Guía del usuario de

AWS Creador de redes de telecomunicaciones

AWS Creador de redes de telecomunicaciones: Guía del usuario de

Table of Contents

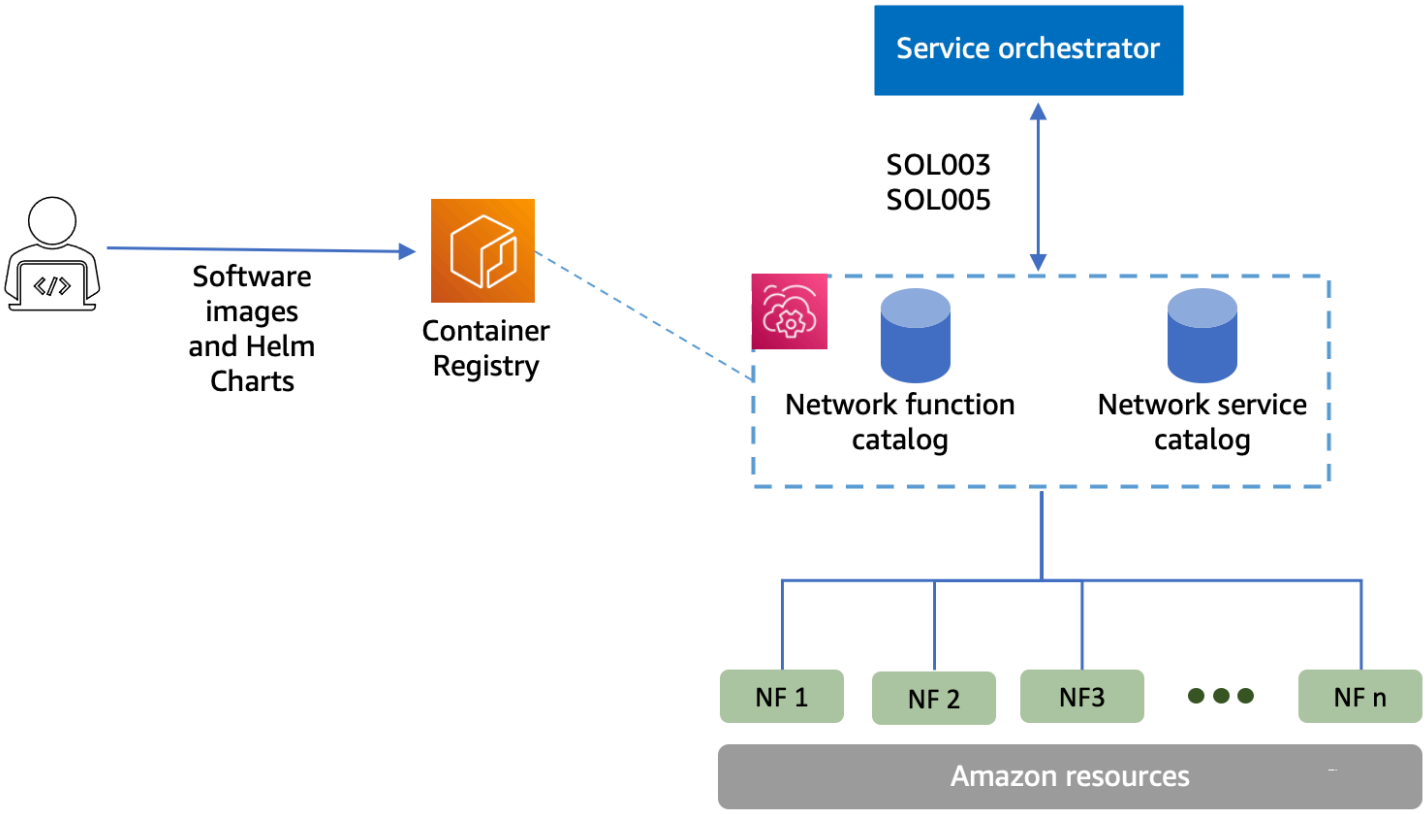
¿Qué es AWS Telco Network Builder?

AWS Telco Network Builder (AWS TNB) es un AWS servicio que proporciona a los proveedores de servicios de comunicación (CSPs) una forma eficiente de implementar, administrar y escalar redes 5G en la infraestructura. AWS

Con AWS TNB, usted despliega redes 5G escalables y seguras Nube de AWS utilizando una imagen de su red de forma automatizada. No necesita aprender nuevas tecnologías, decidir qué servicio de cómputo usar ni saber cómo aprovisionar y configurar AWS los recursos.

En su lugar, debe describir la infraestructura de su red y proporcionar las imágenes de software de las funciones de la red de sus socios proveedores de software independientes (ISV). AWS TNB se integra con orquestadores de AWS servicios y servicios de terceros para aprovisionar automáticamente la AWS infraestructura necesaria, implementar funciones de red en contenedores y configurar la administración de redes y accesos para crear un servicio de red totalmente operativo.

El siguiente diagrama ilustra las integraciones lógicas entre AWS TNB y los orquestadores de servicios para implementar funciones de red mediante interfaces estándar basadas en el Instituto Europeo de Normas de Telecomunicación (ETSI).



Temas

AWS¿Nuevo en?

Si es la primera vez que AWS conoce los productos y servicios, comience a aprender más con los siguientes recursos:

- Introducción a AWS
- Empezando con AWS

¿Para quién es AWS TNB?

AWS TNB CSPs busca aprovechar la rentabilidad, la agilidad y la elasticidad que Nube de AWS ofrece sin tener que escribir ni mantener scripts y configuraciones personalizados para diseñar, implementar y administrar los servicios de red. AWS TNB aprovisiona automáticamente la AWS infraestructura necesaria, despliega funciones de red en contenedores y configura la administración de redes y accesos para crear servicios de red totalmente operativos basados en los descriptores de servicios de red definidos por el CSP y en las funciones de red que el CSP desea implementar.

AWS Características del TNB

Los siguientes son algunos de los motivos por los que un CSP querría utilizar AWS el TNB:

Ayuda a simplificar tareas

> Proporcione una mayor eficiencia a sus operaciones de red, como la implementación de nuevos servicios, la actualización y mejora de las funciones de la red y el cambio de las topologías de la infraestructura de red.

