使用者指南

AWS 電信網路建置器

AWS 電信網路建置器: 使用者指南

Table of Contents

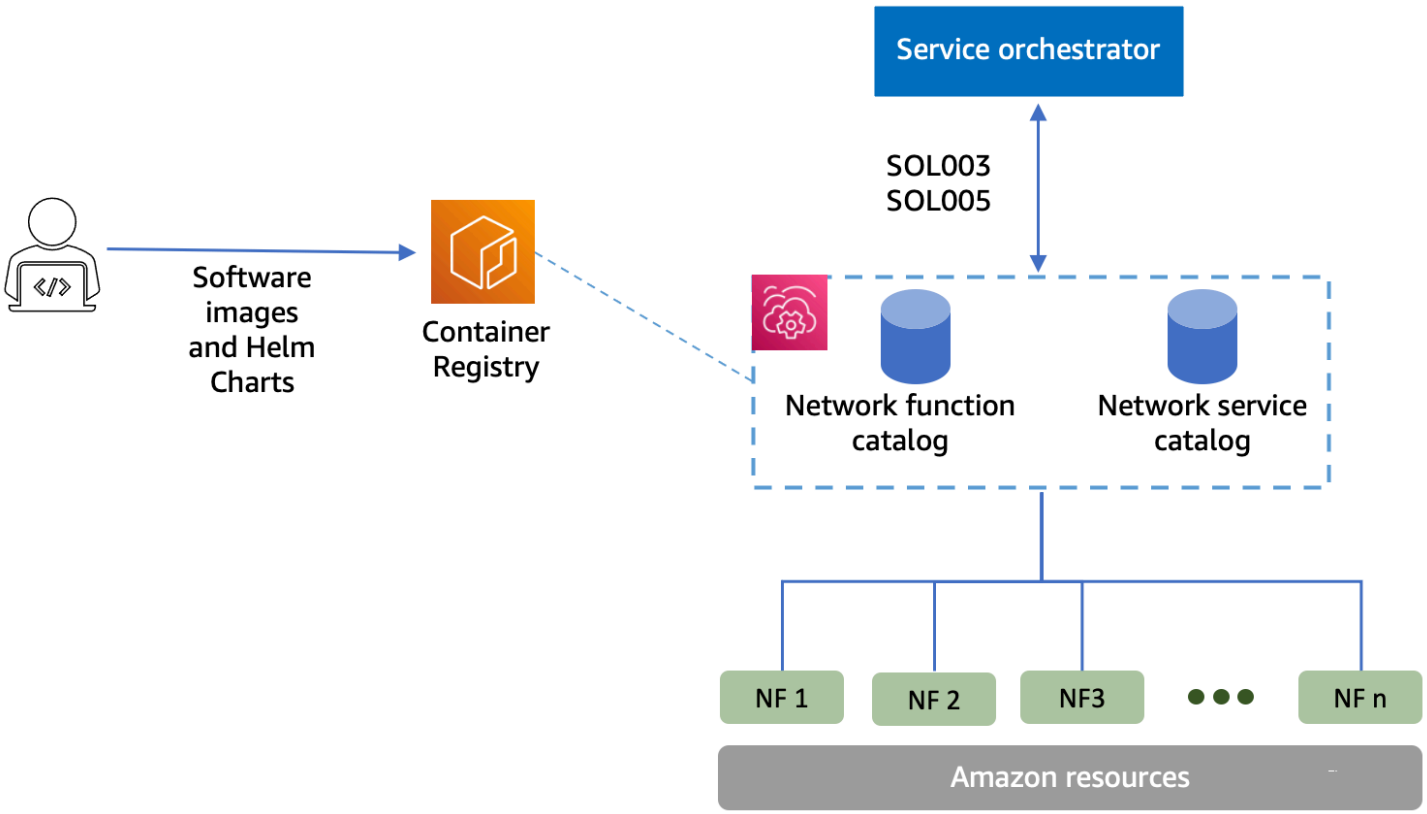
什麼是 AWS Telco Network Builder？

AWS Telco Network Builder (AWS TNB) 是一項 AWS 服務，可提供通訊服務供應商 (CSPs) 在基礎設施上 AWS 部署、管理和擴展 5G 網路的有效方式。

使用 AWS TNB，您可以自動化方式， AWS 雲端 使用網路映像在 中部署可擴展且安全的 5G 網路。您不需要學習新技術、決定要使用的運算服務，或知道如何佈建和設定 AWS 資源。

反之，您會描述網路的基礎設施，並提供獨立軟體廠商 (ISV) 合作夥伴的網路功能軟體映像。 AWS TNB 會與第三方服務協調人員 AWS 和服務整合，以自動佈建必要的 AWS 基礎設施、部署容器化網路功能，以及設定聯網和存取管理，以建立完全運作的網路服務。

下圖說明 AWS TNB 和服務協調器之間的邏輯整合，以使用歐洲電信標準協會 (ETSI) 型標準界面部署網路功能。



主題

- 初次使用 AWS？
- AWS TNB 適用於誰？
- AWS TNB 功能

- 存取 AWS TNB
- AWS TNB 定價
- 下一步

初次使用 AWS？

如果您是 AWS 產品和服務的新手，請使用下列資源開始進一步了解：

- 簡介 AWS
- 入門 AWS

AWS TNB 適用於誰？

AWS TNB 適用於希望利用成本效益的 CSPs， 敏捷性、 和 彈性 AWS 雲端 提供，無需撰寫和維護自訂指令碼和組態，即可設計、 部署、 和管理網路服務。 AWS TNB 會自動佈建必要的 AWS 基礎設施， 部署容器化網路函數、 和 設定聯網和存取管理，以根據 CSP 定義的網路服務描述項建立完全操作的網路服務， 和 CSP 想要部署的網路函數。

AWS TNB 功能

以下是 CSP 想要使用 AWS TNB 的一些原因：

協助簡化任務

為您的網路操作提供更高的效率，例如部署新服務、更新和升級網路功能，以及變更網路基礎設施拓撲。

與協調器整合

AWS TNB 與 ETSI 相容的熱門第三方服務協調器整合。

擴展

您可以設定 AWS TNB 來擴展基礎 AWS 資源，以滿足流量需求、更有效率地執行網路函數更新、推出網路基礎設施拓撲變更，並將新 5G 服務的部署時間從數天縮短為數小時。

檢查和監控 AWS 資源

AWS TNB 可讓您在單一儀表板上檢查和監控支援網路 AWS 的資源，例如 Amazon VPC、Amazon EC2 和 Amazon EKS。

支援服務範本

AWS TNB 可讓您為所有電信工作負載 (RAN、Core、IMS) 建立服務範本。您可以建立新的服務定義、重複使用現有的範本，或與持續整合和持續交付 (CI/CD) 管道整合，以發佈新的定義。

追蹤網路部署的變更

當您變更網路函數部署的基礎組態時，例如，變更 Amazon EC2 執行個體類型的執行個體類型，您可以以可重複且可擴展的方式追蹤變更。手動執行此操作需要管理網路狀態、建立和刪除資源，以及注意所需的變更順序。當您使用 AWS TNB 來管理網路函數的生命週期時，您只會對描述網路函數的網路服務描述項進行變更。 AWS TNB 會自動以正確的順序進行必要的變更。

簡化網路函數生命週期

您可以管理網路函數的第一個和所有後續版本，並指定升級的時間。您也可以以相同方式管理您的 RAN、Core、IMS 和網路應用程式。

存取 AWS TNB

您可以使用下列任一界面來建立、存取和管理 AWS TNB 資源：

- AWS TNB 主控台 — 提供用於管理網路的 Web 界面。
- AWS TNB API — 提供執行 AWS TNB 動作的 RESTful API。如需詳細資訊，請參閱 AWS TNB API 參考
- AWS Command Line Interface (AWS CLI) — 為廣泛的 AWS 服務提供命令，包括 AWS TNB。Windows、macOS 和 Linux 支援此功能。如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》https://docs.aws.amazon.com/cli/latest/userguide/。
- AWS SDKs – 提供特定語言APIs，並完成許多連線詳細資訊。包括計算簽章、處理請求重試和錯誤處理。如需詳細資訊，請參閱 AWS 開發套件。

AWS TNB 定價

AWS TNB 可協助 CSPs 上自動化電信網路的部署和管理 AWS。使用 AWS TNB 時，您需要支付以下兩個維度的費用：

- 依受管網路函數項目 (MNFI) 小時。
- 依 API 請求數量。

當您使用其他 AWS 服務搭配 AWS TNB 時，也會產生額外費用。如需詳細資訊，請參閱 AWS TNB 定價。

若要檢視您的帳單，請前往 AWS 帳單與成本管理 主控台中的帳單與成本管理儀表板。您的帳單內含用量報告的連結，可提供帳單的其他詳細資訊。如需 AWS 帳戶帳單的詳細資訊，請參閱AWS 帳戶帳單。

如果您對 AWS 帳單、帳戶和事件有任何疑問，請聯絡 AWS Support。

AWS Trusted Advisor 是一項服務，可用來協助最佳化 AWS 環境的成本、安全性和效能。如需詳細資訊，請參閱 AWS Trusted Advisor。

下一步

如需如何開始使用 AWS TNB 的詳細資訊，請參閱下列主題：

- 設定 AWS TNB – 完成先決條件步驟。
- AWS TNB 入門 – 部署您的第一個網路函數，例如集中式單元 (CU)、存取與行動管理函數 (AMF)、使用者平面函數 (UPF) 或完整的 5G 核心。

AWS TNB 的運作方式

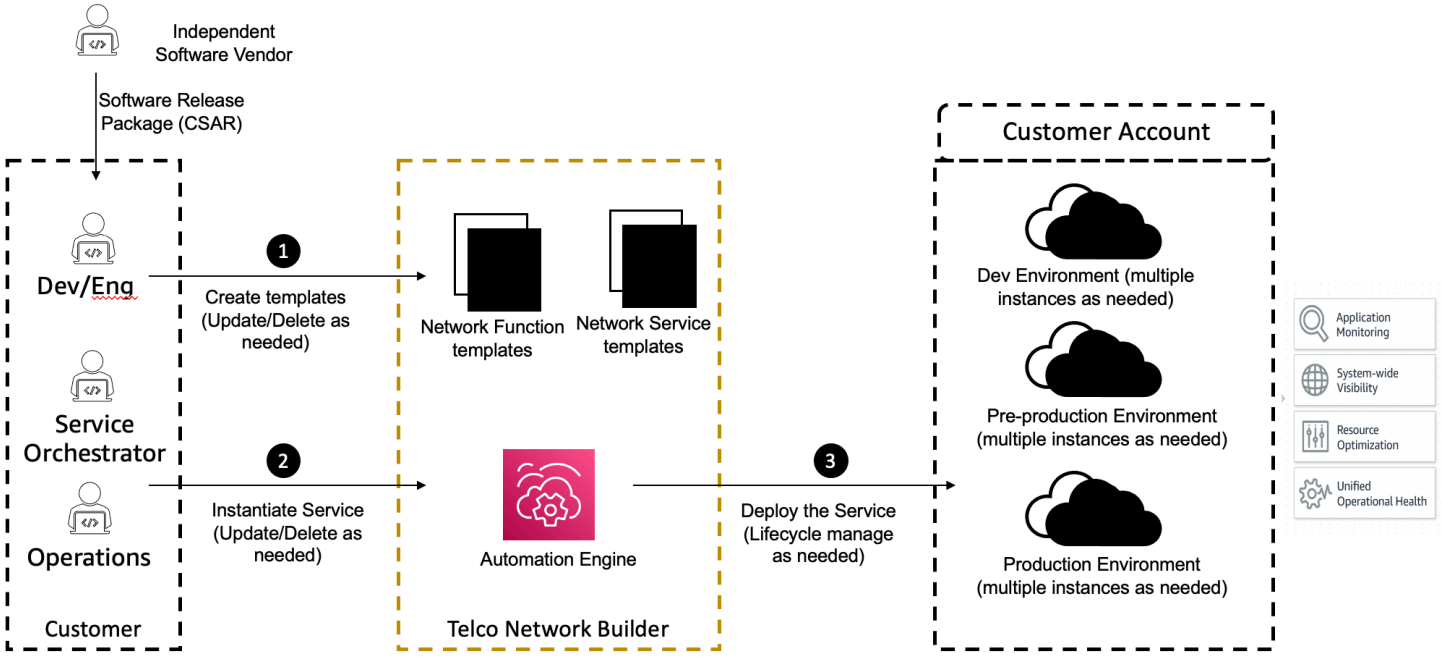
AWS TNB 與標準化end-to-end協調器 AWS 和資源整合，以操作完整的 5G 網路。

AWS TNB 可讓您擷取網路函數套件和網路服務描述項 (NSDs)，並提供自動化引擎來操作您的網路。您可以使用end-to-end協調器並與 AWS TNB APIs 整合，或使用 AWS TNB SDKs建置您自己的自動化流程。如需詳細資訊，請參閱AWS TNB 架構。

主題

AWS TNB 架構

AWS TNB 可讓您透過 AWS 管理主控台、 AWS CLI AWS TNB REST API 和 SDKs 執行生命週期管理操作。這可讓工程、操作和程式設計系統團隊等不同的 CSP 角色利用 AWS TNB。您建立並上傳網路函數套件做為 Cloud Service Archive (CSAR) 檔案。CSAR 檔案包含 Helm Chart、軟體映像和網路函數描述項 (NFD)。您可以使用 範本重複部署該套件的多個組態。您可以建立網路服務範本，定義要部署的基礎設施和網路函數。您可以使用參數覆寫，在不同位置部署不同的組態。然後，您可以使用範本來執行個體化網路，並在 AWS 基礎設施上部署網路功能。 AWS TNB 為您提供部署的可見性。



與 整合 AWS 服務

5G 網路由部署在數千個 Kubernetes 叢集的一組互連容器化網路函數組成。 AWS TNB 將下列 AWS 服務 整合為電信特定 APIs，以建立完全運作的網路服務：

- Amazon Elastic Container Registry (Amazon ECR) 可存放獨立軟體廠商 (ISVs) 網路函數成品。
- Amazon Elastic Kubernetes Service (Amazon EKS) 來設定叢集。
- 適用於聯網建構的 Amazon VPC。
- 使用 的安全群組 CloudFormation。
- AWS CodePipeline 適用於跨 AWS 區域、 AWS Local Zones 和 的部署目標 AWS Outposts。
- 定義角色的 IAM。
- AWS Organizations 以控制對 AWS TNB APIs存取。
- Health 儀板表 和 AWS CloudTrail 來監控運作狀態和張貼指標。

AWS TNB 資源配額

您的 AWS 帳戶 具有每個 的預設配額，先前稱為限制 AWS 服務。除非另有說明，否則每個配額都專屬於 AWS 區域。您可以要求提高某些配額，但並非所有配額都能提高。

若要檢視 AWS TNB 的配額，請開啟 Service Quotas 主控台。在導覽窗格中，選擇 AWS 服務，然後選取 AWS TNB。

若要請求提高配額，請參閱《Service Quotas 使用者指南》中的請求提高配額。

您的 AWS 帳戶 具有與 AWS TNB 相關的下列配額。

資源配額	Description	預設值	是否可調整？
網路服務執行個體	一個區域中網路服務執行個體的數量上限。	800	是
並行持續的網路服務操作	一個區域中並行進行中網路服務操作的數量上限。	40	是
網路套件	一個區域中的網路套件數量上限。	40	是
函數套件	一個區域中的函數套件數目上限。	200	是

AWS TNB 概念

本主題說明協助您開始使用 AWS TNB 的基本概念。

目錄

網路函數的生命週期

AWS TNB 會在網路函數的整個生命週期中協助您。網路函數生命週期包含下列階段和活動：

規劃

1. 透過識別要部署的網路函數來規劃您的網路。
2. 將網路函數軟體映像放入容器映像儲存庫。
3. 建立要部署或升級的 CSAR 套件。
4. 使用 AWS TNB 上傳定義網路函數（例如 CU AMF 和 UPF) 的 CSAR 套件，並與持續整合和持續交付 (CI/CD) 管道整合，以協助您建立新的 CSAR 套件版本，做為新的網路函數軟體映像或客戶指令碼可用。

組態

1. 識別部署所需的資訊，例如運算類型、網路函數版本、IP 資訊和資源名稱。
2. 使用資訊建立您的網路服務描述項 (NSD)。
3. 擷取 NSDs定義網路函數和網路函數執行個體化所需的資源。

執行個體化

1. 建立網路函數所需的基礎設施。
2. 執行個體化（或佈建）NSD 中定義的網路函數，並開始攜帶流量。
3. 驗證資產。

生產

在網路函數的生命週期期間，您將完成生產操作，例如：

- 更新網路函數組態，例如，更新已部署網路函數中的值。
- 使用新的網路套件和參數值更新網路執行個體。例如，更新網路套件中的 Amazon EKS `version` 參數。

使用標準化界面

AWS TNB 與歐洲電信標準協會 (ETSI) 相容服務協調器整合，可讓您簡化網路服務的部署。服務協調器可以使用 AWS TNB SDKs、CLI 或 APIs 來啟動操作，例如執行個體化或將網路函數升級至新版本。

AWS TNB 支援下列規格。

規格	發行版本	說明
ETSI SOL001	v3.6.1	定義允許 TOSCA 型網路函數描述項的標準。
ETSI SOL002	v3.6.1	定義網路函數管理的相關模型。
ETSI SOL003	v3.6.1	定義網路函數生命週期管理的標準。
ETSI SOL004	v3.6.1	定義網路函數套件的 CSAR 標準。
ETSI SOL005	v3.6.1	定義網路服務套件和網路服務生命週期管理的標準。
ETSI SOL007	v3.5.1	定義允許 TOSCA 型網路服務描述項的標準。

函數套件

使用 AWS TNB，您可以將符合 ETSI SOL001/SOL004 的函數套件存放在函數目錄中。然後，您可以上傳包含描述虛擬網路函數成品的 Cloud Service Archive (CSAR) 套件。

- 虛擬網路函數描述項 – 定義套件加入和虛擬網路函數管理的中繼資料。您必須將此檔案命名為 `vnfd.yaml`。
- 軟體映像 – 參考虛擬網路函數容器映像。Amazon Elastic Container Registry (Amazon ECR) 可以充當虛擬網路函數映像儲存庫。
- 其他檔案 – 用來管理虛擬網路函數，例如指令碼和 Helm Chart。

CSAR 是由 OASIS TOSCA 標準定義的套件，包含符合 OASIS TOSCA YAML 規格的網路/服務描述項。如需所需 YAML 規格的資訊，請參閱 TNB 的 TOSCA AWS 參考。

以下是虛擬網路函數描述項的範例。

```
tosca_definitions_version: tnb_simple_yaml_1_0

topology_template:

  node_templates:

    SampleNF:
      type: tosca.nodes.AWS.VNF
      properties:
        descriptor_id: "SampleNF-descriptor-id"
        descriptor_version: "2.0.0"
        descriptor_name: "NF 1.0.0"
        provider: "SampleNF"
      requirements:
        helm: HelmChart

    HelmChart:
      type: tosca.nodes.AWS.Artifacts.Helm
      properties:
        implementation: "./SampleNF"
```

網路套件

網路套件是 CSAR （雲端服務封存） 格式.zip的檔案。它定義了您要部署的函數套件，以及您要部署它們的 AWS 基礎設施。

網路套件包含下列檔案：

- TOSCA 格式的網路描述項檔案 (nsd.yaml)，如 ETSI SOL007 所述。

  nsd.yaml 檔案包含已上傳函數套件的參考及其描述項 IDs。
- 使用者資料指令碼，如果有的話。
- 生命週期掛鉤指令碼，如果有的話。
- 外掛程式的values.yaml組態檔案，如果有的話。

AWS TNB 支援以 TOSCA 語言建立資源模型的 ETSI 標準，例如網路、服務和函數。 AWS TNB 可讓您以 ETSI 相容服務協調器可以理解的方式建立資源模型， AWS 服務 藉此更有效率地使用它們。

AWS TNB 的網路服務描述項

網路服務描述項 (NSD) 是網路套件中的.yaml檔案，使用 TOSCA 標準來描述您要部署的網路函數，以及 AWS 您要部署網路函數的基礎設施。若要定義 NSD 並設定基礎資源和網路生命週期操作，您必須了解 AWS TNB 支援的 NSD TOSCA 結構描述。

您的 NSD 檔案分為下列部分：

1. TOSCA 定義版本 – 這是 NSD YAML 檔案的第一行，包含版本資訊，如下列範例所示。

```
tosca_definitions_version: tnb_simple_yaml_1_0
```

2. VNFD – NSD 包含要在其中執行生命週期操作的網路函數定義。每個網路函數都必須以下列值識別：
   - 的唯一 IDdescriptor_id。ID 必須符合網路函數 CSAR 套件中的 ID。
   - 的唯一名稱namespace。名稱必須與唯一 ID 相關聯，以便更輕鬆地參考整個 NSD YAML 檔案，如下列範例所示。

```
vnfds:
  - descriptor_id: "61465757-cb8f-44d8-92c2-b69ca0de025b"
    namespace: "amf"
```

