歐洲 (倫敦)、中東 (巴林) 和南美洲 (聖保羅)。如需有關支援功能的詳細資訊，請參閱比較部署選項。

主題

- 在 Outpost 上建立本機叢集
- Amazon EKS 本機叢集平台版本
- Amazon EKS 本機叢集 VPC 與子網路要求和考量事項
- 為網路連線中斷做好準備
- 容量考量事項
- AWS Outposts 上 Amazon EKS 本機叢集疑難排解

在 Outpost 上建立本機叢集

本主題概述了在 Outpost 上執行本機叢集時要考量的事項。本主題也提供如何在 Outpost 上部署本機叢集的指示。

考量事項

> **Important**
>
> - 這些考量事項並不是相關 Amazon EKS 文件的重複內容。如果其他 Amazon EKS 文件主題與此處的考量衝突，則請遵循此處的考量事項。
> - 這些考量事項隨時會變更，而且可能會經常變更。因此，我們建議您定期檢閱此主題。
> - 許多考量事項與在 AWS 雲端上建立叢集的考量事項不同。

- 本機叢集僅支援 Outpost 機架。單一本機叢集可以在包含單一邏輯 Outpost 的多個實體 Outpost 機架上執行。單一本機叢集無法跨多個邏輯 Outpost 執行。每個邏輯 Outpost 都有一個單一的 Outpost ARN。
- 本機叢集在 Outpost 上執行和管理您帳戶中的 Kubernetes 控制平面。您無法在 Kubernetes 控制平面執行個體上執行工作負載，或修改 Kubernetes 控制平面元件。這些節點由 Amazon EKS 服務管理。執行自動 Amazon EKS 管理動作 (例如修補) 後，對 Kubernetes 控制平面的變更不會持續存在。
- 本機叢集支援自我管理的附加元件和自我管理的 Amazon Linux 節點群組。Amazon VPC CNI plugin for Kubernetes、`kube-proxy` 及 CoreDNS 附加元件會自動安裝在本機叢集上。
- 本機叢集需要在 Outpost 上使用 Amazon EBS。您的 Outpost 必須具有可用於 Kubernetes 控制平面儲存的 Amazon EBS。
- 本機叢集會在 Outpost 上使用 Amazon EBS。您的 Outpost 必須具有可用於 Kubernetes 控制平面儲存的 Amazon EBS。Outpost 僅支援 Amazon EBS gp2 磁碟區。
- 如果使用 Amazon EBS CSI 驅動程式，可支援 Amazon EBS 所支援的 Kubernetes `PersistentVolumes`。

必要條件

- 熟悉 Outpost 部署選項、容量考量事項 及 Amazon EKS 本機叢集 VPC 與子網路要求和考量事項。
- 現有的 Outpost。如需詳細資訊，請參閱 什麼是 AWS Outposts。

- 已在電腦或 AWS CloudShell上的 kubectl 安裝命令列工具。版本可以與您的叢集 Kubernetes 版本相同，或是為最多比該版本更舊一版或更新一版的次要版本。例如，如果您的叢集版本為 1.29，則可以搭配使用 kubectl 1.28、1.29 或 1.30 版。若要安裝或升級 kubectl，請參閱 安裝或更新 kubectl。
- 您裝置上安裝和設定的 AWS Command Line Interface (AWS CLI) 的版本1.27.160或更新版本、版本或更新版本或更新版本或更新版本 AWS CloudShell。2.12.3若要檢查您目前的版本，請使用 **aws --version | cut -d / -f2 | cut -d ' ' -f1**。如 yum、apt-get 或適用於 macOS 的 Homebrew 等套件管理工具通常比最新版本的 AWS CLI落後數個版本之多。若要安裝最新版本，請參閱《AWS Command Line Interface 使用者指南》中的安裝、更新和解除安裝 AWS CLI以及使用 aws configure 的快速組態。安裝在中的 AWS CLI 版本也 AWS CloudShell 可能是最新版本之後的幾個版本。若要進行更新，請參閱《AWS CloudShell 使用者指南》中的〈安裝 AWS CLI 到主目錄〉。
- 具有 create 和 describe Amazon EKS 叢集之許可的 IAM 主體 (使用者或角色)。如需詳細資訊，請參閱 在 Outpost 上建立本機 Kubernetes 叢集 及 所有叢集的清單或描述。

建立本機 Amazon EKS 叢集後，系統會永久新增建立叢集的 IAM 主體。主體會專門新增至 Kubernetes RBAC 授權資料表作為管理員。此實體具有 system:masters 許可。此實體的標識在您的叢集組態中不可見。因此，請務必注意建立叢集的實體，並確保永遠不會將其刪除。一開始，只有建立伺服器的主體可以使用 kubectl 對 Kubernetes API 伺服器進行呼叫。如果您使用主控台建立叢集，請確定在叢集上執行kubectl命令時， AWS SDK 認證鏈中有相同的 IAM 登入資料。建立叢集之後，您可以授予其他 IAM 主體存取您的叢集。

建立 Amazon EKS 本機叢集

您可以使用 eksctl、 AWS Management Console、AWS CLI、Amazon EKS API、AWS 開發套件、AWS CloudFormation 或 Terraform 來建立本機叢集。

1. 建立本機叢集。

   eksctl

   先決條件

   已在裝置或 AWS CloudShell上安裝版本 0.183.0 或更新版本的 eksctl 命令列工具。如需有關安裝或更新 eksctl 的指示，請參閱 eksctl 文件中的安裝一節。

使用 **eksctl** 建立您的叢集

1. 將隨後的內容複製到您的裝置。取代以下值，然後執行修改後的命令來建立 outpost-control-plane.yaml 檔案：
   - 使用您希望在其中建立叢集的受支援 AWS 區域取代 *region-code*。
   - 使用您的叢集名稱取代 *my-cluster*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - 將*vpc-ExampleID1*和*subnet-ExampleID1*取代為現有 VPC 和子網路的 ID。VPC 和子網路必須符合 Amazon EKS 本機叢集 VPC 與子網路要求和考量事項 的要求。
   - 替換*uniqueid*為您的前哨的 ID。
   - 使用您 Outpost 上可用的執行個體類型取代 *m5.large*。選擇執行個體類型之前，請先參閱容量考量事項。部署了三個控制平面執行個體。您無法變更此數字。

```
cat >outpost-control-plane.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "1.24"

vpc:
  clusterEndpoints:
    privateAccess: true
  id: "vpc-vpc-ExampleID1"
  subnets:
    private:
      outpost-subnet-1:
        id: "subnet-subnet-ExampleID1"

outpost:
  controlPlaneOutpostARN: arn:aws:outposts:region-code:111122223333:outpost/
op-uniqueid
  controlPlaneInstanceType: m5.large
EOF
```

如需所有可用選項和預設值的完整清單，請參閱AWS Outposts 文件中的 eksctl 支援和組態檔案結構描述。

2. 使用您在上一步驟中建立的組態檔案建立叢集。eksctl 會在 Outpost 上建立 VPC 和一個子網路，以便在其中部署叢集。

```
eksctl create cluster -f outpost-control-plane.yaml
```

叢集佈建需要幾分鐘的時間。建立叢集時，會出現幾行輸出。輸出的最後一行類似於下面的範例行。

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

> **Tip**
>
> 若要查看使用 eksctl 建立叢集時可指定的大部分選項，請使用 **eksctl create cluster --help** 命令。若要查看所有可用的選項，您可使用 config 檔案。如需詳細資訊，請參閱 eksctl 文件中的使用組態檔與組態檔結構描述。您可以在 GitHub 上找到組態檔範例。

Eksctl 會自動為建立叢集的 IAM 主體 (使用者或角色) 建立存取項目，並向該 IAM 主體授予對叢集上的 Kubernetes 物件的管理員許可。如果您不希望叢集建立者對叢集上的 Kubernetes 物件具有管理員存取權，請將下列文字新增至先前的組態檔：**bootstrapClusterCreatorAdminPermissions: false** (與 metadata、vpc 和 outpost 處於同一層級)。如果您新增了該選項，則在建立叢集後，您需要為至少一個 IAM 主體建立存取項目，否則任何 IAM 主體均將無法存取叢集上的 Kubernetes 物件。

AWS Management Console

先決條件

現有 VPC 和子網路符合 Amazon EKS 要求。如需詳細資訊，請參閱 Amazon EKS 本機叢集 VPC 與子網路要求和考量事項。

若要使用 AWS Management Console

1. 如果您已擁有本機叢集 IAM 角色，或者您要使用 eksctl 來建立叢集，則可以略過此步驟。根據預設，eksctl 會為您建立角色。

   a. 執行下列命令以建立 IAM 信任政策 JSON 檔案。

```
cat >eks-local-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

   b. 建立 Amazon EKS 叢集 IAM 角色。若要建立 IAM 角色，必須為建立角色的 IAM 主體指派以下 iam:CreateRole 動作 (許可)。

```
aws iam create-role --role-name myAmazonEKSLocalClusterRole --assume-role-
policy-document file://"eks-local-cluster-role-trust-policy.json"
```

   c. 將名為 AmazonEKSLocalOutpostClusterPolicy 的 Amazon EKS 受管 IAM 政策連接到角色。若要將 IAM 政策連接至 IAM 主體，必須為連接政策的 IAM 實體指派以下 IAM 動作之一 (許可)：iam:AttachUserPolicy 或 iam:AttachRolePolicy。

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSLocalOutpostClusterPolicy --role-name myAmazonEKSLocalClusterRole
```

2. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
3. 請確定您已在主控台畫面頂端選取了受支援的 AWS 區域。
4. 選取 Add cluster (新增叢集)，然後選取 Create (建立)。
5. 在 Configure cluster (設定叢集) 頁面上，輸入或選取下列欄位的值：

- Kubernetes 控制平面位置 – 選擇 AWS Outposts。
- Outpost ID：選擇要在其上建立控制平面的 Outpost 的 ID。
- Instance type (執行個體類型)：選取執行個體類型。只會顯示 Outpost 中可用的執行個體類型。在下拉式清單中，每個執行個體類型都會說明為其建議的節點數量。選擇執行個體類型之前，請先參閱容量考量事項。所有複本均使用相同的執行個體類型進行部署。叢集建立後，就無法再變更執行個體類型。部署了三個控制平面執行個體。您無法變更此數字。
- Name (名稱)：叢集的名稱。它在您的 AWS 帳戶. 此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
- Kubernetes 版本 – 選擇您要用於叢集的 Kubernetes 版本。我們建議選取最新版本，除非您需要使用較早版本。
- 叢集服務角色 — 選擇您在上一個步驟中建立的 Amazon EKS 叢集 IAM 角色，以允許 Kubernetes控制平面管理 AWS 資源。
- Kubernetes 叢集管理員存取權：如果您希望建立叢集的 IAM 主體 (角色或使用者) 對叢集上的 Kubernetes 物件具有管理員存取權，請接受預設值 (允許)。Amazon EKS 會為該 IAM 主體建立存取項目，並向該存取項目授予叢集管理員許可。如需存取項目的詳細資訊，請參閱 管理存取項目。

  如果您不希望建立叢集的 IAM 主體擁有對 Kubernetes 叢集物件的管理員存取權，而是希望其他主體擁有此權限，請選擇「不允許」選項。建立叢集後，任何具有建立存取項目 IAM 許可的 IAM 主體，都可以為需要存取 Kubernetes 叢集物件的任何 IAM 主體新增存取項目。如需所需 IAM 許可的詳細資訊，請參閱《服務授權參考》中的 Amazon Elastic Kubernetes Service 定義的動作一節。如果您選擇「不允許」選項且不建立任何存取項目，則任何 IAM 主體均將無法存取叢集上的 Kubernetes 物件。
- Tags (標籤) – (選用) 將任何標籤新增到您的叢集。如需詳細資訊，請參閱 為您的 Amazon EKS 資源加上標籤。

完成此頁面後，請選擇下一步。

6. 在 Specify networking (指定網路) 頁面上，選取下列欄位的值：
   - VPC：選擇現有的 VPC。針對您要建立的叢集、任何節點和其他 Kubernetes 資源，VPC 必須有足夠數目的 IP 地址。您的 VPC 必須符合 VPC 要求和注意事項 中的要求。
   - Subnets (子網路)：根據預設，前一個欄位指定的 VPC 中的所有可用子網路會預先選取。您選擇的子網路必須符合 子網需求和注意事項 的要求。

安全群組:(選用) 指定一或多個您希望 Amazon EKS 與其建立的網路介面相關聯的安全群組。Amazon EKS 會自動建立一個安全群組，以支援您的叢集和 VPC 之間的通訊。Amazon EKS 將此安全群組及您選擇的任何群組與其建立的網路介面相關聯。如需有關 Amazon EKS 建立的叢集安全群組的詳細資訊，請參閱 Amazon EKS 安全群組與考量。您可以修改 Amazon EKS 建立的叢集安全群組中的規則。如果您選擇新增自己的安全群組，在叢集建立後，即無法變更選擇的安全群組。若要使內部部署主機與叢集端點通訊，您必須允許來自叢集安全群組的傳入流量。對於沒有輸入和輸出網際網路連線 (亦稱為私有叢集) 的叢集，您必須執行以下其中一項：

- 新增與所需 VPC 端點關聯的安全群組。如需有關所需端點的詳細資訊，請參閱 AWS 服務 的子網路存取權 中的介面 VPC 端點。
- 修改 Amazon EKS 建立的安全群組，以允許來自與 VPC 端點關聯之安全群組的流量。

完成此頁面後，請選擇下一步。

7. 在設定可觀測性頁面上，您可以選擇性地選擇要開啟的指標和控制平面日誌記錄選項。根據預設，系統會關閉每個日誌類型。
   - 如需 Prometheus 指標選項的詳細資訊，請參閱 建立叢集時開啟 Prometheus 指標。
   - 如需控制平面日誌記錄選項的詳細資訊，請參閱 Amazon EKS 控制平面記錄。

   完成此頁面後，請選擇下一步。
8. 在 Review and create (檢閱並建立) 頁面上，檢閱您在先前頁面上輸入或選取的資訊。如需變更，請選擇 Edit (編輯)。當您感到滿意時，請選擇 Create (建立)。在佈建叢集時，Status (狀態) 欄位顯示 CREATING (正在建立)。

   叢集佈建需要幾分鐘的時間。

2. 叢集建立之後，您便可以檢視已建立的 Amazon EC2 控制平面執行個體。

```
aws ec2 describe-instances --query 'Reservations[*].Instances[*].{Name:Tags[?
Key==`Name`]|[0].Value}' | grep my-cluster-control-plane
```

範例輸出如下。

```
"Name": "my-cluster-control-plane-id1"
"Name": "my-cluster-control-plane-id2"
"Name": "my-cluster-control-plane-id3"
```

每個執行個體均受到 node-role.eks-local.amazonaws.com/control-plane 污染，因此沒有在控制平面執行個體上排程工作負載。如需有關污點的詳細資訊，請參閱 Kubernetes 文件中的 Taints and Tolerations (污點和容差)。Amazon EKS 會持續監控本機叢集的狀態。我們會執行自動管理動作，如安全性修補程式和修復運作狀態不良的執行個體。如果本機叢集與雲端中斷連線，我們會完成動作，以確保在重新連線時將叢集修復為健全狀態。