Se integra con los orquestadores

AWS TNB se integra con populares orquestadores de servicios de terceros que cumplen con la ETSI.

Escalas

Puede configurar el AWS TNB para escalar AWS los recursos subyacentes a fin de satisfacer la demanda de tráfico, actualizar las funciones de la red de manera más eficiente, implementar cambios en la topología de la infraestructura de la red y reducir el tiempo de implementación de los nuevos servicios 5G de días a horas.

Inspecciona y monitorea los recursos AWS

AWS TNB le permite inspeccionar y supervisar los AWS recursos que dan soporte a su red en un único panel, como Amazon VPC, Amazon y EC2 Amazon EKS.

Admite plantillas de servicio

AWS TNB le permite crear plantillas de servicios para todas las cargas de trabajo de telecomunicaciones (RAN, Core, IMS). Puede crear una nueva definición de servicio, reutilizar una plantilla existente o integrarla con una canalización de integración y entrega continuas (CI/CD) para publicar una nueva definición.

Realiza un seguimiento de los cambios en las implementaciones de red

Al cambiar la configuración subyacente de un despliegue de funciones de red, por ejemplo, al cambiar el tipo de instancia de un tipo de EC2 instancia de Amazon, puede realizar un seguimiento de los cambios de forma repetible y escalable. Hacerlo manualmente requeriría administrar el estado de la red, crear y eliminar recursos y prestar atención al orden de los cambios necesarios. Cuando utilizas AWS TNB para gestionar el ciclo de vida de la función de red, solo realizas los cambios en los descriptores de los servicios de red que describen la función de red. AWS A continuación, TNB realizará automáticamente los cambios necesarios en el orden correcto.

Simplifica el ciclo de vida de las funciones de red

Puede administrar la primera versión y todas las versiones posteriores de una función de red y especificar cuándo realizar la actualización. También puede administrar sus aplicaciones RAN, Core, IMS y de red de la misma manera.

Acceder a TNB AWS

Puede crear, acceder y administrar sus recursos de AWS TNB mediante cualquiera de las siguientes interfaces:

- AWS Consola TNB: proporciona una interfaz web para administrar la red.
- AWS API de TNB: proporciona una RESTful API para realizar acciones de AWS TNB. Para obtener más información, consulte Referencia de la API de AWS.
- AWS Command Line Interface (AWS CLI): proporciona comandos para un amplio conjunto de AWS servicios, incluido AWS TNB. Es compatible con Windows, macOS y Linux. Para obtener más información, consulte la AWS Command Line Interface Guía del usuario de .
- AWS SDKs— Proporciona un idioma específico APIs y completa muchos de los detalles de la conexión. Incluyen cálculos de firmas, control de reintentos de solicitud y control de errores. Para obtener más información, consulte AWS SDKs.

Precios para TNB AWS

AWS TNB ayuda a CSPs automatizar el despliegue y la administración de sus redes de telecomunicaciones en. AWS AI utilizar AWS TNB, paga por las dos dimensiones siguientes:

- Por horas de elementos de función de red gestionados (MNFI).
- Por número de solicitudes de API.

También incurrirá en cargos adicionales al utilizar otros AWS servicios junto con AWS TNB. Para obtener más información, consulte Precios de AWS TNB.

Para ver su factura, vaya al Panel de Billing and Cost Management en la consola de Administración de facturación y costos de AWS. La factura contiene enlaces a informes de uso que ofrecen detalles sobre la cuenta. Para obtener más información sobre la facturación de la AWS cuenta, consulte Facturación de la AWS cuenta.

Si tienes preguntas sobre la AWS facturación, las cuentas y los eventos, ponte en contacto con AWS Support.

AWS Trusted Advisor es un servicio que puede utilizar para ayudar a optimizar los costes, la seguridad y el rendimiento de su AWS entorno. Para obtener más información, consulte AWS Trusted Advisor.

Siguientes pasos

Para obtener más información acerca de cómo empezar a utilizar AWS TNB, consulte los temas siguientes:

- Configuración AWS TNB: completar los pasos de requisitos previos.
- Cómo empezar con AWS TNB: implemente su primera función de red, como la unidad centralizada (CU), la función de gestión del acceso y la movilidad (AMF), la función de plano de usuario (UPF) o un núcleo 5G completo.

Cómo AWS funciona TNB

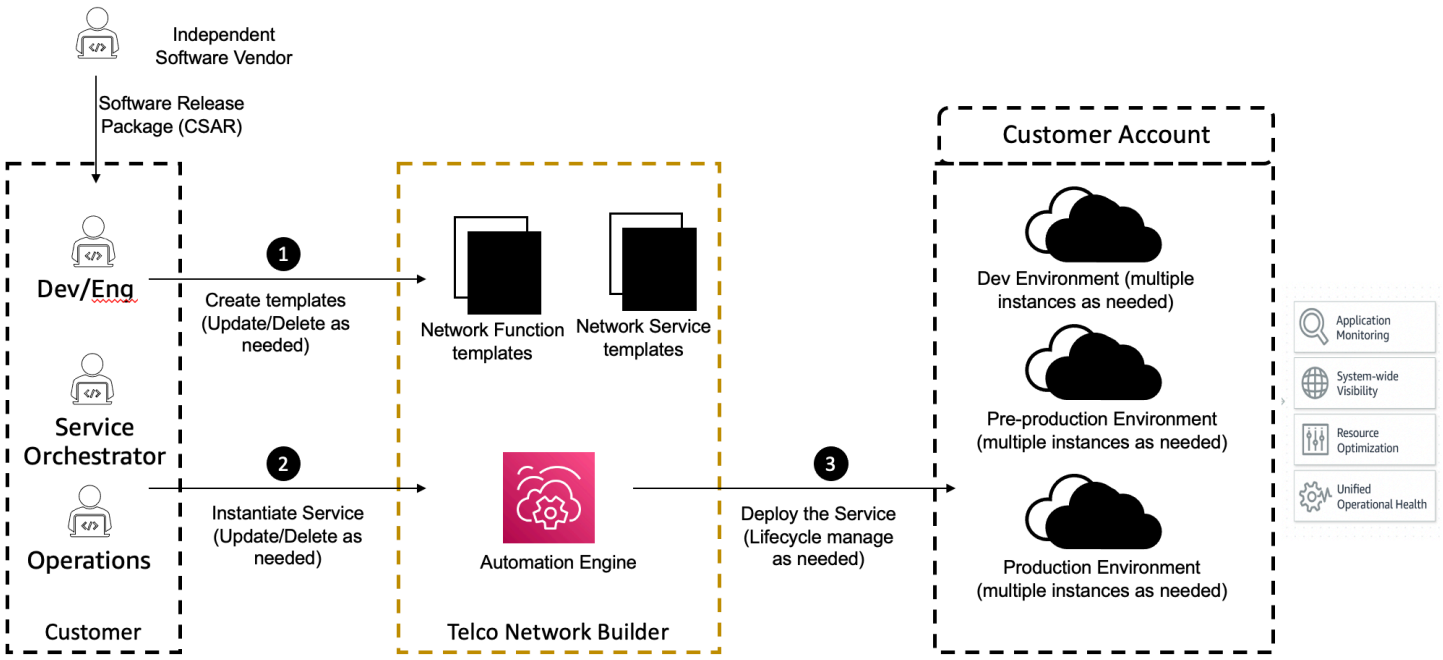
AWS TNB se integra con end-to-end orquestadores y AWS recursos estandarizados para operar redes 5G completas.