3. 拓撲範本 – 定義要部署的資源、網路函數部署，以及生命週期掛鉤等任何自訂指令碼。如以下範例所示。

```
topology_template:

  node_templates:

    SampleNS:
      type: tosca.nodes.AWS.NS
      properties:
        descriptor_id: "<Sample Identifier>"
        descriptor_version: "<Sample nversion>"
        descriptor_name: "<Sample name>"
```

4. 其他節點 – 每個建模資源都有屬性和需求的區段。屬性說明資源的選用或必要屬性，例如 版本。這些要求說明必須做為引數提供的相依性。例如，若要建立 Amazon EKS 節點群組資源，必須在 Amazon EKS 叢集內建立。如以下範例所示。

```
SampleEKSNode:
  type: tosca.nodes.AWS.Compute.EKSManagedNode
  properties:
    node_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/SampleRole"
  capabilities:
    compute:
      properties:
        ami_type: "AL2_x86_64"
        instance_types:
          - "t3.xlarge"
        key_pair: "SampleKeyPair"
    scaling:
      properties:
        desired_size: 1
        min_size: 1
        max_size: 1
  requirements:
    cluster: SampleEKS
    subnets:
      - SampleSubnet
    network_interfaces:
      - SampleENI01
      - SampleENI02
```

範例 NSD

以下是 NSD 的程式碼片段，示範如何建立模型 AWS 服務。網路函數將部署在具有 Kubernetes 1.27 版的 Amazon EKS 叢集上。應用程式子網路為 Subnet01 和 Subnet02。然後，您可以使用 Amazon Machine Image (AMI)、執行個體類型和自動擴展組態為您的應用程式定義 NodeGroups。

```
tosca_definitions_version: tnb_simple_yaml_1_0

SampleNFEKS:
  type: tosca.nodes.AWS.Compute.EKS
  properties:
    version: "1.27"
    access: "ALL"
```

```
    cluster_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/SampleClusterRole"
  capabilities:
    multus:
      properties:
        enabled: true
  requirements:
    subnets:
      - Subnet01
      - Subnet02

SampleNFEKSNode01:
  type: tosca.nodes.AWS.Compute.EKSManagedNode
  properties:
    node_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/SampleNodeRole"
  capabilities:
    compute:
      properties:
        ami_type: "AL2_x86_64"
        instance_types:
          - "t3.xlarge"
        key_pair: "SampleKeyPair"
    scaling:
      properties:
        desired_size: 3
        min_size: 2
        max_size: 6
  requirements:
    cluster: SampleNFEKS
    subnets:
      - Subnet01
    network_interfaces:
      - ENI01
      - ENI02
```

AWS TNB 的管理和操作

使用 AWS TNB，您可以根據 ETSI SOL003 和 SOL005，使用標準化管理操作來管理網路。您可以使用 AWS TNB APIs 來執行生命週期操作，例如：

- 執行個體化您的網路函數。
- 終止您的網路函數。
- 更新您的網路函數以覆寫 Helm 部署。

- 使用新的網路套件和參數值更新執行個體或更新的網路執行個體。
- 管理網路函數套件的版本。
- 管理 NSDs的版本。
- 擷取已部署網路函數的相關資訊。

設定 AWS TNB

完成本主題中所述的任務來設定 AWS TNB。

任務

- 註冊 AWS 帳戶
- 選擇 AWS 區域
- 記下服務端點
- （選用）安裝 AWS CLI
- 設定 AWS TNB 角色

註冊 AWS 帳戶

若要開始使用 AWS，您需要 AWS 帳戶。如需建立 的詳細資訊 AWS 帳戶，請參閱《AWS 帳戶管理參考指南》中的 入門 AWS 帳戶。

選擇 AWS 區域

若要檢視 AWS TNB 的可用區域清單，請參閱AWS 區域服務清單。若要檢視用於程式設計存取的端點清單，請參閱 中的 AWS TNB 端點AWS 一般參考。

記下服務端點

若要以程式設計方式連線至 AWS 服務，您可以使用 端點。除了標準 AWS 端點之外，某些 AWS 服務還在所選區域中提供 FIPS 端點。如需詳細資訊，請參閱 AWS 服務端點。

區域名稱	區域	端點	通訊協定	
美國東部 (維吉尼亞北部)	us-east-1	tnb.us-east-1.amazonaws.com	HTTPS	
美國西部 (奧勒岡)	us-west-2	tnb.us-west-2.amazonaws.com	HTTPS	

區域名稱	區域	端點	通訊協定
亞太區域 (首爾)	ap-northeast-2	tnb.ap-northeast-2.amazonaws.com	HTTPS
亞太地區 (悉尼)	ap-southeast-2	tnb.ap-southeast-2.amazonaws.com	HTTPS
加拿大 (中部)	ca-central-1	tnb.ca-central-1.amazonaws.com	HTTPS
歐洲 (法蘭克福)	eu-central-1	tnb.eu-central-1.amazonaws.com	HTTPS
歐洲 (巴黎)	eu-west-3	tnb.eu-west-3.amazonaws.com	HTTPS
歐洲 (西班牙)	eu-south-2	tnb.eu-south-2.amazonaws.com	HTTPS
歐洲 (斯德哥爾摩)	eu-north-1	tnb.eu-north-1.amazonaws.com	HTTPS
南美洲 (聖保羅)	sa-east-1	tnb.sa-east-1.amazonaws.com	HTTPS

（選用）安裝 AWS CLI

AWS Command Line Interface (AWS CLI) 為廣泛的 AWS 產品提供命令，並支援 Windows、macOS 和 Linux。您可以使用 存取 AWS TNB AWS CLI。若要開始使用，請參閱《AWS Command Line Interface 使用者指南》。如需 AWS TNB 命令的詳細資訊，請參閱《 AWS CLI 命令參考》中的 tnb。

設定 AWS TNB 角色

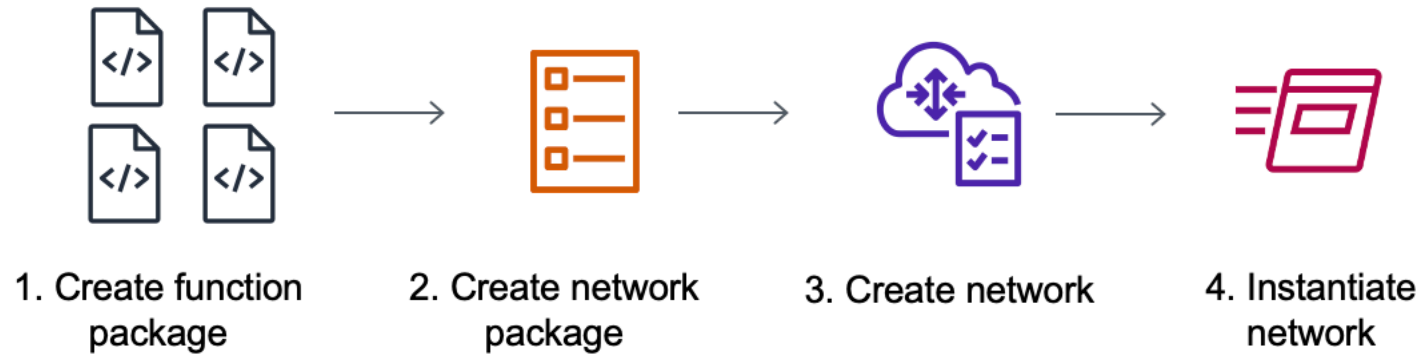
您必須建立 IAM 服務角色來管理 AWS TNB 解決方案的不同部分。 AWS TNB 服務角色可以代表您對其他服務進行 API 呼叫 AWS CloudFormation， AWS 例如 AWS CodeBuild和各種運算和儲存服務，以執行個體化和管理部署的資源。

如需 AWS TNB 服務角色的詳細資訊，請參閱 AWS TNB 的身分和存取管理。

AWS TNB 入門

本教學課程示範如何使用 AWS TNB 部署網路函數，例如集中式單位 (CU)、存取與行動性管理函數 (AMF) 或 5G 使用者平面函數 (UPF)。

下圖說明部署程序：



任務

- 先決條件
- 建立函數套件
- 建立網路套件
- 建立和執行個體化網路執行個體
- 清除

先決條件

您必須先擁有下列項目，才能成功執行部署：

- AWS 商業支援計劃。
- 透過 IAM 角色的許可。
- 符合 ETSI SOL001/SOL004 的網路函數 (NF) 套件。
- 符合 ETSI SOL007 的網路服務描述項 (NSD) 範本。

您可以從 AWS TNB GitHub 網站的範例套件使用範例函數套件或網路套件。 GitHub

建立函數套件

網路函數套件是 Cloud Service Archive (CSAR) 檔案。CSAR 檔案包含 Helm Chart、軟體映像和網路函數描述項 (NFD)。

建立函數套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇函數套件。
3. 選擇建立函數套件。
4. 在上傳函數套件下，選擇選擇檔案，然後將每個 CSAR 套件上傳為`.zip`檔案。您最多可以上傳 10 個檔案。
5. （選用）在標籤下，選擇新增標籤，然後輸入索引鍵和值。您可以使用標籤來搜尋和篩選資源或追蹤 AWS 成本。
6. 選擇 Next (下一步)。
7. 檢閱套件詳細資訊，然後選擇建立函數套件。

建立網路套件

網路套件會指定您要部署的網路函數，以及您要如何將它們部署到目錄中。

建立網路套件

1. 在導覽窗格中，選擇網路套件。
2. 選擇建立網路套件。
3. 在上傳網路套件下，選擇選擇檔案，然後將每個 NSD 上傳為`.zip`檔案。您最多可以上傳 10 個檔案。
4. （選用）在標籤下，選擇新增標籤，然後輸入索引鍵和值。您可以使用標籤來搜尋和篩選資源或追蹤 AWS 成本。
5. 選擇 Next (下一步)。
6. 選擇建立網路套件。

建立和執行個體化網路執行個體

網路執行個體是在 AWS TNB 中建立的單一網路，可以部署。您必須建立網路執行個體並將其執行個體化。當您執行個體化網路執行個體時， AWS TNB 會佈建必要的 AWS 基礎設施、部署容器化網路函數，以及設定聯網和存取管理，以建立完全運作的網路服務。

建立和執行個體化網路執行個體

1. 在導覽窗格中，選擇網路。
2. 選擇建立網路執行個體。
3. 輸入網路的名稱和描述，然後選擇下一步。
4. 選擇網路套件。驗證詳細資訊，然後選擇下一步。
5. 選擇建立網路執行個體。初始狀態為 Created。

   網路頁面隨即出現，顯示Not instantiated處於 狀態的新網路執行個體。
6. 選取網路執行個體，選擇動作和執行個體化。

   網路執行個體頁面隨即出現。
7. 檢閱詳細資訊並更新參數值。參數值的更新僅適用於此網路執行個體。NSD 和 VNFD 套件中的參數不會變更。
8. 選擇執行個體化網路。

   部署狀態頁面隨即出現。
9. 使用重新整理圖示來追蹤網路執行個體的部署狀態。您也可以在部署任務區段中啟用自動重新整理，以追蹤每個任務的進度。

清除

您現在可以刪除您為此教學課程建立的資源。

清除您的資源

1. 在導覽窗格中，選擇網路。
2. 選擇網路的 ID，然後選擇終止。
3. 出現確認提示時，請輸入網路 ID，然後選擇終止。
4. 使用重新整理圖示來追蹤網路執行個體的狀態。

5. （選用）選取網路，然後選擇刪除。

AWS TNB 的函數套件

函數套件是 CSAR (Cloud Service Archive) 格式的 .zip 檔案，其中包含網路函數 (ETSI 標準電信應用程式） 和函數套件描述項，其使用 TOSCA 標準來描述網路函數應如何在您的網路上執行。

任務

- 在 AWS TNB 中建立函數套件
- 在 AWS TNB 中檢視函數套件
- 從 AWS TNB 下載函數套件
- 從 AWS TNB 刪除函數套件

在 AWS TNB 中建立函數套件

了解如何在 AWS TNB 網路函數目錄中建立函數套件。建立函數套件是在 AWS TNB 中建立網路的第一步。上傳函數套件之後，您可以建立網路套件。

Console

使用主控台建立函數套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇函數套件。
3. 選擇建立函數套件。
4. 選擇選擇檔案，並將每個 CSAR 套件上傳為 `.zip` 檔案。您最多可以上傳 10 個檔案。
5. 選擇下一步。
6. 檢閱套件詳細資訊。
7. 選擇建立函數套件。

AWS CLI

使用 建立函數套件 AWS CLI

1. 使用 create-sol-function-package 命令建立新的函數套件：

```
aws tnb create-sol-function-package
```

2. 使用 put-sol-function-package-content 命令上傳函數套件內容。例如：

```
aws tnb put-sol-function-package-content \
--vnf-pkg-id ^fp-[a-f0-9]{17}$ \
--content-type application/zip \
--file "fileb://valid-free5gc-udr.zip" \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

在 AWS TNB 中檢視函數套件

了解如何檢視函數套件的內容。

Console

使用主控台檢視函數套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇函數套件。
3. 使用搜尋方塊尋找函數套件

AWS CLI

使用 檢視函數套件 AWS CLI

1. 使用 list-sol-function-packages 命令列出函數套件。

```
aws tnb list-sol-function-packages
```

2. 使用 get-sol-function-package 命令來檢視函數套件的詳細資訊。

```
aws tnb get-sol-function-package \
--vnf-pkg-id ^fp-[a-f0-9]{17}$ \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

從 AWS TNB 下載函數套件

了解如何從 AWS TNB 網路函數目錄下載函數套件。

Console

使用主控台下載函數套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在主控台左側的導覽窗格中，選擇函數套件。
3. 使用搜尋方塊尋找函數套件
4. 選擇函數套件
5. 選擇動作、下載。

AWS CLI

使用 下載函數套件 AWS CLI

使用 get-sol-function-package-content 命令下載函數套件。

```
aws tnb get-sol-function-package-content \
--vnf-pkg-id ^fp-[a-f0-9]{17}$ \
--accept "application/zip" \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

從 AWS TNB 刪除函數套件

了解如何從 AWS TNB 網路函數目錄中刪除函數套件。若要刪除函數套件，套件必須處於停用狀態。

Console

使用主控台刪除函數套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇函數套件。
3. 使用搜尋方塊來尋找函數套件。

4. 選擇函數套件。
5. 選擇 Actions (動作)、Disable (停用)。
6. 選擇 動作、刪除。

AWS CLI

使用 刪除函數套件 AWS CLI

1. 使用 update-sol-function-package 命令來停用函數套件。

```
aws tnb update-sol-function-package --vnf-pkg-id ^fp-[a-f0-9]{17}$ ---
operational-state DISABLED
```

2. 使用 delete-sol-function-package 命令來刪除函數套件。

```
aws tnb delete-sol-function-package \
--vnf-pkg-id ^fp-[a-f0-9]{17}$ \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

AWS TNB 的網路套件

網路套件是 CSAR （雲端服務封存） 格式的 .zip 檔案。它定義了您要部署的函數套件，以及您要部署它們的 AWS 基礎設施。

網路套件包含下列檔案：

- TOSCA 格式的網路描述項檔案 (nsd.yaml)，如 ETSI SOL007 所述。

  nsd.yaml 檔案包含已上傳函數套件的參考及其描述項 IDs。
- 使用者資料指令碼，如果有的話。
- 生命週期掛鉤指令碼，如果有的話。
- 外掛程式的values.yaml組態檔案，如果有的話。

任務

- 在 AWS TNB 中建立網路套件
- 在 AWS TNB 中檢視網路套件
- 從 AWS TNB 下載網路套件
- 從 AWS TNB 刪除網路套件

在 AWS TNB 中建立網路套件

網路套件包含網路服務描述項 (NSD) 檔案 （必要） 和任何其他檔案 （選用），例如符合您需求的指令碼。例如，如果您的網路套件中有多個函數套件，您可以使用 NSD 來定義哪些網路函數應該在特定 VPCs、子網路或 Amazon EKS 叢集中執行。

在建立函數套件之後建立網路套件。建立網路套件後，您需要建立網路執行個體。

Console

使用主控台建立網路套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路套件。
3. 選擇建立網路套件。
4. 選擇選擇檔案，並將每個 NSD 上傳為.zip檔案。您最多可以上傳 10 個檔案。

5. 選擇下一步。
6. 檢閱套件詳細資訊。
7. 選擇建立網路套件。

AWS CLI

使用 建立網路套件 AWS CLI

1. 使用 create-sol-network-package 命令來建立網路套件。

```
aws tnb create-sol-network-package
```

2. 使用 put-sol-network-package-content 命令上傳網路套件內容。例如：

```
aws tnb put-sol-network-package-content \
--nsd-info-id ^np-[a-f0-9]{17}$ \
--content-type application/zip \
--file "fileb://free5gc-core-1.0.9.zip" \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

在 AWS TNB 中檢視網路套件

了解如何檢視網路套件的內容。

Console

使用主控台檢視網路套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路套件。
3. 使用搜尋方塊尋找網路套件。

AWS CLI

使用 檢視網路套件 AWS CLI

1. 使用 list-sol-network-packages 命令列出您的網路套件。

```
aws tnb list-sol-network-packages
```

2. 使用 get-sol-network-package 命令來檢視網路套件的詳細資訊。

```
aws tnb get-sol-network-package \
--nsd-info-id ^np-[a-f0-9]{17}$ \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

從 AWS TNB 下載網路套件

了解如何從 AWS TNB 網路服務目錄下載網路套件。

Console

使用主控台下載網路套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路套件。
3. 使用搜尋方塊尋找網路套件
4. 選擇網路套件。
5. 選擇動作、下載。

AWS CLI

使用 下載網路套件 AWS CLI

- 使用 get-sol-network-package-content 命令下載網路套件。

```
aws tnb get-sol-network-package-content \
--nsd-info-id ^np-[a-f0-9]{17}$ \
--accept "application/zip" \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

從 AWS TNB 刪除網路套件

了解如何從 AWS TNB 網路服務目錄中刪除網路套件。若要刪除網路套件，套件必須處於停用狀態。

Console

使用主控台刪除網路套件

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路套件。
3. 使用搜尋方塊尋找網路套件
4. 選擇網路套件
5. 選擇 Actions (動作)、Disable（停用)。
6. 選擇 動作、刪除。

AWS CLI

使用 刪除網路套件 AWS CLI

1. 使用 update-sol-network-package 命令來停用網路套件。

```
aws tnb update-sol-network-package --nsd-info-id ^np-[a-f0-9]{17}$ --nsd-
operational-state DISABLED
```

2. 使用 delete-sol-network-package 命令來刪除網路套件。

```
aws tnb delete-sol-network-package \
--nsd-info-id ^np-[a-f0-9]{17}$ \
--endpoint-url "https://tnb.us-west-2.amazonaws.com" \
--region us-west-2
```

AWS TNB 的網路執行個體

網路執行個體是在 AWS TNB 中建立的單一網路，可以部署。

任務

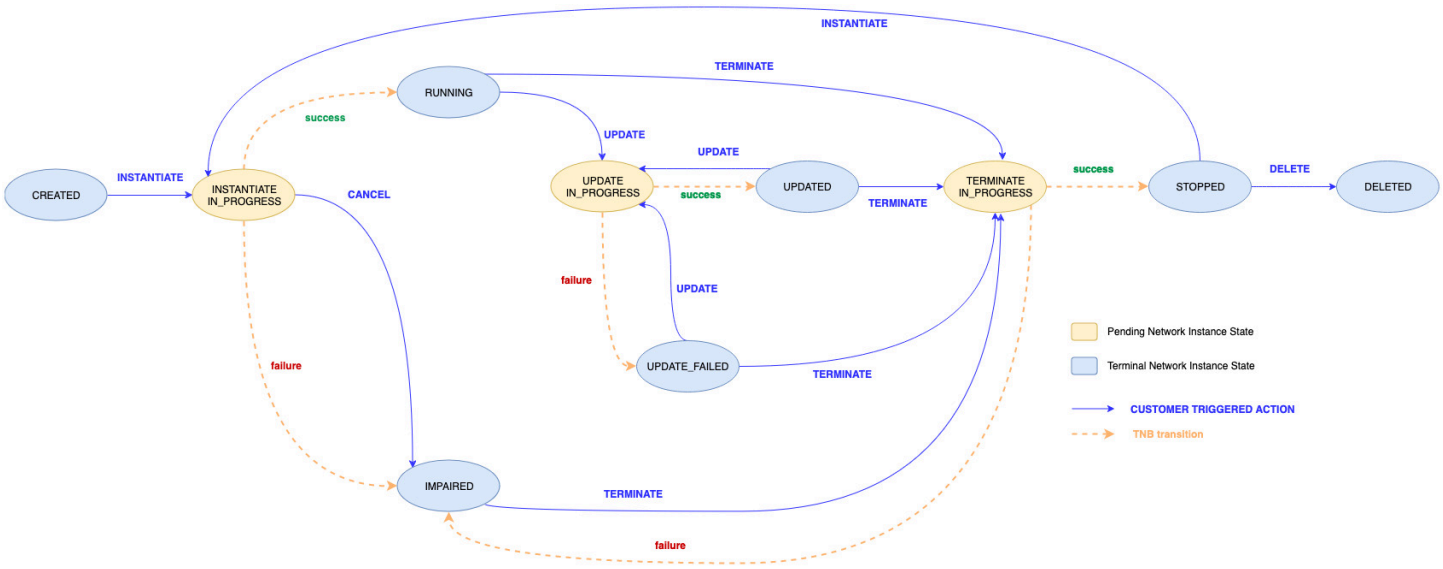
- 網路執行個體的生命週期操作
- 使用 AWS TNB 建立網路執行個體
- 使用 AWS TNB 實例化網路執行個體
- 在 AWS TNB 中更新函數執行個體
- 在 AWS TNB 中更新網路執行個體
- 在 AWS TNB 中檢視網路執行個體
- 從 AWS TNB 終止和刪除網路執行個體

網路執行個體的生命週期操作

AWS TNB 可讓您使用與 ETSI SOL003 和 SOL005 整合的標準化管理操作，輕鬆管理網路。您可以執行下列生命週期操作：

- 建立網路
- 執行個體化網路
- 更新網路函數
- 更新網路執行個體
- 檢視網路詳細資訊和狀態
- 終止網路

下圖顯示網路管理操作：



使用 AWS TNB 建立網路執行個體

您在建立網路套件後建立網路執行個體。建立網路執行個體之後，請將其執行個體化。

Console

使用主控台建立網路執行個體

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路。
3. 選擇建立網路執行個體。
4. 輸入執行個體的名稱和描述，然後選擇下一步。
5. 選取網路套件，驗證詳細資訊，然後選擇下一步。
6. 選擇建立網路執行個體。

新的網路執行個體會出現在網路頁面上。接著，執行個體化此網路執行個體。

AWS CLI

使用 建立網路執行個體 AWS CLI

- 使用 create-sol-network-instance 命令來建立網路執行個體。