3. 如果您使用 eksctl 建立叢集，則可以略過此步驟。eksctl 會為您完成此步驟。透過向 kubectl config 檔案新增內容，使 kubectl 能夠與您的叢集通訊。如需如何建立和更新檔案的說明，請參閱 建立或更新 Amazon EKS 叢集的 kubeconfig 檔案。

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

範例輸出如下。

```
Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/
username/.kube/config
```

4. 若要連線至您的本機叢集的 Kubernetes API 伺服器，請存取子網路的本機閘道，或從 VPC 內進行連線。如需將 Outpost 機架連接至內部部署網路的詳細資訊，請參閱 AWS Outposts 使用者指南中的機架本機閘道如何運作。如果使用直接 VPC 路由，且 Outpost 子網路具有通往您本機閘道的路由，則 Kubernetes 控制平面執行個體的私有 IP 地址會自動透過您的區域網路廣播。本機叢集的 Kubernetes API 伺服器端點託管在 Amazon Route 53 (Route 53) 中。API 服務端點可透過公有 DNS 伺服器將其解析為 Kubernetes API 伺服器的私有 IP 地址。

本機叢集的 Kubernetes 控制平面執行個體設定為具有固定私有 IP 地址的靜態彈性網路介面，這些 IP 地址在叢集生命週期內不會變更。在網路連線中斷期間，與 Kubernetes API 伺服器互動的機器可能無法連線至 Route 53。如果是這種情況，建議使用靜態私有 IP 地址來設定 /etc/hosts 以繼續操作。我們還建議您設定本機 DNS 伺服器，並將其連接至 Outpost。如需詳細資訊，請參閱 AWS Outposts 文件。執行下列命令以確認與您的叢集建立通訊。

```
kubectl get svc
```

範例輸出如下。

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

5. (選用) 在本機叢集處於與 AWS 雲端中斷連線的狀態時測試本機叢集的驗證。如需說明，請參閱為網路連線中斷做好準備。

內部資源

Amazon EKS 會在您的叢集上建立下列資源。這些資源供 Amazon EKS 內部使用。為了讓您的叢集正常運作，請勿編輯或修改這些資源。

- 下列鏡射 Pods：
  - `aws-iam-authenticator-node-hostname`
  - `eks-certificates-controller-node-hostname`
  - `etcd-node-hostname`
  - `kube-apiserver-node-hostname`
  - `kube-controller-manager-node-hostname`
  - `kube-scheduler-node-hostname`
- 下列自我管理的附加元件：
  - `kube-system/coredns`
  - `kube-system/kube-proxy` (在新增第一個節點之前不會建立)
  - `kube-system/aws-node` (在新增第一個節點之前不會建立)。本機叢集會使用 Amazon VPC CNI plugin for Kubernetes 外掛程式進行叢集聯網。請勿變更控制平面執行個體 (名為 `aws-node-controlplane-*` 的 Pod) 的組態。外掛程式建立新網路介面時，您可以使用組態變數來變更預設值。如需詳細資訊，請參閱上的文件 GitHub。
- 下列服務：
  - `default/kubernetes`
  - `kube-system/kube-dns`
- 名為 `eks.system` 的 `PodSecurityPolicy`
- 名為 `eks:system:podsecuritypolicy` 的 `ClusterRole`
- 名為 `eks:system` 的 `ClusterRoleBinding`
- 預設PodSecurity策略
- 除了叢集安全群組之外，Amazon EKS 還會在您 AWS 帳戶 命`eks-local-internal-do-not-use-or-edit-cluster-name-uniqueid`名的安全群組中建立一個安全群組。此安全群組允許流量在控制平面執行個體上執行的 Kubernetes 元件之間自由流動。

建議的後續步驟：

- 授與建立叢集的 IAM 主體檢視Kubernetes資源所需的權限 AWS Management Console
- 授予 IAM 實體對叢集的存取權。如果您希望實體在 Amazon EKS 主控台中檢視 Kubernetes 資源，請對實體授予 所需的許可。
- 設定叢集的日誌
- 熟悉網路連線中斷期間發生的情況。
- 將節點新增至叢集
- 考慮為您的 etcd 設定備份計劃。Amazon EKS 不支援本機叢集的 etcd 自動備份和還原。如需詳細資訊，請參閱 Kubernetes 文件中的備份 etcd 叢集。兩個主要選項是使用 etcdctl 自動拍攝快照或使用 Amazon EBS 儲存磁碟區備份。

Amazon EKS 本機叢集平台版本

本機叢集平台版本代表 Amazon EKS 叢集在 AWS Outposts的功能。這些版本包括在 Kubernetes 控制平面上執行的元件，已啟用 Kubernetes API 伺服器旗標。其也包含目前 Kubernetes 修補程式版本。每個 Kubernetes 次要版本皆有一或多個相關的平台版本。適用於不同 Kubernetes 次要版本的平台版本都是彼此獨立。雲端中本機叢集和 Amazon EKS 叢集的平台版本皆各自獨立。

當本機叢集推出新的 Kubernetes 次要版本 (例如 1.28) 時，該 Kubernetes 次要版本的初始平台版本便會從 eks-local-outposts.1 開始。但是，Amazon EKS 會定期發佈新平台版本，以啟用新的 Kubernetes 控制平面設定，以及提供安全性問題修正。

當有新的本機叢集平台版本可供次要版本使用時：

- 平台版本編號將會遞增 (eks-local-outposts.n+1)。
- Amazon EKS 會自動將所有現有的本機叢集更新到與其對應之 Kubernetes 次要版本的最新平台版本。現有平台版本的自動更新將會逐步發佈。發行程序可能需要一些時間。如果您立即需要最新平台版本的功能，建議您建立新的本機叢集。
- Amazon EKS 可能會發佈具有對應之修補程式版本的新節點 AMI。針對單一 Kubernetes 次要版本，所有修補程式版本在 Kubernetes 控制平面和節點 AMI 之間皆相容。

新的 平台版本不會帶來重大改變或導致服務中斷。

本機叢集一律會透過指定 Kubernetes 版本的最新可用平台版本 (eks-local-outposts.n) 建立。

的目前及最新平台版本如下表所示。

Kubernetes 版本 **1.28**

下列許可控制器已為所有 1.28 平台版本啟用：CertificateApproval、CertificateSigning、CertificateSubjectRestriction、DefaultIngressClass、DefaultStorageClass、DefaultTolerationSeconds、LimitRanger、MutatingAdmissionWebhook、NamespaceLifecycle、NodeRestriction、PersistentVolumeClaimResize、PodSecurity、Priority、ResourceQuota、RuntimeClass、ServiceAccount、StorageObjectInUseProtection、TaintNodesByCondition、ValidatingAdmissionPolicy 以及 ValidatingAdmissionWebhook。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.28.6	eks-local-outposts.5	將包含最新的錯誤修正更新至 v1.19.3，以支援 Outposts 的本機開機。	2024年4月18日
1.28.6	eks-local-outposts.4	具有安全性修正和增強功能的新平台版本。在 Outposts 恢復支持或本地啟動。降級Bottlerocket版本為以確保相容v1.15.1性。	2024年4月2日
1.28.6	eks-local-outposts.3	具有安全性修正和增強功能的新平台版本。	2024年3月22日
1.28.6	eks-local-outposts.2	具有安全性修正和增強功能 kube-proxy 的新平台版本已更新為. v1.28.6 AWS IAM 身份驗證器已更新為。v0.6.17出於兼容性原因，針對 Kubernetes 的 Amazon VPC CNI 插件降級為。v1.13.2將Bottlerocket版本更新為v1.19.2.	2024年3月8日
1.28.1	eks-local-outposts.1	Kubernetes 版本的初始版本v1.28。適用於前哨站上的本地 Amazon EKS 叢集	2023 年 10 月 4 日

Kubernetes 版本 **1.27**

下列許可控制器已為所有 1.27 平台版本啟用：CertificateApproval、CertificateSigning、CertificateSubjectRestriction、DefaultIngressClass、DefaultStorageClass、DefaultTolerationSeconds、LimitRanger、MutatingAdmissionWebhook、NamespaceLifecycle、NodeRestriction、PersistentVolumeClaimResize、PodSecurity、Priority、ResourceQuota、RuntimeClass、ServiceAccount、StorageObjectInUseProtection、TaintNodesByCondition、ValidatingAdmissionPolicy 以及 ValidatingAdmissionWebhook。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.27.10	eks-local-outposts.5	具有安全修復和增強功能的新平台。	2024年4月2日
1.27.10	eks-local-outposts.4	具有安全性修正和增強功能的新平台。kube-proxy 已更新至. v1.27.10 AWS IAM 身份驗證器已更新為。v0.6.17將Bottlerocket版本更新為v1.19.2.	2024年3月22日
1.27.3	eks-local-outposts.3	具有安全性修正和增強功能的新平台版本。kube-proxy 已更新至 v1.27.3。Kubernetes 專用 Amazon VPC CNI 外掛程式已更新至 v1.13.2。	2023 年 7 月 14 日
1.27.1	eks-local-outposts.2	將 CoreDNS 映像更新至 v1.10.1	2023 年 6 月 22 日
1.27.1	eks-local-outposts.1	1.27針對 Outposts 上本地 Amazon EKS 叢集的 Kubernetes 版本的初始版本。	2023 年 5 月 30 日

Kubernetes 版本 **1.26**

下列許可控制器已為所有 1.26 平台版本啟用：CertificateApproval、CertificateSigning、CertificateSubjectRestriction、DefaultIngressClass、DefaultStorageClass、DefaultTolerationSeconds、LimitRanger、MutatingAdmissionWebhook、NamespaceLifecycle、NodeRestriction、PersistentVolumeClaimResize、PodSecurity、Priority、ResourceQuota、RuntimeClass、ServiceAccount、StorageObjectInUseProtection、TaintNodesByCondition、ValidatingAdmissionPolicy 以及 ValidatingAdmissionWebhook。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.26.13	eks-local-outposts.5	具有安全性修正和增強功能的新平台版本。kube-proxy 已更新至. v1.26.13 AWS IAM 身份驗證器已更新為 。v0.6.17將Bottlerocket版本更新為v1.19.2.	2024年3月22日

Kubernetes 版本 **1.25**

下列許可控制器已為所有 1.25 平台版本啟用：CertificateApproval、CertificateSigning、CertificateSubjectRestriction、Defaul 以及 ValidatingAdmissionWebhook。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.25.16	eks-local-outposts.7	具有安全性修正和增強功能的新平台版本。kube-proxy 已更新至. v1.25.16 AWS IAM 身份驗證器已更新為 。v0.6.17將Bottlerocket版本更新為v1.19.2.	2024年3月22日
1.25.11	eks-local-outposts.6	具有安全性修正和增強功能的新平台版本。kube-proxy 已更新至 v1.25.11。Kubernetes 專用 Amazon VPC CNI 外掛程式已更新至 v1.13.2。	2023 年 7 月 14 日
1.25.9	eks-local-outposts.5	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 13 日

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.25.6	eks-local-outposts.4	將 Bottlerocket 版本更新至 1.13.2	2023 年 5 月 2 日
1.25.6	eks-local-outposts.3	Amazon EKS 控制平面實例操作系統已更新為保安版本，v1.13.1並將 Amazon VPC CNI 插件的 Kubernetes 更新為版本。v1.12.6	2023 年 4 月 14 日
1.25.6	eks-local-outposts.2	改善 Kubernetes 控制平面執行個體的診斷集合。	2023 年 3 月 8 日
1.25.6	eks-local-outposts.1	1.25針對 Outposts 上本地 Amazon EKS 叢集的 Kubernetes 版本的初始版本。	2023 年 3 月 1 日

Kubernetes 版本 **1.24**

下列許可控制器已為所有 1.24 平台版本啟用：DefaultStorageClass、DefaultTolerationSeconds、LimitRanger、MutatingAdmissionW以及 DefaultIngressClass。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.24.17	eks-local-outposts.7	具有安全性修正和增強功能的新平台版本。kube-proxy 已更新至. v1.25.16 AWS IAM 身份驗證器已更新v0.6.17。將Bottlerocket版本更新為v1.19.2.	2024年3月22日
1.24.15	eks-local-outposts.6	具有安全性修正和增強功能的新平台版本。kube-proxy 已	2023 年 7 月 14 日

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
		更新至 v1.24.15。Kubernetes 專用 Amazon VPC CNI 外掛程式已更新至 v1.13.2。	
1.24.13	eks-local-outposts.5	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 13 日
1.24.9	eks-local-outposts.4	將 Bottlerocket 版本更新至 1.13.2	2023 年 5 月 2 日
1.24.9	eks-local-outposts.3	Amazon EKS 控制平面實例操作系統已更新為保安版本，v1.13.1並將 Amazon VPC CNI 插件的 Kubernetes 更新為版本。v1.12.6	2023 年 4 月 14 日
1.24.9	eks-local-outposts.2	改善 Kubernetes 控制平面執行個體的診斷集合。	2023 年 3 月 8 日
1.24.9	eks-local-outposts.1	1.24針對 Outposts 上本地 Amazon EKS 叢集的 Kubernetes 版本的初始版本。	2023 年 1 月 17 日

Kubernetes 版本 **1.23**

下列許可控制器已為所有 1.23 平台版本啟用：DefaultStorageClass、DefaultTolerationSeconds、LimitRanger、MutatingAdmissionW以及 DefaultIngressClass。

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.23.17	eks-local-outposts.6	具有安全性修正和增強功能的新平台版本。	2023 年 7 月 13 日

Kubernetes 版本	Amazon EKS 平台版本	版本備註	版本日期
1.23.17	eks-local-outposts.5	具有安全性修正和增強功能的新平台版本。kube-proxy 已更新至. v1.23.17 將Bottlerocket版本更新為v1.14.1.	2023 年 7 月 6 日
1.23.15	eks-local-outposts.4	Amazon EKS 控制平面實例操作系統已更新為保安版本，v1.13.1並將 Amazon VPC CNI 插件的 Kubernetes 更新為版本。v1.12.6	2023 年 4 月 14 日
1.23.15	eks-local-outposts.3	改善 Kubernetes 控制平面執行個體的診斷集合。	2023 年 3 月 8 日
1.23.15	eks-local-outposts.2	1.23針對 Outposts 上本地 Amazon EKS 叢集的 Kubernetes 版本的初始版本。	2023 年 1 月 17 日

Amazon EKS 本機叢集 VPC 與子網路要求和考量事項

建立本機叢集時，您需要指定 VPC 與至少一個在 Outpost 上執行的私人子網路。本主題概述了本機叢集的 VPC 和子網路要求和考量事項。

VPC 要求和注意事項

建立本機叢集時，您指定的 VPC 必須符合下列要求和考量事項：

- 確保 VPC 具有足夠的 IP 地址用於本機叢集、任何節點以及您要建立的其他 Kubernetes 資源。如果要使用的 VPC 沒有足夠的 IP 地址，請增加可用 IP 地址的數目。您可以透過將其他無類別域間路由 (CIDR) 區塊與 VPC 關聯來完成此操作。您可以在建立叢集的之前或之後，將私有 (RFC 1918) 和公有 (非 RFC 1918) CIDR 區塊與 VPC 關聯。叢集可能最多需要 5 小時才能辨識您與 VPC 關聯的 CIDR 區塊。
- VPC 無法指派 IP 字首或 IPv6 CIDR 區塊。由於這些限制，增加 Amazon EC2 節點的可用 IP 地址數量 和 IPv6叢集的位址Pods、和 services 中涵蓋的資訊不適用於您的 VPC。