AWS TNB le permite incorporar paquetes de funciones de red y descriptores de servicios de red (NSDs) y le proporciona el motor de automatización necesario para operar sus redes. Puede usar su end-to-end orquestador e integrarlo con AWS TNB APIs, o usar AWS TNB SDKs para crear su propio flujo de automatización. Para obtener más información, consulte AWS Arquitectura TNB.

Temas

AWS Arquitectura TNB

AWS TNB le brinda la capacidad de realizar operaciones de administración del ciclo de vida a través de la Consola de administración de AWS API REST de AWS TNB y. AWS CLI SDKs Esto permite a las distintas personas de CSP, como los miembros de los equipos de ingeniería, operaciones y sistemas programáticos, aprovechar las ventajas de AWS TNB. Puede crear y subir un paquete de funciones de red como un archivo de servicios en la nube (CSAR). El archivo CSAR contiene gráficos de Helm, imágenes de software y un descriptor de funciones de red (NFD). Puede utilizar plantillas para implementar varias configuraciones de ese paquete de forma repetida. Puede crear plantillas de servicios de red que definen la infraestructura y las funciones de red que desea implementar. Puede utilizar las anulaciones de parámetros para implementar diferentes configuraciones en diferentes ubicaciones. A continuación, puede crear instancias de una red mediante las plantillas e implementar las funciones de la red en la infraestructura. AWS AWS TNB le proporciona la visibilidad de sus despliegues.



Integración con Servicios de AWS

Una red 5G se compone de un conjunto de funciones de red en contenedores interconectadas que se despliegan en miles de clústeres de Kubernetes. AWS TNB se integra con lo siguiente Servicios de AWS como elemento específico de las telecomunicaciones para crear un servicio de red totalmente operativo APIs :

- Amazon Elastic Container Registry (Amazon ECR) para almacenar artefactos de funciones de red de proveedores de software independientes ISVs ().
- Amazon Elastic Kubernetes Service (Amazon EKS) para configurar clústeres.
- Amazon VPC para constructos de redes.
- Grupos de seguridad que utilizan CloudFormation.
- AWS CodePipeline para objetivos de despliegue en todas Regiones de AWS las Zonas AWS Locales y AWS Outposts.
- IAM para definir roles.
- AWS Organizations para controlar el acceso a AWS TNB APIs.
- Panel de estado y AWS CloudTrail para monitorear el estado y publicar las métricas.

AWS Cuotas de recursos de TNB

Cuenta de AWS Tiene cuotas predeterminadas, anteriormente denominadas límites, para cada una Servicio de AWS de ellas. A menos que se indique lo contrario, cada cuota es específica de un Región de AWS. Puede solicitar el aumento de algunas cuotas, pero no de todas.

Para ver las cuotas de AWS TNB, abra la consola Service Quotas. En el panel de navegación, elija Servicios de AWS y seleccione AWS TNB.

Para solicitar un aumento de cuota, consulte Solicitud de un aumento de cuota en la Guía de usuario de Service Quotas.

Cuenta de AWS Tiene las siguientes cuotas relacionadas con AWS TNB.

Cuota de recursos	Description (Descripción)	Predeterminado	¿Ajustable?
Instancias de servicios de red	El número máximo de instancias de servicios de red por región.	800	Sí
Operaciones de servicios de red simultáneas y continuas	Establece el número máximo de operaciones de red simultáneas en curso en una región.	40	Sí
Paquetes de red	Establece el número máximo de paquetes de red en una región.	40	Sí
Paquetes de funciones	El número máximo de paquetes de funciones en una región.	200	Sí

AWS Conceptos de TNB

En este tema se describen los conceptos esenciales que le ayudarán a empezar a utilizar el AWS TNB.

Contenido

Ciclo de vida de una función de red

AWS TNB le ayuda durante todo el ciclo de vida de las funciones de su red. El ciclo de vida de las funciones de red incluye las siguientes etapas y actividades:

Planificación

1. Planifique su red identificando las funciones de red que se van a implementar.
2. Coloque las imágenes del software de funciones de red en un repositorio de imágenes contenedor.
3. Cree los paquetes CSAR que vaya a implementar o actualizar.
4. Utilice AWS TNB para cargar el paquete CSAR que define la función de su red (por ejemplo, CU, AMF y UPF) e intégrelo con una canalización de integración y entrega continuas (CI/CD) que pueda ayudarlo a crear nuevas versiones de su paquete CSAR a medida que estén disponibles nuevas imágenes de software de funciones de red o scripts de clientes.

Configuración

1. Identifique la información necesaria para la implementación, como el tipo de cómputo, la versión de la función de red, la información de IP y los nombres de los recursos.
2. Utilice la información para crear el descriptor de servicio de red (NSD).
3. Tenga en cuenta NSDs que las funciones de su red y los recursos necesarios definen la función de red para instanciarse.