```
aws tnb create-sol-network-instance --nsd-info-id ^np-[a-f0-9]{17}$ --ns-name
 "SampleNs" --ns-description "Sample"
```

接著，執行個體化此網路執行個體。

使用 AWS TNB 實例化網路執行個體

建立網路執行個體之後，您必須將其執行個體化。當您執行個體化網路執行個體時， AWS TNB 會佈建必要的 AWS 基礎設施、部署容器化網路函數，以及設定聯網和存取管理，以建立完全運作的網路服務。

Console

使用主控台執行個體化網路執行個體

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路。
3. 選取您要執行個體化的網路執行個體。
4. 選擇動作，然後執行個體化。
5. 在 Instantiate 網路頁面上，檢閱詳細資訊，並選擇性地更新參數值。

   參數值的更新僅適用於此網路執行個體。NSD 和 VNFD 套件中的參數不會變更。
6. 選擇執行個體化網路。

   部署狀態頁面隨即出現。
7. 使用重新整理圖示來追蹤網路執行個體的部署狀態。您也可以在部署任務區段中啟用自動重新整理，以追蹤每個任務的進度。

   當部署狀態變更為 時`Completed`，網路執行個體會執行個體化。

AWS CLI

使用 執行個體化網路執行個體 AWS CLI

1. 使用 instantiate-sol-network-instance 命令來執行個體化網路執行個體。

```
aws tnb instantiate-sol-network-instance --ns-instance-id ^ni-[a-f0-9]{17}$ --
additional-params-for-ns "{\"param1\": \"value1\", \"param2\": \"value2\"}"
```

2. 接著，檢視網路操作狀態。

在 AWS TNB 中更新函數執行個體

執行個體化網路執行個體之後，您可以在網路執行個體中更新函數套件。

Console

使用主控台更新函數執行個體

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路。
3. 選取網路執行個體。只有在網路執行個體的狀態為 時，您才能更新網路執行個體Instantiated。

   網路執行個體頁面隨即出現。
4. 從函數索引標籤中，選取要更新的函數執行個體。
5. 選擇更新。
6. 輸入您的更新覆寫。
7. 選擇更新。

AWS CLI

使用 CLI 更新函數執行個體

使用 update-sol-network-instance 命令搭配MODIFY_VNF_INFORMATION更新類型來更新網路執行個體中的函數執行個體。

```
aws tnb update-sol-network-instance --ns-instance-id ^ni-[a-f0-9]{17}$ --update-type
 MODIFY_VNF_INFORMATION --modify-vnf-info ...
```

在 AWS TNB 中更新網路執行個體

執行個體化網路執行個體之後，您可能需要更新基礎設施或應用程式。若要這樣做，請更新網路執行個體的網路套件和參數值，並部署更新操作以套用變更。

考量事項

- 您可以更新處於 Instantiated或 Updated 狀態的網路執行個體。
- 當您更新網路執行個體時，UpdateSolNetworkServiceAPI 會使用新的網路套件和參數值來更新網路執行個體的拓撲。
- AWS TNB 會驗證網路執行個體中的 NSD 和 VNFD 參數數量不超過 200。強制執行此限制，以防止惡意人士傳遞影響服務的錯誤或巨型承載。

您可以更新的參數

您可以在更新執行個體化網路執行個體時更新下列參數：

參數	Description	範例：之前	範例：之後

Amazon EKS 叢集版本	您可以將 Amazon EKS 叢集控制平面version參數的值更新為下一個次要版本。您無法降級版本。	`EKSCluster:`
		`  type: tosca.nodes.AWS.Compute.EKS`
		`  properties:`
		`  version: "1.28"`
			`EKSC`
			`r:`
			
			`typ`
			`tos`
			`es.A`
			`mput`
			
			`pro`
			`s:`

參數	Description	範例：之前	範例：之後
			ver "1.

參數	Description	範例：之前	範例：之後

Amazon EKS 工作者節點	您可以更新 EKSManagedNode kubernetes_version 參數的值，將節點群組升級至較新的 Amazon EKS 版本，或更新 ami_id 參數，將節點群組升級至最新的 EKS 最佳化 AMI。
	
	您可以更新 的 AMI IDEKSSelfManagedNode 。AMI 的 Amazon EKS 版本必須與 Amazon EKS 叢集版本相同或低於 2 個版本。例如，如果 Amazon EKS 叢集版本為 1.31，則 Amazon EKS AMI 版本必須為 1.31、1.30 或 1.29。
		`EKSManagedNodeGroup01:`
		`  ...`
		`  properties:`
		`  kubernetes_version: "  1.28"`
		`  EKSSelfManagedNode01:`
		`  compute:`
		`  compute:`
		`  properties:`
		`  ami_id: "ami-1231230LD "`
			`EKSM`
			`dNoc`
			`p01:`
			`...`
			`pro`
			`s:`
			`kub`
			`s_ve`
			`:`
			`"1.`
			`EKS`
			`nage`
			`01:`
			`com`

參數	Description	範例：之前	範例：之後
			com pro s: ami "*am* *3NEW*

參數	Description	範例：之前	範例：之後

Amazon EKS 節點群組	您可以根據您的運算需求新增或移除節點群組。
	
	刪除現有節點群組並新增節點群組時，請確保新節點群組的 IDs 與已刪除節點群組不同，否則操作會被視為節點群組修改，而不是刪除和新增。請注意，對於現有的節點群組，只能更新一組有限的參數。捲動此資料表以查看您可以更新的參數。
		`Free5GCEKSNode01:`
		`  type: tosca.nod`
		`es.AWS.Compute.EKS`
		`ManagedNode`
		`  ...`
		`    scaling:`
		`      properties:`
		`        desired_size: 1`
		`        min_size: 1`
		`        max_size: 1`
		`  ...`
		`Free5GCEKSNode02 : #`
		` Deleted Nodegroup`
		`  type: tosca.nod`
		`es.AWS.Compute.EKS`
		`ManagedNode`
		`  ...`
		`    scaling:`
		`      properties:`
		`        desired_size: 1`
		`        min_size: 1`
		`        max_size: 1`
		`  ...`
		`Free5GCEKSNode03 : #`
		` Deleted Nodegroup`
		`  type: tosca.nod`
		`es.AWS.Compute.EKS`
		`SelfManagedNode`
		`  ...`
		`    scaling:`
		`      properties:`
		`        desired_size: 1`
		`        min_size: 1`
		`        max_size: 1`
		`  ...`
			`Free`
			`SNod`
			
			` typ`
			` tos`
			`es.A`
			`mput`
			`Mana`
			`de`
			
			` ...`
			
			` sca`
			
			` pro`
			`s:`
			
			` des`
			`ize:`
			` 1`

參數	Description	範例：之前	範例：之後
			mir 1 max 1 ... Free SNod # New Nod typ tos es.A mput Self edNo ... sca

參數	Description	範例：之前	範例：之後
			pro s: des ize: 1 mir 1 max 1 ... *Free* *SNod* # New Nod

參數	Description	範例：之前	範例：之後
			typ tos es.A mput Mana de ... sca pro s: des ize: 1

參數	Description	範例：之前	範例：之後
			mir 1 max 1 ...

參數	Description	範例：之前	範例：之後

擴展屬性	您可以更新 EKSManagedNode 和 EKSSelfManagedNode TOSCA 節點的擴展屬性。	`EKSNodeGroup01:`
		`  ...`
		`  scaling:`
		`  properties:`
		`  desired_size: 1`
		`  min_size: 1`
		`  max_size: 1`
			`EKSN`
			`oup0`
			`...`
			`sca`
			`pro`
			`s:`
			`des`
			`ize:`

參數	Description	範例：之前	範例：之後
			mir max

參數	Description	範例：之前	範例：之後

Amazon EBS CSI 外掛程式屬性	您可以在 Amazon EKS 叢集上啟用或停用 Amazon EBS CSI 外掛程式。您也可以變更外掛程式版本。	`EKSCluster:`
		`  capabilities:`
		`  ...`
		`  ebs_csi:`
		`  properties:`
		`  enabled: false`
			`EKSC`
			`r:`
			`cap`
			`ies:`
			`...`
			`ebs`
			`pro`
			`s:`
			`ena`
			`ver`
			`"v1`
			`e`

參數	Description	範例：之前	範例：之後
			ksbu "

參數	Description	範例：之前	範例：之後
根磁碟區大小	您可以新增、移除或更新 EKSManagedNode 和 EKSSelfManagedNode TOSCA 節點的根磁碟區大小屬性。	Free5GCEKSNode01: ... capabilities: compute: properties: root_volume_size: 50	Free SNod ... cap ies: com pro s:

參數	Description	範例：之前	範例：之後
			roo me_s

參數	Description	範例：之前	範例：之後
VNF	您可以在 NSD 中參考 VNFs並使用 VNFDeployment TOSCA 節點將其部署至在 NSD 中建立的叢集。在更新過程中，您將能夠新增、更新和刪除網路的 VNFs。	vnfds: - descriptor_id: "43c012fa-2616-41a8-a833-0dfd4c5a049e " namespace: " vnf1" - descriptor_id: "64222f98-ecd6-4871-bf94-7354b53f3ee5 " namespace: "vnf2" // Deleted VNF ... SampleVNF1HelmDeploy: type: tosca.nodes.AWS.Deployment.VNFDeployment requirements: cluster: EKSCluster vnfs: - vnf1.SampleVNF1 - vnf2.SampleVNF2	vnfd - des r_id "59 79e9 - be53 2ad0 " nam : "vn Upd VNF - des r_id "b7 839d -916 a166 " nam : "vn Add VNF ....

參數	Description	範例：之前	範例：之後
			Sample mpleElmD: typ tos es.A ploy VNFD ment rec nts: clu EKS r

參數	Description	範例：之前	範例：之後
			vnf - v leVN - v leVN

參數	Description	範例：之前	範例：之後

勾點	若要在建立網路函數之前和之後執行生命週期操作，請將 pre_create 和 post_create 掛鉤新增至VNFDeployment 節點。
	
	在此範例中，PreCreate Hook 掛鉤會在 執行個體化vnf3.SampleVNF3 之前執行，而掛PostCreateHook 鉤會在 vnf3.SampleVNF3 執行個體化之後執行。
		vnfds:
		  - descriptor_id: "*43c012fa-2616-41a8-a833-0dfd4c5a049e* "
		  namespace: " *vnf1*"
		  - descriptor_id: "*64222f98-ecd6-4871-bf94-7354b53f3ee5* "
		  namespace: " *vnf2*"
		...
		SampleVNF1HelmDeploy:
		  type: tosca.nodes.AWS.Deployment.VNFDeployment
		  requirements:
		  cluster: EKSCluster
		  vnfs:
		  - *vnf1.SampleVNF1*
		  - *vnf2.SampleVNF2 // Removed during update*
			vnfd
			
			-
			des
			r_id
			"43
			2616
			-
			a833
			d4c5
			"
			
			nam
			:
			"vr
			
			-
			des
			r_id
			"b7
			839d
			-916
			a166
			"
			
			nam
			:
			"vr
			....
			S
			ampl
			Helm
			y:

參數	Description	範例：之前	範例：之後
			typ tos es.A ploy VNFD ment req nts: clu EKS r vnf - v leVN No cha to thi fur as the nam and uui rem

參數	Description	範例：之前	範例：之後
			the sam - v leVN New VNF as the nam , vnf was not pre y pre int s: Hoo pos te: eHoo

參數	Description	範例：之前	範例：之後
			pre e: *Hook*

參數	Description	範例：之前	範例：之後
勾點	若要在更新網路函數之前和之後執行生命週期操作，您可以將pre_update 勾點和post_update 勾點新增至VNFDeployment 節點。 在此範例中， PreUpdate Hook 會在 更新前執行vnf1.SampleVNF1 ，PostUpdateHook 並在 vnf1.SampleVNF1 更新至命名空間 vnf1 更新 所指示uuid的vnf套件後執行。	vnfds: - descriptor_id: "*43c012fa-2616-41a8-a833-0dfd4c5a049e* " namespace: " *vnf1*" - descriptor_id: "*64222f98-ecd6-4871-bf94-7354b53f3ee5* " namespace: " *vnf2*" ... SampleVNF1HelmDeploy: type: tosca.nodes.AWS.Deployment.VNFDeployment requirements: cluster: EKSCluster vnfs: - *vnf1.SampleVNF1* - *vnf2.SampleVNF2*	vnfc - des r_ic "0e bd87 - b8a1 4666 " nam : "vr - des r_ic "64 ecd6 - bf94 4b53 " nam : "vr ... S ampl

參數	Description	範例：之前	範例：之後
			Helmy: typ tos es.A ploy VNFD ment req nts: clu EKS r vnf - v leVN A VNF upd as the uui cha for

參數	Description	範例：之前	範例：之後
			nam "vr - v leVN No cha to thi fur as nam and uui rem the sam int s: Hoo pre e: Hook

參數	Description	範例：之前	範例：之後
			pos te: *eHoc*

參數	Description	範例：之前	範例：之後
子網路	您可以從網路新增和刪除子網路。刪除子網路之前，請確認網路中的任何資源都不會使用該子網路。	*Free5GCSubnet01* : #Deleted Subnet type: tosca.nodes.AWS.Networking.Subnet properties: type: "PUBLIC" availability_zone: { get_input: subnet_01_az } cidr_block: { get_input: subnet_01_cidr_block } requirements: route_table: Free5GCRouteTable vpc: Free5GCVPC	*Free* *bnet* #Ne Sub typ tos es.A twor Subn pro s: typ "PU ava ity_ { g : sub _az cid k: { g : sub _cid ck }

參數	Description	範例：之前	範例：之後
			rec nts: rou le: Fre uteT vpc Fre C

參數	Description	範例：之前	範例：之後
Security groups (安全群組)	您可以從網路新增和刪除安全群組。刪除安全群組之前，請確認網路中的任何資源都不會使用該安全群組。	*Free5GCSecurityGroup01* : #Deleted Security Group type: tosca.nodes.AWS.Networking.SecurityGroup properties: description: "SecurityGroup for Free5GC cluster" name: "Free5GCSecurityGroup01" tags: - "Name=Free5GCEKSAdditionalSecurityGroup" requirements: vpc: Free5GCVPC *Free5GCSecurityGroupEgressRule01* : #Deleted Security Group Egress Node type: tosca.nodes.AWS.Networking.SecurityGroupEgressRule properties: ip_protocol: "tcp" from_port: 8000 to_port: 9000 description: "Egress Rule for free5GC cluster" cidr_ip : "172.10.10.1/24" requirements:	*Free* *curi* *up02* #Ne Sec Gro typ tos es.A twor Secu roup pro s: des on: "Se Grou for Fre clu nam "Fr ecur oup0 tag

參數	Description	範例：之前	範例：之後
		security_group: Free5GCSecurityGroup01 *Free5GCSecurityGroupIngressRule01* : #Deleted Security Group Ingress Node type: tosca.nodes.AWS.Networking.SecurityGroupIngressRule properties: ip_protocol: "tcp" from_port: 8000 to_port: 9000 description: "Ingress Rule for free5GC cluster" cidr_ip: "172.10.10.1/24" requirements: security_group: Free5GCSecurityGroup01	- "Na e5GC diti ecur oup" rec nts: vpc Fre C *Free curi upEg ule0* #Ne Sec Gro Egr Noc typ tos es.A twor Secu

參數	Description	範例：之前	範例：之後
			roup sRul pro s: ip_ ol: "to fro : 800 to_ 900 des on: "Eg Rul for fre clu cid "17 0.1/

參數	Description	範例：之前	範例：之後
			rec nts: sec grou Fre curi up02 *Free* *curi* *upIr* *Rule* #Ne Sec Gro Ing Noc typ tos es.A twor Secu roup ssRu pro s: ip_

參數	Description	範例：之前	範例：之後
			ol: "to fro : 800 to_ 900 des on: "In Rul for fre clu cid "17 0.1/ rec nts: sec grou Fre

參數	Description	範例：之前	範例：之後
			curi up02

參數	Description	範例：之前	範例：之後
網路介面	您可以從網路新增、修改和刪除 ENIs。	Free5GCENI01: #Modified ENI type: tosca.nodes.AWS.Networking.ENI properties: device_index: 2 requirements: subnet: *Free5GCENISubnet01* security_groups: - Free5GCSecurityGroup01 Free5GCENI02: #Modified ENI type: tosca.nodes.AWS.Networking.ENI properties: device_index: 3 source_dest_check: true requirements: subnet: Free5GCENISubnet01 *Free5GCENI04* : #Deleted ENI type: tosca.nodes.AWS.Networking.ENI properties: device_index: 4 source_dest_check: true requirements: subnet: Free5GCENISubnet01	Free I01: #Mo ENI typ tos es.A twor ENI pro s: dev dex: 2 req nts: sub *ISub* sec grou

參數	Description	範例：之前	範例：之後
			- Fre curi up01 Fre e5GC : #Mc ENI typ tos es.A twor ENI pro s: dev dex: 3 sou st_c tru rec nts: sub

參數	Description	範例：之前	範例：之後
			Fre ISub se grou - Fre curi up01 Free I03 #Ne ENI typ tos es.A twor ENI pro s: dev dex: 3 rec nts:

參數	Description	範例：之前	範例：之後
			sub Fre bnet sec grou - Fre curi up01

更新網路執行個體

Console

使用主控台更新網路執行個體

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路。
3. 選取網路執行個體。只有在網路執行個體的狀態為 Instantiated或 時，您才能更新網路執行個體Updated。
4. 選擇動作和更新。

   更新執行個體頁面會顯示網路詳細資訊，以及目前基礎設施中的參數清單。
5. 選擇新的網路套件。

新網路套件中的參數會出現在更新參數區段中。

6. 或者，更新更新參數區段中的參數值。如需您可以更新的參數值清單，請參閱 您可以更新的參數。
7. 選擇更新網路。

   AWS TNB 會驗證請求並啟動部署。部署狀態頁面隨即出現。
8. 使用重新整理圖示來追蹤網路執行個體的部署狀態。您也可以在部署任務區段中啟用自動重新整理，以追蹤每個任務的進度。

   當部署狀態變更為 時Completed，網路執行個體會更新。
9.
   - 如果驗證失敗，網路執行個體會保持與請求更新之前相同的狀態 - Instantiated或Updated。
   - 如果更新失敗，網路執行個體狀態會顯示 Update failed。選擇每個失敗任務的連結，以判斷原因。
   - 如果更新成功，網路執行個體狀態會顯示 Updated。

AWS CLI

使用 CLI 更新網路執行個體

使用 update-sol-network-instance 命令搭配UPDATE_NS更新類型來更新網路執行個體。

```
aws tnb update-sol-network-instance --ns-instance-id ^ni-[a-f0-9]{17}$ --
update-type UPDATE_NS --update-ns "{\"nsdInfoId\":\"^np-[a-f0-9]{17}$\",
 \"additionalParamsForNs\": {\"param1\": \"value1\"}}"
```

在 AWS TNB 中檢視網路執行個體

了解如何檢視網路執行個體。

Console

使用主控台檢視網路執行個體

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路執行個體。

3. 使用搜尋方塊來尋找網路執行個體。

AWS CLI

使用 檢視網路執行個體 AWS CLI

1. 使用 list-sol-network-instances 命令列出您的網路執行個體。