- VPC 已啟用 DNS 主機名稱和 DNS 解析。如果沒有這些功能，本機叢集無法建立，您需要啟用這些功能並重新建立叢集。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 的 DNS 屬性。
- 若要透過區域網路存取本機叢集，VPC 必須與 Outpost 的本機閘道路由表關聯。如需詳細資訊，請參閱 AWS Outposts 使用者指南中的 VPC 關聯。

子網需求和注意事項

在建立叢集時，指定至少一個私有子網路。如果您指定多個子網路，Kubernetes 控制平面執行個體會平均分佈在子網路上。如果指定多個子網路，則子網路必須存在於相同的 Outpost 上。此外，子網路還必須具有適當的路由和安全群組許可，才能互相通訊。在建立本機叢集時，您指定的子網路必須符合下列要求：

- 子網路都位於相同的邏輯 Outpost 上。
- 這些子網路一起至少有三個可用於 Kubernetes 控制平面執行個體的 IP 地址。如果指定三個子網路，每個子網路必須至少有一個可用的 IP 地址。如果指定兩個子網路，每個子網路必須至少有兩個可用的 IP 地址。如果指定一個子網路，該子網路必須至少有三個可用的 IP 地址。
- 子網路具有 Outpost 機架本機閘道的路由，以存取您區域網路上的 Kubernetes API 伺服器。如果子網路沒有 Outpost 機架本機閘道的路由，您必須在 VPC 內與 Kubernetes API 伺服器進行通訊。
- 子網必須使用 IP 地址型命名。Amazon EKS 不支援 Amazon EC2 資源型命名。

AWS 服務 的子網路存取權

Outposts 上的本機叢集私有子網路必須能與區域性 AWS 服務 通訊。為此，您可以使用 NAT 閘道進行網際網路對外存取，或者，如果您想要所有流量在 VPC 內保持私密，可使用介面 VPC 端點。

使用 NAT 閘道

Outposts 上的本機叢集私有子網路必須具有關聯的路由表，此路由表具有 Outpost 父可用區域公有子網路中 NAT 閘道的路由。公有子網路必須擁有到網際網路閘道的路由。此 NAT 閘道會啟用傳出網際網路存取，並防止來自網際網路未經要求的傳入連線連接到 Outpost 上的執行個體。

使用 介面 VPC 端點

如果 Outposts 上的本機叢集私有子網路沒有對外網際網路連線，或者如果您想要所有流量在 VPC 內保持私密，則在建立叢集之前，您必須在區域子網路中建立下列介面 VPC 端點和閘道端點。

端點	端點類型
com.amazonaws. *region-code* .ssm	界面
com.amazonaws. *region-code* .ssmmessages	界面
com.amazonaws. *region-code* .ec2messages	界面
com.amazonaws. *region-code* .ec2	界面
com.amazonaws. *region-code* .secretsmanager	界面
com.amazonaws. *region-code* .logs	界面
com.amazonaws. *region-code* .sts	界面
com.amazonaws. *region-code* .ecr.api	界面
com.amazonaws. *region-code* .ecr.dkr	界面
com.amazonaws. *region-code* .s3	閘道

端點必須符合下列需求：

- 在位於 Outpost 父可用區域的私有子網路中建立
- 已啟用私有 DNS 名稱
- 擁有連接的安全群組，該群組允許來自私有 Outpost 子網路之 CIDR 範圍的輸入 HTTPS 流量。

建立端點會產生費用。如需詳細資訊，請參閱 AWS PrivateLink 定價。如果您的 Pods 需要存取其他 AWS 服務，則您需要建立其他端點。如需完整的端點清單，請參閱與 AWS PrivateLink 整合的 AWS 服務。

建立 VPC

您可以使用下列其中一個 AWS CloudFormation 範本建立符合先前需求的 VPC：

- 範本 1 – 此範本會建立 VPC，其包含的一個私有子網路在 Outpost 上，一個公有子網路在 AWS 區域 中。私有子網路具有透過駐留在 AWS 區域 中公有子網路中的 NAT 閘道連至網際網路的路由。此範本可用於在具有輸出網際網路存取權的子網路中建立本機叢集。
- 範本 2 – 此範本會在 Outpost 上建立具有一個私有子網路的 VPC，以及在沒有輸入或輸出網際網路存取權 (亦稱為私有子網路) 的子網路中建立本機叢集所需的最基本的一組 VPC 端點。

為網路連線中斷做好準備

如果您的區域網路失去與 AWS 雲端 之間的連線，您可以繼續使用 Outpost 上的本機 Amazon EKS 叢集。本主題包含如何準備本機叢集以應對網路連線中斷和相關考量事項。

準備本機叢集以應對網路連線中斷時的考量事項：

- 本機叢集可在出現暫時且意外的網路連線中斷時提供穩定性和持續運作。AWS Outposts 仍為完整連線方案，可作為您資料中心中 AWS 雲端 的擴充功能運作。如果 Outpost 與 AWS 雲端 之間的網路連線中斷，則我們建議您嘗試恢復連線。如需說明，請參閱《AWS Outposts 使用者指南》中的 AWS Outposts 機架網路疑難排解檢查清單。如需有關如何針對本機叢集問題進行疑難排解的詳細資訊，請參閱 AWS Outposts 上 Amazon EKS 本機叢集疑難排解。
- Outpost 會發射 `ConnectedStatus` 指標，您可用此指標來監控 Outpost 的連線狀態。如需詳細資訊，請參閱 AWS Outposts 使用者指南中的 Outpost 指標。
- 本機叢集會使用 IAM 作為預設身分驗證機制，該機制使用 Kubernetes 的 AWS Identity and Access Management 驗證器。IAM 在網路連線中斷期間無法使用。因此，本機叢集會使用 `x.509` 憑證支援替代身分驗證機制，您可使用這些憑證在網路連線中斷期間連線至叢集。如需有關如何取得和使用您叢集 `x.509` 憑證的資訊，請參閱 在網路連線中斷期間針對本機叢集進行身分驗證。
- 如果您無法在網路連線中斷期間存取 Route 53，請考慮在內部部署環境中使用本機 DNS 伺服器。Kubernetes 控制平面執行個體會使用靜態 IP 地址。您可使用端點主機名稱和 IP 地址來設定用來連線至叢集的主機，以此作為使用本機 DNS 伺服器的替代方法。如需詳細資訊，請參閱《AWS Outposts使用者指南》中的 DNS。
- 如果您預期應用程式流量會在網路連線中斷期間增加，則可在連線至雲端時在叢集中佈建備用運算容量。AWS Outposts 的價格已包含 Amazon EC2 執行個體。因此，執行備用執行個體不會影響您的 AWS 用量成本。
- 在網路連線中斷期間，若要啟用工作負載的建立、更新和擴展操作，則必須可透過區域網路存取應用程式的容器映像，且叢集須有足夠容量。本機叢集不會為您託管容器登錄檔。如果先前已在這些節點上執行 Pods，則系統會在節點上快取容器映像。如果您通常會從雲端中的 Amazon ECR 提取應用程式的容器映像，則請考慮執行本機快取或登錄檔。如果您在網路連線中斷期間需要建立、更新和擴展工作負載資源的操作，則本機快取或登錄會很有幫助。

- 本機叢集會使用 Amazon EBS 作為持久性磁碟區的預設儲存類別，並使用 Amazon EBS CSI 驅動程式管理 Amazon EBS 持久性磁碟區的生命週期。在網路連線中斷期間，無法建立、更新或擴展由 Amazon EBS 支援的 Pods。這是因為這些操作需要呼叫雲端中的 Amazon EBS API。如果您要在本機叢集上部署具狀態的工作負載，且在網路連線中斷期間需要建立、更新或擴展操作，則請考慮使用替代儲存機制。
- 如果 AWS Outposts 無法存取相關的 AWS 區域內 API (例如適用於 Amazon EBS 或 Amazon S3 的 API)，就無法建立或刪除 Amazon EBS 快照。
- 將 ALB (輸入) 與 AWS Certificate Manager (ACM) 整合時，會將憑證推送並儲存在 AWS Outposts ALB 運算執行個體的記憶體中。如果與 AWS 區域 中斷連線，目前的 TLS 終止將繼續運作。在此內容中變動操作將會失敗 (例如新的輸入定義、新的 ACM 型憑證 API 操作、ALB 運算規模或憑證輪換)。如需詳細資訊，請參閱 AWS Certificate Manager 使用者指南中的針對受管憑證續約進行疑難排解。
- Amazon EKS 控制平面日誌會在網路連線中斷期間於 Kubernetes 控制平面執行個體上進行本機快取。重新連線後，記錄檔會傳送至父系中的 CloudWatch 記錄檔AWS 區域。您可以使用 Prometheus、Grafana 或 Amazon EKS 合作夥伴解決方案，以透過 Kubernetes API 伺服器的指標端點或使日誌的 Fluent Bit 在本機上監控叢集。
- 如果您正為應用程式流量使用 Outposts 上的 AWS Load Balancer Controller，前端是 AWS Load Balancer Controller 的現有 Pods 會在網路連線中斷期間持續接收流量。在 Outpost 重新連線至 AWS 雲端 之前，在網路連線中斷期間建立的新 Pods 皆不會接收流量。請考慮在連線至 AWS 雲端時為您的應用程式設定複本計數，以滿足網路連線中斷期間的擴展需求。
- Amazon VPC CNI plugin for Kubernetes 預設為次要 IP 模式。其使用 `WARM_ENI_TARGET=1` 來設定，這會允許外掛程式保留可用 IP 地址的「完整彈性網路介面」。請依據中斷連線狀態期間的擴展需求來考慮變更 `WARM_ENI_TARGET`、`WARM_IP_TARGET` 和 `MINIMUM_IP_TARGET` 值。如需詳細資訊，請參閱上的外掛程式`readme`檔案 GitHub。有關每個執行個體類型所支援Pods的最大數目清單，請參閱（詳見）的`eni-max-pods.txt`檔案 GitHub。

在網路連線中斷期間針對本機叢集進行身分驗證

AWS Identity and Access Management (IAM) 在網路連線中斷期間無法使用。中斷連線時，您無法使用 IAM 憑證對本機叢集進行身分驗證。但是，您可以在中斷連線時使用 `x509` 憑證，透過區域網路連接至您的叢集。您需要下載並存放客戶端 `X509` 憑證，以在中斷連線期間使用。本此主題中，您將了解如何在中斷連線時建立和使用憑證對叢集進行身分驗證。

1. 建立憑證簽署請求。

    a. 產生憑證簽署請求。

```
openssl req -new -newkey rsa:4096 -nodes -days 365 \
    -keyout admin.key -out admin.csr -subj "/CN=admin"
```

b. 在 Kubernetes 中建立憑證簽署請求。

```
BASE64_CSR=$(cat admin.csr | base64 -w 0)
cat << EOF > admin-csr.yaml
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: admin-csr
spec:
  signerName: kubernetes.io/kube-apiserver-client
  request: ${BASE64_CSR}
  usages:
  - client auth
EOF
```

2. 使用 kubectl 建立憑證簽署請求。

```
kubectl create -f admin-csr.yaml
```

3. 檢查憑證簽署請求的狀態。

```
kubectl get csr admin-csr
```

範例輸出如下。

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Pending
```

Kubernetes 已建立憑證簽署請求。

4. 核准憑證簽署請求。

```
kubectl certificate approve admin-csr
```

5. 重新檢查核准的憑證簽署請求狀態。

```
kubectl get csr admin-csr
```

範例輸出如下。

```
NAME       AGE   REQUESTOR                     CONDITION
admin-csr  11m   kubernetes-admin              Approved
```

6. 擷取並驗證憑證。

   a. 擷取憑證。

   ```
   kubectl get csr admin-csr -o jsonpath='{.status.certificate}' | base64 --decode
    > admin.crt
   ```

   b. 驗證憑證。

   ```
   cat admin.crt
   ```

7. 建立與 admin 使用者綁定的叢集角色。

```
kubectl create clusterrolebinding admin --clusterrole=cluster-admin \
    --user=admin --group=system:masters
```

8. 產生中斷連線狀態的使用者範圍 kubeconfig。

   您可以使用下載的 admin 憑證來產生 kubeconfig 檔案。取代下列命令中的 *my-cluster* 和 *apiserver-endpoint*。

```
aws eks describe-cluster --name my-cluster \
    --query "cluster.certificateAuthority" \
    --output text | base64 --decode > ca.crt
```

```
kubectl config --kubeconfig admin.kubeconfig set-cluster my-cluster \
    --certificate-authority=ca.crt --server apiserver-endpoint --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-credentials admin \
    --client-certificate=admin.crt --client-key=admin.key --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-context admin@my-cluster \
    --cluster my-cluster --user admin
```

```
kubectl config --kubeconfig admin.kubeconfig use-context admin@my-cluster
```

9. 檢視您的 kubeconfig 檔案。