Instanciación

1. Cree la infraestructura que requieren las funciones de la red.
2. Cree una instancia (o aprovisione) la función de red tal como se define en su NSD y comience a transportar tráfico.
3. Valide los activos.

Producción

Durante el ciclo de vida de la función de red, completará operaciones de producción tales como:

- Actualice la configuración de la función de red, por ejemplo, actualice un valor en la función de red implementada.
- Actualice la instancia de red con un nuevo paquete de red y valores de parámetros. Por ejemplo, actualice el `version` parámetro Amazon EKS en el paquete de red.

Utilizar interfaces estandarizadas

AWS TNB se integra con orquestadores de servicios compatibles con el Instituto Europeo de Normas de Telecomunicación (ETSI), lo que le permite simplificar la implementación de sus servicios de red. Los orquestadores de servicios pueden usar la AWS TNB, SDKs la CLI o la APIs para iniciar operaciones, como crear instancias o actualizar una función de red a una nueva versión.

AWS TNB admite las siguientes especificaciones.

Especificación	Release	Description (Descripción)
ETSI SOL001	v3.6.1	Define los estándares para permitir los descriptores de funciones de red basados en TOSCA.
ETSI SOL002	v3.6.1	Define modelos en torno a la gestión de funciones de red.
ETSI SOL003	v3.6.1	Define los estándares para la gestión del ciclo de vida de las funciones de red.
ETSI SOL004	v3.6.1	Define los estándares CSAR para los paquetes de funciones de red.

Especificación	Release	Description (Descripción)
ETSI SOL005	v3.6.1	Define los estándares para la gestión del paquete de servicios de red y del ciclo de vida de los servicios de red.
ETSI SOL007	v3.5.1	Define los estándares para permitir los descriptores de servicios de red basados en TOSCA.

Paquete de funciones

Con AWS TNB, puede almacenar paquetes de funciones que cumplen con la ETSI SOL001/SOL004 en un catálogo de funciones. A continuación, puede cargar paquetes de Cloud Service Archive (CSAR) que contengan artefactos que describan el funcionamiento de su red virtual.

- Descriptor de función de red virtual: define los metadatos para la incorporación de paquetes y la administración de funciones de red virtual. Debe asignar un nombre a este archivo `vnfd.yaml`.
- Imágenes de software: hace referencia a la función de red virtual Container Images. Amazon Elastic Container Registry (Amazon ECR) puede actuar como repositorio de imágenes de funciones de red virtual.
- Archivos adicionales: utilícelos para administrar la función de la red virtual; por ejemplo, scripts y gráficos de Helm.

El CSAR es un paquete definido por el estándar OASIS TOSCA e incluye un descriptor de red/servicio que cumple con la especificación OASIS TOSCA YAML. Para obtener información sobre la especificación YAML requerida, consulte. Referencia TOSCA para TNB AWS

A continuación se muestra un ejemplo de un descriptor de función de red virtual.

```
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"
```

Paquete de red

Un paquete de red es un `.zip` archivo en formato CSAR (Cloud Service Archive). Define los paquetes de funciones que desea implementar y la AWS infraestructura en la que desea implementarlos.

El paquete de red contiene los siguientes archivos:

- Un archivo descriptor de red (`nsd.yaml`) en formato TOSCA, tal como se describe en ETSI SOL007.

  El `nsd.yaml` archivo contiene referencias a los paquetes de funciones cargados con su descriptor. IDs
- Scripts de datos de usuario, si los hay.
- Scripts de enlaces de ciclo de vida, si los hay.
- Archivos de `values.yaml` configuración de los complementos, si los hay.

AWS TNB es compatible con los estándares de la ETSI para el modelado de recursos, como redes, servicios y funciones, en el lenguaje TOSCA. AWS TNB hace que su uso Servicios de AWS sea más eficiente al modelarlos de una manera que su orquestador de servicios compatible con la ETSI pueda entender.

Descriptores de servicios de red para TNB AWS

Un descriptor de servicio de red (NSD) es un archivo `.yaml` de un paquete de red que utiliza el estándar TOSCA para describir las funciones de red que desea implementar y la infraestructura AWS

en la que desea implementar las funciones de red. Para definir su NSD y configurar sus recursos subyacentes y las operaciones del ciclo de vida de la red, debe comprender el esquema TOSCA del NSD compatible con TNB. AWS

El archivo NSD está dividido en las partes siguientes:

1. Versión de definición de TOSCA: es la primera línea del archivo NSD YAML y contiene la información de la versión, que se muestra en el siguiente ejemplo.

```
tosca_definitions_version: tnb_simple_yaml_1_0
```

2. VNFD: el NSD contiene la definición de la función de red en la que se deben realizar las operaciones del ciclo de vida. Cada función de la red debe identificarse mediante los siguientes valores:
   - Un identificador único para `descriptor_id`. El identificador debe coincidir con el identificador del paquete CSAR de la función de red.
   - Un nombre exclusivo para `namespace`. El nombre debe estar asociado a un identificador único para poder consultarlo más fácilmente en todo el archivo NSD YAML, como se muestra en el siguiente ejemplo.

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

3. Plantilla de topología: define los recursos que se van a implementar, la implementación de la función de red y cualquier guion personalizado, como los enlaces de ciclo de vida. Esto se muestra en el siguiente ejemplo.

```
topology_template:

  node_templates:

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

4. Nodos adicionales: cada recurso modelado tiene secciones para propiedades y requisitos. Las propiedades describen los atributos opcionales u obligatorios de un recurso, como la versión. Los requisitos describen las dependencias que se deben proporcionar como argumentos. Por ejemplo, para crear un recurso de grupo de nodos de Amazon EKS, debe crearse dentro de un clúster de Amazon EKS. Esto se muestra en el siguiente ejemplo.

```
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
```

Ejemplo de NSD

El siguiente es un fragmento de un NSD que muestra cómo modelar. Servicios de AWS La función de red se implementará en un clúster de Amazon EKS con la versión 1.27 de Kubernetes. Las subredes de las aplicaciones son Subnet01 y Subnet02. A continuación, puede definir las opciones NodeGroups para sus aplicaciones con una imagen de máquina de Amazon (AMI), un tipo de instancia y una configuración de escalado automático.