```
aws tnb list-sol-network-instances
```

2. 使用 get-sol-network-instance 命令來檢視特定網路執行個體的詳細資訊。

```
aws tnb get-sol-network-instance --ns-instance-id ^ni-[a-f0-9]{17}$
```

從 AWS TNB 終止和刪除網路執行個體

若要刪除網路執行個體，執行個體必須處於終止狀態。

Console

使用主控台終止和刪除網路執行個體

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路。
3. 選取網路執行個體的 ID。
4. 選擇終止。
5. 出現確認提示時，請輸入 ID 並選擇終止。
6. 重新整理以追蹤網路執行個體的狀態。
7. （選用）選取網路執行個體，然後選擇刪除。

AWS CLI

使用 終止和刪除網路執行個體 AWS CLI

1. 使用 terminate-sol-network-instance 命令來終止網路執行個體。

```
aws tnb terminate-sol-network-instance --ns-instance-id ^ni-[a-f0-9]{17}$
```

2. （選用）使用 delete-sol-network-instance 命令來刪除網路執行個體。

```
aws tnb delete-sol-network-instance --ns-instance-id ^ni-[a-f0-9]{17}$
```

AWS TNB 的網路操作

網路操作是對網路執行的任何操作，例如網路執行個體執行個體執行個體執行個體化或終止。

任務

- 檢視 AWS TNB 網路操作
- 取消 AWS TNB 網路操作

檢視 AWS TNB 網路操作

檢視網路操作的詳細資訊，包括網路操作中涉及的任務，以及任務的狀態。

Console

使用主控台檢視網路操作

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路執行個體。
3. 使用搜尋方塊尋找網路執行個體。
4. 在部署索引標籤上，選擇網路操作。

AWS CLI

使用 檢視網路操作 AWS CLI

1. 使用 list-sol-network-operations 命令列出所有網路操作。

```
aws tnb list-sol-network-operations
```

2. 使用 get-sol-network-operation 命令來檢視網路操作的詳細資訊。

```
aws tnb get-sol-network-operation --ns-lcm-op-occ-id ^no-[a-f0-9]{17}$
```

取消 AWS TNB 網路操作

了解如何取消網路操作。

Console

使用主控台取消網路操作

1. 在 https：//https://console.aws.amazon.com/tnb/ 開啟 AWS TNB 主控台。
2. 在導覽窗格中，選擇網路。
3. 選取網路的 ID 以開啟其詳細資訊頁面。
4. 在部署索引標籤上，選擇網路操作。
5. 選擇取消操作。

AWS CLI

使用 取消網路操作 AWS CLI

使用 cancel-sol-network-operation 命令取消網路操作。

```
aws tnb cancel-sol-network-operation --ns-lcm-op-occ-id ^no-[a-f0-9]{17}$
```

TNB 的 TOSCA AWS 參考

雲端應用程式的拓撲和協調規格 (TOSCA) 是一種宣告式語法，CSPs 會用來描述雲端型 Web 服務的拓撲、其元件、關係以及管理它們的程序。CSPs 描述連線點、連線點之間的邏輯連結，以及 TOSCA 範本中的親和性和安全性等政策。然後，CSPs會將範本上傳到 AWS TNB，TNB 會合成跨 AWS 可用區域建立正常運作 5G 網路所需的資源。

目錄

VNFD 範本

定義虛擬網路函數描述項 (VNFD) 範本。

語法

```
tosca_definitions_version: tnb_simple_yaml_1_0

topology_template:

  inputs:
    SampleInputParameter:
      type: String
      description: "Sample parameter description"
      default: "DefaultSampleValue"

  node_templates:
    SampleNode1: tosca.nodes.AWS.VNF
```

拓撲範本

`node_templates`

TOSCA AWS 節點。可能的節點包括：

- AWS.VNF

- AWS.Artifacts.Helm

AWS.VNF

定義 AWS 虛擬網路函數 (VNF) 節點。

語法

```
tosca.nodes.AWS.VNF:
  properties:
    descriptor_id: String
    descriptor_version: String
    descriptor_name: String
    provider: String
  requirements:
    helm: String
```

屬性

`descriptor_id`

描述項的 UUID。

必要：是

類型：字串

模式：`[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}`

`descriptor_version`

VNFD 的版本。

必要：是

類型：字串

模式：`^[0-9]{1,5}\\.[0-9]{1,5}\\.[0-9]{1,5}.*`

`descriptor_name`

描述項的名稱。

必要：是

類型：字串

`provider`

VNFD 的作者。

必要：是

類型：字串

要求

`helm`

定義容器成品的 Helm 目錄。這是 AWS.Artifacts.Helm 的參考。

必要：是

類型：字串

範例

```
SampleVNF:
  type: tosca.nodes.AWS.VNF
  properties:
    descriptor_id: "6a792e0c-be2a-45fa-989e-5f89d94ca898"
    descriptor_version: "1.0.0"
    descriptor_name: "Test VNF Template"
    provider: "Operator"
  requirements:
    helm: SampleHelm
```

AWS.Artifacts.Helm

定義 AWS Helm 節點。

語法

```
tosca.nodes.AWS.Artifacts.Helm:
  properties:
    implementation: String
```

屬性

`implementation`

CSAR 套件內包含 Helm Chart 的本機目錄。

必要：是

類型：字串

範例

```
SampleHelm:
  type: tosca.nodes.AWS.Artifacts.Helm
  properties:
    implementation: "./vnf-helm"
```

網路服務描述項範本

定義網路服務描述項 (NSD) 範本。

語法

```
tosca_definitions_version: tnb_simple_yaml_1_0

vnfds:
  - descriptor_id: String
    namespace: String

topology_template:

  inputs:
    SampleInputParameter:
      type: String
      description: "Sample parameter description"
      default: "DefaultSampleValue"

  node_templates:
    SampleNode1: tosca.nodes.AWS.NS
```

使用定義的參數

當您想要動態傳遞參數，例如 VPC { get_input: *input-parameter-name* } 節點的 CIDR 區塊時，您可以使用 語法，並在 NSD 範本中定義參數。然後在相同的 NSD 範本中重複使用 參數。

下列範例示範如何定義和使用參數：

```
tosca_definitions_version: tnb_simple_yaml_1_0

topology_template:

  inputs:
    cidr_block:
      type: String
      description: "CIDR Block for VPC"
      default: "10.0.0.0/24"

  node_templates:
    ExampleSingleClusterNS:
      type: tosca.nodes.AWS.NS
      properties:
        descriptor_id: "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
         .....

    ExampleVPC:
      type: tosca.nodes.AWS.Networking.VPC
      properties:
        cidr_block: { get_input: cidr_block }
```

VNFD 匯入

`descriptor_id`

描述項的 UUID。

必要：是

類型：字串

模式：`[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}`

`namespace`

唯一名稱。

必要：是

類型：字串

拓撲範本

`node_templates`

可能的 TOSCA AWS 節點包括：

- AWS.NS
- AWS.Compute.EKS
- AWS.Compute.EKS.AuthRole
- AWS.Compute.EKSManagedNode
- AWS.Compute.EKSSelfManagedNode
- AWS.Compute.PlacementGroup
- AWS.Compute.UserData
- AWS.Networking.SecurityGroup
- AWS.Networking.SecurityGroupEgressRule
- AWS.Networking.SecurityGroupIngressRule
- AWS.Resource.Import
- AWS.Networking.ENI
- AWS.HookExecution
- AWS.Networking.InternetGateway
- AWS.Networking.RouteTable
- AWS.Networking.Subnet
- AWS.Deployment.VNFDeployment
- AWS.Networking.VPC
- AWS.Networking.NATGateway
- AWS.Networking.Route

AWS.NS

定義 AWS 網路服務 (NS) 節點。

語法

```
tosca.nodes.AWS.NS:
  properties:
    descriptor_id: String
    descriptor_version: String
    descriptor_name: String
```

Properties

`descriptor_id`

描述項的 UUID。

必要：是

類型：字串

模式：`[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}`

`descriptor_version`

NSD 的版本。

必要：是

類型：字串

模式：`^[0-9]{1,5}\\.[0-9]{1,5}\\.[0-9]{1,5}.*`

`descriptor_name`

描述項的名稱。

必要：是

類型：字串

範例

```
SampleNS:
  type: tosca.nodes.AWS.NS
  properties:
    descriptor_id: "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

```
    descriptor_version: "1.0.0"
    descriptor_name: "Test NS Template"
```

AWS.Compute.EKS

提供叢集的名稱、所需的 Kubernetes 版本，以及允許 Kubernetes 控制平面管理 NFs 所需 AWS 資源的角色。Multus 容器網路介面 (CNI) 外掛程式已啟用。您可以連接多個網路介面，並將進階網路組態套用至以 Kubernetes 為基礎的網路函數。您也可以指定叢集端點存取和叢集的子網路。

語法

```
tosca.nodes.AWS.Compute.EKS:
  capabilities:
    multus:
      properties:
        enabled: Boolean
        multus_role: String
    ebs_csi:
      properties:
        enabled: Boolean
        version: String
  properties:
    version: String
    access: String
    cluster_role: String
    tags: List
    ip_family: String
  requirements:
    subnets: List
```

功能

`multus`

選用。定義 Multus 容器網路介面 (CNI) 用量的屬性。

如果您包含 `multus`，請指定 `enabled`和 `multus_role` 屬性。

`enabled`

指出是否已啟用預設 Multus 功能。

必要：是

類型：布林值

`multus_role`

Multus 網路介面管理的角色。

必要：是

類型：字串

`ebs_csi`

定義安裝在 Amazon EKS 叢集中 Amazon EBS 容器儲存介面 (CSI) 驅動程式的屬性。

啟用此外掛程式以在 AWS Outposts AWS 本機區域或 上使用 Amazon EKS 自我管理節點 AWS 區域。如需詳細資訊，請參閱《Amazon EKS 使用者指南》中的 Amazon Elastic Block Store CSI 驅動程式。

`enabled`

指出是否已安裝預設 Amazon EBS CSI 驅動程式。

必要：否

類型：布林值

`version`

Amazon EBS CSI 驅動程式附加元件的版本。版本必須符合 DescribeAddonVersions 動作傳回的其中一個版本。如需詳細資訊，請參閱《Amazon EKS API 參考》中的 DescribeAddonVersions

必要：否

類型：字串

Properties

`version`

叢集的 Kubernetes 版本。 AWS Telco Network Builder 支援 Kubernetes 版本 1.27 到 1.34。

必要：是

類型：字串

可能的值：1.27 | 1.28 | 1.29 | 1.30 | 1.31 | 1.32 | 1.33 | 1.34

`access`

叢集端點存取。

必要：是

類型：字串

可能的值：PRIVATE | PUBLIC | ALL

`cluster_role`

叢集管理的角色。

必要：是

類型：字串

`tags`

要連接到資源的標籤。

必要：否

類型：清單

`ip_family`

指出叢集中服務和 Pod 地址的 IP 系列。

允許的值：IPv4、 IPv6

預設值：IPv4

必要：否

類型：字串

要求

`subnets`

AWS.Networking.Subnet 節點。

必要：是

類型：清單

範例

```
SampleEKS:
  type: tosca.nodes.AWS.Compute.EKS
  properties:
    version: "1.26"
    access: "ALL"
    cluster_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/SampleRole"
    ip_family: "IPv6"
    tags:
      - "Name=SampleVPC"
      - "Environment=Testing"
  capabilities:
    multus:
      properties:
        enabled: true
        multus_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/MultusRole"
    ebs_csi:
      properties:
        enabled: true
        version: "v1.16.0-eksbuild.1"
  requirements:
    subnets:
    - SampleSubnet01
    - SampleSubnet02
```

AWS.Compute.EKS.AuthRole

AuthRole 可讓您將 IAM 角色新增至 Amazon EKS 叢集，aws-authConfigMap讓使用者可以使用 IAM 角色存取 Amazon EKS 叢集。

語法

```
tosca.nodes.AWS.Compute.EKS.AuthRole:
  properties:
    role_mappings: List
      arn: String
```

```
      groups: List
  requirements:
    clusters: List
```

Properties

`role_mappings`

定義需要新增至 Amazon EKS `aws-auth` 叢集 之 IAM 角色的映射清單`ConfigMap`。

`arn`

IAM 角色的 ARN。

必要：是

類型：字串

`groups`

要指派給 中定義之角色的 Kubernetes 群組`arn`。

必要：否

類型：清單

要求

`clusters`

AWS.Compute.EKS 節點。

必要：是

類型：清單

範例

```
EKSAuthMapRoles:
    type: tosca.nodes.AWS.Compute.EKS.AuthRole
    properties:
```

```
        role_mappings:
        - arn: arn:aws:iam::${AWS::TNB::AccountId}:role/TNBHookRole1
          groups:
          - system:nodes
          - system:bootstrappers
        - arn: arn:aws:iam::${AWS::TNB::AccountId}:role/TNBHookRole2
          groups:
          - system:nodes
          - system:bootstrappers
    requirements:
         clusters:
         - Free5GCEKS1
         - Free5GCEKS2
```

AWS.Compute.EKSManagedNode

AWS TNB 支援 EKS 受管節點群組，以自動化 Amazon EKS Kubernetes 叢集節點 (Amazon EC2 執行個體) 的佈建和生命週期管理。若要建立 EKS 節點群組，請執行下列動作：

- 提供 AMI 的 ID 或 AMI 類型，為您的叢集工作者節點選擇 Amazon Machine Image (AMI)。
- 提供 SSH 存取的 Amazon EC2 金鑰對，以及節點群組的擴展屬性。
- 確保您的節點群組與 Amazon EKS 叢集相關聯。
- 提供工作者節點的子網路。
- 或者，將安全群組、節點標籤和置放群組連接到節點群組。

語法

```
tosca.nodes.AWS.Compute.EKSManagedNode:
  capabilities:
    compute:
      properties:
        ami_type: String
        ami_id: String
        instance_types: List
        key_pair: String
        root_volume_encryption: Boolean
        root_volume_encryption_key_arn: String
        root_volume_size: Integer
    scaling:
      properties:
```

```
        desired_size: Integer
        min_size: Integer
        max_size: Integer
  properties:
    node_role: String
    tags: List
    kubernetes_version: String
  requirements:
    cluster: String
    subnets: List
    network_interfaces: List
    security_groups: List
    placement_group: String
    user_data: String
    labels: List
```

功能

compute

定義 Amazon EKS 受管節點群組運算參數的屬性，例如 Amazon EC2 執行個體類型和 Amazon EC2 執行個體 AMIs。

ami_type

Amazon EKS 支援的 AMI 類型。

必要：是

類型：字串

可能的值： AL2_x86_64 | AL2_x86_64_GPU | AL2_ARM_64 | AL2023_x86_64 | AL2023_ARM_64 | AL2023_x86_64_NVIDIA | AL2023_x86_64_NEURON | CUSTOM | BOTTLEROCKET_ARM_64 | BOTTLEROCKET_x86_64 | BOTTLEROCKET_ARM_64_NVIDIA | BOTTLEROCKET_x86_64_NVIDIA

ami_id

AMI 的 ID。

必要：否

類型：字串

> **Note**
>
> 如果在範本中同時指定 `ami_id` `ami_type`和 ， AWS TNB 只會使用 `ami_id`值來建立 `EKSManagedNode`。

`instance_types`

執行個體大小。

必要：是

類型：清單

`key_pair`

啟用 SSH 存取的 EC2 金鑰對。

必要：是

類型：字串

`root_volume_encryption`

啟用 Amazon EBS 根磁碟區的 Amazon EBS 加密。如果未提供此屬性， AWS TNB 預設會加密 Amazon EBS 根磁碟區。

必要：否

預設：true

類型：布林值

`root_volume_encryption_key_arn`

key AWS KMS . AWS TNB 的 ARN 支援一般金鑰 ARN、多區域金鑰 ARN 和別名 ARN。

必要：否

類型：字串

> **Note**
>
> - 如果 `root_volume_encryption`為 false，請勿包含 `root_volume_encryption_key_arn`。

- AWS TNB 支援 Amazon EBS 後端 AMI 的根磁碟區加密。
- 如果 AMI 的根磁碟區已加密，您必須包含`root_volume_encryption_key_arn`適用於 AWS TNB 的 ，才能重新加密根磁碟區。
- 如果 AMI 的根磁碟區未加密， AWS TNB 會使用 `root_volume_encryption_key_arn`來加密根磁碟區。

  如果您不包含 `root_volume_encryption_key_arn`， AWS TNB 會使用 提供的預設金鑰 AWS Key Management Service 來加密根磁碟區。
- AWS TNB 不會解密加密的 AMI。

`root_volume_size`

Amazon Elastic Block Store 根磁碟區的大小，以 GiBs為單位。

必要：否

預設：20

類型：整數

可能的值：1 到 16，384

`scaling`

定義 Amazon EKS 受管節點群組擴展參數的屬性，例如所需的 Amazon EC2 執行個體數量，以及節點群組中 Amazon EC2 執行個體的最小和最大數量。

`desired_size`

此 NodeGroup 中的執行個體數量。

必要：是

類型：整數

`min_size`

此 NodeGroup 中的執行個體數量下限。

必要：是

類型：整數

`max_size`

此 NodeGroup 中的執行個體數量上限。

必要：是

類型：整數

Properties

`node_role`

連接至 Amazon EC2 執行個體之 IAM 角色的 ARN。

必要：是

類型：字串

`tags`

要連接到資源的標籤。

必要：否

類型：清單

`kubernetes_version`

Managed Node 群組的 Kubernetes 版本。 AWS TNB 支援 Kubernetes 版本 1.27 到 1.34。考慮下列各項：

- 指定 `kubernetes_version`或 `ami_id`。不要同時指定兩者。
- `kubernetes_version` 必須小於或等於 AWS.Compute.EKSManagedNode 版本。
- AWS.Compute.EKSManagedNode 版本和 之間可能會有 3 個版本的差異。`kubernetes_version`
- 如果未`ami_id`指定 `kubernetes_version`或 ， AWS TNB 將使用`AWS.Compute.EKSManagedNode`版本的最新 AMI 來建立 `EKSManagedNode`

必要：否

類型：字串

可能的值：1.27 | 1.28 | 1.29 | 1.30 | 1.31 | 1.32 | 1.33 | 1.34

要求

`cluster`

AWS.Compute.EKS 節點。

必要：是

類型：字串

`subnets`

AWS.Networking.Subnet 節點。

必要：是

類型：清單

`network_interfaces`

AWS.Networking.ENI 節點。確保網路介面和子網路設定為相同的可用區域，否則執行個體化將會失敗。

當您設定 時`network_interfaces`，如果您在 AWS.Compute.EKS 節點中包含 `multus` 屬性，AWS TNB 會從 `multus_role` 屬性取得與 ENIs 相關的許可。否則， AWS TNB 會從 node_role 屬性取得與 ENIs相關的許可。

必要：否

類型：清單

`security_groups`

AWS.Networking.SecurityGroup 節點。

必要：否

類型：清單

`placement_group`

tosca.nodes AWS。Compute.PlacementGroup 節點。

必要：否

類型：字串

`user_data`

tosca.nodes AWS。Compute.UserData 節點參考。使用者資料指令碼會傳遞至受管節點群組啟動的 Amazon EC2 執行個體。將執行自訂使用者資料所需的許可新增至傳遞至節點群組的 node_role。

必要：否

類型：字串

`labels`

節點標籤的清單。節點標籤必須具有名稱和值。使用以下條件建立標籤：

- 名稱和值必須以 分隔=。
- 名稱和值的長度上限為 63 個字元。
- 標籤可以包含字母 (A-Z、a-z)、數字 (0-9) 和下列字元：`[-, _, ., *, ?]`
- 名稱和值必須以英數字元?、 或 * 字元開頭和結尾。

例如 `myLabelName1=*NodeLabelValue1`

必要：否

類型：清單

範例

```
SampleEKSManagedNode:
  type: tosca.nodes.AWS.Compute.EKSManagedNode
  capabilities:
    compute:
      properties:
        ami_type: "AL2_x86_64"
        instance_types:
          - "t3.xlarge"
        key_pair: "SampleKeyPair"
        root_volume_encryption: true
```

```
        root_volume_encryption_key_arn: "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        root_volume_size: 1500
    scaling:
      properties:
        desired_size: 1
        min_size: 1
        max_size: 1
  properties:
    node_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/SampleRole"
    tags:
      - "Name=SampleVPC"
      - "Environment=Testing"
    kubernetes_version:
      - "1.30"
  requirements:
    cluster: SampleEKS
    subnets:
      - SampleSubnet
    network_interfaces:
      - SampleENI01
      - SampleENI02
    security_groups:
      - SampleSecurityGroup01
      - SampleSecurityGroup02
    placement_group: SamplePlacementGroup
    user_data: CustomUserData
    labels:
      - "sampleLabelName001=sampleLabelValue001"
      - "sampleLabelName002=sampleLabelValue002"
```