```
kubectl get nodes --kubeconfig admin.kubeconfig
```

10. 如果您在 Outpost 上已有生產中的服務，請跳過此步驟。如果 Amazon EKS 是您 Outpost 上唯一執行的服務，且 Outpost 目前尚未處於生產中，則您可模擬網路連線中斷。在您使用本機叢集進入生產之前，模擬中斷連線以確保在中斷連線狀態下可存取叢集。

    a. 在連接 Outpost 至 AWS 區域 的聯網裝置上套用防火牆規則。這會中斷 Outpost 連結的服務。您無法建立任何新的執行個體。目前執行中的執行個體會失去與 AWS 區域 和網際網路的連線。

    b. 您可在中斷連線時使用 x509 憑證，測試連結至您本機叢集的連線。請確保將您的 kubeconfig 變更為您在上一步中建立的 admin.kubeconfig。使用您本機叢集的名稱取代 *my-cluster*。

    ```
    kubectl config use-context admin@my-cluster --kubeconfig admin.kubeconfig
    ```

    如果您在本機叢集處於中斷連線狀態時發現任何問題，建議您開啟支援票證。

容量考量事項

本主題提供有關為 Outpost 上本機 Amazon EKS 叢集選取 Kubernetes 控制平面執行個體類型以及 (選用) 使用置放群組滿足該叢集的高可用性需求的指導。

選取要用於 Outposts 上本機叢集 Kubernetes 控制平面的執行個體類型 (例如 m5、c5 或 r5) 之前，請確認 Outpost 組態上可用的執行個體類型。識別可用的執行個體類型之後，根據工作負載所需的節點數量，來選取執行個體大小 (例如 large、xlarge 或 2xlarge)。下表提供有關選擇執行個體大小的建議。

> **Note**
>
> 執行個體大小必須在您的 Outposts 上設定好插槽。確保在本機叢集的生命週期內，您有足夠的容量可容納 Outposts 上可用大小的三個執行個體。如需可用 Amazon EC2 執行個體類型的清單，請參閱AWS Outposts 機架功能中的運算和儲存區段。

節點數量。	Kubernetes 控制平面執行個體大小
1–20	`large`
21–100	`xlarge`
101–250	`2xlarge`
251–500	`4xlarge`

Kubernetes 控制平面的儲存空間需要每個本機叢集 246 GB 的 Amazon EBS 儲存空間，才能滿足 `etcd` 所需 IOPS 佔用的空間。建立本機叢集時，系統會自動為您佈建 Amazon EBS 磁碟區。

控制平面置放

若您未使用該 `OutpostConfig.ControlPlanePlacement.GroupName` 屬性指定置放群組，則為您的 Kubernetes 控制平面所佈建的 Amazon EC2 執行個體不會在 Outpost 上可用的基礎容量上收到任何特定硬體置放的強制執行。

您可以使用置放群組來滿足 Outpost 上本機 Amazon EKS 叢集的高可用性需求。透過在叢集建立期間指定置放群組，可影響 Kubernetes 控制平面執行個體的置放。執行個體分布於獨立的基礎硬體 (機架或主機)，這樣可將相關執行個體對硬體故障事件的影響降至最低。

要求

您能夠設定的分布類型取決於部署中所擁有的 Outpost 機架數量。

- 單一邏輯 Outpost 中具有一或兩個實體機架的部署：您必須擁有至少三台主機，這些主機使用您為 Kubernetes 控制平面執行個體選擇的執行個體類型進行設定。使用主機層級分布的分布置放群組可確保所有 Kubernetes 控制平面執行個體在 Outpost 部署中可用的基礎機架內的不同主機上執行。
- 單一邏輯 Outpost 中具有三個或更多實體機架的部署：您必須擁有至少三台主機，這些主機使用您為 Kubernetes 控制平面執行個體選擇的執行個體類型進行設定。使用機架層級分布的分布置放群組可確保所有 Kubernetes 控制平面執行個體在 Outpost 部署中的不同機架上執行。或者，如前一個選項中所述，您也可以使用主機層級分布置放群組。

您負責建立所需的置放群組。您可以在呼叫 `CreateCluster` API 時指定置放群組。如需有關放置群組以及如何建立放置群組的詳細資訊，請參閱 Amazon EC2 使用者指南中的放置群組。

考量事項

- 指定放置群組後，Outpost 上必須有可用的開槽容量，才能成功建立本機 Amazon EKS 叢集。容量會根據您使用的是主機還是機架分布類型而有所不同。若容量不足，則叢集會維持在 `Creating` 狀態。您可以檢查 `DescribeCluster` API 回應於運作狀態欄位的 `Insufficient Capacity Error`。您必須釋放容量才能進行建立程序。
- 在 Amazon EKS 本機叢集平台和版本更新期間，系統會使用滾動式更新策略，將叢集中的 Kubernetes 控制平面執行個體取代為新的執行個體。在此取代過程中，每個控制平面執行個體皆會終止，從而釋放各自的插槽。新更新的執行個體已佈建在其位置中。更新後的執行個體可能會被置放於已釋放的插槽中。若插槽已被另一個不相關的執行個體使用，且沒有剩餘的容量符合所需的分布拓撲需求，則叢集會維持在 `Updating` 狀態。您可以在 `DescribeCluster` API 回應的運作狀態欄位上看到對應的 `Insufficient Capacity Error`。您必須釋放容量，才能進行更新程序並重新建立先前的高可用性層級。
- 每個帳戶最多可以建立 500 個放置群組 AWS 區域。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的一般規則和限制。

AWS Outposts 上 Amazon EKS 本機叢集疑難排解

本主題涵蓋一些使用本機叢集時可能遇到的常見錯誤與排除這些錯誤的方法。雖然本機叢集與雲端中的 Amazon EKS 叢集相似，但 Amazon EKS 管理本機叢集的方式仍有些許差異。

API 行為

本機叢集透過 Amazon EKS API 建立，但會以非同步方式執行。這表示本機叢集對 Amazon EKS API 的請求會立即傳回。但是，這些請求可能會成功、因輸入驗證錯誤而快速檢錯，或失敗並附上描述性驗證錯誤。此行為與 Kubernetes API 相似。

本機叢集不會變為 `FAILED` 狀態。Amazon EKS 會嘗試以連續的方式使叢集狀態與使用者請求的所需狀態一致。因此，本機叢集可能會長期保持在 `CREATING` 狀態，直到基礎問題解決為止。

描述叢集運作狀態欄位

本機叢集問題可使用 `describe-cluster` Amazon EKS AWS CLI 命令發現。本機叢集問題可透過 `describe-cluster` 命令回應的 `cluster.health` 欄位顯露。此欄位內含的訊息包括錯誤代碼、描述性訊息和相關資源 ID。此資訊僅可透過 Amazon EKS API 和 AWS CLI 提供。在以下範例中，使用您本機叢集的名稱取代 *`my-cluster`*。

```
aws eks describe-cluster --name my-cluster --query 'cluster.health'
```

範例輸出如下。

```
{
    "issues": [
        {
            "code": "ConfigurationConflict",
            "message": "The instance type 'm5.large' is not supported in Outpost 'my-outpost-arn'.",
            "resourceIds": [
                "my-cluster-arn"
            ]
        }
    ]
}
```

如果問題無法解決，您可能需要刪除本機叢集並建立新叢集。例如，嘗試使用在 Outpost 上未提供的執行個體類型佈建叢集。下表包含與運作狀態相關的常見錯誤。

錯誤情況	代碼	訊息	ResourceIds
找不到提供的子網路。	ResourceNotFound	The subnet ID *subnet-id* does not exist	所有提供的子網路 ID
提供的子網路不屬於同一 VPC。	ConfigurationConflict	Subnets specified must belong to the same VPC	所有提供的子網路 ID
某些提供的子網路不屬於指定的 Outpost。	ConfigurationConflict	Subnet *subnet-id* expected to be in *outpost-arn*, but is in *other-outpost-arn*	有問題的子網路 ID
某些提供的子網路不屬於任何 Outpost。	ConfigurationConflict	Subnet *subnet-id* is not part of any Outpost	有問題的子網路 ID

錯誤情況	代碼	訊息	ResourceIds
某些提供的子網路沒有足夠的可用地址為控制平面執行個體建立彈性網路介面。	ResourceLimitExceeded	The specified subnet does not have enough free addresses to satisfy the request.	有問題的子網路 ID
您的 Outpost 不支援指定的控制平面執行個體類型。	ConfigurationConflict	The instance type *type* is not supported in Outpost *outpost-arn*	叢集 ARN
您已終止控制平面 Amazon EC2 執行個體，或 run-instance 已成功，但觀察到的狀態變更為 Terminated 。您的 Outpost 重新連線後一段時間可能會發生此情況，且 Amazon EBS 內部錯誤會導致 Amazon EC2 內部工作流程失敗。	InternalFailure	EC2 instance state "Terminated" is unexpected	叢集 ARN
您在 Outpost 上的容量不足。如果 Outpost 與 AWS 區域 中斷連線後，在建立叢集時也會發生此情況。	ResourceLimitExceeded	There is not enough capacity on the Outpost to launch or start the instance.	叢集 ARN
您的帳戶已超過您的安全群組配額。	ResourceLimitExceeded	Amazon EC2 API 傳回的錯誤訊息	目標 VPC ID

錯誤情況	代碼	訊息	ResourceIds
您的帳戶已超過您的彈性網路介面配額。	`ResourceLimitExceeded`	Amazon EC2 API 傳回的錯誤訊息	目標子網路 ID
控制平面執行個體無法透過 AWS Systems Manager 連線。如需解決方案，請參閱控制平面執行個體無法透過 AWS Systems Manager 連線。	`ClusterUnreachable`	Amazon EKS 控制平面執行個體無法透過 SSM 連線。請確認您的 SSM 和網路組態，並參考 Outpost 上的 EKS 疑難排解文件。	Amazon EC2 執行個體 ID
取得受管安全群組或彈性網路介面的詳細資訊時發生錯誤。	根據 Amazon EC2 用戶端錯誤代碼。	Amazon EC2 API 傳回的錯誤訊息	所有受管的安全群組 ID
授權或撤銷安全群組輸入規則時發生錯誤。這適用於叢集和控制平面安全群組。	根據 Amazon EC2 用戶端錯誤代碼。	Amazon EC2 API 傳回的錯誤訊息	有問題的安全群組 ID
刪除控制平面執行個體的彈性網路介面時發生錯誤。	根據 Amazon EC2 用戶端錯誤代碼。	Amazon EC2 API 傳回的錯誤訊息	有問題的彈性網路介面 ID

下表列出 `describe-cluster` 回應的運作狀態欄位中呈現的其他 AWS 服務 的錯誤。

Amazon EC2 錯誤代碼	叢集運作狀態問題代碼	描述
`AuthFailure`	`AccessDenied`	基於各種原因，可能會發生此錯誤。最常見的原因是您意外移除了服務用於從控制平面縮小服務連結角色政策範圍的標籤。如果發生這種情況，Amazon EKS 將無法再管理和監控這些 AWS 資源。

Amazon EC2 錯誤代碼	叢集運作狀態問題代碼	描述
UnauthorizedOperation	AccessDenied	基於各種原因，可能會發生此錯誤。最常見的原因是您意外移除了服務用於從控制平面縮小服務連結角色政策範圍的標籤。如果發生這種情況，Amazon EKS 將無法再管理和監控這些 AWS 資源。
InvalidSubnetID.NotFound	ResourceNotFound	找不到安全群組輸入規則的子網路 ID 時，會發生此錯誤。
InvalidPermission.NotFound	ResourceNotFound	安全群組輸入規則的許可不正確時，會發生此錯誤。
InvalidGroup.NotFound	ResourceNotFound	找不到安全群組輸入規則的群組時，會發生此錯誤。
InvalidNetworkInterfaceID.NotFound	ResourceNotFound	找不到安全群組輸入規則的網路介面 ID 時，會發生此錯誤。
InsufficientFreeAddressesInSubnet	ResourceLimitExceeded	超過子網路資源配額時，會發生此錯誤。
InsufficientCapacityOnOutpost	ResourceLimitExceeded	超過 Outpost 容量配額時，會發生此錯誤。
NetworkInterfaceLimitExceeded	ResourceLimitExceeded	超過彈性網路介面配額時，會發生此錯誤。
SecurityGroupLimitExceeded	ResourceLimitExceeded	超過安全群組配額時，會發生此錯誤。

Amazon EC2 錯誤代碼	叢集運作狀態問題代碼	描述
VcpuLimitExceeded	ResourceLimitExceeded	在新帳戶中建立 Amazon EC2 執行個體時，便會觀察到這種狀況。此錯誤可能類似以下內容："You have requested more vCPU capacity than your current vCPU limit of 32 allows for the instance bucket that the specified instance type belongs to. Please visit http://aws.amazon.com/contact-us/ec2-request to request an adjustment to this limit."
InvalidParameterValue	ConfigurationConflict	如果 Outpost 不支援指定的執行個體類型，Amazon EC2 會傳回此錯誤代碼。
所有其他失敗	InternalFailure	無

無法建立或修改叢集

本機叢集需要不同於雲端中託管的 Amazon EKS 叢集的許可和政策。當叢集無法建立並產生InvalidPermissions錯誤時，請仔細檢查您使用的叢集角色是否已附加 AmazonEks LocalOutpostClusterPolicy 管理政策。所有其他 API 呼叫需要與雲端中 Amazon EKS 叢集相同的許可集。

叢集卡在 **CREATING** 狀態

建立本機叢集所需的時間量因數個因素而異。這些因素包括您的網路組態、Outpost 組態以及叢集的組態。一般而言，本機叢集會在 15-20 分鐘內建立並變更為 ACTIVE 狀態。如果本機叢集仍處於 CREATING 狀態，則您可以在 cluster.health 輸出欄位中呼叫 describe-cluster 以取得有關原因的資訊。

最常見的問題如下：

AWS Systems Manager (Systems Manager) 遇到下列問題：

- 您的叢集無法從 Systems Manager 所在的 AWS 區域 連接至控制平面執行個體。您可以從區域內的堡壘主機呼叫 `aws ssm start-session --target instance-id` 進行驗證。如果該命令不起作用，則請檢查 Systems Manager 是否在控制平面執行個體上執行。或者，另一個解決方法是刪除叢集，然後重新建立叢集。
- Systems Manager 控制平面執行個體可能無法存取網際網路。檢查您在建立叢集時提供的子網路是否具有 NAT 閘道和具有網際網路閘道的 VPC。使用 VPC Reachability Analyzer 確認控制平面執行個體是否可連線至網際網路閘道。如需詳細資訊，請參閱 Getting started with VPC Reachability Analyzer (VPC Reachability Analyzer 入門)。
- 您提供的角色 ARN 缺少政策。檢查是否已從角色移除 AWS 管理策略：AmazonEKS LocalOutpostClusterPolicy。如果 AWS CloudFormation 堆疊設定錯誤，則也可能發生這種情況。

建立叢集時錯誤設定與指定多個子網路：

- 所有提供的子網路必須與相同的 Outpost 關聯，並且必須互相連線。如果在建立叢集時指定多個子網路，Amazon EKS 會嘗試將控制平面執行個體散佈到多個子網路。
- Amazon EKS 受管的安全群組將套用至彈性網路介面。但是，其他組態元素 (例如 NACL 防火牆規則) 可能會與彈性網路介面的規則衝突。

VPC 和子網路 DNS 組態設定錯誤或遺失

檢閱 Amazon EKS 本機叢集 VPC 與子網路要求和考量事項。

無法將節點加入叢集

常見原因：

- AMI 問題：
  - 您正在使用不支援的 AMI。您必須使用 Amazon EKS 最佳化的 Amazon Linux AMI Amazon EKS 最佳化 Amazon Linux 的 v20220620 或更新版本。
  - 如果您已使用 AWS CloudFormation 範本建立您的節點，確保其沒有使用不支援的 AMI。
- 遺失 AWS IAM Authenticator `ConfigMap` – 如果遺失，您必須建立它。如需詳細資訊，請參閱將 `aws-auth` `ConfigMap` 套用至您的叢集。

- 使用錯誤的安全群組 – 確保將 `eks-cluster-sg-`*`cluster-name`*`-`*`uniqueid`* 用於您工作節點的安全群組。選取的安全群組是由 AWS CloudFormation 變更，以在每次使用堆疊時允許新的安全群組。
- 依照未預期的私人連結 VPC 步驟：傳遞了錯誤的 CA 資料 (`--b64-cluster-ca`) 或 API 端點 (`--apiserver-endpoint`)。
- 設定錯誤的 Pod 安全政策：
  - CoreDNS 和 Amazon VPC CNI plugin for Kubernetes Daemonsets 必須在加入並與叢集進行通訊的節點上執行。
  - Amazon VPC CNI plugin for Kubernetes 需要某些有權限設定的聯網功能，才能正常運作。您可以使用下列命令檢視有權限設定的聯網功能：`kubectl describe psp eks.privileged`。

  建議您不要修改預設 Pod 安全政策。如需詳細資訊，請參閱Pod 安全政策。

收集日誌

當 Outpost 與其關聯之 AWS 區域 的連線中斷時，Kubernetes 叢集可能會繼續正常運作。但是，如果叢集無法正常運作，請依照 為網路連線中斷做好準備 中的疑難排解步驟執行。如果您遇到其他問題，則請聯絡 AWS Support。AWS Support 可以指導您下載和執行日誌收集工具。如此一來，您就可以從 Kubernetes 叢集控制平面執行個體收集日誌，並將其傳送至 AWS Support 支援以供進一步調查。

控制平面執行個體無法透過 AWS Systems Manager 連線

Amazon EKS 控制平面執行個體無法透過 AWS Systems Manager (Systems Manager) 連線時，Amazon EKS 會顯示下列叢集錯誤。