```
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
```

Administración y operaciones de TNB AWS

Con AWS TNB, puede gestionar su red mediante operaciones de gestión estandarizadas de acuerdo con las normas ETSI SOL003 y SOL005. Puede utilizar el AWS TNB para realizar operaciones del ciclo de vida, APIs tales como:

- Creación de instancias de las funciones de su red.
- Finalización de las funciones de su red.
- Actualizar las funciones de su red para anular implementaciones de Helm.
- Actualizar una instancia de red instanciada o actualizada con un nuevo paquete de red y valores de parámetros.
- Administrar las versiones de sus paquetes de funciones de red.
- Administrar versiones de su. NSDs
- Recuperar información sobre las funciones de red implementadas.

Configuración AWS TNB

Configure el AWS TNB realizando las tareas descritas en este tema.

Tareas

- Inscríbase en una Cuenta de AWS
- Elija un AWS Region
- Observe el punto de conexión de servicio
- (Opcional) Instale el AWS CLI
- Configuración AWS Funciones de TNB

Inscríbase en una Cuenta de AWS

Para empezar AWS, necesitas un Cuenta de AWS. Para obtener información sobre cómo crear un Cuenta de AWS, consulte Cómo empezar con un Cuenta de AWS en la Guía de AWS Account Management referencia.

Elija un AWS Region

Para ver la lista de regiones disponibles para AWS TNB, consulte la Lista de servicios AWS regionales. Para ver la lista de puntos de conexión para el acceso programático, consulte puntos de conexión de AWS TNB en Referencia general de AWS.

Observe el punto de conexión de servicio

Para conectarse mediante programación a un AWS servicio, utilice un punto final. Además de los AWS puntos finales estándar, algunos AWS servicios ofrecen puntos finales FIPS en determinadas regiones. Para obtener más información, consulte puntos de conexión de servicio de AWS.

Nombre de la región	Región	Punto de conexión	Protocolo	
Este de EE. UU.	us-east-1	tnb.us-east-1.amazonaws.com	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
(Norte de Virginia)				
Oeste de EE. UU. (Oregón)	us-west-2	tnb.us-west-2.amazonaws.com	HTTPS	
Asia-Pacífico (Seúl)	ap-northeast-2	tnb.ap-northeast-2.amazonaws.com	HTTPS	
Asia-Pacífico (Sídney)	ap-southeast-2	tnb.ap-southeast-2.amazonaws.com	HTTPS	
Canadá (centro)	ca-central-1	tnb.ca-central-1.amazonaws.com	HTTPS	
Europa (Fráncfort)	eu-central-1	tnb.eu-central-1.amazonaws.com	HTTPS	
Europa (París)	eu-west-3	tnb.eu-west-3.amazonaws.com	HTTPS	
Europa (España)	eu-south-2	tnb.eu-south-2.amazonaws.com	HTTPS	
Europa (Estocolmo)	eu-north-1	tnb.eu-north-1.amazonaws.com	HTTPS	

Nombre de la región	Región	Punto de conexión	Protocolo	
América del Sur (São Paulo)	sa-east-1	tnb.sa-east-1.amazonaws.com	HTTPS	

(Opcional) Instale el AWS CLI

El AWS Command Line Interface (AWS CLI) proporciona comandos para un amplio conjunto de AWS productos y es compatible con Windows, macOS y Linux. Puede acceder a AWS TNB mediante. AWS CLI Para empezar, consulte la AWS Command Line Interface Guía del usuario de . Para obtener más información sobre los comandos de AWS TNB, consulte tnb en la AWS CLI Referencia de comandos.

Configuración AWS Funciones de TNB

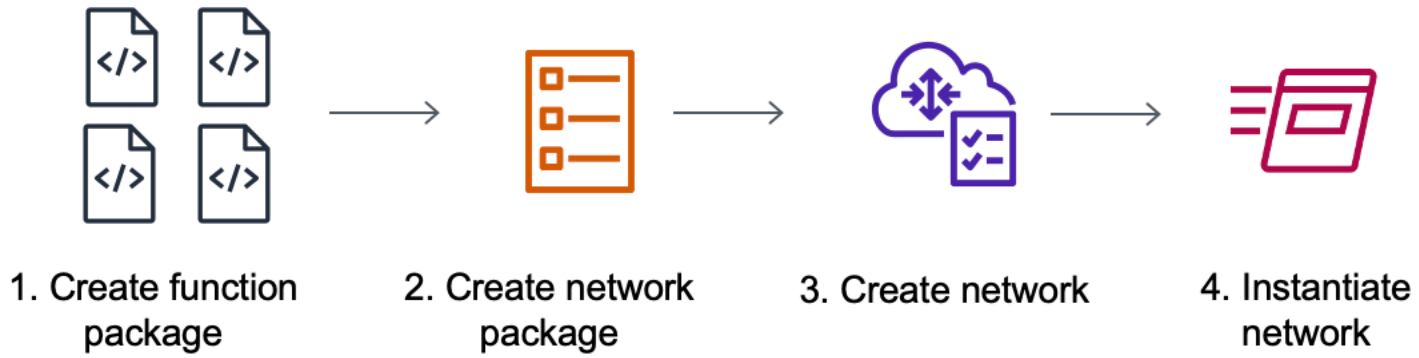
Debe crear un rol de servicio de IAM para administrar diferentes partes de su solución AWS TNB. AWS Los roles de servicio de TNB pueden realizar llamadas a la API a otros AWS servicios, como AWS CloudFormation AWS CodeBuild, y a varios servicios de computación y almacenamiento, en su nombre, a fin de crear instancias y administrar los recursos para su implementación.

Para obtener más información sobre la función de servicio de AWS TNB, consulte. Administración de identidad y acceso para AWS TNB

Cómo empezar con AWS TNB

En este tutorial, se muestra cómo se utiliza el AWS TNB para implementar una función de red, por ejemplo, la unidad centralizada (CU), la función de gestión del acceso y la movilidad (AMF) o la función de plano de usuario de 5G (UPF).

En el diagrama siguiente se ilustra el proceso de implementación:



Tareas

- Requisitos previos
- Cree un paquete de funciones
- Crear un paquete de red
- Crear e instanciar una instancia de red
- Limpieza

Requisitos previos

Para poder realizar una implementación exitosa, debe disponer de lo siguiente:

- Un plan AWS Business Support.
- Permisos mediante funciones de IAM.
- Un paquete de funciones de red (NF) que cumple con la norma ETSI SOL001/SOL004.
- Plantillas de descriptores de servicios de red (NSD) que cumplen con la norma ETSI SOL007.

Puede utilizar un paquete de funciones o un paquete de red de ejemplo del sitio de paquetes de muestra para TNB. AWS GitHub

Cree un paquete de funciones

Un paquete de funciones de red es un archivo de Cloud Service Archive (CSAR). El archivo CSAR contiene gráficos de Helm, imágenes de software y un descriptor de funciones de red (NFD).