AWS.Compute.EKSSelfManagedNode

AWS TNB 支援 Amazon EKS 自我管理節點，以自動化 Amazon EKS Kubernetes 叢集節點 (Amazon EC2 執行個體） 的佈建和生命週期管理。若要建立 Amazon EKS 節點群組，請執行下列動作：

- 透過提供 AMI 的 ID，為您的叢集工作者節點選擇 Amazon Machine Image (AMI)。
- 為 SSH 存取提供 Amazon EC2 金鑰對。
- 確保您的節點群組與 Amazon EKS 叢集相關聯。
- 提供執行個體類型和所需的大小、大小下限和大小上限。
- 提供工作者節點的子網路。

- 或者，將安全群組、節點標籤和置放群組連接到節點群組。

語法

```
tosca.nodes.AWS.Compute.EKSSelfManagedNode:
  capabilities:
    compute:
      properties:
        ami_id: String
        instance_type: String
        key_pair: String
        root_volume_encryption: Boolean
        root_volume_encryption_key_arn: String
        root_volume_size: Integer
    scaling:
      properties:
        desired_size: Integer
        min_size: Integer
        max_size: Integer
  properties:
    node_role: String
    tags: List
  requirements:
    cluster: String
    subnets: List
    network_interfaces: List
    security_groups: List
    placement_group: String
    user_data: String
    labels: List
```

功能

compute

定義 Amazon EKS 自我管理節點運算參數的屬性，例如 Amazon EC2 執行個體類型和 Amazon EC2 執行個體 AMIs。

`ami_id`

用來啟動執行個體的 AMI ID。 AWS TNB 支援利用 IMDSv2 的執行個體。如需詳細資訊，請參閱IMDS 版本。

> **Note**
>
> 您可以更新 的 AMI IDEKSSelfManagedNode。AMI 的 Amazon EKS 版本必須與 Amazon EKS 叢集版本相同或低於 2 個版本。例如，如果 Amazon EKS 叢集版本為 1.31，則 Amazon EKS AMI 版本必須為 1.31、1.30 或 1.29。

必要：是

類型：字串

`instance_type`

執行個體大小。

必要：是

類型：字串

`key_pair`

啟用 SSH 存取的 Amazon EC2 金鑰對。

必要：是

類型：字串

`root_volume_encryption`

啟用 Amazon EBS 根磁碟區的 Amazon EBS 加密。如果未提供此屬性， AWS TNB 預設會加密 Amazon EBS 根磁碟區。

必要：否

預設：true

類型：布林值

`root_volume_encryption_key_arn`

key AWS KMS . AWS TNB 的 ARN 支援一般金鑰 ARN、多區域金鑰 ARN 和別名 ARN。

必要：否

類型：字串

> **Note**
>
> - 如果 `root_volume_encryption`為 false，請勿包含 `root_volume_encryption_key_arn`。
> - AWS TNB 支援 Amazon EBS 後端 AMI 的根磁碟區加密。
> - 如果 AMI 的根磁碟區已加密，您必須包含`root_volume_encryption_key_arn`適用於 AWS TNB 的 ，才能重新加密根磁碟區。
> - 如果 AMI 的根磁碟區未加密， AWS TNB 會使用 `root_volume_encryption_key_arn`來加密根磁碟區。
>
> 如果您不包含 `root_volume_encryption_key_arn`， AWS TNB 會使用 AWS Managed Services 來加密根磁碟區。
> - AWS TNB 不會解密加密的 AMI。

`root_volume_size`

Amazon Elastic Block Store 根磁碟區的大小，以 GiBs為單位。

必要：否

預設：20

類型：整數

可能的值：1 到 16，384

`scaling`

定義 Amazon EKS 自我管理節點擴展參數的屬性，例如所需的 Amazon EC2 執行個體數量，以及節點群組中 Amazon EC2 執行個體的數量下限和上限。

`desired_size`

此 NodeGroup 中的執行個體數量。

必要：是

類型：整數

`min_size`

此 NodeGroup 中的執行個體數量下限。

必要：是

類型：整數

`max_size`

此 NodeGroup 中的執行個體數量上限。

必要：是

類型：整數

Properties

`node_role`

連接至 Amazon EC2 執行個體之 IAM 角色的 ARN。

必要：是

類型：字串

`tags`

要連接到資源的標籤。標籤將傳播到 資源建立的執行個體。

必要：否

類型：清單

要求

`cluster`

AWS.Compute.EKS 節點。

必要：是

類型：字串

`subnets`

AWS.Networking.Subnet 節點。

必要：是

類型：清單

`network_interfaces`

AWS.Networking.ENI 節點。確保網路介面和子網路設定為相同的可用區域，否則執行個體化將會失敗。

當您設定 時`network_interfaces`，如果您在 AWS.Compute.EKS 節點中包含 `multus` 屬性，AWS TNB 會從 `multus_role` 屬性取得與 ENIs 相關的許可。否則， AWS TNB 會從 node_role 屬性取得與 ENIs相關的許可。

必要：否

類型：清單

`security_groups`

AWS.Networking.SecurityGroup 節點。

必要：否

類型：清單

`placement_group`

tosca.nodes AWS。Compute.PlacementGroup 節點。

必要：否

類型：字串

`user_data`

tosca.nodes AWS。Compute.UserData 節點參考。使用者資料指令碼會傳遞至自我管理節點群組啟動的 Amazon EC2 執行個體。將執行自訂使用者資料所需的許可新增至傳遞至節點群組的 node_role。

必要：否

類型：字串

`labels`

節點標籤的清單。節點標籤必須具有名稱和值。使用以下條件建立標籤：

- 名稱和值必須以 分隔=。
- 名稱和值的長度上限為 63 個字元。
- 標籤可以包含字母 (A-Z、a-z、)、數字 (0-9) 和下列字元：`[-, _, ., *, ?]`
- 名稱和值必須以英數字元?、 或 * 字元開頭和結尾。

例如 `myLabelName1=*NodeLabelValue1`

必要：否

類型：清單

範例

```
SampleEKSSelfManagedNode:
  type: tosca.nodes.AWS.Compute.EKSSelfManagedNode
  capabilities:
    compute:
      properties:
        ami_id: "ami-123123EXAMPLE"
        instance_type: "c5.large"
        key_pair: "SampleKeyPair"
        root_volume_encryption: true
        root_volume_encryption_key_arn: "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        root_volume_size: 1500
    scaling:
      properties:
        desired_size: 1
        min_size: 1
        max_size: 1
  properties:
    node_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/SampleNodeRole"
    tags:
      - "Name=SampleVPC"
      - "Environment=Testing"
  requirements:
    cluster: SampleEKSCluster
    subnets:
```

```
      - SampleSubnet
    network_interfaces:
      - SampleNetworkInterface01
      - SampleNetworkInterface02
    security_groups:
      - SampleSecurityGroup01
      - SampleSecurityGroup02
    placement_group: SamplePlacementGroup
    user_data: CustomUserData
    labels:
      - "sampleLabelName001=sampleLabelValue001"
      - "sampleLabelName002=sampleLabelValue002"
```

AWS.Compute.PlacementGroup

PlacementGroup 節點支援放置 Amazon EC2 執行個體的不同策略。

當您啟動新的 Amazon EC2instance時，Amazon EC2 服務會嘗試以將您的所有執行個體分散到基礎硬體的方式放置執行個體，以將相關故障降至最低。不過，您可以使用 置放群組 來影響一組 互相依存執行個體的置放，以符合您的工作負載需求。

語法

```
tosca.nodes.AWS.Compute.PlacementGroup
  properties:
    strategy: String
    partition_count: Integer
    tags: List
```

Properties

`strategy`

用來放置 Amazon EC2 執行個體的策略。

必要：是

類型：字串

可能的值：CLUSTER | PARTITION | SPREAD_HOST | SPREAD_RACK

- CLUSTER – 將執行個體封裝在可用區域內。此策略可讓工作負載達到高效能運算 (HPC) 應用程式典型緊密耦合節點對節點通訊所需的低延遲網路效能。

- PARTITION – 將您的執行個體分散到邏輯分割區，讓一個分割區中的執行個體群組不會與不同分割區中的執行個體群組共用基礎硬體。大量分散和複寫的工作負載 (例如 Hadoop、Cassandra 和 Kafka) 通常採取此策略。
- SPREAD_RACK – 跨不同的基礎硬體放置一小組執行個體，以減少相互關聯的故障。
- SPREAD_HOST – 僅用於 Outpost 置放群組。跨不同的基礎硬體放置一小組執行個體，以減少相互關聯的故障。

`partition_count`

分割區數。

必要：只有在 `strategy` 設定為 時才需要PARTITION。

類型：整數

可能的值：1 | 2 |3 | 4 | 5 | 6 | 7

`tags`

您可以連接到置放群組資源的標籤。

必要：否

類型：清單

範例

```
ExamplePlacementGroup:
  type: tosca.nodes.AWS.Compute.PlacementGroup
  properties:
    strategy: "PARTITION"
    partition_count: 5
    tags:
      - tag_key=tag_value
```

AWS.Compute.UserData

AWS TNB 支援透過 Network Service Descriptor (NSD) 中的 UserData 節點，使用自訂使用者資料啟動 Amazon EC2 執行個體。如需自訂使用者資料的詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用者資料和 shell 指令碼。

在網路執行個體化期間， AWS TNB 會透過使用者資料指令碼將 Amazon EC2 執行個體註冊提供給叢集。同時提供自訂使用者資料時， AWS TNB 會合併兩個指令碼，並將它們做為多mime 指令碼傳遞給 Amazon EC2。自訂使用者資料指令碼會在 Amazon EKS 註冊指令碼之前執行。

若要在使用者資料指令碼中使用自訂變數，請在開啟大括號 !後新增驚嘆號{。例如，若要在指令碼MyVariable中使用 ，請輸入： {!MyVariable}

> **Note**
>
> - AWS TNB 支援大小上限為 7 KB 的使用者資料指令碼。
> - 由於 AWS TNB 使用 CloudFormation 來處理和轉譯multimime使用者資料指令碼，因此請確保指令碼符合所有 CloudFormation 規則。

語法

```
tosca.nodes.AWS.Compute.UserData:
  properties:
    implementation: String
    content_type: String
```

Properties

`implementation`

使用者資料指令碼定義的相對路徑。格式必須為： `./scripts/script_name.sh`

必要：是

類型：字串

`content_type`

使用者資料指令碼的內容類型。

必要：是

類型：字串

可能的值： `x-shellscript`

範例

```
ExampleUserData:
  type: tosca.nodes.AWS.Compute.UserData
  properties:
    content_type: "text/x-shellscript"
    implementation: "./scripts/customUserData.sh"
```

AWS.Networking.SecurityGroup

AWS TNB 支援安全群組自動佈建 Amazon EC2 安全群組，您可以將這些群組連接到 Amazon EKS Kubernetes 叢集節點群組。

語法

```
tosca.nodes.AWS.Networking.SecurityGroup
  properties:
    description: String
    name: String
    tags: List
  requirements:
    vpc: String
```

Properties

`description`

安全群組的描述。您最多可以使用 255 個字元來描述群組。您只能包含字母 (A-Z 和 a-z)、數字 (0-9)、空格和下列特殊字元：._-：/()#，@【】+=&；{}！$*

必要：是

類型：字串

`name`

安全群組的名稱。名稱最多可使用 255 個字元。您只能包含字母 (A-Z 和 a-z)、數字 (0-9)、空格和下列特殊字元：._-：/()#，@【】+=&；{}！$*

必要：是

類型：字串

`tags`

您可以連接到安全群組資源的標籤。

必要：否

類型：清單

要求

`vpc`

AWS.Networking.VPC 節點。

必要：是

類型：字串

範例

```
SampleSecurityGroup001:
  type: tosca.nodes.AWS.Networking.SecurityGroup
  properties:
        description: "Sample Security Group for Testing"
        name: "SampleSecurityGroup"
        tags:
          - "Name=SecurityGroup"
          - "Environment=Testing"
      requirements:
        vpc: SampleVPC
```

AWS.Networking.SecurityGroupEgressRule

AWS TNB 支援安全群組輸出規則，以自動佈建可連接到 AWS.Networking.SecurityGroup 的 Amazon EC2 安全群組輸出規則。請注意，您必須提供 cidr_ip/destination_security_group/destination_prefix_list 作為輸出流量的目的地。

語法

```
AWS.Networking.SecurityGroupEgressRule
```

```
  properties:
    ip_protocol: String
    from_port: Integer
    to_port: Integer
    description: String
    destination_prefix_list: String
    cidr_ip: String
    cidr_ipv6: String
  requirements:
    security_group: String
    destination_security_group: String
```

Properties

`cidr_ip`

CIDR 格式的 IPv4 地址範圍。您必須指定允許輸出流量的 CIDR 範圍。

必要：否

類型：字串

`cidr_ipv6`

CIDR 格式的 IPv6 地址範圍，用於輸出流量。您必須指定目標安全群組 (`destination_security_group` 或 `destination_prefix_list`) 或 CIDR 範圍 (`cidr_ip` 或 `cidr_ipv6`)。

必要：否

類型：字串

`description`

輸出 (傳出) 安全群組規則的描述。您最多可以使用 255 個字元來描述規則。

必要：否

類型：字串

`destination_prefix_list`

現有 Amazon VPC 受管字首清單的字首清單 ID。這是來自與安全群組相關聯之節點群組執行個體的目的地。如需受管字首清單的詳細資訊，請參閱《Amazon VPC 使用者指南》中的受管字首清單。

必要：否

類型：字串

`from_port`

如果通訊協定是 TCP 或 UDP，這是連接埠範圍的開頭。如果通訊協定是 ICMP 或 ICMPv6，這是類型編號。值 -1 表示所有 ICMP/ICMPv6 類型。若您指定所有 ICMP/ICMPv6 類型，您必須指定所有 ICMP/ICMPv6 代碼。

必要：否

類型：整數

`ip_protocol`

IP 通訊協定名稱 (tcp、udp、icmp、icmpv6) 或通訊協定編號。使用 -1 指定所有通訊協定。授權安全群組規則時，不論您指定的連接埠範圍為何，指定 tcp、udp、icmp 或 icmpv6 以外的 -1 或通訊協定號碼，都允許所有連接埠上的流量。對於 tcp、udp 和 icmp，您必須指定連接埠範圍。對於 icmpv6，連接埠範圍是選用的；如果您省略連接埠範圍，則允許所有類型和代碼的流量。

必要：是

類型：字串

`to_port`

如果通訊協定是 TCP 或 UDP，這是連接埠範圍的結尾。如果通訊協定是 ICMP 或 ICMPv6，這是代碼。值 -1 表示所有 ICMP/ICMPv6 代碼。若您指定所有 ICMP/ICMPv6 類型，您必須指定所有 ICMP/ICMPv6 代碼。

必要：否

類型：整數

要求

`security_group`

要新增此規則的安全群組 ID。

必要：是

類型：字串

`destination_security_group`

允許輸出流量目的地安全群組的 ID 或 TOSCA 參考。

必要：否

類型：字串

範例

```
SampleSecurityGroupEgressRule:
      type: tosca.nodes.AWS.Networking.SecurityGroupEgressRule
      properties:
        ip_protocol: "tcp"
        from_port: 8000
        to_port: 9000
        description: "Egress Rule for sample security group"
        cidr_ipv6: "2600:1f14:3758:ca00::/64"
      requirements:
        security_group: SampleSecurityGroup001
        destination_security_group: SampleSecurityGroup002
```

AWS.Networking.SecurityGroupIngressRule

AWS TNB 支援安全群組傳入規則，以自動佈建可連接到 AWS.Networking.SecurityGroup 的 Amazon EC2 安全群組傳入規則。請注意，您必須提供 cidr_ip/source_security_group/source_prefix_list 作為輸入流量的來源。

語法

```
AWS.Networking.SecurityGroupIngressRule
  properties:
    ip_protocol: String
    from_port: Integer
    to_port: Integer
    description: String
    source_prefix_list: String
    cidr_ip: String
    cidr_ipv6: String
  requirements:
    security_group: String
```

```
    source_security_group: String
```

Properties

`cidr_ip`

CIDR 格式的 IPv4 地址範圍。您必須指定允許輸入流量的 CIDR 範圍。

必要：否

類型：字串

`cidr_ipv6`

輸入流量的 CIDR 格式 IPv6 地址範圍。您必須指定來源安全群組 (`source_security_group` 或 `source_prefix_list`) 或 CIDR 範圍 (`cidr_ip` 或 `cidr_ipv6`)。

必要：否

類型：字串

`description`

傳入（傳入）安全群組規則的描述。您最多可以使用 255 個字元來描述規則。

必要：否

類型：字串

`source_prefix_list`

現有 Amazon VPC 受管字首清單的字首清單 ID。這是允許與安全群組相關聯之節點群組執行個體接收流量的來源。如需受管字首清單的詳細資訊，請參閱《Amazon VPC 使用者指南》中的受管字首清單。

必要：否

類型：字串

`from_port`

如果通訊協定是 TCP 或 UDP，這是連接埠範圍的開頭。如果通訊協定是 ICMP 或 ICMPv6，這是類型編號。值 -1 表示所有 ICMP/ICMPv6 類型。若您指定所有 ICMP/ICMPv6 類型，您必須指定所有 ICMP/ICMPv6 代碼。

必要：否

類型：整數

`ip_protocol`

IP 通訊協定名稱 (tcp、udp、icmp、icmpv6) 或通訊協定編號。使用 -1 指定所有通訊協定。授權安全群組規則時，不論您指定的連接埠範圍為何，指定 tcp、udp、icmp 或 icmpv6 以外的 -1 或通訊協定號碼，都允許所有連接埠上的流量。對於 tcp、udp 和 icmp，您必須指定連接埠範圍。對於 icmpv6，連接埠範圍是選用的；如果您省略連接埠範圍，則允許所有類型和代碼的流量。

必要：是

類型：字串

`to_port`

如果通訊協定是 TCP 或 UDP，這是連接埠範圍的結尾。如果通訊協定是 ICMP 或 ICMPv6，這是代碼。值 -1 表示所有 ICMP/ICMPv6 代碼。若您指定所有 ICMP/ICMPv6 類型，您必須指定所有 ICMP/ICMPv6 代碼。

必要：否

類型：整數

要求

`security_group`

要新增此規則的安全群組 ID。

必要：是

類型：字串

`source_security_group`

允許輸入流量之來源安全群組的 ID 或 TOSCA 參考。

必要：否

類型：字串

範例

```
SampleSecurityGroupIngressRule:
```

```
      type: tosca.nodes.AWS.Networking.SecurityGroupIngressRule
      properties:
        ip_protocol: "tcp"
        from_port: 8000
        to_port: 9000
        description: "Ingress Rule for free5GC cluster on IPv6"
        cidr_ipv6: "2600:1f14:3758:ca00::/64"
      requirements:
        security_group: SampleSecurityGroup1
        source_security_group: SampleSecurityGroup2
```

AWS.Resource.Import

您可以將下列 AWS 資源匯入 AWS TNB：

- VPC
- 子網路
- 路由表
- 網際網路閘道
- 安全群組

語法

```
tosca.nodes.AWS.Resource.Import
  properties:
    resource_type: String
    resource_id: String
```

Properties

`resource_type`

匯入至 AWS TNB 的資源類型。

必要：否

類型：清單

`resource_id`

匯入至 AWS TNB 的資源 ID。

必要：否

類型：清單

範例

```
SampleImportedVPC:
  type: tosca.nodes.AWS.Resource.Import
  properties:
    resource_type: "tosca.nodes.AWS.Networking.VPC"
    resource_id: "vpc-123456"
```

AWS.Networking.ENI

網路界面是 VPC 中代表虛擬網路卡的邏輯聯網元件。網路界面會根據其子網路自動或手動指派 IP 地址。在子網路中部署 Amazon EC2 執行個體之後，您可以將網路介面連接至子網路，或從該 Amazon EC2 執行個體分離網路介面，然後重新連接至該子網路中的另一個 Amazon EC2 執行個體。裝置索引會以連接順序識別位置。

語法

```
tosca.nodes.AWS.Networking.ENI:
  properties:
    device_index: Integer
    source_dest_check: Boolean
    tags: List
  requirements:
    subnet: String
    security_groups: List
```

Properties

`device_index`

裝置索引必須大於零。

必要：是

類型：整數

`source_dest_check`

指出網路界面是否執行來源/目的地檢查。`true` 值表示啟用檢查，`false` 值表示停用檢查。

允許的值：true、false

預設：true

必要：否

類型：布林值

`tags`

要連接到資源的標籤。

必要：否

類型：清單

要求

`subnet`

AWS.Networking.Subnet 節點。

必要：是

類型：字串

`security_groups`

AWS.Networking.SecurityGroup 節點。

必要：否

類型：字串

範例

```
SampleENI:
  type: tosca.nodes.AWS.Networking.ENI
```

```
  properties:
    device_index: 5
    source_dest_check: true
    tags:
      - "Name=SampleVPC"
      - "Environment=Testing"
  requirements:
    subnet: SampleSubnet
    security_groups:
      - SampleSecurityGroup01
      - SampleSecurityGroup02
```