```
Amazon EKS control plane instances are not reachable through SSM. Please verify your
 SSM and network configuration, and reference the EKS on Outposts troubleshooting
 documentation.
```

若要解決此問題，請確保您的 VPC 和子網路符合 Amazon EKS 本機叢集 VPC 與子網路要求和考量事項 中的要求，且您已完成《AWS Systems Manager 使用者指南》中設定工作階段管理員中的步驟。

啟動 Outpost 上自我管理的 Amazon Linux 節點

本主題會說明如何啟動 Outpost 上已向 Amazon EKS 叢集註冊的 Amazon Linux 節點的 Auto Scaling 群組。叢集可以位於前哨 AWS 雲端 或上。

必要條件

- 現有的 Outpost。如需詳細資訊，請參閱 什麼是 AWS Outposts。
- 現有 Amazon EKS 叢集。若要在上部署叢集 AWS 雲端，請參閱建立 Amazon EKS 叢集。若要在 Outpost 上部署叢集，請參閱AWS Outposts 上的 Amazon EKS 本機叢集。
- 假設您正在上的叢集中建立節點 AWS Outposts， AWS Wavelength而 AWS 雲端 且您已啟用或 L AWS ocal Zones 的子網路。 AWS 區域 然後，在您建立叢集時，不得傳入這些子網路。如果您要在 Outpost 上的叢集中建立節點，則必須在建立叢集時便已傳入 Outpost 子網路。
- (建議在上的叢集使用 AWS 雲端) 使用自己的 IAM 角色設定的附Amazon VPC CNI plugin for Kubernetes加元件，並附加了必要的 IAM 政策。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。本機叢集不支援服務帳戶的 IAM 角色。

您可以使用`eksctl`或 AWS Management Console (使用 AWS CloudFormation 範本) 建立自我管理的 Amazon Linux 節點群組。您也可使用 Terraform。

eksctl

先決條件

已在裝置或 AWS CloudShell上安裝版本 `0.183.0` 或更新版本的 `eksctl` 命令列工具。如需有關安裝或更新 `eksctl` 的指示，請參閱 `eksctl` 文件中的安裝一節。

使用 **`eksctl`** 啟動自我管理的 Linux 節點

1. 若您的叢集位於 AWS 雲端 上，且 AmazonEKS_CNI_Policy 受管 IAM 政策已連接至您的 Amazon EKS 節點 IAM 角色，建議您改為將其指派給您與 Kubernetes `aws-node` 服務帳戶相關聯的 IAM 角色。如需詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。若您的叢集位於 Outpost 上，則政策必須連接至您的節點角色。
2. 以下命令會在現有的叢集建立節點群組。叢集必須使用 `eksctl` 來建立。將 *`al-nodes`* 取代為您的節點群組名稱。節點群組名稱不可超過 63 個字元。它必須以字母或數字開頭，但剩餘字元也可以包含連字符和底線。使用您叢集的名稱取代 *`my-cluster`*。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。若您的叢集存在於 Outpost 上，請使用 Outpost 子網路的 ID 來取代 *`id`*。如果您的叢集存在於 AWS 雲端，請以建立叢集時未指定的子網路 ID 取*`id`*代。使用您 Outpost 支援的執行個體類型來取代 *`instance-type`*。以您自己的值取代其餘的 *`example values`*。依預設，會使用與控制平面相同的 Kubernetes 版本來建立節點。

使用您 Outpost 上可用的執行個體類型取代 *instance-type*。

使用 Amazon EC2 金鑰對或公有金鑰的名稱取代 *my-key*。此金鑰會在節點啟動後用於將 SSH 套用至節點。如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 金鑰對。

使用下列命令來建立您的節點群組。

```
eksctl create nodegroup --cluster my-cluster --name al-nodes --node-
type instance-type \
    --nodes 3 --nodes-min 1 --nodes-max 4 --managed=false --node-volume-type gp2
 --subnet-ids subnet-id
```

如果您的叢集部署在 AWS 雲端：

- 您部署的節點群組可以從與執行個體不同的 CIDR 區塊將 IPv4 地址指派給 Pods。如需詳細資訊，請參閱 Pod 的自訂聯網。
- 您部署的節點群組不需要對外網際網路存取。如需詳細資訊，請參閱 私有叢集要求。

如需所有可用選項和預設值的完整清單，請參閱 eksctl 文件中的 AWS Outposts 支援。

若節點無法加入叢集，則請參閱 Amazon EKS 故障診斷中的節點無法加入叢集以及 AWS Outposts 上 Amazon EKS 本機叢集疑難排解中的無法將節點加入叢集。

範例輸出如下。建立節點時，會有數行輸出。輸出的最後幾行之一類似於以下的範例行。

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (選用) 部署範例應用程式以測試您的叢集和 Linux 節點。

AWS Management Console

步驟 1：若要啟動自我管理的 Amazon Linux 節點，請使用 AWS Management Console

1. 下載最新版本的 AWS CloudFormation 模板。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/
amazon-eks-nodegroup.yaml
```

2. 開啟主 AWS CloudFormation 控台，網址為 https://console.aws.amazon.com/cloudformation。
3. 選擇 Create stack (建立堆疊)，然後選取 With new resources (standard) (使用新資源 (標準))。
4. 針對 Specify template (指定範本)，選取 Upload a template file (上傳範本檔案)，然後選取 Choose file (選擇檔案)。選取在上一步驟中下載的 `amazon-eks-nodegroup.yaml` 檔案，然後選取 Next (下一步)。
5. 在 Specify stack details (指定堆疊詳細資訊) 頁面上，據此填寫下列參數，然後選擇 Next (下一步)：

   - Stack name：為您的 AWS CloudFormation 堆疊選擇堆疊名稱。例如，您可以稱它為 ***al-nodes***。此名稱僅能使用英數字元 (區分大小寫) 和連字號。它必須以字母數字字元開頭，且長度不得超過 100 個字元。名稱在中必須是唯一的，而 AWS 區域 AWS 帳戶 且您要在中建立叢集。
   - ClusterName：輸入叢集的名稱。若此名稱與您叢集的名稱不符，則您的節點便無法加入叢集。
   - ClusterControlPlaneSecurity群組：從建立 VPC 時產生的 AWS CloudFormation 輸出中選擇SecurityGroups值。

     下列步驟顯示擷取適用群組的一種操作。

     1. 在以下網址開啟 Amazon EKS 主控台：https://console.aws.amazon.com/eks/home#/clusters。
     2. 選擇叢集的名稱。
     3. 選擇 Networking (網路) 索引標籤。
     4. 從「群組」下拉式清單中選取時，請使用其他安全性群組值作為參考。ClusterControlPlaneSecurity
   - NodeGroup名稱：輸入節點群組的名稱。此名稱稍後可用以識別為您節點建立的 Auto Scaling 節點群組。
   - NodeAutoScalingGroupMinSize：輸入節點「自動調整」群組可擴充至的最小節點數。
   - NodeAutoScalingGroupDesiredCapacity：輸入堆疊建立時要擴展到的所需節點數。
   - NodeAutoScalingGroupMaxSize：輸入節點「自動調整」群組可擴充至的節點數目上限。
   - NodeInstance類型：選擇節點的執行個體類型。如果您的叢集正在上執行 AWS 雲端，則如需詳細資訊，請參閱選擇 Amazon EC2 執行個體類型。若您的叢集正於 Outpost 上執行，則您僅可選取 Outpost 上可用的執行個體類型。

- NodeImageIDSSmParam：針對可變版本的最新 Amazon EKS 優化 AMI 預先填充了 Amazon EC2 Systems Manager 參數。Kubernetes若要使用 Amazon EKS 支援不同的 Kubernetes 次要版本，請將 *1.XX* 取代為不同的受支援版本。建議指定與叢集相同的 Kubernetes 版本。

  若要使用 Amazon EKS 最佳化加速 AMI，請將 *amazon-linux-2* 取代為 **amazon-linux-2-gpu**。若要使用 Amazon EKS 最佳化 Arm AMI，請將 *amazon-linux-2* 取代為 **amazon-linux-2-arm64**。

  > **Note**
  >
  > Amazon EKS 節點 AMI 基於 Amazon Linux。您可以透過 Amazon Linux 安全中心或是訂閱關聯的 RSS 摘要，追蹤 Amazon Linux 2 的安全或隱私權事件。安全與隱私權事件包含問題的概觀、哪些套件受到影響，以及如何更新您的執行個體以修正問題。

- NodeImageID：(選用) 如果您使用自己的自訂 AMI (而不是 Amazon EKS 最佳化 AMI)，請輸入您 AWS 區域的節點 AMI ID。如果您在此處指定一個值，它會取代 NodeImageIDSSmParam 欄位中的任何值。
- NodeVolume大小：指定節點的根磁碟區大小 (以 GiB 為單位)。
- NodeVolume類型：指定節點的根磁碟區類型。
- KeyName：輸入 Amazon EC2 安全殼層 key pair 的名稱，您可以在啟動後使用 SSH 連線到節點。如果您還沒有 Amazon EC2 金鑰對，可以在 AWS Management Console中建立一個。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 金鑰對。

  > **Note**
  >
  > 如果您未在此處提供 key pair， AWS CloudFormation 堆疊建立會失敗。

- BootstrapArguments：您可以將幾個可選參數傳遞給節點。如需詳細資訊，請檢視 GitHub 上的 bootstrap script usage information (啟動程序指令碼使用資訊)。如果要將節點新增至 Amazon EKS 本機叢集 AWS Outposts (執行Kubernetes控制平面執行個體的位置 AWS Outposts)，且叢集沒有輸入和輸出網際網路連線 (也稱為私有叢集)，則必須提供下列啟動引數 (以單行形式)。

```
--b64-cluster-ca ${CLUSTER_CA} --apiserver-endpoint https://
${APISERVER_ENDPOINT} --enable-local-outpost true --cluster-id ${CLUSTER_ID}
```

- DisableIMDSv1：預設情況下，每個節點都支援執行個體中繼資料服務版本 1 (IMDSv1) 和 IMDSv2。您可以停用 IMDSv1。若要防止節點群組中的未來節點和 Pods 使用 IMDSv1，請將 DisableIMDSv1 設定為 true。如需 IMDS 的詳細資訊，請參閱設定執行個體中繼資料服務。如需在節點上限制存取的詳細資訊，請參閱限制存取指派給工作節點的執行個體設定檔。
- VpcId：輸入您建立之 VPC 的識別碼。在選擇 VPC 之前，請先檢閱 VPC 要求和注意事項。
- 子網路：如果叢集位於 Outpost 上，則請至少在 VPC 中選擇一個私有子網路。在選擇子網路之前，請先檢閱子網路需求和注意事項。您可以看到哪些子網是私有子網，方法是從叢集的 Networking (聯網) 標籤打開每一個子網連結。

6. 請在 Configure stack options (設定堆疊選項) 頁面上選取所需的選項，然後選擇 Next (下一頁)。
7. 選取我了解 AWS CloudFormation 會建立 IAM 資源。左側的核取方塊，然後選擇 Create stack (建立堆疊)。
8. 當當堆疊已完成建立時，從主控台將其選取，然後選擇 Outputs (輸出)。
9. 記錄已建立之節點群組的「NodeInstance角色」。當您設定 Amazon EKS 節點時會需要此值。

步驟 2：讓節點加入叢集

1. 檢查以瞭解是否有 `aws-auth ConfigMap`。

```
kubectl describe configmap -n kube-system aws-auth
```

2. 如果您看到 `aws-auth ConfigMap`，請視需要更新之。

   a. 開啟 `ConfigMap` 進行編輯。

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. 視需要新增 `mapRoles` 個項目。將`rolearn`值設定為您在上一個程序中記錄的「NodeInstance角色」值。

```
[...]
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
```

```
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

c. 儲存檔案並結束您的文字編輯器。

3. 如果您收到一個錯誤，說明 "Error from server (NotFound): configmaps "aws-auth" not found，那麼請套用股票 ConfigMap。

a. 下載組態對應。

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm.yaml
```

b. 在aws-auth-cm.yaml檔案中，將設定rolearn為您在上一個程序中記錄的「NodeInstance角色」值。您可以使用文字編輯器來完成此操作，或者透過替換 *my-node-instance-role* 並執行以下命令：

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-
instance-role|' aws-auth-cm.yaml
```

c. 套用組態。此命令可能需要幾分鐘的時間來完成。