Para crear un paquete de funciones

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Paquetes de funciones en el panel de navegación.
3. Elija Crear paquete de funciones.
4. En Cargar paquete de funciones, elija Elegir archivos y cargue cada paquete CSAR como un `.zip` archivo. Puede cargar un máximo de 10 archivos.
5. (Opcional) En Etiquetas, selecciona Añadir nueva etiqueta e introduce una clave y un valor. Puede usar etiquetas para buscar y filtrar sus recursos o realizar un seguimiento de sus AWS costos.
6. Elija Next (Siguiente).
7. Revise los detalles del paquete y, a continuación, seleccione Crear paquete de funciones.

Crear un paquete de red

Un paquete de red especifica las funciones de red que desea implementar y cómo desea implementarlas en el catálogo.

Para crear un paquete de red

1. Seleccione Paquetes de red en el panel de navegación.
2. Seleccione Crear paquete de red.
3. En Cargar paquete de red, elija Elegir archivos y cargue cada NSD como un `.zip` archivo. Puedes cargar un máximo de 10 archivos.
4. (Opcional) En Etiquetas, selecciona Añadir nueva etiqueta e introduce una clave y un valor. Puede usar etiquetas para buscar y filtrar sus recursos o realizar un seguimiento de sus AWS costos.

5. Elija Next (Siguiente).
6. Seleccione Crear paquete de red.

Crear e instanciar una instancia de red

Una instancia de red es una red única creada en AWS TNB que se puede implementar. Debe crear una instancia de red e instanciarla. Al crear una instancia de red, AWS TNB aprovisiona la AWS infraestructura necesaria, despliega funciones de red en contenedores y configura la administración de redes y accesos para crear un servicio de red totalmente operativo.

Para crear e instanciar una instancia de red

1. En el panel de navegación, elija Redes.
2. Elija Crear instancia de red.
3. Introduzca un nombre y una descripción para la red y, a continuación, elija Siguiente.
4. Elija un paquete de red. Compruebe los detalles y seleccione Siguiente.
5. Elija Crear instancia de red. El estado inicial es `Created`.

   Aparece la página Redes que muestra la nueva instancia de red en el `Not instantiated` estado.
6. Seleccione la instancia de red, elija Acciones e Instanciar.

   Aparece la página de creación de instancias de red.
7. Revise los detalles y actualice los valores de los parámetros. Las actualizaciones de los valores de los parámetros solo se aplican a esta instancia de red. Los parámetros de los paquetes NSD y VNFD no cambian.
8. Elija Instanciar red.

   Aparece la página de estado del despliegue.
9. Utilice el icono de actualización para realizar un seguimiento del estado de despliegue de la instancia de red. También puede activar la actualización automática en la sección Tareas de despliegue para realizar un seguimiento del progreso de cada tarea.

Limpieza

Ahora puede eliminar los recursos que creó para este tutorial.

Para limpiar los recursos

1. En el panel de navegación, elija Redes.
2. Elija el identificador de la red y, a continuación, elija Finalizar.
3. Cuando se le solicite confirmación, introduzca el identificador de red y, a continuación, elija Finalizar.
4. Utilice el icono de actualización para realizar un seguimiento del estado de la instancia de red.
5. (Opcional) Seleccione la red y elija Eliminar.

Paquetes de funciones para AWS TNB

Un paquete de funciones es un archivo .zip en formato CSAR (Cloud Service Archive) que contiene una función de red (una aplicación de telecomunicaciones estándar de la ETSI) y un descriptor de paquete de funciones que utiliza el estándar TOSCA para describir cómo deben ejecutarse las funciones en su red.

Tareas

- Cree un paquete de funciones en TNB AWS
- Vea un paquete de funciones en TNB AWS
- Descargue un paquete de funciones de AWS TNB
- Elimine un paquete de funciones de TNB AWS

Cree un paquete de funciones en TNB AWS

Aprenda a crear un paquete de funciones en el catálogo de funciones de red AWS TNB. Crear un paquete de funciones es el primer paso para crear una red en AWS TNB. Después de cargar un paquete de funciones, puede crear un paquete de red.

Console

Cree un paquete de funciones mediante la consola

1. Abra la consola AWS TNB en https://console.aws.amazon.com/tnb/.
2. Seleccione Paquetes de funciones en el panel de navegación.
3. Elija Crear paquete de funciones.
4. Elija Elegir archivos y cargue cada paquete CSAR como un `.zip` archivo. Puede cargar un máximo de 10 archivos.
5. Elija Siguiente.
6. Revise los detalles del paquete.
7. Elija Crear paquete de funciones.

AWS CLI

Para crear un paquete de funciones mediante el AWS CLI

1. Utilice el create-sol-function-packagecomando para crear un nuevo paquete de funciones:

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

2. Utilice el comando put-sol-function-package-content para cargar el contenido del paquete de funciones. Por ejemplo:

```
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
```

Vea un paquete de funciones en TNB AWS

Aprenda a ver el contenido de un paquete de funciones.

Console

Para ver un paquete de funciones mediante la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Paquetes de funciones en el panel de navegación.
3. Utilice el cuadro de búsqueda para encontrar el paquete de funciones

AWS CLI

Para ver un paquete de funciones mediante el AWS CLI

1. Utilice el list-sol-function-packagescomando para enumerar los paquetes de funciones.

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

2. Utilice el get-sol-function-packagecomando para ver los detalles de un paquete de funciones.

```
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
```

Descargue un paquete de funciones de AWS TNB

Aprenda a descargar un paquete de funciones del catálogo de funciones de red AWS TNB.

Console

Para descargar un paquete de funciones mediante la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. En el panel de navegación del lado izquierdo de la consola, elija Paquetes de funciones.
3. Utilice el cuadro de búsqueda para encontrar el paquete de funciones
4. Elija el paquete de funciones
5. En Acciones, elija Descargar.

AWS CLI

Para descargar un paquete de funciones mediante el AWS CLI

Utilice el comando get-sol-function-package-content para descargar un paquete de funciones.

```
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
```

Elimine un paquete de funciones de TNB AWS

Aprenda a eliminar un paquete de funciones del catálogo de funciones de red AWS TNB. Para eliminar un paquete de funciones, el paquete debe estar desactivado.

Console

Para eliminar un paquete de funciones mediante la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Paquetes de funciones en el panel de navegación.
3. Utilice el cuadro de búsqueda para encontrar el paquete de funciones.
4. Elija un paquete de funciones.
5. Elija Acciones, Desactivar.
6. Elija Acciones, Eliminar.

AWS CLI

Para eliminar un paquete de funciones mediante el AWS CLI

1. Utilice el update-sol-function-packagecomando para deshabilitar un paquete de funciones.

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

2. Utilice el delete-sol-function-packagecomando para eliminar un paquete de funciones.

```
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
```

Paquetes de red para AWS TNB

Un paquete de red es un archivo.zip en formato CSAR (Cloud Service Archive). Define los paquetes de funciones que desea implementar y la AWS infraestructura en la que desea implementarlos.

El paquete de red contiene los siguientes archivos:

- Un archivo descriptor de red (`nsd.yaml`) en formato TOSCA, tal como lo describe la ETSI. SOL007

  El `nsd.yaml` archivo contiene referencias a los paquetes de funciones cargados con su descriptor. IDs
- Scripts de datos de usuario, si los hay.
- Scripts de enlaces de ciclo de vida, si los hay.
- Archivos de `values.yaml` configuración de los complementos, si los hay.

Tareas

- Cree un paquete de red en TNB AWS
- Vea un paquete de red en TNB AWS
- Descargue un paquete de red de AWS TNB
- Elimine un paquete de red de TNB AWS

Cree un paquete de red en TNB AWS

Un paquete de red consta de un archivo descriptor del servicio de red (NSD) (obligatorio) y cualquier archivo adicional (opcional), como los guiones específicos que se adapten a sus necesidades. Por ejemplo, si tiene varios paquetes de funciones en su paquete de red, puede usar el NSD para definir qué funciones de red deben ejecutarse en determinadas VPCs subredes o clústeres de Amazon EKS.

Cree un paquete de red después de crear los paquetes de funciones. Una vez que haya creado un paquete de red, debe crear una instancia de red.

Console

Para crear un paquete de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Paquetes de red en el panel de navegación.
3. Seleccione Crear paquete de red.
4. Elija Elegir archivos y cargue cada NSD como un `.zip` archivo. Puedes cargar un máximo de 10 archivos.
5. Elija Siguiente.
6. Revise los detalles del paquete.
7. Seleccione Crear paquete de red.

AWS CLI

Para crear un paquete de red mediante AWS CLI

1. Utilice el create-sol-network-packagecomando para crear un paquete de red.

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

2. Utilice el comando put-sol-network-package-content para cargar el contenido del paquete de red. Por ejemplo:

```
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
```

Vea un paquete de red en TNB AWS

Aprenda a ver el contenido de un paquete de red.

Console

Para ver un paquete de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Paquetes de red en el panel de navegación.
3. Utilice el cuadro de búsqueda para encontrar el paquete de red.

AWS CLI

Para ver un paquete de red mediante el AWS CLI

1. Utilice el list-sol-network-packagescomando para enumerar los paquetes de red.

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

2. Utilice el get-sol-network-packagecomando para ver los detalles de un paquete de red.

```
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
```

Descargue un paquete de red de AWS TNB

Aprenda a descargar un paquete de red del catálogo de servicios de red de AWS TNB.

Console

Para descargar un paquete de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Paquetes de red en el panel de navegación.
3. Utilice el cuadro de búsqueda para encontrar el paquete de red
4. Seleccione el paquete de red.
5. En Acciones, elija Descargar.

AWS CLI

Para descargar un paquete de red mediante AWS CLI

- Utilice el comando get-sol-network-package-content para descargar un paquete de red.

```
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
```

Elimine un paquete de red de TNB AWS

Aprenda a eliminar un paquete de red del catálogo de servicios de red AWS TNB. Para eliminar un paquete de red, el paquete debe estar desactivado.

Console

Para eliminar un paquete de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Paquetes de red en el panel de navegación.
3. Utilice el cuadro de búsqueda para encontrar el paquete de red
4. Seleccione el paquete de red
5. Elija Acciones, Desactivar.
6. Elija Acciones, Eliminar.

AWS CLI

Para eliminar un paquete de red mediante el AWS CLI

1. Utilice el update-sol-network-packagecomando para deshabilitar un paquete de red.

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

2. Utilice el delete-sol-network-packagecomando para eliminar un paquete de red.