AWS.HookExecution

生命週期關聯可讓您執行自己的指令碼，做為基礎設施和網路執行個體的一部分。

語法

```
tosca.nodes.AWS.HookExecution:
  capabilities:
    execution:
      properties:
        type: String
  requirements:
    definition: String
    vpc: String
```

功能

execution

執行勾點指令碼之勾點執行引擎的屬性。

`type`

勾點執行引擎類型。

必要：否

類型：字串

可能的值： `CODE_BUILD`

要求

`definition`

AWS.HookDefinition.Bash 節點。

必要：是

類型：字串

`vpc`

AWS.Networking.VPC 節點。

必要：是

類型：字串

範例

```
SampleHookExecution:
  type: tosca.nodes.AWS.HookExecution
  requirements:
    definition: SampleHookScript
    vpc: SampleVPC
```

AWS.Networking.InternetGateway

定義 AWS 網際網路閘道節點。

語法

```
tosca.nodes.AWS.Networking.InternetGateway:
  capabilities:
    routing:
      properties:
        dest_cidr: String
        ipv6_dest_cidr: String
  properties:
    tags: List
    egress_only: Boolean
  requirements:
```

```
    vpc: String
    route_table: String
```

功能

routing

定義 VPC 內路由連線的屬性。您必須包含 dest_cidr或 ipv6_dest_cidr 屬性。

dest_cidr

用於目的地比對的 IPv4 CIDR 區塊。此屬性用於在 中建立路由，RouteTable其值會用作 DestinationCidrBlock。

必要：如果您包含 ipv6_dest_cidr 屬性，則為否。

類型：字串

ipv6_dest_cidr

用於目的地比對的 IPv6 CIDR 區塊。

必要：如果您包含 dest_cidr 屬性，則為否。

類型：字串

Properties

tags

要連接到資源的標籤。

必要：否

類型：清單

egress_only

IPv6-specific 屬性。指出網際網路閘道是否僅用於輸出通訊。當 egress_only 為 true 時，您必須定義 ipv6_dest_cidr 屬性。

必要：否

類型：布林值

要求

vpc

AWS.Networking.VPC 節點。

必要：是

類型：字串

route_table

AWS.Networking.RouteTable 節點。

必要：是

類型：字串

範例

```
Free5GCIGW:
  type: tosca.nodes.AWS.Networking.InternetGateway
  properties:
    egress_only: false
  capabilities:
    routing:
      properties:
        dest_cidr: "0.0.0.0/0"
        ipv6_dest_cidr: "::/0"
  requirements:
    route_table: Free5GCRouteTable
    vpc: Free5GCVPC
Free5GCEGW:
  type: tosca.nodes.AWS.Networking.InternetGateway
  properties:
    egress_only: true
  capabilities:
    routing:
      properties:
        ipv6_dest_cidr: "::/0"
  requirements:
    route_table: Free5GCPrivateRouteTable
    vpc: Free5GCVPC
```

AWS.Networking.RouteTable

路由表包含一組稱為路由的規則，用於判斷來自 VPC 或閘道內子網路的網路流量導向位置。您必須將路由表與 VPC 建立關聯。

語法

```
tosca.nodes.AWS.Networking.RouteTable:
  properties:
    tags: List
  requirements:
    vpc: String
```

Properties

`tags`

要連接到資源的標籤。

必要：否

類型：清單

要求

`vpc`

AWS.Networking.VPC 節點。

必要：是

類型：字串

範例

```
SampleRouteTable:
  type: tosca.nodes.AWS.Networking.RouteTable
  properties:
    tags:
      - "Name=SampleVPC"
```

```
      - "Environment=Testing"
  requirements:
    vpc: SampleVPC
```

AWS.Networking.Subnet

子網路是 VPC 中的 IP 地址範圍，且必須完全位於一個可用區域內。您必須為子網路指定 VPC、CIDR 區塊、可用區域和路由表。您還必須定義子網路是私有還是公有。

語法

```
tosca.nodes.AWS.Networking.Subnet:
  properties:
    type: String
    availability_zone: String
    cidr_block: String
    ipv6_cidr_block: String
    ipv6_cidr_block_suffix: String
    outpost_arn: String
    tags: List
  requirements:
    vpc: String
    route_table: String
```

Properties

`type`

指示在此子網路中啟動的執行個體是否會收到公有 IPv4 地址。

必要：是

類型：字串

可能的值：`PUBLIC | PRIVATE`

`availability_zone`

子網路的可用區域。此欄位支援 AWS 區域內的 AWS 可用區域，例如 `us-west-2`（美國西部（奧勒岡））。它也支援可用區域內的 AWS 本機區域，例如 `us-west-2-lax-1a`。

必要：是

類型：字串

`cidr_block`

子網路的 CIDR 區塊。

必要：否

類型：字串

`ipv6_cidr_block`

用來建立 IPv6 子網路的 CIDR 區塊。如果您包含此屬性，請勿包含 `ipv6_cidr_block_suffix`。

必要：否

類型：字串

`ipv6_cidr_block_suffix`

透過 Amazon VPC 建立之子網路的 IPv6 CIDR 區塊的 2 位數十六進位尾碼。使用下列格式：*2-digit hexadecimal*`::/`*subnetMask*

如果您包含此屬性，請勿包含 `ipv6_cidr_block`。

必要：否

類型：字串

`outpost_arn`

要在 AWS Outposts 其中建立子網路的 ARN。如果您想要在 上啟動 Amazon EKS 自我管理節點，請將此屬性新增至 NSD 範本 AWS Outposts。如需詳細資訊，請參閱《Amazon EKS 使用者指南 AWS Outposts》中的 上的 Amazon EKS。

如果您將此屬性新增至 NSD 範本，則必須將 `availability_zone` 屬性的值設定為 的可用區域 AWS Outposts。

必要：否

類型：字串

`tags`

要連接到資源的標籤。

必要：否

類型：清單

要求

vpc

[AWS.Networking.VPC](#) 節點。

必要：是

類型：字串

route_table

[AWS.Networking.RouteTable](#) 節點。

必要：是

類型：字串

範例

```
SampleSubnet01:
  type: tosca.nodes.AWS.Networking.Subnet
  properties:
    type: "PUBLIC"
    availability_zone: "us-east-1a"
    cidr_block: "10.100.50.0/24"
    ipv6_cidr_block_suffix: "aa::/64"
    outpost_arn: "arn:aws:outposts:region:accountId:outpost/op-11223344EXAMPLE"
    tags:
      - "Name=SampleVPC"
      - "Environment=Testing"
  requirements:
    vpc: SampleVPC
    route_table: SampleRouteTable

SampleSubnet02:
  type: tosca.nodes.AWS.Networking.Subnet
  properties:
```

```
    type: "PUBLIC"
    availability_zone: "us-west-2b"
    cidr_block: "10.100.50.0/24"
    ipv6_cidr_block: "2600:1f14:3758:ca00::/64"
  requirements:
    route_table: SampleRouteTable
    vpc: SampleVPC
```

AWS.Deployment.VNFDeployment

NF 部署的建模方式是提供基礎設施和與其相關聯的應用程式。叢集屬性會指定 EKS 叢集來託管您的 NFs。vnfs 屬性會指定您部署的網路函數。您也可以提供 pre_create 和 post_create 類型的選用生命週期關聯操作，以執行您部署的特定指示，例如呼叫庫存管理系統 API。

語法

```
tosca.nodes.AWS.Deployment.VNFDeployment:
  requirements:
    deployment: String
    cluster: String
    vnfs: List
  interfaces:
    Hook:
      pre_create: String
      post_create: String
```

要求

`deployment`

AWS.Deployment.VNFDeployment 節點。

必要：否

類型：字串

`cluster`

AWS.Compute.EKS 節點。

必要：是

類型：字串

vnfs

AWS.VNF 節點。

必要：是

類型：字串

介面

勾點

定義生命週期掛鉤執行時的階段。

pre_create

AWS.HookExecution 節點。此掛鉤會在VNFDeployment節點部署之前執行。

必要：否

類型：字串

post_create

AWS.HookExecution 節點。此掛鉤會在VNFDeployment節點部署之後執行。

必要：否

類型：字串

範例

```
SampleHelmDeploy:
  type: tosca.nodes.AWS.Deployment.VNFDeployment
  requirements:
    deployment: SampleHelmDeploy2
    cluster: SampleEKS
    vnfs:
      - vnf.SampleVNF
  interfaces:
    Hook:
      pre_create: SampleHook
```

AWS.Networking.VPC

您必須為虛擬私有雲端 (VPC) 指定 CIDR 區塊。

語法

```
tosca.nodes.AWS.Networking.VPC:
  properties:
    cidr_block: String
    ipv6_cidr_block: String
    dns_support: String
    tags: List
```

Properties

`cidr_block`

VPC 的 IPv4 網路範圍 (以 CIDR 表示法表示)。

必要：是

類型：字串

`ipv6_cidr_block`

用來建立 VPC 的 IPv6 CIDR 區塊。

允許的值： `AMAZON_PROVIDED`

必要：否

類型：字串

`dns_support`

指示 VPC 中啟動的執行個體是否會收到 DNS 主機名稱。

必要：否

類型：布林值

預設：`false`

`tags`

要連接到資源的標籤。

必要：否

類型：清單

範例

```
SampleVPC:
  type: tosca.nodes.AWS.Networking.VPC
  properties:
    cidr_block: "10.100.0.0/16"
    ipv6_cidr_block: "AMAZON_PROVIDED"
    dns_support: true
    tags:
      - "Name=SampleVPC"
      - "Environment=Testing"
```

AWS.Networking.NATGateway

您可以透過子網路定義公有或私有 NAT Gateway 節點。對於公有閘道，如果您不提供彈性 IP 配置 ID， AWS TNB 會為您的 帳戶配置彈性 IP，並將其與閘道建立關聯。

語法

```
tosca.nodes.AWS.Networking.NATGateway:
  requirements:
    subnet: String
    internet_gateway: String
  properties:
    type: String
    eip_allocation_id: String
    tags: List
```

Properties

`subnet`

AWS.Networking.Subnet 節點參考。

必要：是

類型：字串

`internet_gateway`

AWS.Networking.InternetGateway 節點參考。

必要：是

類型：字串

Properties

`type`

指出閘道是公有還是私有。

允許的值：`PUBLIC`、 `PRIVATE`

必要：是

類型：字串

`eip_allocation_id`

代表彈性 IP 地址配置的 ID。

必要：否

類型：字串

`tags`

要連接到資源的標籤。

必要：否

類型：清單

範例

```
Free5GCNatGateway01:
  type: tosca.nodes.AWS.Networking.NATGateway
```

```
    requirements:
       subnet: Free5GCSubnet01
       internet_gateway: Free5GCIGW
    properties:
       type: PUBLIC
       eip_allocation_id: eipalloc-12345
```

AWS.Networking.Route

您可以定義路由節點，將目的地路由關聯至 NAT Gateway 做為目標資源，並將路由新增至關聯的路由表。

語法

```
tosca.nodes.AWS.Networking.Route:
  properties:
    dest_cidr_blocks: List
  requirements:
    nat_gateway: String
    route_table: String
```

Properties

`dest_cidr_blocks`

目的地 IPv4 路由至目標資源的清單。

必要：是

類型：清單

成員類型：字串

要求

`nat_gateway`

AWS.Networking.NATGateway 節點參考。

必要：是

類型：字串

`route_table`

AWS.Networking.RouteTable 節點參考。

必要：是

類型：字串

範例

```
Free5GCRoute:
  type: tosca.nodes.AWS.Networking.Route
  properties:
    dest_cidr_blocks:
      - 0.0.0.0/0
      - 10.0.0.0/28
  requirements:
    nat_gateway: Free5GCNatGateway01
    route_table: Free5GCRouteTable
```

AWS.Store.SSMParameters

您可以透過 AWS TNB 建立 SSM 參數。您建立的 SSM 參數是在 SSM 中建立，字首為 AWS TNB 網路執行個體 ID。這可防止在使用相同 NSD 範本執行個體化和升級多個執行個體時，參數值遭到覆寫。

語法

```
tosca.nodes.AWS.Store.SSMParameters
  properties:
    parameters:
      name: String
      value: String
    tags: List
```

Properties

Parameters

`name`

ssm 屬性的名稱。使用下列格式：`^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`

每個參數的名稱必須少於 256 個字元。

必要：是

類型：字串

`value`

ssm 屬性的值。請使用下列其中一個格式：

- 對於沒有參考的值： `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`
- 對於靜態參考： `^\$\{[a-zA-Z0-9]+\.(properties|capabilities|requirements)(\.([a-zA-Z0-9\-_]+))+\}$`
- 對於動態參考： `^\$\{[a-zA-Z0-9]+\.(name|id|arn)\}$`

每個參數的值必須小於 4 KB。

必要：是

類型：字串

`tags`

您可以連接到 SSM 屬性的標籤。

必要：否

類型：清單

範例

```
SampleSSM
    type: tosca.nodes.AWS.Store.SSMParameters
    properties:
        parameters:
            - name: "Name1"
              value: "Value1"
            - name: "EKS_VERSION"
              value: "${SampleEKS.properties.version}"
            - name: "VPC_ID"
              value: "${SampleVPC.id}
            - name: "REGION"
```

```
                value: "${AWS::Region}
        tags:
            - "tagKey=tagValue"
```

常見節點

定義 NSD 和 VNFD 的節點。

- AWS.HookDefinition.Bash

AWS.HookDefinition.Bash

在 中定義 an AWS HookDefinitionbash。

語法

```
tosca.nodes.AWS.HookDefinition.Bash:
  properties:
    implementation: String
    environment_variables: List
    execution_role: String
```

屬性

`implementation`

勾點定義的相對路徑。格式必須為： `./hooks/script_name.sh`

必要：是

類型：字串

`environment_variables`

hook bash 指令碼的環境變數。使用下列格式：**envName**=**envValue**搭配下列 regex 模式：

- 對於沒有參考的值： `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`
- 對於靜態參考： `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=\$\{[a-zA-Z0-9]+\.(properties|capabilities|requirements)(\.([a-zA-Z0-9\-_]+))+\}$`

- 對於動態參考： `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=\$\{[a-zA-Z0-9]+\.(name|id|arn)\}$`

請確定該**envName**=**envValue**值符合下列條件：

- 請勿使用空格。
- 從字母 (A-Z 或 a-z) 或數字 (0-9) **envName**開始。
- 請勿使用下列 AWS TNB 預留關鍵字 （ 不區分大小寫 ） 啟動環境變數名稱：
  - CODEBUILD
  - TNB
  - 首頁
  - AWS
- 您可以使用任意數量的字母 (A-Z 或 a-z)、數字 (0-9) 和特殊字元，**envName**以及 _ -和 **envValue**。
- 每個環境變數 （ 每個 **envName=envValue**) 必須少於 128 個字元。

範例：`A123-45xYz=Example_789`

必要：否

類型：清單

`execution_role`

勾點執行的角色。

必要：是

類型：字串

範例

```
SampleHookScript:
  type: tosca.nodes.AWS.HookDefinition.Bash
  properties:
    implementation: "./hooks/myhook.sh"
    environment_variables:
      - "variable01=value01"
      - "variable02=value02"
    execution_role: "arn:aws:iam::${AWS::TNB::AccountId}:role/SampleHookPermission"
```

AWS TNB 的安全性

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構是為了滿足最安全敏感組織的需求而建置。

安全性是 AWS 與您之間的共同責任。共同責任模式將其描述為雲端的安全性，和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 中執行 AWS 服務的基礎設施 AWS 雲端。 AWS 也為您提供可安全使用的服務。在AWS 合規計劃中，第三方稽核人員會定期測試和驗證我們安全的有效性。若要了解適用於 AWS Telco Network Builder 的合規計劃，請參閱AWS 合規計劃的 服務範圍。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您了解如何在使用 AWS TNB 時套用共同責任模型。下列主題說明如何設定 AWS TNB 以符合您的安全與合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 AWS TNB 資源。

目錄

AWS TNB 中的資料保護

AWS 共同責任模型適用於 Telco Network Builder AWS 中的資料保護。如此模型所述， AWS 負責保護執行所有 的全域基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。 如需歐洲資料保護的詳細資訊，請參閱一般資料保護規則 (GDPR) 中心。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 CloudTrail 追蹤。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-3。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 AWS TNB 或使用主控台、API AWS CLI或其他 AWS 服務 AWS SDKs 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

標籤處理

當您關閉 AWS 帳戶時， AWS TNB 會將您的資料標記為刪除，並將其從任何使用中移除。如果您在 90 天內重新啟用 AWS 帳戶， AWS TNB 會還原您的資料。120 天後， AWS TNB 會永久刪除您的資料。 AWS TNB 也會終止您的網路，並刪除您的函數套件和網路套件。

靜態加密

AWS TNB 一律會加密存放在服務中的所有靜態資料，而不需要任何額外的組態。此加密會透過 自動執行 AWS Key Management Service。

傳輸中加密

AWS TNB 使用 Transport Layer Security (TLS) 1.2 保護傳輸中的所有資料。

您有責任加密模擬代理程式與其用戶端之間的資料。

網際網路流量隱私權

AWS TNB 運算資源位於所有客戶共用的虛擬私有雲端 (VPC) 中。所有內部 AWS TNB 流量都會保留在 AWS 網路中，而不會周遊網際網路。模擬代理程式與其用戶端之間的連線會透過網際網路路由。

AWS TNB 的身分和存取管理

AWS Identity and Access Management (IAM) 是 AWS 服務 ，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可控制誰可以進行身分驗證 （ 登入 ） 和授權 （ 具有許可 ） 來使用 AWS TNB 資源。IAM 是您可以免費使用 AWS 服務 的 。

目錄

目標對象

使用方式 AWS Identity and Access Management (IAM) 會根據您的角色而有所不同：

- 服務使用者 — 若無法存取某些功能，請向管理員申請所需許可 (請參閱 對 AWS Telco Network Builder 身分和存取進行故障診斷)
- 服務管理員 — 負責設定使用者存取權並提交相關許可請求 (請參閱 AWS TNB 如何與 IAM 搭配使用)
- IAM 管理員 — 撰寫政策以管理存取控制 (請參閱 Telco Network Builder AWS 的身分型政策範例)

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色身分進行身分驗證。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的如何登入您的 AWS 帳戶。

對於程式設計存取， AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的 API 請求的AWS 第 4 版簽署程序。

AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分具有對所有 AWS 服務 和資源的完整存取權。強烈建議不要使用根使用者來執行日常任務。有關需要根使用者憑證的任務，請參閱《IAM 使用者指南》中的需要根使用者憑證的任務。

聯合身分

最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 服務 使用臨時憑證存取 。

聯合身分是您企業目錄、Web 身分提供者的使用者，或是 AWS 服務 使用身分來源的登入資料 Directory Service 存取的使用者。聯合身分會擔任角色，而該角色會提供臨時憑證。

若需集中化管理存取權限，建議使用 AWS IAM Identity Center。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

IAM 使用者和群組

IAM 使用者https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html是一種身分具備單人或應用程式的特定許可權。建議以臨時憑證取代具備長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 。

IAM 群組會指定 IAM 使用者集合，使管理大量使用者的許可權更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 使用者的使用案例。

IAM 角色

IAM 角色https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html的身分具有特定許可權，其可以提供臨時憑證。您可以透過從使用者切換到 IAM 角色（主控台）或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的擔任角色的方法。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權與跨服務存取，以及在 Amazon EC2 執行的應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策定義與身分或資源相關聯的許可。當委託人提出請求時 AWS ， 會評估這些政策。大多數政策會以 JSON 文件 AWS 形式存放在 中。如需進一步了解 JSON 政策文件，請參閱《IAM 使用者指南》中的 JSON 政策概觀。

管理員會使用政策，透過定義哪些主體可在哪些條件下對哪些資源執行動作，以指定可存取的範圍。

預設情況下，使用者和角色沒有許可。IAM 管理員會建立 IAM 政策並將其新增至角色，供使用者後續擔任。IAM 政策定義動作的許可，無論採用何種方式執行。

身分型政策

身分型政策是附加至身分 (使用者、使用者群組或角色) 的 JSON 許可政策文件。這類政策控制身分可對哪些資源執行哪些動作，以及適用的條件。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

身分型政策可分為內嵌政策 (直接內嵌於單一身分) 與受管政策 (可附加至多個身分的獨立政策)。如需了解如何在受管政策及內嵌政策之間做選擇，請參閱《IAM 使用者指南》中的在受管政策與內嵌政策之間選擇。

資源型政策

資源型政策是附加到資源的 JSON 政策文件。範例包括 IAM 角色信任政策與 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。您必須在資源型政策中指定主體。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