```
kubectl apply -f aws-auth-cm.yaml
```

4. 查看節點的狀態，並等待他們到達 Ready 狀態。

```
kubectl get nodes --watch
```

輸入 Ctrl+C 傳回 Shell 提示。

> **Note**
>
> 如果您收到任何授權或資源類型錯誤，請參閱故障診斷主題中的未經授權或存取遭拒 (kubectl)。

若節點無法加入叢集，則請參閱 Amazon EKS 故障診斷中的節點無法加入叢集以及 AWS Outposts 上 Amazon EKS 本機叢集疑難排解中的無法將節點加入叢集。

5. 安裝 Amazon EBS CSI 驅動程式。如需詳細資訊，請參閱安裝於 GitHub。在 Set up driver permission (設定驅動程式權限) 區段中，請務必依照 Using IAM instance profile (使用 IAM 執行個體設定檔) 選項指示操作。您必須使用 gp2 儲存類別。不支援 gp3 儲存類別。

   若要在叢集上建立 gp2 儲存類別，請完成以下步驟。

   1. 執行下列命令以建立 gp2-storage-class.yaml 檔案。

   ```
   cat >gp2-storage-class.yaml <<EOF
   apiVersion: storage.k8s.io/v1
   kind: StorageClass
   metadata:
     annotations:
       storageclass.kubernetes.io/is-default-class: "true"
     name: ebs-sc
   provisioner: ebs.csi.aws.com
   volumeBindingMode: WaitForFirstConsumer
   parameters:
     type: gp2
     encrypted: "true"
   allowVolumeExpansion: true
   EOF
   ```

   2. 將清單檔案套用至叢集。

   ```
   kubectl apply -f gp2-storage-class.yaml
   ```

6. (僅限 GPU 節點) 若選擇了 GPU 執行個體類型以及 Amazon EKS 最佳化之加速 AMI，您就必須套用 Kubernetes 專用 NVIDIA 裝置外掛程式作為叢集上的 DaemonSet。請先以您想要的 NVIDIA/k8s-device-plugin 版本來取代 *vX.X.X*，再執行下列命令。

   ```
   kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-
   plugin/vX.X.X/nvidia-device-plugin.yml
   ```

步驟 3：其他動作

1. (選用) 部署範例應用程式以測試您的叢集和 Linux 節點。
2. 若您的叢集部署在 Outpost 上，請跳過此步驟。如果您的叢集部署在上 AWS 雲端，則下列資訊為選擇性資訊。如果 AmazonEKS_CNI_Policy 受管 IAM 政策已連接至您的 Amazon EKS 節點 IAM 角色，建議您改為將其指派給您與 Kubernetes aws-node 服務帳戶相關聯的 IAM 角

色。如需更多詳細資訊，請參閱 設定為Amazon VPC CNI plugin for Kubernetes使用服務帳戶的 IAM 角色 (IRSA)。

相關專案

這些開源專案可擴展在 AWS 上或之外執行的 Kubernetes 叢集 (包括 Amazon EKS 管理的叢集) 的功能。

管理工具

Amazon EKS 和 Kubernetes 叢集的相關管理工具。

eksctl

`eksctl` 是一種簡單的 CLI 工具，可在 Amazon EKS 上建立叢集。

- 專案 URL
- 專案文件
- AWS 開放原始碼部落格：eksctl：以單一命令建立 Amazon EKS 叢集

Kubernetes 專用 AWS 控制器

藉由 Kubernetes 專用 AWS 控制器，您可以直接從 Kubernetes 叢集建立和管理 AWS 資源。

- 專案 URL
- AWS 開放原始碼部落格：Kubernetes 專用 AWS Service Operator 現已推出

Flux CD

Flux 是一個工具，您可以用於使用 Git 來管理您的叢集組態。其使用叢集中的運算子來觸發 Kubernetes 內部的部署。如需有關運算子的詳細資訊，請參閱 GitHub 上的 OperatorHub.io。

- 專案 URL
- 專案文件

Kubernetes 專用 CDK

使用 Kubernetes 專用 CDK (cdk8s)，您可以使用熟悉的程式設計語言定義 Kubernetes 應用程式和元件。cdk8s 應用程式會合成為標準的 Kubernetes 清單檔案，可套用至任何 Kubernetes 叢集。

- 專案 URL
- 專案文件
- AWS 容器部落格：cdk8s+ 簡介：適用於 Kubernetes 物件的意向驅動型 API

聯網

Amazon EKS 和 Kubernetes 叢集的相關聯網專案。

Amazon VPC CNI plugin for Kubernetes

Amazon EVPC 透過 Amazon VPC CNI plugin for Kubernetes 支援原生 VPC 聯網。外掛程式會將 VPC 中的 IP 地址指派給每個 Pod。

- 專案 URL
- 專案文件

適用於 Kubernetes 的 AWS Load Balancer Controller

AWS Load Balancer Controller 可幫助管理適用於 Kubernetes 叢集的 AWS Elastic Load Balancer。透過佈建 AWS Application Load Balancer 符合 Kubernetes 傳入資源。透過佈建 AWS Network Load Balancer 符合 Kubernetes 服務資源。

- 專案 URL
- 專案文件

ExternalDNS

ExternalDNS 將已公開的 Kubernetes 服務和輸入與 DNS 供應商 (包括 Amazon Route 53 和 AWS 服務探索) 進行同步。

- 專案 URL
- 專案文件

機器學習

Amazon EKS 和 Kubernetes 叢集的相關機器學習專案。

Kubeflow

Kubernetes 專用機器學習工具組。

- 專案 URL
- 專案文件
- AWS 開放原始碼部落格：Amazon EKS 上的 Kubeflow

Auto Scaling

Amazon EKS 和 Kubernetes 叢集的相關自動調整規模專案。

Cluster Autoscaler

Cluster Autoscaler 是一種工具，可根據 CPU 和記憶體壓力，自動調整 Kubernetes 叢集的大小。

- 專案 URL
- 專案文件
- Amazon EKS 研討會：https://www.eksworkshop.com/

Escalator

Escalator 是 Kubernetes 專用的批次或任務最佳化水平自動擴展工具。

- 專案 URL
- 專案文件

監控

Amazon EKS 和 Kubernetes 叢集的相關監控專案。

Prometheus

Prometheus 是一種開放原始碼系統監控和警示工具組。

- 專案 URL

- 專案文件
- Amazon EKS 研討會：https://eksworkshop.com/intermediate/240_monitoring/

持續整合 / 持續部署

Amazon EKS 和 Kubernetes 叢集的相關 CI/CD 專案。

Jenkins X

Amazon EKS 和 Kubernetes 叢集上適用於現代雲端應用程式的 CI/CD 解決方案。

- 專案 URL
- 專案文件

Amazon EKS 新功能和藍圖

若要了解 Amazon EKS 新功能，請捲動至 AWS 最新資訊頁面上的最新資訊摘要。您也可以在 GitHub 上檢視藍圖，讓您了解即將推出的功能和優先順序，以便規劃未來使用 Amazon EKS 的方式。您可以向我們提供有關藍圖優先順序的直接意見回饋。

Amazon EKS 的文件歷史記錄

下表說明主要更新及《Amazon EKS 使用者指南》的新內容。我們也會經常更新文件，以處理您傳送給我們的意見回饋。

變更	描述	日期
Kubernetes 版本 1.30	新增 Kubernetes 1.30 版支援新的叢集和版本升級。	2024年5月23 日
Amazon EKS 平台版本更新	這是具有安全性修正和增強功能的新平台版本。這包括新的修補程式版本的 Kubernetes 1.29.4、、1.28.9和。1.27.13	2024 年 5 月 14 日
CoreDNS 自動調度	CoreDNS自動配置器將根據節點和 CPU 核心的數量動態調整 EKS 叢集中CoreDNS部署的複本數量。此功能適用於 EKS 發行版本CoreDNSv1.9及更高版本的最新平台版本1.25。	2024 年 5 月 14 日
https://issues.amazon.com/issues/ 支 CloudWatch Container InsightsWindows	Amazon CloudWatch Observability Operator附加元件現在也可以在叢集中的Windows工作節點Container Insights上使用。	2024年4月10日
https://sim.amazon.com/issues/ Kubernetes	新增了新的庫伯尼特概念主題。	2024年4月5 日
重組存取和 IAM 內容	將與存取和 IAM 主題相關的現有頁面，例如驗證設定對應、存取項目、Pod ID 和 IRSA 移至新區段。修改概覽內容。	2024年4月2日

https://sim.amazon.com/issues/ 對 Amazon S3 序的Bottlerocket操作系統支持	Amazon S3 CSI 驅動程式的掛載點現在與. Bottlerocket	2024年3月13日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2024年3月4日
https://sim.amazon.com/issues/ https://docs.aws.amazon.com/eks/latest/userguide/eks-optimized-ami.html#al2023	Amazon Linux 2023 (AL2023) 是一種以 Linux 為基礎的全新作業系統，旨在為您的雲端應用程式提供安全、穩定且高效能的環境。	2024 年 2 月 29 日
EKS 波德身份和 IRSA 支持邊車 Kubernetes1.29	在中 Kubernetes s1.29，Amazon EKS 叢集中提供了邊車容器。服務帳戶的 IAM 角色或 EKS Pod 身分支援並行容器。如需側車的Kubernetes詳細資訊，請參閱文件中的「並行容器」。	2024年2月26日
Kubernetes 版本 1.29	新增 Kubernetes 1.29 版支援新的叢集和版本升級。	2024 年 1 月 23 日
完整版本：適Kubernetes用於版本的 Amazon EKS 擴展 Support	Kubernetes 版本的延伸支援可讓您使用特定 Kubernetes 版本超過 14 個月。	2024年1月16日

雲端中的 AWS Amazon EKS 叢集健康狀態偵測	Amazon EKS 可偵測 Amazon EKS 叢集的問題，以及叢集運作狀態中叢集先決條件的基礎設施。您可以在 EKS API 中檢視叢集中 AWS Management Console 和叢集中`health`的 EKS 叢集的問題。這些問題是主控台偵測到且由主控台顯示之問題之外的問題。之前，叢集健全狀況僅適用於上的本機叢集 AWS Outposts。	2023 年 12 月 28 日
https://sim.amazon.com/issues/ 展 AWS 區域	加拿大西部 (卡加利) (`ca-west-1`) AWS 區域現可使用 Amazon EKS。	2023 年 12 月 20 日
https://issues.amazon.com/issues/ https://docs.aws.amazon.com/eks/latest/userguide/cluster-insights.html	您現在可以從定期檢查取得有關您的叢集的建議。	2023 年 12 月 20 日
您現在可以使用存取項目向 IAM 角色和使用者授予對叢集的存取權	在引進存取項目之前，您透過在 `aws-auth ConfigMap` 中新增項目來授予 IAM 角色和使用者對叢集的存取權。現在每個叢集都有一種存取模式，您可以根據排程轉變為使用存取項目。切換模式後，您可以透過在、和 AWS SDK 中新增存取項目來新增使用者。 AWS CLI AWS CloudFormation	2023 年 12 月 18 日

Amazon EKS 平台版本更新	這是具有安全性修正和增強功能的新平台版本。這包括 Kubernetes 1.28.4、1.27.8、1.26.11 和 1.25.16 的新修補程式版本。	2023 年 12 月 12 日
適用於 Amazon S3 CSI 驅動程式的 Mountpoint	您現在可以在 Amazon EKS 叢集上安裝適用於 Amazon S3 CSI 驅動程式的 Mountpoint。	2023 年 11 月 27 日
建立叢集時開啟 Prometheus 指標	在中 AWS Management Console，您現在可以在建立叢集時開啟Prometheus指標。您還可以在可觀測性索引標籤中檢視 Prometheus 湊集器的詳細資訊。	2023 年 11 月 26 日
Amazon EKS Pod 身分識別	Amazon EKS Pod 身分識別會將 IAM 角色和 Kubernetes 服務帳戶建立關聯。透過此功能，您不再需要提供延伸許可給節點 IAM 角色。這樣，Pods在該節點上可以呼叫 AWS API。與服務帳戶的 IAM 角色不同，EKS Pod 身分識別完全位於 EKS 內部；您不需要 OIDC 身分識別提供者。	2023 年 11 月 26 日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2023 年 11 月 26 日
CSI 快照控制器	您現在可以安裝 CSI 快照控制器，以便與相容的 CSI 驅動程式 (例如 Amazon EBS CSI 驅動程式) 搭配使用。	2023 年 11 月 17 日

ADOT 運算子主題重寫	ADOT 運算子的 Amazon EKS 附加元件支援區段和 AWS Distro for OpenTelemetry 文件為冗餘。我們將剩餘的重要資訊遷移到該資源中，以減少過時和不一致的資訊。	2023 年 11 月 14 日
適用於 Prometheus 的 CoreDNS EKS 附加元件支援指標	適用於 CoreDNS 的 EKS 附加元件的 `v1.10.1-eksbuild.5` 、`v1.9.3-eksbuild.9` 和 `v1.8.7-eksbuild.8` 版本，會在 `kube-dns` 服務中公開CoreDNS 發布指標的連接埠。這樣可以更輕鬆地在監控系統中包含 CoreDNS 指標。	2023 年 11 月 10 日
Amazon EKS CloudWatch 可觀測運算符附加	增加了 Amazon EKS CloudWatch 可觀測運算符頁面。	2023 年 11 月 6 日
美國東部 (俄亥俄) 推出適用於自我管理 P5 執行個體的容量區塊	在美國東部 (俄亥俄)，您現在可以將容量區塊用於自我管理 P5 執行個體。	2023 年 10 月 31 日
叢集支援修改子網路和安全群組	您可以更新叢集，變更叢集使用的子網路和安全群組。您可以從 AWS Management Console、、和版本的最新版本`v0.164.0-rc.0` 或更新`eksctl`版本進行更新。AWS CLI AWS CloudFormation這麼做才能為子網路提供更多可用 IP 地址，以成功升級叢集版本。	2023 年 10 月 24 日

叢集角色和受管節點群組角色支援客戶管理的 AWS Identity and Access Management 政策	您可以在叢集角色上使用自訂 IAM 政策，而不是受AmazonEKSClusterPolicy AWS 管政策。此外，您可以在受管節點群組中的節點角色上使用自訂 IAM 政策，而不是受AmazonEKSWorkerNodePolicy AWS 管政策。為符合嚴格的合規要求，請建立具有最低權限的政策。	2023 年 10 月 23 日
修復 eksctl 的安裝連結	在頁面移動後修復 eksctl 的安裝連結。	2023 年 10 月 6 日
預覽版本：Amazon EKS Kubernetes 版本的延伸支援	Kubernetes 版本的延伸支援可讓您使用特定 Kubernetes 版本超過 14 個月。	2023 年 10 月 4 日
移除 AWS App Mesh 整合的參照	Amazon EKS 整合僅適 AWS App Mesh 用於 App Mesh 的現有客戶。	2023 年 9 月 29 日
Kubernetes 版本 1.28	新增 Kubernetes 1.28 版支援新的叢集和版本升級。	2023 年 9 月 26 日
現有叢集支援在 Amazon VPC CNI plugin for Kubernetes 強制執行 Kubernetes 網路政策	您可搭配 Amazon VPC CNI plugin for Kubernetes 在現有叢集使用 Kubernetes網路政策，而不需要第三方解決方案。	2023 年 9 月 15 日

CoreDNS Amazon EKS 附加元件支援修改 PDB	您可在版本 `v1.9.3-eksbuild.7` 與更新版本，以及版本 `v1.10.1-eksbuild.4` 與更新版本，針對 CoreDNS 的 EKS 附加元件修改 `PodDisruptionBudget` 。	2023 年 9 月 15 日
Amazon EKS 支援共用子網路	用來在共用子網路中製作 Amazon EKS 叢集的新共用子網路要求與考量。	2023 年 9 月 7 日