其他政策類型

AWS 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

- 許可界限 — 設定身分型政策可授與 IAM 實體的最大許可。如需詳細資訊，請參閱《 IAM 使用者指南》中的 IAM 實體許可界限。
- 服務控制政策 (SCP) — 為 AWS Organizations中的組織或組織單位指定最大許可。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的服務控制政策。
- 資源控制政策 (RCP) — 設定您帳戶中資源可用許可的上限。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的資源控制政策 (RCP)。
- 工作階段政策 — 在以程式設計方式為角色或聯合身分使用者建立臨時工作階段時，以參數形式傳遞的進階政策。如需詳細資訊，請參《IAM 使用者指南》中的工作階段政策。

多種政策類型

當多種類型的政策適用於請求時，產生的許可會更複雜而無法理解。若要了解如何 AWS 在涉及多個政策類型時決定是否允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

AWS TNB 如何與 IAM 搭配使用

在您使用 IAM 管理對 AWS TNB 的存取之前，請先了解哪些 IAM 功能可與 AWS TNB 搭配使用。

您可以搭配 Telco Network Builder AWS 使用的 IAM 功能

IAM 功能	AWS TNB 支援
身分型政策	是
資源型政策	否
政策動作	是
政策資源	是
政策條件索引鍵	是
ACL	否
ABAC (政策中的標籤)	是
臨時憑證	是
主體許可	是
服務角色	否
服務連結角色	否

若要全面了解 AWS TNB 和其他 AWS 服務如何與大多數 IAM 功能搭配使用，請參閱《IAM 使用者指南》中的AWS 與 IAM 搭配使用的 服務。

AWS TNB 的身分型政策

支援身分型政策：是

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。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

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。如要了解您在 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。

AWS TNB 的身分型政策範例

若要檢視 AWS TNB 身分型政策的範例，請參閱 Telco Network Builder AWS 的身分型政策範例。

AWS TNB 中的資源型政策

支援資源型政策：否

資源型政策是附加到資源的 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。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

如需啟用跨帳戶存取權，您可以在其他帳戶內指定所有帳戶或 IAM 實體作為資源型政策的主體。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

AWS TNB 的政策動作

支援政策動作：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。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。

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。政策會使用動作來授予執行相關聯動作的許可。

若要查看 AWS TNB 動作清單，請參閱服務授權參考中的 AWS Telco Network Builder 定義的動作。

AWS TNB 中的政策動作在動作之前使用下列字首：

```
tnb
```

如需在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [
```

```
      "tnb:CreateSolFunctionPackage",
      "tnb:DeleteSolFunctionPackage"
]
```

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 `List` 文字的所有動作，請包含以下動作：

```
"Action": "tnb:List*"
```

若要檢視 AWS TNB 身分型政策的範例，請參閱 Telco Network Builder AWS 的身分型政策範例。

AWS TNB 的政策資源

支援政策資源：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。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。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。若動作不支援資源層級許可，使用萬用字元 (*) 表示該陳述式適用於所有資源。

```
"Resource": "*"
```

若要查看 AWS TNB 資源類型及其 ARNs的清單，請參閱服務授權參考中的 AWS Telco Network Builder 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 Telco Network Builder AWS 定義的動作。

若要檢視 AWS TNB 身分型政策的範例，請參閱 Telco Network Builder AWS 的身分型政策範例。

AWS TNB 的政策條件索引鍵

支援服務特定政策條件金鑰：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。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。

`Condition` 元素會根據定義的條件，指定陳述式的執行時機。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。若要查看所有 AWS 全域條件索引鍵，請參閱《IAM 使用者指南》中的AWS 全域條件內容索引鍵。

若要查看 AWS TNB 條件索引鍵的清單，請參閱《服務授權參考》中的 AWS Telco Network Builder 的條件索引鍵。若要了解您可以使用條件金鑰的動作和資源，請參閱 Telco Network Builder AWS 定義的動作。

若要檢視 AWS TNB 身分型政策的範例，請參閱 Telco Network Builder AWS 的身分型政策範例。

AWS TNB ACLs

支援 ACL：否

存取控制清單 (ACL) 可控制哪些主體 (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。

ABAC 與 AWS TNB

支援 ABAC (政策中的標籤)：是

屬性型存取控制 (ABAC) 是一種授權策略，依據稱為標籤的屬性來定義許可。您可以將標籤連接至 IAM 實體 AWS 和資源，然後設計 ABAC 政策，以便在委託人的標籤符合資源上的標籤時允許操作。

如需根據標籤控制存取，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。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的使用 ABAC 授權定義許可。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的使用屬性型存取控制 (ABAC)。

搭配 AWS TNB 使用臨時登入資料

支援臨時憑證：是

臨時登入資料提供 AWS 資源的短期存取權，當您使用聯合身分或切換角色時會自動建立。 AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的臨時安全憑證與可與 IAM 搭配運作的AWS 服務。

AWS TNB 的跨服務主體許可

支援轉寄存取工作階段 (FAS)：是

轉送存取工作階段 (FAS) 使用呼叫 的委託人許可 AWS 服務，結合 AWS 服務 請求向下游服務提出請求。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。

AWS TNB 的服務角色

支援服務角色：否

服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務。

AWS TNB 的服務連結角色

支援服務連結角色：否

服務連結角色是連結至 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

Telco Network Builder AWS 的身分型政策範例

根據預設，使用者和角色沒有建立或修改 AWS TNB 資源的許可。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。

如需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》中的建立 IAM 政策 (主控台)。

如需 AWS TNB 定義的動作和資源類型的詳細資訊，包括每種資源類型的 ARNs 格式，請參閱《服務授權參考》中的 AWS Telco Network Builder 的動作、資源和條件索引鍵。

目錄

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 AWS TNB 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 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。我們建議您定

義特定於使用案例 AWS 的客戶受管政策，進一步減少許可。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。

- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。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。如果透過特定 例如 使用服務動作 AWS 服務，您也可以使用條件來授予其存取權 CloudFormation。如需詳細資訊，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，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM Access Analyzer 驗證政策。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的透過 MFA 的安全 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 AWS TNB 主控台

若要存取 AWS Telco Network Builder 主控台，您必須擁有一組最低許可。這些許可必須允許您列出和檢視 中 AWS TNB 資源的詳細資訊 AWS 帳戶。如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (使用者或角色) 而言，主控台就無法如預期運作。

對於僅呼叫 AWS CLI 或 AWS API 的使用者，您不需要允許最低主控台許可。反之，只需允許存取符合他們嘗試執行之 API 操作的動作就可以了。

服務角色政策範例

身為管理員，您擁有和管理 AWS TNB 建立的資源，如環境和服務範本所定義。您必須將 IAM 服務角色連接到您的帳戶，以允許 AWS TNB 為您的網路生命週期管理建立資源。

IAM 服務角色可讓 AWS TNB 代表您呼叫 資源，以執行個體化和管理網路。如果您指定服務角色，AWS TNB 會使用該角色的登入資料。

您使用 IAM 服務建立服務角色及其許可政策。如需建立服務角色的詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務。

AWS TNB 服務角色

身為平台團隊的成員，身為管理員，您可以建立 AWS TNB 服務角色並將其提供給 AWS TNB。此角色允許 AWS TNB 呼叫其他 服務，例如 Amazon Elastic Kubernetes Service CloudFormation ，並為您的網路佈建必要的基礎設施，並佈建 NSD 中定義的網路函數。

建議您針對 AWS TNB 服務角色使用下列 IAM 角色和信任政策。縮小此政策的許可範圍時，請記住，AWS TNB 可能會失敗，導致拒絕存取錯誤導致從您的政策中剔除的資源。

下列程式碼顯示 AWS TNB 服務角色政策：

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "sts:GetCallerIdentity"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "AssumeRole"
        },
        {
            "Action": [
                "tnb:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "TNBPolicy"
        },
        {
            "Action": [
                "iam:AddRoleToInstanceProfile",
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:TagInstanceProfile",
                "iam:UntagInstanceProfile"
            ],
            "Resource": "*",
```

```
            "Effect": "Allow",
            "Sid": "IAMPolicy"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "eks.amazonaws.com",
                        "eks-nodegroup.amazonaws.com"
                    ]
                }
            },
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "TNBAccessSLRPermissions"
        },
        {
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:CreateOrUpdateTags",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:DeleteTags",
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeScalingActivities",
                "autoscaling:DescribeTags",
                "autoscaling:UpdateAutoScalingGroup",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateLaunchTemplate",
                "ec2:CreateLaunchTemplateVersion",
                "ec2:CreateSecurityGroup",
                "ec2:DeleteLaunchTemplateVersions",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:DeleteLaunchTemplate",
                "ec2:DeleteSecurityGroup",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeTags",
                "ec2:GetLaunchTemplateData",
                "ec2:RevokeSecurityGroupEgress",
```

```
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RunInstances",
                "ec2:AssociateRouteTable",
                "ec2:AttachInternetGateway",
                "ec2:CreateInternetGateway",
                "ec2:CreateNetworkInterface",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:CreateSubnet",
                "ec2:CreateTags",
                "ec2:CreateVpc",
                "ec2:DeleteInternetGateway",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteRoute",
                "ec2:DeleteRouteTable",
                "ec2:DeleteSubnet",
                "ec2:DeleteTags",
                "ec2:DeleteVpc",
                "ec2:DetachNetworkInterface",
                "ec2:DescribeInstances",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroupRules",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DetachInternetGateway",
                "ec2:DisassociateRouteTable",
                "ec2:ModifySecurityGroupRules",
                "ec2:ModifySubnetAttribute",
                "ec2:ModifyVpcAttribute",
                "ec2:AllocateAddress",
                "ec2:AssignIpv6Addresses",
                "ec2:AssociateAddress",
                "ec2:AssociateNatGatewayAddress",
                "ec2:AssociateVpcCidrBlock",
                "ec2:CreateEgressOnlyInternetGateway",
                "ec2:CreateNatGateway",
                "ec2:DeleteEgressOnlyInternetGateway",
                "ec2:DeleteNatGateway",
                "ec2:DescribeAddresses",
                "ec2:DescribeEgressOnlyInternetGateways",
                "ec2:DescribeNatGateways",
```

```
                "ec2:DisassociateAddress",
                "ec2:DisassociateNatGatewayAddress",
                "ec2:DisassociateVpcCidrBlock",
                "ec2:ReleaseAddress",
                "ec2:UnassignIpv6Addresses",
                "ec2:DescribeImages",
                "eks:CreateCluster",
                "eks:ListClusters",
                "eks:RegisterCluster",
                "eks:TagResource",
                "eks:DescribeAddonVersions",
                "events:DescribeRule",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "TNBAccessComputePerms"
        },
        {
            "Resource": "*",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Condition": {
                "StringEquals": {
                   "iam:PassedToService": [
                       "ec2.amazonaws.com",
                       "eks.amazonaws.com",
                       "eks-nodegroup.amazonaws.com",
                       "events.amazonaws.com",
                       "autoscaling.amazonaws.com",
                       "codebuild.amazonaws.com"
                   ]
                }
            }
        },
        {
            "Action": [
                "codebuild:BatchDeleteBuilds",
                "codebuild:BatchGetBuilds",
                "codebuild:CreateProject",
                "codebuild:DeleteProject",
```

```
                "codebuild:ListBuildsForProject",
                "codebuild:StartBuild",
                "codebuild:StopBuild",
                "events:DeleteRule",
                "events:PutRule",
                "events:PutTargets",
                "events:RemoveTargets",
                "s3:CreateBucket",
                "s3:GetBucketAcl",
                "s3:GetObject",
                "eks:DescribeNodegroup",
                "eks:DeleteNodegroup",
                "eks:AssociateIdentityProviderConfig",
                "eks:CreateNodegroup",
                "eks:DeleteCluster",
                "eks:DeregisterCluster",
                "eks:UpdateAddon",
                "eks:UpdateClusterVersion",
                "eks:UpdateNodegroupConfig",
                "eks:UpdateNodegroupVersion",
                "eks:DescribeUpdate",
                "eks:UntagResource",
                "eks:DescribeCluster",
                "eks:ListNodegroups",
                "eks:CreateAddon",
                "eks:DeleteAddon",
                "eks:DescribeAddon",
                "eks:DescribeAddonVersions",
                "s3:PutObject",
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStackResources",
                "cloudformation:DescribeStacks",
                "cloudformation:ListStackResources",
                "cloudformation:UpdateStack",
                "cloudformation:UpdateTerminationProtection",
                "ssm:PutParameter",
                "ssm:GetParameters",
                "ssm:GetParameter",
                "ssm:DeleteParameter",
                "ssm:AddTagsToResource",
                "ssm:ListTagsForResource",
                "ssm:RemoveTagsFromResource"
            ],
```

```
            "Resource": [
                "arn:aws:events:*:*:rule/tnb*",
                "arn:aws:codebuild:*:*:project/tnb*",
                "arn:aws:logs:*:*:log-group:/aws/tnb*",
                "arn:aws:s3:::tnb*",
                "arn:aws:eks:*:*:addon/tnb*/*/*",
                "arn:aws:eks:*:*:cluster/tnb*",
                "arn:aws:eks:*:*:nodegroup/tnb*/tnb*/*",
                "arn:aws:cloudformation:*:*:stack/tnb*",
                "arn:aws:ssm:*:*:parameter/tnb/*"
            ],
            "Effect": "Allow",
            "Sid": "TNBAccessInfraResourcePerms"
        },
        {
            "Sid": "CFNTemplatePerms",
            "Effect": "Allow",
            "Action": [
              "cloudformation:GetTemplateSummary"
            ],
            "Resource": "*"
         },
        {
            "Sid": "ImageAMISSMPerms",
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:*::parameter/aws/service/eks/optimized-ami/*",
                "arn:aws:ssm:*::parameter/aws/service/bottlerocket/*"
            ]
        },
        {
            "Action": [
                "tag:GetResources"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "TaggingPolicy"
        },
        {
            "Action": [
                "outposts:GetOutpost"
```

```
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "OutpostPolicy"
        }
    ]
}
```

下列程式碼顯示 AWS TNB 服務信任政策：

JSON

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

```
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "tnb.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

AWS Amazon EKS 叢集的 TNB 服務角色

當您在 NSD 中建立 Amazon EKS 資源時，請提供 `cluster_role` 屬性來指定要用來建立 Amazon EKS 叢集的角色。

下列範例顯示為 Amazon EKS 叢集政策建立 AWS TNB 服務角色的 AWS CloudFormation 範本。

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  TNBEKSClusterRole:
    Type: "AWS::IAM::Role"
    Properties:
      RoleName: "TNBEKSClusterRole"
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - eks.amazonaws.com
            Action:
              - "sts:AssumeRole"
      Path: /
      ManagedPolicyArns:
        - !Sub "arn:${AWS::Partition}:iam::aws:policy/AmazonEKSClusterPolicy"
```

如需使用 AWS CloudFormation 範本之 IAM 角色的詳細資訊，請參閱AWS CloudFormation 《使用者指南》中的下列章節：

- AWS::IAM::Role
- 選取堆疊範本

AWS Amazon EKS 節點群組的 TNB 服務角色

當您在 NSD 中建立 Amazon EKS 節點群組資源時，請提供 node_role 屬性來指定要用來建立 Amazon EKS 節點群組的角色。

下列範例顯示為 Amazon EKS 節點群組政策建立 AWS TNB 服務角色的 CloudFormation 範本。

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  TNBEKSNodeRole:
    Type: "AWS::IAM::Role"
    Properties:
      RoleName: "TNBEKSNodeRole"
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action:
              - "sts:AssumeRole"
      Path: /
      ManagedPolicyArns:
        - !Sub "arn:${AWS::Partition}:iam::aws:policy/AmazonEKSWorkerNodePolicy"
        - !Sub "arn:${AWS::Partition}:iam::aws:policy/AmazonEKS_CNI_Policy"
        - !Sub "arn:${AWS::Partition}:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly"
        - !Sub "arn:${AWS::Partition}:iam::aws:policy/service-role/
AmazonEBSCSIDriverPolicy"
      Policies:
        - PolicyName: EKSNodeRoleInlinePolicy
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
                Action:
                  - "logs:DescribeLogStreams"
                  - "logs:PutLogEvents"
                  - "logs:CreateLogGroup"
                  - "logs:CreateLogStream"
                Resource: "arn:aws:logs:*:*:log-group:/aws/tnb/tnb*"
        - PolicyName: EKSNodeRoleIpv6CNIPolicy
          PolicyDocument:
```

```
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
                Action:
                  - "ec2:AssignIpv6Addresses"
                Resource: "arn:aws:ec2:*:*:network-interface/*"
```

如需使用 AWS CloudFormation 範本之 IAM 角色的詳細資訊，請參閱AWS CloudFormation 《使用者指南》中的下列章節：

- AWS::IAM::Role
- 選取堆疊範本

AWS Multus 的 TNB 服務角色

當您在 NSD 中建立 Amazon EKS 資源，並想要在部署範本中管理 Multus 時，必須提供 `multus_role` 屬性來指定要用於管理 Multus 的角色。

下列範例顯示為 Multus 政策建立 AWS TNB 服務角色的 CloudFormation 範本。

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  TNBMultusRole:
    Type: "AWS::IAM::Role"
    Properties:
      RoleName: "TNBMultusRole"
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - events.amazonaws.com
            Action:
              - "sts:AssumeRole"
          - Effect: Allow
            Principal:
              Service:
                - codebuild.amazonaws.com
            Action:
              - "sts:AssumeRole"
      Path: /
```

```
      Policies:
        - PolicyName: MultusRoleInlinePolicy
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
                Action:
                  - "codebuild:StartBuild"
                  - "logs:DescribeLogStreams"
                  - "logs:PutLogEvents"
                  - "logs:CreateLogGroup"
                  - "logs:CreateLogStream"
                Resource:
                  - "arn:aws:codebuild:*:*:project/tnb*"
                  - "arn:aws:logs:*:*:log-group:/aws/tnb/*"
              - Effect: Allow
                Action:
                  - "ec2:CreateNetworkInterface"
                  - "ec2:ModifyNetworkInterfaceAttribute"
                  - "ec2:AttachNetworkInterface"
                  - "ec2:DeleteNetworkInterface"
                  - "ec2:CreateTags"
                  - "ec2:DetachNetworkInterface"
                Resource: "*"
```

如需使用 AWS CloudFormation 範本之 IAM 角色的詳細資訊，請參閱AWS CloudFormation 《使用者指南》中的下列章節：

- AWS::IAM::Role
- 選取堆疊範本

AWS 生命週期掛鉤政策的 TNB 服務角色

當您的 NSD 或網路函數套件使用生命週期掛鉤時，您需要一個服務角色，以允許您建立環境來執行生命週期掛鉤。

> **Note**
>
> 您的生命週期掛鉤政策應該根據您的生命週期掛鉤嘗試執行的操作而定。

下列範例顯示為生命週期掛鉤政策建立 AWS TNB 服務角色的 CloudFormation 範本。

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  TNBHookRole:
    Type: "AWS::IAM::Role"
    Properties:
      RoleName: "TNBHookRole"
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - codebuild.amazonaws.com
            Action:
              - "sts:AssumeRole"
      Path: /
      ManagedPolicyArns:
        - !Sub "arn:${AWS::Partition}:iam::aws:policy/AdministratorAccess"
```

如需使用 AWS CloudFormation 範本之 IAM 角色的詳細資訊，請參閱AWS CloudFormation 《使用者指南》中的下列章節：

- AWS::IAM::Role
- 選取堆疊範本

允許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 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在主控台或使用 或 AWS CLI AWS API 以程式設計方式完成此動作的許可。

```
{
    "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": "*"
        }
    ]
}
```

對 AWS Telco Network Builder 身分和存取進行故障診斷

使用以下資訊來協助您診斷和修正使用 AWS TNB 和 IAM 時可能遇到的常見問題。

問題

- 我無權在 AWS TNB 中執行動作
- 我未獲得執行 iam:PassRole 的授權
- 我想要允許 以外的人員 AWS 帳戶 存取我的 AWS TNB 資源

我無權在 AWS TNB 中執行動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 tnb:*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 tnb:GetWidget on resource: my-example-widget
```

在此情況下，必須更新 Mateo 政策，允許他使用 `tnb:GetWidget` 動作存取 *my-example-widget* 資源。

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。

我未獲得執行 iam:PassRole 的授權

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，以允許您將角色傳遞給 AWS TNB。

有些 AWS 服務 可讓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 的 IAM `marymajor` 使用者嘗試使用主控台在 AWS TNB 中執行動作時，會發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞給服務的許可。

```
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 以外的人員 AWS 帳戶 存取我的 AWS TNB 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。

如需進一步了解，請參閱以下內容：

- 若要了解 AWS TNB 是否支援這些功能，請參閱 AWS TNB 如何與 IAM 搭配使用。
- 若要了解如何 AWS 帳戶 在您擁有的 資源間提供存取權，請參閱《IAM 使用者指南》中的在您擁有 AWS 帳戶 的另一個 IAM 使用者中提供存取權。
- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《IAM 使用者指南》中的將存取權提供給第三方 AWS 帳戶 擁有。

- 如需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)。
- 如需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 中的跨帳戶資源存取。

AWS TNB 的合規驗證

若要了解 是否 AWS 服務 在特定合規計劃範圍內，請參閱AWS 服務 合規計劃範圍內然後選擇您感興趣的合規計劃。如需一般資訊，請參閱AWS 合規計劃。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱下載報告 in AWS Artifact

您使用 時的合規責任 AWS 服務 取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。如需使用 時合規責任的詳細資訊 AWS 服務，請參閱 AWS 安全文件。

AWS TNB 中的彈性

AWS 全球基礎設施是以 AWS 區域 和 可用區域為基礎建置。 AWS 區域 提供多個實體隔離和隔離的可用區域，這些可用區域與低延遲、高輸送量和高備援聯網連接。透過可用區域，您可以設計與操作的應用程式和資料庫，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、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域 和可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

AWS TNB 會在您選擇的 AWS 區域中的虛擬私有雲端 (VPC) 中執行 Network Service on EKS 叢集。

AWS TNB 中的基礎設施安全性

身為受管服務， AWS Telco Network Builder 受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及如何 AWS 保護基礎設施的資訊，請參閱AWS 雲端安全。若要使用基礎設施安全的最佳實務來設計您的 AWS 環境，請參閱安全支柱 AWS Well-Architected Framework 中的基礎設施保護。

您可以使用 AWS 發佈的 API 呼叫，透過網路存取 AWS TNB。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。

以下是一些共同責任的範例：

- AWS 負責保護支援 AWS TNB 的元件，包括：
  - 運算執行個體 （ 也稱為工作者)
  - 內部資料庫
  - 內部元件之間的網路通訊
  - AWS TNB 應用程式程式設計界面 (API)
  - AWS 軟體開發套件 (SDK)
- 您有責任保護對 AWS 資源和工作負載元件的存取，包括 （ 但不限於 ）：
  - IAM 使用者、群組、角色和政策
  - 用於存放 AWS TNB 資料的 S3 儲存貯體
  - 您用來支援您透過 AWS TNB 佈建之網路服務的其他 AWS 服務 和資源
  - 您的應用程式程式碼
  - 您透過 AWS TNB 佈建的網路服務與其用戶端之間的連線

> **Important**
>
> 您負責實作災難復原計劃，以有效復原您透過 AWS TNB 佈建的網路服務。

網路連線安全模型

您透過 AWS TNB 佈建的網路服務，會在您所選 AWS 區域中虛擬私有雲端 (VPC) 內的運算執行個體上執行。VPC 是 AWS 雲端中的虛擬網路，可依工作負載或組織實體隔離基礎設施。VPCs內的運算執行個體之間的通訊會保留在 AWS 網路中，而且不會透過網際網路傳輸。有些內部服務通訊會跨網際網路並加密。透過 AWS TNB 為在相同區域中執行的所有客戶佈建的網路服務共用相同的 VPC。透過 AWS TNB 為不同客戶佈建的網路服務會使用相同 VPC 中的個別運算執行個體。

AWS TNB 中的網路服務用戶端與網路服務之間的通訊周遊網際網路。 AWS TNB 不會管理這些連線。保護用戶端連線是您的責任。

您透過 AWS 管理主控台、 AWS Command Line Interface (AWS CLI) 和 SDK 與 AWS TNB 的連線會加密。 AWS SDKs

IMDS 版本

AWS TNB 支援利用執行個體中繼資料服務第 2 版 (IMDSv2) 的執行個體，這是一種工作階段導向的方法。IMDSv2 的安全性高於 IMDSV1。如需詳細資訊，請參閱為 Amazon EC2 執行個體中繼資料服務新增對開放防火牆、反向代理和 SSRF 漏洞的深度防禦。

啟動執行個體時，您必須使用 IMDSv2。如需 IMDSv2 的詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用 IMDSv22。 Amazon EC2

監控 AWS TNB

監控是維護 AWS TNB 和其他 AWS 解決方案的可靠性、可用性和效能的重要部分。 AWS 提供監 AWS CloudTrail 看 AWS TNB、在發生錯誤時回報，以及適時採取自動動作。

使用 CloudTrail 擷取對 AWS APIs發出的呼叫詳細資訊。您可以將這些呼叫儲存為 Amazon S3 中的日誌檔案。您可以使用這些 CloudTrail 日誌來判斷進行了哪些呼叫、呼叫的來源 IP 地址、進行呼叫的人員以及進行呼叫的時間等資訊。

CloudTrail 日誌包含對 AWS TNB API 動作呼叫的相關資訊。它們也包含從 Amazon EC2 和 Amazon EBS 等服務呼叫 API 動作的資訊。

使用 AWS 記錄 Telco Network Builder API 呼叫 AWS CloudTrail

AWS Telco Network Builder 已與 整合AWS CloudTrail，此服務可提供使用者、角色或 所採取動作的記錄 AWS 服務。CloudTrail 會將 AWS TNB 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自 AWS TNB 主控台的呼叫，以及對 AWS TNB API 操作的程式碼呼叫。使用 CloudTrail 收集的資訊，您可以判斷對 AWS TNB 提出的請求、提出請求的 IP 地址、提出請求的時間，以及其他詳細資訊。

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是使用根使用者還是使用者憑證提出。
- 請求是否代表 IAM Identity Center 使用者提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 該請求是否由另一項 AWS 服務服務提出。

當您建立帳戶 AWS 帳戶 時CloudTrail 會在 中處於作用中狀態，而且您會自動存取 CloudTrail 事件歷史記錄。CloudTrail 事件歷史記錄為 AWS 區域中過去 90 天記錄的管理事件，提供可檢視、可搜尋、可下載且不可變的記錄。如需詳細資訊，請參閱「AWS CloudTrail 使用者指南」中的使用 CloudTrail 事件歷史記錄。檢視事件歷史記錄不會產生 CloudTrail 費用。

如需 AWS 帳戶 過去 90 天內持續記錄的事件，請建立線索或 CloudTrail Lake 事件資料存放區。

CloudTrail 追蹤

線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。使用 建立的所有線索 AWS 管理主控台 都是多區域。您可以使用 AWS CLI建立單一或多區域追蹤。建議您建立多區域追蹤，因為您

擷取 AWS 區域 帳戶中所有 的活動。如果您建立單一區域追蹤，您只能檢視追蹤 AWS 區域中記錄的事件。如需追蹤的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的為您的 AWS 帳戶建立追蹤和為組織建立追蹤。

您可以透過建立追蹤，免費將持續管理事件的一個複本從 CloudTrail 傳遞至您的 Amazon S3 儲存貯體，但這樣做會產生 Amazon S3 儲存費用。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。如需 Amazon S3 定價的相關資訊，請參閱 Amazon S3 定價。

CloudTrail Lake 事件資料存放區

CloudTrail Lake 讓您能夠對事件執行 SQL 型查詢。CloudTrail Lake 會將分列式 JSON 格式的現有事件轉換為 Apache ORC 格式。ORC 是一種單欄式儲存格式，針對快速擷取資料進行了最佳化。系統會將事件彙總到事件資料存放區中，事件資料存放區是事件的不可變集合，其依據為您透過套用進階事件選取器選取的條件。套用於事件資料存放區的選取器控制哪些事件持續存在並可供您查詢。如需 CloudTrail Lake 的詳細資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 AWS CloudTrail Lake。

CloudTrail Lake 事件資料存放區和查詢會產生費用。建立事件資料存放區時，您可以選擇要用於事件資料存放區的定價選項。此定價選項將決定擷取和儲存事件的成本，以及事件資料存放區的預設和最長保留期。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。

AWS TNB 事件範例

一個事件代表任何來源提出的單一請求，並包含請求 API 操作的相關資訊、操作的日期和時間、請求參數等。CloudTrail 日誌檔案不是公有 API 呼叫的已排序堆疊追蹤，因此事件不會以任何特定順序顯示。

以下範例顯示的 CloudTrail 事件會示範 `CreateSolFunctionPackage` 操作。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:example",
        "arn": "arn:aws:sts::111222333444:assumed-role/example/user",
        "accountId": "111222333444",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
```

```
                "arn": "arn:aws:iam::111222333444:role/example",
                "accountId": "111222333444",
                "userName": "example"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-02-02T01:42:39Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-02-02T01:43:17Z",
    "eventSource": "tnb.amazonaws.com",
    "eventName": "CreateSolFunctionPackage",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "XXX.XXX.XXX.XXX",
    "userAgent": "userAgent",
    "requestParameters": null,
    "responseElements": {
        "vnfPkgArn": "arn:aws:tnb:us-east-1:111222333444:function-package/
fp-12345678abcEXAMPLE",
        "id": "fp-12345678abcEXAMPLE",
        "operationalState": "DISABLED",
        "usageState": "NOT_IN_USE",
        "onboardingState": "CREATED"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111222333444",
    "eventCategory": "Management"
}
```


如需有關 CloudTrail 記錄內容的資訊，請參閱《AWS CloudTrail 使用者指南》中的 CloudTrail record contents。

AWS TNB 部署任務

了解部署任務，以有效監控部署並更快地採取行動。

下表列出 AWS TNB 部署任務：

2024 年 3 月 7 日之前開始的部署任務名稱	2024 年 3 月 7 日當天和之後開始部署的任務名稱	Task description (任務描述)
AppInstallation	ClusterPluginInstall	在 Amazon EKS 叢集上安裝 Multus 外掛程式。
AppUpdate	名稱沒有變更	更新已安裝在網路執行個體中的網路函數。
-	ClusterPluginUninstall	在 Amazon EKS 叢集上解除安裝外掛程式。
ClusterStorageClassConfiguration	名稱沒有變更	在 Amazon EKS 叢集上設定儲存體方案 (CSI 驅動程式）。
FunctionDeletion	名稱沒有變更	從 AWS TNB 資源刪除網路函數。
FunctionInstantiation	FunctionInstall	使用 HELM 部署網路函數。
FunctionUninstallation	FunctionUninstall	從 Amazon EKS 叢集解除安裝網路函數。
HookExecution	名稱沒有變更	執行 NSD 中定義的生命週期關聯。
InfrastructureCancellation	名稱沒有變更	取消網路服務。
InfrastructureInstantation	名稱沒有變更	代表使用者佈建 AWS 資源。
InfrastructureTermination	名稱沒有變更	取消佈建透過 AWS TNB 叫用 AWS 的資源。
-	InfrastructureUpdate	更新代表使用者佈建 AWS 的資源。
InventoryDeregistration	名稱沒有變更	從 AWS TNB 取消註冊 AWS 資源。
-	InventoryRegistration	在 AWS TNB 中註冊 AWS 資源。
KubernetesClusterConfiguration	ClusterConfiguration	設定 Kubernetes 叢集，並將其他 IAM 角色新增至 NSD 中定義的 Amazon EKS AuthMap。

2024 年 3 月 7 日之前開始的部署任務名稱	2024 年 3 月 7 日當天和之後開始部署的任務名稱	Task description (任務描述)
NetworkServiceFinalization	名稱沒有變更	完成網路服務並提供成功或失敗狀態更新。
NetworkServiceInstantiation	名稱沒有變更	初始化網路服務。
SelfManagedNodesConfiguration	名稱沒有變更	使用 Amazon EKS 和 Kubernetes 控制平面引導自我管理節點。
-	ValidateNetworkServiceUpdate	在更新網路執行個體之前執行驗證。

AWS TNB 的服務配額

服務配額也稱為限制，是 AWS 您的帳戶的服務資源或操作數量上限。如需詳細資訊，請參閱《AWS 》中的 Amazon Web Services 一般參考服務配額。

以下是 AWS TNB 的服務配額。

Name	預設	可調整	說明
並行持續的網路服務操作	每個受支援的區域：40	是	一個區域中並行進行中網路服務操作的數量上限。
函數套件	每個受支援的區域：200	是	一個區域中的函數套件數目上限。
網路套件	每個受支援的區域：40	是	一個區域中的網路套件數量上限。
網路服務執行個體	每個支援的區域：800	是	一個區域中網路服務執行個體的數量上限。

AWS TNB 使用者指南的文件歷史記錄

下表說明 AWS TNB 的文件版本。

變更	描述	日期
Amazon EKS 節點群組網路組態的更新	新增和刪除子網路和安全群組。從網路新增、修改和刪除 ENIs。如需詳細資訊，請參閱您可以更新的參數。	2025 年 9 月 10 日
在現有叢集中新增和刪除 Amazon EKS 節點群組	AWS TNB 現在支援新增節點群組，以及從 Amazon EKS 叢集移除現有的節點群組。如需詳細資訊，請參閱您可以更新的參數。	2025 年 6 月 4 日
根磁碟區大小	您可以透過 .AWS Compute.EKSManagedNode 和 .Compute.AWS EKSSelfManagedNode TOSCA 節點中的 `root_volume_size` 欄位，指定 Amazon EKS 工作者節點的基礎 Amazon EBS 根磁碟區大小。	2025 年 5 月 19 日
參考指令碼中的資源	您可以參考 AWS TNB 建立的資源，在 Lifecycle Hook 指令碼和使用者資料指令碼中進行設定。	2025 年 5 月 2 日
Amazon EKS 節點和受管節點群組現在支援 Kubernetes 1.32 版。	AWS TNB 支援適用於 AWS.Compute.EKS 和 AWS.Compute.EKSManagedNode 的 Kubernetes 1.32 版。	2025 年 4 月 24 日

Amazon EKS 節點和受管節點群組不再支援 Kubernetes 1.24 版	AWS TNB 不再支援 .AWS Compute.EKS 和 .AWS Compute.EKSManagedNode 的 Kubernetes 1.24 版。	2025 年 4 月 17 日
對 Amazon EKS 受管節點的 AL2023 AMI 支援	AWS TNB 支援 AWS.Compute.EKSManagedNode 的 AL2023 AMI 類型。	2025 年 4 月 17 日
Amazon EKS 節點和受管節點群組不再支援 Kubernetes 1.23 版	AWS TNB 不再支援 .AWS Compute.EKS 和 .AWS Compute.EKSManagedNode 的 Kubernetes 1.23 版。	2025 年 4 月 4 日
可以更新 AMI ID	您現在可以在 `UpdateSolNetworkService` API 呼叫期間更新 ami_id 欄位。	2025 年 3 月 31 日
Amazon EKS 節點和受管節點群組現在支援 Kubernetes 1.31 版。	AWS TNB 支援適用於 AWS.Compute.EKS 和 .AWS Compute.EKSManagedNode 的 Kubernetes 1.31 版。	2025 年 2 月 18 日
for AWS.Compute.EKSManagedNode 的 Kubernetes 版本	AWS TNB 支援 Kubernetes 版本 1.23 到 1.30 來建立 Amazon EKS 受管節點群組。	2025 年 1 月 28 日
叢集的 Kubernetes 版本	AWS TNB 現在支援 Kubernetes 1.30 版來建立 Amazon EKS 叢集。	2024 年 8 月 19 日

AWS TNB 支援管理網路生命週期的額外操作。	您可以使用新的網路套件和參數值來更新執行個體化或先前更新的網路執行個體。請參閱：
	• 生命週期操作
	• 更新網路執行個體
	• AWS TNB 服務角色範例：
	  • 新增這些 Amazon EKS 動作：eks:UpdateAddon 、eks:UpdateClusterVersion 、eks:UpdateNodegroupConfig 、eks:UpdateNodegroupVersion 、eks:DescribeUpdate
	  • 新增此 CloudFormation 動作： cloudformation:UpdateStack
	• 新的部署任務：InfrastructureUpdate 、InventoryRegistration 、ValidateNetworkServiceUpdate
	• API 更新：GetSolNetworkOperation、ListSolNetworkOperations 和 UpdateSolNetworkInstance
		2024 年 7 月 30 日

現有任務的新任務和新任務名稱	有可用的新任務。截至 2024 年 3 月 7 日，為了清楚起見，某些現有任務具有新的名稱。	2024 年 5 月 7 日
叢集的 Kubernetes 版本	AWS TNB 現在支援 Kubernetes 1.29 版來建立 Amazon EKS 叢集。	2024 年 4 月 10 日
支援網路界面 `security_groups`	您可以將安全群組連接到 AWS.Networking.ENI 節點。	2024 年 4 月 2 日
支援 Amazon EBS 根磁碟區加密	您可以為 Amazon EBS 根磁碟區啟用 Amazon EBS 加密。若要啟用，請在 AWS.Compute.EKSManagedNode 或 AWS.Compute.EKSSelfManagedNode 節點中新增屬性。	2024 年 4 月 2 日
節點的支援 `labels`	您可以在 AWS.Compute.EKSManagedNode 或 AWS.Compute.EKSSelfManagedNode 節點中將節點標籤連接至節點群組。	2024 年 3 月 19 日
支援網路界面 `source_dest_check`	您可以指出是否要透過 AWS.Networking.ENI 節點啟用或停用網路介面來源/目的地檢查。	2024 年 1 月 25 日
支援具有自訂使用者資料的 Amazon EC2 執行個體	您可以透過 AWS.Compute.UserData 節點使用自訂使用者資料啟動 Amazon EC2 執行個體。	2024 年 1 月 16 日
支援安全群組	AWS TNB 可讓您匯入安全群組 AWS 資源。	2024 年 1 月 8 日

更新 的描述 network_interfaces	當network_interfaces 屬性包含在 AWS.Compute.EKSManagedNode 或 AWS.Compute.EKSSelfManagedNode 節點中時，AWS TNB 會從可用multus_role 屬性或從 node_role 屬性取得與 ENIs 相關的許可。	2023 年 12 月 18 日
支援私有叢集	AWS TNB 現在支援私有叢集。若要指示私有叢集，請將 access 屬性設定為 PRIVATE。	2023 年 12 月 11 日
叢集的 Kubernetes 版本	AWS TNB 現在支援 Kubernetes 1.28 版來建立 Amazon EKS 叢集。	2023 年 12 月 11 日
AWS TNB 支援置放群組	新增 AWS.Compute.EKSManagedNode 和 AWS.Compute.EKSSelfManagedNode 節點定義的置放群組。	2023 年 12 月 11 日

AWS TNB 新增對 IPv6 的支援	AWS TNB 現在支援使用 IPv6 基礎設施建立網路執行個體。檢查適用於 IPv6 組態的節點 AWS.Networking.VPC、AWS.Networking.Subnet、AWS.Networking.InternetGateway、AWS.Networking.SecurityGroupIngressRule、AWS.Networking.SecurityGroupEgressRule 和 AWS.Compute.EKS。我們也為 NAT64 組態新增了節點 AWS.Networking.NATGateway 和 AWS.Networking.Route。我們已針對 IPv6 許可更新 AWS Amazon EKS 節點群組的 TNB 服務角色和 AWS TNB 服務角色。請參閱服務角色政策範例。	2023 年 11 月 16 日
新增對 AWS TNB 服務角色政策的許可	我們已將許可新增至 Amazon S3 的 AWS TNB 服務角色政策 CloudFormation ，並啟用基礎設施執行個體化。	2023 年 10 月 23 日
AWS 在更多區域中啟動的 TNB	AWS TNB 現已在亞太區域（首爾）、加拿大（中部）、歐洲（西班牙）、歐洲（斯德哥爾摩）和南美洲（聖保羅）區域提供。	2023 年 9 月 27 日
適用於 AWS.Compute.EKSSelfManagedNode 的標籤	AWS TNB 現在支援`AWS.Compute.EKSSelfManagedNode` 節點定義的標籤。	2023 年 8 月 22 日

AWS TNB 支援利用 IMDSv2 的執行個體	啟動執行個體時，您必須使用 IMDSv2。	2023 年 8 月 14 日
已更新 MultusRoleInlinePolicy 的許可	MultusRoleInlinePolicy 現在包含 `ec2:DeleteNetworkInterface` 許可。	2023 年 8 月 7 日
叢集的 Kubernetes 版本	AWS TNB 現在支援 Kubernetes 1.27 版來建立 Amazon EKS 叢集。	2023 年 7 月 25 日
AWS.Compute.EKS.AuthRole	AWS TNB 支援 AuthRole，可讓您將 IAM 角色新增至 Amazon EKS 叢集，`aws-authConfigMap` 讓使用者可以使用 IAM 角色存取 Amazon EKS 叢集。	2023 年 7 月 19 日
AWS TNB 支援安全群組。	已將 AWS.Networking.SecurityGroup、AWS.Networking.SecurityGroupEgressRule 和 AWS.Networking.SecurityGroupIngressRule 新增至 NSD 範本。	2023 年 7 月 18 日
叢集的 Kubernetes 版本	AWS TNB 支援 Kubernetes 版本 1.22 到 1.26 來建立 Amazon EKS 叢集。 AWS TNB 不再支援 Kubernetes 1.21 版。	2023 年 5 月 11 日
AWS.Compute.EKSSelfManagedNode	您可以在區域內、 AWS 本機區域和 上建立自我管理的工作者節點 AWS Outposts。	2023 年 3 月 29 日
初始版本	這是 AWS TNB 使用者指南的第一個版本。	2023 年 2 月 21 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。