「什麼是Amazon EKS？」的更新內容	加入了新的常見使用案例和架構主題。重新整理其他主題。	2023 年 9 月 6 日
在 Amazon VPC CNI plugin for Kubernetes 中強制執行 Kubernetes 網路政策	您可以搭配 Amazon VPC CNI plugin for Kubernetes 來使用 Kubernetes 網路政策，代替第三方解決方案的需要。	2023 年 8 月 29 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已可在以色列 (特拉維夫) (`il-central-1`) AWS 區域取得。	2023 年 8 月 1 日
適用於 Fargate 的可設定暫時性儲存	您可以增加在 Amazon EKS Fargate 上執行的每個 Pod 之暫時性儲存的總量。	2023 年 7 月 31 日
Amazon EFS CSI 驅動程式的附加元件支援	您現在可以使用 AWS Management Console AWS CLI、和 API 來管理 Amazon EFS CSI 驅動程式。	2023 年 7 月 26 日
AWS 受管策略更新-新策略	Amazon EKS 添加了新的 AWS 受管政策。	2023 年 7 月 26 日

Kubernetes1.27、1.26、1.25 和 1.24 的版本更新現在可供以下位置的本機叢集使用： AWS Outposts	Kubernetes1.27.3、1.26.6、1.25.11 和 1.24.15 的版本更新現在可供下列本機叢集使用： AWS Outposts	2023 年 7 月 20 日
IP 字首支援 Windows 節點	相較於將個別的次要 IP 地址指派給節點，將 IP 字首指派給節點可讓您在節點上託管更多數量的 Pods。	2023 年 7 月 6 日
Amazon FSx for OpenZFS CSI 驅動程式	您現在可以在 Amazon EKS 叢集上安裝 Amazon FSx for OpenZFS CSI 驅動程式。	2023 年 6 月 30 日
IPv4 叢集中 Linux 節點上的 Pods 現在可與 IPv6 端點進行通訊。	將 IPv6 地址指派給節點後，您 Pods 的 IPv4 地址就是將網路地址轉換為其執行所在節點的 IPv6 地址。	2023 年 6 月 19 日
Windows受管理節點群組 AWS GovCloud (US) Regions	在中 AWS GovCloud (US) Regions，Amazon EKS 受管節點群組現在可以執行Windows容器。	2023 年 5 月 30 日
Kubernetes 版本 1.27	新增 Kubernetes 1.27 版支援新的叢集和版本升級。	2023 年 5 月 24 日
Kubernetes 版本 1.26	新增 Kubernetes 1.26 版支援新的叢集和版本升級。	2023 年 4 月 11 日
無網域 gMSA	您現在可以將無網域 gMSA 與 Windows Pods 搭配使用。	2023 年 3 月 27 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在亞太區域（墨爾本）（ap-southeast-4 ）AWS 區域上市。	2023 年 3 月 10 日

Amazon File Cache CSI 驅動程式	您現在可以在 Amazon EKS 叢集上安裝 Amazon File Cache CSI 驅動程式。	2023 年 3 月 3 日
Kubernetes版本 1.25 現在可用於以下位置的本地叢集 AWS Outposts	現在使用 Kubernetes 版本 1.22 – 1.25，可在 Outpost 上建立 Amazon EKS 本機叢集。	2023 年 3 月 1 日
Kubernetes 版本 1.25	新增 Kubernetes 1.25 版支援新的叢集和版本升級。	2023 年 2 月 22 日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2023 年 2 月 7 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在亞太區域 (海德拉巴) (ap-south-2)、歐洲 (蘇黎世) (eu-centra 1-2) 和歐洲 (西班牙) (eu-south-2) AWS 區域推出。	2023 年 2 月 6 日
Kubernetes版本 1.21 — 現1.24在可用於上的本機叢集 AWS Outposts。	現在使用 Kubernetes 版本 1.21 – 1.24，可在 Outpost 上建立 Amazon EKS 本機叢集。以前，只有 1.21 版本可用。	2023 年 1 月 17 日
Amazon EKS 現在支持 AWS PrivateLink	您可以使 AWS PrivateLink 用在 VPC 和 Amazon EKS 之間建立私有連接。	2022 年 12 月 16 日
受管節點群組 Windows 支援	您現在可以將 Windows 用於 Amazon EKS 受管節點群組。	2022 年 12 月 15 日
來自獨立軟體廠商的 Amazon EKS 附加元件現在可於 AWS Marketplace中取得	您現在可以透過 AWS Marketplace來瀏覽和訂閱來自獨立軟體廠商的 Amazon EKS 附加元件。	2022 年 11 月 28 日

AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2022 年 11 月 17 日
Kubernetes 版本 1.24	新增 Kubernetes 1.24 版支援新的叢集和版本升級。	2022 年 11 月 15 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在中東（阿聯酋）（me-central-1 ）AWS 區域提供。	2022 年 11 月 3 日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2022 年 10 月 24 日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2022 年 10 月 20 日
上的本機叢集現 AWS Outposts 在可供使用	您現在可以在 Outpost 上建立 Amazon EKS 本機叢集。	2022 年 9 月 19 日
Fargate vCPU 型配額	Fargate 正在從 Pod 型配額轉換為 vCPU 型配額。	2022 年 9 月 8 日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2022 年 8 月 31 日
成本監控	Amazon EKS 現在支援 Kubecost，可讓您監控按 Kubernetes 資源 (包括 Pods、節點、命名空間和標籤) 細分的成本。	2022 年 8 月 24 日
AWS 受管策略更新-新策略	Amazon EKS 添加了新的 AWS 受管政策。	2022 年 8 月 24 日
AWS 受管策略更新-新策略	Amazon EKS 添加了新的 AWS 受管政策。	2022 年 8 月 23 日

帳單的標籤資源	對所有叢集新增了 `aws:eks:cluster-name` 產生的成本分配標籤支援。	2022 年 8 月 16 日
Fargate 設定檔萬用字元	在命名空間、標籤索引鍵和標籤值的選擇器條件中，新增了對 Fargate 設定檔萬用字元的支援。	2022 年 8 月 16 日
Kubernetes 版本 `1.23`	新增 Kubernetes `1.23` 版支援新的叢集和版本升級。	2022 年 8 月 11 日
檢視中的Kubernetes資源 AWS Management Console	您現在可以使用 AWS Management Console來檢視有關部署到叢集的 Kubernetes 資源的資訊。	2022 年 5 月 3 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在亞太區域（雅加達）（`ap-southeast-3`）AWS 區域提供。	2022 年 5 月 2 日
可觀測性頁面和 ADOT 附加元件支援	為（ADOT）添加了可觀察性頁面和發 AWS OpenTelemetry 行版。	2022 年 4 月 21 日
Kubernetes 版本 `1.22`	新增 Kubernetes `1.22` 版支援新的叢集和版本升級。	2022 年 4 月 4 日
AWS 受管策略更新-新策略	Amazon EKS 添加了新的 AWS 受管政策。	2022 年 4 月 4 日
新增 Fargate Pod 修補詳細資訊	升級 Fargate Pods 時，Amazon EKS 會先嘗試根據您的 Pod 中斷預算移出 Pods。您可以建立事件規則，以便在刪除 Pods 之前對失敗的移出事件做出反應。	2022 年 4 月 1 日

完整版本：Amazon EBS CSI 驅動程式的附加元件支援	您現在可以使用 AWS Management Console AWS CLI、和 API 來管理 Amazon EBS CSI 驅動程式。	2022 年 3 月 31 日
AWS Outposts 內容更新	在 AWS Outposts上部署 Amazon EKS 叢集。	2022 年 3 月 22 日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2022 年 3 月 21 日
Windows `containerd` 支援	您現在可以選擇 Windows 節點的 `containerd` 執行時間。	2022 年 3 月 14 日
在安全文件中新增 Amazon EKS 連接器考量事項	描述與連接叢集相關的共同責任模型。	2022 年 2 月 25 日
指派 IPv6 地址至您的 Pods 和服務	您現在可以建立 `1.21` 或更新版本的叢集，其會將 IPv6 地址指派至您的 Pods 和服務。	2022 年 1 月 6 日
AWS 受管策略更新-現有策略的更新	Amazon EKS 更新了現有的 AWS 受管政策。	2021 年 12 月 13 日
預覽版本：Amazon EBS CSI 驅動程式的附加元件支援	您現在可以使用 AWS Management Console、和 API 進行預覽 AWS CLI，以管理 Amazon EBS CSI 驅動程式。	2021 年 12 月 9 日
Karpenter Autoscaler 支援	您現在可以使用 Karpenter 開放原始碼專案來自動調整節點。	2021 年 11 月 29 日
Fargate 記錄支援 Fluent Bit Kubernetes 篩選條件	您現在可以將 Fargate 記錄與 Fluent Bit Kubernetes 篩選條件搭配使用。	2021 年 11 月 10 日

控制平面提供 Windows 支援	您現可在控制平面中使用 Windows 支援。您不再需要於資料平面中將其啟用。	2021 年 11 月 9 日
新增 Bottlerocket 作為受管節點群組的 AMI 類型	Bottlerocket 以往僅能作為自我管理節點選項使用。現在可將其設定為受管節點群組，進而減少為滿足節點合規性要求要耗費的精力。	2021 年 10 月 28 日
DL1 驅動程式支援	自訂 Amazon Linux AMI 現在支援 Amazon Linux 2 深度學習工作負載。此支援操作允許一般內部部署或雲端基準組態設定。	2021 年 10 月 25 日
VT1 視訊支援	自訂 Amazon Linux AMI 現在支援某些發行版本的 VT1。此支援功能會在您的 Amazon EKS 叢集上公告 Xilinx U30 裝置。	2021 年 9 月 13 日
Amazon EKS Connector 現已推出	您可以使用 Amazon EKS 連接器在 Amazon EKS 主控台中註冊和連接任何一致性Kubernetes叢集， AWS 並將其視覺化。	2021 年 9 月 8 日
Amazon EKS Anywhere 現已推出	Amazon EKS Anywhere 是 Amazon EKS 的新部署選項，您可輕鬆地在內部部署建立和操作 Kubernetes 叢集。	2021 年 9 月 8 日
適用於 NetApp ONTAP CSI 驅動程式的 Amazon FSx	已新增概述適用於 NetApp ONTAP CSI 驅動程式的 Amazon FSx 的主題，並提供其他參考資料的連結。	2021 年 9 月 2 日

受管節點群組現在會自動計算 Amazon EKS 建議的節點 Pods 數量上限	受管節點群組現在會針對您在沒有啟動範本的情況下部署的節點，或使用未在其中指定 AMI ID 的啟動範本所部署的節點，自動計算其 Amazon EKS 最大 Pods 數。	2021 年 8 月 30 日
移除附加元件設定的 Amazon EKS 管理，而不移除 Amazon EKS 附加元件軟體	您現在可以移除 Amazon EKS 附加元件，而無需從叢集移除附加元件軟體。	2021 年 8 月 20 日
使用 Multus 建立多重主目錄 Pods	您現在可以使用 Multus 將多個網路介面新增至 Pod。	2021 年 8 月 2 日
將更多 IP 地址新增至您的 Linux Amazon EC2 節點	您現在可以將更多的 IP 地址新增至您的 Linux Amazon EC2 節點。這表示您可以在每個節點上執行更多數量的 Pods。	2021 年 7 月 27 日
`containerd` 執行階段啟動程序	Amazon EKS 最佳化加速 Amazon Linux Amazon Machine Image (AMI) 現在包含啟動程序旗標，您可啟用 Amazon EKS 最佳化和 Bottlerocket AMI 中的 `containerd` 執行時間。此旗標可用於所有支援 Kubernetes 版本的 AMI 中。	2021 年 7 月 19 日
Kubernetes 版本 `1.21`	已新增 Kubernetes 版本 `1.21` 支援。	2021 年 7 月 19 日
新增受管政策主題	自 2021 年 6 月 17 日起所有 Amazon EKS IAM 受管政策以及已對其進行變更的清單。	2021 年 6 月 17 日

將 Pods 安全群組搭配 Fargate 使用	除了搭配 Amazon EC2 節點使用外，您現在可以將 Pods 安全群組與 Fargate 搭配使用。	2021 年 6 月 1 日
已新增 CoreDNS 和 kube-proxy Amazon EKS 附加元件	Amazon EKS 現在可以幫助您管理叢集的 CoreDNS 和 kube-proxy Amazon EKS 附加元件。	2021 年 5 月 19 日
Kubernetes 版本 1.20	新增 Kubernetes 1.20 版支援新的叢集和版本升級。	2021 年 5 月 18 日
發行 AWS Load Balancer Controller2.2.0	您現在可以使用 AWS Load Balancer Controller 建立可使用執行個體或 IP 目標的 Elastic Load Balancer。	2021 年 5 月 14 日
受管節點群組的節點污點	Amazon EKS 現在支援將備註污點新增至受管節點群組。	2021 年 5 月 11 日
現有叢集的秘密加密	Amazon EKS 現在支援為現有叢集新增秘密加密。	2021 年 2 月 26 日
Kubernetes 版本 1.19	新增 Kubernetes 1.19 版支援新的叢集和版本升級。	2021 年 2 月 16 日
Amazon EKS 現在支援 OpenID Connect (OIDC) 身分提供者，以做為對 1.16 或更新版本叢集驗證使用者身分的方法。	OIDC 身分提供者可搭配使用，或作為 AWS Identity and Access Management (IAM) 的備用選項。	2021 年 2 月 12 日
檢視節點和工作負載資源 AWS Management Console	您現在可以在 AWS Management Console中檢視有關您的受管、自我管理和 Fargate 節點以及已部署 Kubernetes 工作負載的詳細資訊。	2020 年 12 月 1 日

在受管節點群組中部署 Spot 執行個體類型	您現在可以將多個 Spot 或隨需執行個體類型部署到受管節點群組。	2020 年 12 月 1 日
Amazon EKS 現在可以管理叢集的特定附加元件	您可以自行管理附加元件，或允許 Amazon EKS 透過 Amazon EKS API 控制附加元件的啟動和版本。	2020 年 12 月 1 日
在多個輸入之間共用 ALB	您現在可以跨多個Kubernetes導入共 Ap AWS plication Load Balancer (ALB)。在過去，您必須為每個輸入部署個別的 ALB。	2020 年 10 月 23 日
NLB IP 目標支援	您現在可以部署具有 IP 目標的 Network Load Balancer。這表示您可使用 NLB 將網路流量負載平衡至 Fargate Pods，並直接負載平衡至 Amazon EC2 節點上執行的 Pods。	2020 年 10 月 23 日
Kubernetes 版本 1.18	新增 Kubernetes 1.18 版支援新的叢集和版本升級。	2020 年 10 月 13 日
為 Kubernetes 服務 IP 地址指派指定自訂 CIDR 區塊。	您現在可以指定 Kubernetes 從中指派服務 IP 地址的自訂 CIDR 區塊。	2020 年 9 月 29 日
將安全群組指派至個別 Pods	您現在可以將不同的安全群組，與在眾多 Amazon EC2 執行個體類型上執行的一些個別 Pods 建立關聯。	2020 年 9 月 9 日
在節點上部署 Bottlerocket	您現在可以部署執行 Bottlerocket 的節點。	2020 年 8 月 31 日

啟動 Arm 節點的功能已正式可用	您現在可以在受管節點群組和自我管理節點群組中啟動 Arm 節點。	2020 年 8 月 17 日
受管節點群組啟動範本和自訂 AMI	現在可以使用 Amazon EC2 啟動範本部署受管節點群組。您可以自行選擇是否為啟動範本指定自訂 AMI。	2020 年 8 月 17 日
EFS 支援 AWS Fargate	您現在可以搭配使用 Amazon EFS AWS Fargate。	2020 年 8 月 17 日
Amazon EKS 平台版本更新	這是具有安全性修正和增強功能的新平台版本。這包括在使用具有 Kubernetes 版本 `1.15` 或更高版本的 Network Load Balancer 時對 `LoadBalancer` 類型服務的 UDP 支援。如需詳細資訊，請參閱上GitHub的 < 允許 AWS Network Load Balancer 的 UDP 問題 > 。	2020 年 8 月 12 日
Amazon EKS 擴張 AWS 區域	非洲 (開普敦) (`af-south-1`) 和歐洲 (米蘭) (`eu-south-1`) AWS 區域現可使用 Amazon EKS。	2020 年 8 月 6 日
Fargate 用量指標	AWS Fargate 提供 CloudWatch 使用指標，可讓您了解您帳戶對 Fargate 隨需資源的使用情況。	2020 年 8 月 3 日
Kubernetes 版本 `1.17`	新增 Kubernetes `1.17` 版支援新的叢集和版本升級。	2020 年 7 月 10 日

透過 Kubernetes 的 App Mesh 控制器建立及管理 Kubernetes 內部的 App Mesh 資源	您可以從 Kubernetes 內部建立和管理 App Mesh 資源。控制器也會自動將 Envoy 代理和初始化容器注入至您部署的 Pods。	2020 年 6 月 18 日
Amazon EKS 現在支援 Amazon EC2 Inf1 節點	您可以將 Amazon EC2 Inf1 節點新增至叢集。	2020 年 6 月 4 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在 AWS GovCloud (美國東部) () 和 (美國西部us-gov-east-1) AWS GovCloud () 提供。us-gov-west-1 AWS 區域	2020 年 5 月 13 日
Amazon EKS 已不再支援 Kubernetes1.12	Amazon EKS 已不再支援 Kubernetes 1.12 版。請將任何 1.12 版叢集更新至 1.13 或更新版本，避免服務中斷。	2020 年 5 月 12 日
Kubernetes 版本 1.16	新增 Kubernetes 1.16 版支援新的叢集和版本升級。	2020 年 4 月 30 日
已新增AWSServiceRoleForAmazonEKS服務連結角色	已新增服AWSServiceRoleForAmazonEKS務連結角色。	2020 年 4 月 16 日
Kubernetes 版本 1.15	新增 Kubernetes 1.15 版支援新的叢集和版本升級。	2020 年 3 月 10 日
Amazon EKS 擴張 AWS 區域	北京 (cn-north-1) 和寧夏 (cn-northwest-1) AWS 區域現可使用 Amazon EKS。	2020 年 2 月 26 日
FSx for Lustre CSI 驅動程式	新增在 Kubernetes 1.14 Amazon EKS 叢集上安裝 FSx for Lustre CSI 驅動程式的相關主題。	2019 年 12 月 23 日

限制對叢集公有存取端點的網路存取	透過此更新，您可以使用 Amazon EKS 來限制與 Kubernetes API 伺服器之公有存取端點通訊的 CIDR 範圍。	2019 年 12 月 20 日
從 VPC 外部解析叢集的私有存取端點地址	透過此更新，您可以使用 Amazon EKS 從 VPC 外部解析 Kubernetes API 伺服器的私有存取端點。	2019 年 12 月 13 日
(Beta 版) Amazon EC2 A1 Amazon EC2 執行個體節點	啟動向您的 Amazon EKS 叢集註冊的 Amazon EC2 A1 Amazon EC2 執行個體節點。	2019 年 12 月 4 日
在 AWS Outposts 上創建集群	Amazon EKS 現在支援在 AWS Outposts上建立叢集。	2019 年 12 月 3 日
AWS Fargate 在 Amazon EKS 上	Amazon EKS Kubernetes 叢集現在支援在 Fargate 上執行 Pods。	2019 年 12 月 3 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在加拿大（中部）（`ca-central-1` ）AWS 區域提供。	2019 年 11 月 21 日
受管節點群組	Amazon EKS 受管節點群組會自動化 Amazon EKS Kubernetes 叢集節點 (Amazon EC2 執行個體) 的佈建和生命週期管理。	2019 年 11 月 18 日
Amazon EKS 平台版本更新	可解決 `CVE-2019-11253` 問題的新平台版本。	2019 年 11 月 6 日
Amazon EKS 已不再支援 Kubernetes`1.11`	Amazon EKS 已不再支援 Kubernetes `1.11` 版。請將任何 `1.11` 版叢集更新至 `1.12` 或更新版本，避免服務中斷。	2019 年 11 月 4 日

Amazon EKS 擴張 AWS 區域	南美洲 (聖保羅) (sa-east-1) AWS 區域現可使用 Amazon EKS。	2019 年 10 月 16 日
Windows 支援	執行 1.14 版 Kubernetes 的 Amazon EKS 叢集現在支援 Windows 工作負載。	2019 年 10 月 7 日
自動擴展	新增章節以涵蓋 Amazon EKS 叢集上支援的一些不同類型的 Kubernetes 自動擴展。	2019 年 9 月 30 日
Kubernetes 儀表板更新	已更新在 Amazon EKS 叢集上安裝 Kubernetes 儀表板以使用 Beta 2.0 版的相關主題。	2019 年 9 月 28 日
Amazon EFS CSI 驅動程式	已新增在 Kubernetes 1.14 Amazon EKS 叢集上安裝 Amazon EFS CSI 驅動程式的相關主題。	2019 年 9 月 19 日
Amazon EKS 最佳化 AMI ID 的 Amazon EC2 Systems Manager 參數	已新增使用 Amazon EC2 Systems Manager 參數擷取 Amazon EKS 最佳化 AMI ID 的相關主題。此參數讓您無需查詢 AMI ID。	2019 年 9 月 18 日
Amazon EKS 資源標記	您可管理 Amazon EKS 叢集的標記。	2019 年 9 月 16 日
Amazon EBS CSI 驅動程式	已新增在 Kubernetes 1.14 Amazon EKS 叢集上安裝 Amazon EBS CSI 驅動程式的相關主題。	2019 年 9 月 9 日

全新 Amazon EKS 最佳化 AMI 為 CVE-2019-9512 和 CVE-2019-9514 進行修補	Amazon EKS 已更新 Amazon EKS 最佳化 AMI，可解決 CVE-2019-9512 和 CVE-2019-9514 問題。	2019 年 9 月 6 日
宣布 Amazon EKS 廢除 Kubernetes1.11	Amazon EKS 已在 2019 年 11 月 4 日停止支援 Kubernetes 1.11 版。	2019 年 9 月 4 日
Kubernetes 版本 1.14	新增 Kubernetes 1.14 版支援新的叢集和版本升級。	2019 年 9 月 3 日
服務帳戶的 IAM 角色	透過 Amazon EKS 叢集上服務帳戶的 IAM 角色，您可以建立 IAM 角色與 Kubernetes 服務帳戶的關聯。透過此功能，您不再需要提供延伸許可給節點 IAM 角色。這樣，Pods在該節點上可以呼叫 AWS API。	2019 年 9 月 3 日
Amazon EKS 擴張 AWS 區域	中東 (巴林) (me-south-1) AWS 區域現可使用 Amazon EKS。	2019 年 8 月 29 日
Amazon EKS 平台版本更新	可解決 CVE-2019-9512 和 CVE-2019-9514 問題的新平台版本。	2019 年 8 月 28 日
Amazon EKS 平台版本更新	可解決 CVE-2019-11247 和 CVE-2019-11249 問題的新平台版本。	2019 年 8 月 5 日
Amazon EKS 區域擴展	亞太區域 (香港) (ap-east-1) AWS 區域現可使用 Amazon EKS。	2019 年 7 月 31 日

Amazon EKS 已不再支援 Kubernetes1.10	Amazon EKS 已不再支援 Kubernetes 1.10 版。請將任何 1.10 版叢集更新至 1.11 或更新版本，避免服務中斷。	2019 年 7 月 30 日
在 ALB 傳入控制器上新增主題	的 AWS ALB 入口控制器Kubernetes是一個控制器，可在建立輸入資源時建立 ALB。	2019 年 7 月 11 日
全新 Amazon EKS 最佳化 AMI	從 AMI 移除不必要的 kubectl 二進位檔。	2019 年 7 月 3 日
Kubernetes 版本 1.13	新增 Kubernetes 1.13 版支援新的叢集和版本升級。	2019 年 6 月 18 日
全新 Amazon EKS 最佳化 AMI 為 AWS-2019-005 進行修補	Amazon EKS 已更新 Amazon EKS 最佳化 AMI，可解決 AWS-2019-005 中所述的安全漏洞。	2019 年 6 月 17 日
宣布 Amazon EKS 停止支援 Kubernetes1.10	Amazon EKS 已在 2019 年 7 月 22 日停止支援 Kubernetes 1.10 版。	2019 年 5 月 21 日
Amazon EKS 平台版本更新	全新推出適用於 Kubernetes 1.11 和 1.10 叢集的平台版本，可供支援 kubelet 憑證中的自訂 DNS 名稱和提升 etcd 效能。	2019 年 5 月 21 日

AWS CLIget-token 命令	aws eks get-token 命令已新增至 AWS CLI。您不再需要安裝 AWS IAM 身份驗證器，即可為Kubernetes叢集 API 伺服器通訊建立用戶端安全性權杖。將您的 AWS CLI 安裝升級到最新版本以使用此新功能。如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的安裝 AWS Command Line Interface。	2019 年 5 月 10 日
eksctl 入門	本入門指南說明如何使用 eksctl 來安裝 Amazon EKS 入門所有所需的資源。這是簡單的命令列公用程式，可用於在 Amazon EKS 上建立和管理 Kubernetes 叢集。	2019 年 5 月 10 日
Amazon EKS 平台版本更新	全新推出 Kubernetes 1.12 叢集的平台版本，可支援 kubelet 憑證的自訂 DNS 名稱和提升 etcd 效能。這個版本修正了造成節點 kubelet 協助程式每隔幾秒便發出請求新憑證的錯誤。	2019 年 5 月 8 日
Prometheus 教學課程	新增了有關將 Prometheus 部署至 Amazon EKS 叢集的主題。	2019 年 4 月 5 日

Amazon EKS 控制平面記錄	透過此更新，您可以從 Amazon EKS 控制窗格中，直接取得稽核和診斷日誌。您可以在帳戶中使用這些 CloudWatch 記錄做為保護和執行叢集的參考。	2019 年 4 月 4 日
Kubernetes 版本 1.12	新增 Kubernetes 1.12 版支援新的叢集和版本升級。	2019 年 3 月 28 日
新增 App Mesh 入門指南	新增 App Mesh 和 Kubernetes 入門文件。	2019 年 3 月 27 日
Amazon EKS API 伺服器端點私有存取	新增有關停用公有存取 Amazon EKS 叢集的 Kubernetes API 伺服器端點的文件。	2019 年 3 月 19 日
新增有關安裝 Kubernetes 指標伺服器的主題	Kubernetes 指標伺服器是叢集中資源使用狀況資料的彙總工具。	2019 年 3 月 18 日
新增相關的開放原始碼專案清單	這些開放原始碼專案擴充執行 Kubernetes叢集的功能 AWS，包括由 Amazon EKS 管理的叢集。	2019 年 3 月 15 日
新增有關本機安裝 Helm 的主題	適用於 Kubernetes 的 `helm` 套件管理工具可協助您在 Kubernetes 叢集上安裝和管理應用程式。本主題展示如何本機安裝和執行 `helm` 和 `tiller` 二進位檔案。這樣，您可以在本機系統上使用 Helm CLI 安裝和管理圖表。	2019 年 3 月 11 日

Amazon EKS 平台版本更新	新平台版本會將 Amazon EKS Kubernetes 1.11 叢集更新為修補程式等級 1.11.8，可解決 CVE-2019-1002100 問題。	2019 年 3 月 8 日
已提高叢集限制	Amazon EKS 已將您可在 AWS 區域 內建立的叢集數量從 3 增加到 50。	2019 年 2 月 13 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在歐洲 (倫敦) (eu-west-2)、歐洲 (巴黎) (eu-west-3) 和亞太區域 (孟買) (ap-south-1) AWS 區域推出。	2019 年 2 月 13 日
全新 Amazon EKS 最佳化 AMI 為 ALAS-2019-1156 進行修補	Amazon EKS 已更新 Amazon EKS 最佳化 AMI，可解決 ALAS-2019-1156 中所述的安全漏洞。	2019 年 2 月 11 日
全新 Amazon EKS 最佳化 AMI 為 ALAS2-2019-1141 進行修補	Amazon EKS 已更新 Amazon EKS 最佳化 AMI，可解決 ALAS2-2019-1141 所述的 CVE 問題。	2019 年 1 月 9 日
Amazon EKS 擴張 AWS 區域	Amazon EKS 現已在亞太區域（首爾）（ap-northeast-2 ）AWS 區域上市。	2019 年 1 月 9 日
Amazon EKS 區域擴展	Amazon EKS 現已推出以下附加產品 AWS 區域：歐洲 (法蘭克福) (eu-central-1)、亞太區域 (東京) (ap-northeast-1)、亞太區域 (新加坡) (ap-southeast-1) 和亞太區域 (雪梨) (ap-southeast-2)。	2018 年 12 月 19 日

Amazon EKS 叢集更新	已針對 Amazon EKS 叢集 Kubernetes 版本更新和節點取代。	2018 年 12 月 12 日
Amazon EKS 擴張 AWS 區域	歐洲 (斯德哥爾摩) (eu-north-1) AWS 區域現可使用 Amazon EKS。	2018 年 12 月 11 日
Amazon EKS 平台版本更新	新的平台版本更新 Kubernetes 至修補程式層級 1.10.11 以解決 CVE-2018-1002105 。	2018 年 12 月 4 日
新增支援 ALB 傳入控制器的 1.0.0 版	ALB 入口控制器發布了正式支持1.0.0的版本。 AWS	2018 年 11 月 20 日
新增對 CNI 網路組態的支援	Amazon VPC CNI plugin for Kubernetes 1.2.1 版現已針對輔助 Pod 網路介面支援自訂網路組態。	2018 年 10 月 16 日
新增對 MutatingAdmissionWebhook 和 ValidatingAdmissionWebhook 的支援	Amazon EKS 平台版本 1.10-eks.2 現已支援 MutatingAdmissionWebhook 和 ValidatingAdmissionWebhook 許可控制器。	2018 年 10 月 10 日
新增合作夥伴 AMI 資訊	Canonical 已與 Amazon EKS 合作，建立了可供您用於叢集的節點 AMI。	2018 年 10 月 3 日
已新增 AWS CLIupdate-kubeconfig 命令的指示	Amazon EKS 已新增update-kubeconfig 至，以簡化建立kubeconfig 檔案 AWS CLI 以存取叢集的程序。	2018 年 9 月 21 日

全新 Amazon EKS 最佳化 AMI	Amazon EKS 已更新 Amazon EKS 最佳化 AMI (支援及未支援 GPU)，提供了各項安全性修正和 AMI 最佳化。	2018 年 9 月 13 日
Amazon EKS 擴張 AWS 區域	歐洲 (愛爾蘭) (`eu-west-1`) 區域現可使用 Amazon EKS。	2018 年 9 月 5 日
Amazon EKS 平台版本更新	新的平台版本支援 Kubernetes 彙整層和 Horizontal Pod Autoscaler(HPA)。	2018 年 8 月 31 日
全新 Amazon EKS 最佳化 AMI 與 GPU 支援	Amazon EKS 已經更新了 Amazon EKS 優化 AMI，以使用新的 AWS CloudFormation 節點模板和引導腳本。此外，還推出了全新的支援 GPU 的 Amazon EKS 最佳化 AMI。	2018 年 8 月 22 日
全新 Amazon EKS 最佳化 AMI 為 `ALAS2-2018-1058` 進行修補	Amazon EKS 已更新 Amazon EKS 最佳化 AMI，可解決 `ALAS2-2018-1058` 所述的 CVE 問題。	2018 年 8 月 14 日
Amazon EKS 最佳化之 AMI 建置指令碼	Amazon EKS 已採用開放原始碼的形式提供建置指令碼，可用於建置 Amazon EKS 最佳化 AMI。這些建置指令碼現可於 GitHub 取得。	2018 年 7 月 10 日
Amazon EKS 初始版本	服務啟動的初始文件	2018 年 6 月 5 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。