```
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
```

Instancias de red para AWS TNB

Una instancia de red es una red única creada en AWS TNB que se puede implementar.

Tareas

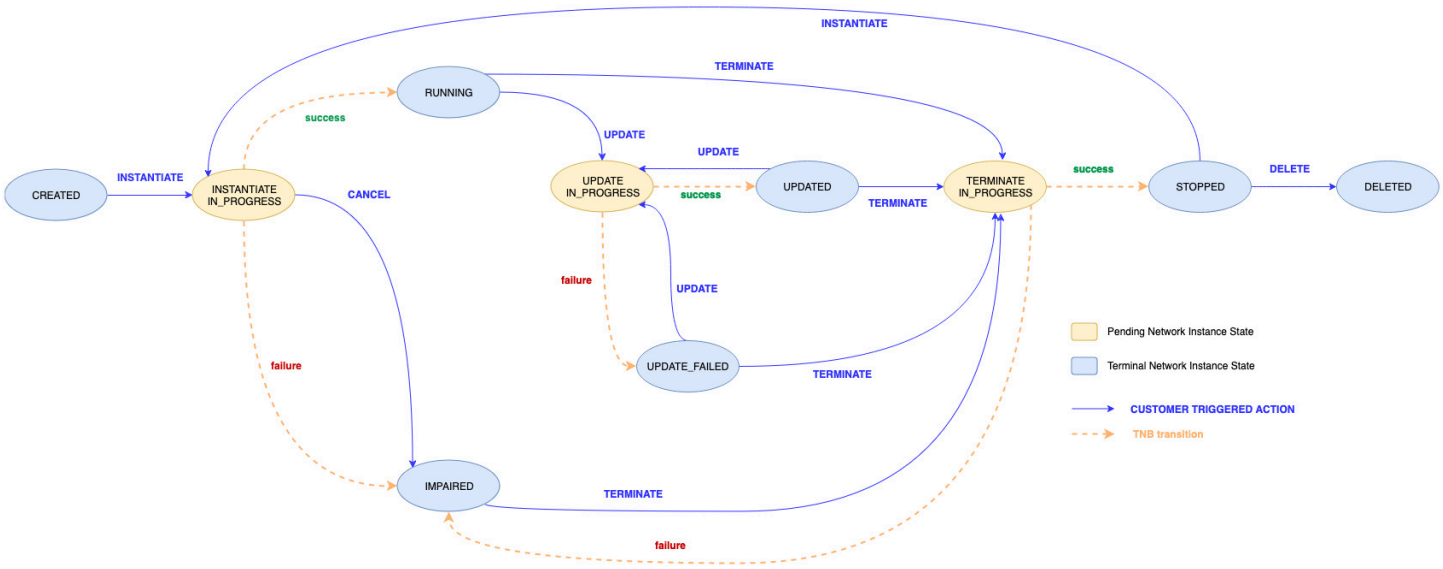
- Operaciones del ciclo de vida de una instancia de red
- Cree una instancia de red mediante TNB AWS
- Cree una instancia de red mediante TNB AWS
- Actualice una instancia de función en AWS TNB
- Actualice una instancia de red en AWS TNB
- Vea una instancia de red en AWS TNB
- Termine y elimine una instancia de red de AWS TNB

Operaciones del ciclo de vida de una instancia de red

AWS TNB le permite administrar fácilmente su red mediante operaciones de administración estandarizadas en línea con ETSI y. SOL003 SOL005 Puede realizar las siguientes operaciones del ciclo de vida:

- Cree la red
- Cree una instancia de la red
- Actualice la función de red
- Actualice la instancia de red
- Vea los detalles y el estado de la red
- Termine la red

La siguiente imagen muestra las operaciones de administración de la red:



Cree una instancia de red mediante TNB AWS

Una instancia de red se crea después de crear un paquete de red. Tras crear una instancia de red, instanciela.

Console

Para crear una instancia de red mediante la consola

1. Abra la consola AWS TNB en https://console.aws.amazon.com/tnb/.
2. En el panel de navegación, elija Redes.
3. Elija Crear instancia de red.
4. Introduzca un nombre y una descripción para la instancia y, a continuación, elija Siguiente.
5. Seleccione el paquete de red, compruebe los detalles y pulse Siguiente.
6. Elija Crear instancia de red.

   La nueva instancia de red aparece en la página Redes. A continuación, cree una instancia de esta instancia de red.

AWS CLI

Para crear una instancia de red mediante AWS CLI

- Utilice el create-sol-network-instancecomando para crear una instancia de red.

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

A continuación, cree una instancia de esta instancia de red.

Cree una instancia de red mediante TNB AWS

Tras crear una instancia de red, debe instanciarla. Al crear una instancia de red, AWS TNB aprovisiona la AWS infraestructura necesaria, implementa funciones de red en contenedores y configura la administración de redes y accesos para crear un servicio de red totalmente operativo.

Console

Para crear una instancia de red mediante la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. En el panel de navegación, elija Redes.
3. Seleccione la instancia de red que desee instanciar.
4. Elija Acciones y, a continuación, Crear una instancia.
5. En la página Instanciar la red, revise los detalles y, si lo desea, actualice los valores de los parámetros.

   Las actualizaciones de los valores de los parámetros solo se aplican a esta instancia de red. Los parámetros de los paquetes NSD y VNFD no cambian.
6. Elija Instanciar red.

   Aparece la página de estado del despliegue.
7. Utilice el icono de actualización para realizar un seguimiento del estado de despliegue de la instancia de red. También puede activar la actualización automática en la sección Tareas de despliegue para realizar un seguimiento del progreso de cada tarea.

   Cuando el estado de la implementación cambia a`Completed`, se crea una instancia de red.

AWS CLI

Para crear una instancia de red mediante el AWS CLI

1. Usa el instantiate-sol-network-instancecomando para crear una instancia de red.

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

2. A continuación, consulte el estado de funcionamiento de la red.

Actualice una instancia de función en AWS TNB

Después de crear una instancia de red, puede actualizar un paquete de funciones en la instancia de red.

Console

Para actualizar una instancia de función mediante la consola

1. Abra la consola AWS TNB en https://console.aws.amazon.com/tnb/.
2. En el panel de navegación, elija Redes.
3. Seleccione la instancia de red. Puede actualizar una instancia de red solo si su estado es`Instantiated`.

   Aparece la página de la instancia de red.
4. En la pestaña Funciones, seleccione la instancia de la función que desee actualizar.
5. Elija Actualizar.
6. Introduzca las anulaciones de actualización.
7. Elija Actualizar.

AWS CLI

Utilice la CLI para actualizar una instancia de función

Utilice el update-sol-network-instancecomando con el tipo de `MODIFY_VNF_INFORMATION` actualización para actualizar una instancia de función en una instancia de red.

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

Actualice una instancia de red en AWS TNB

Después de crear una instancia de red, es posible que necesite actualizar la infraestructura o la aplicación. Para ello, debe actualizar el paquete de red y los valores de los parámetros de la instancia de red e implementar la operación de actualización para aplicar los cambios.

Consideraciones

- Puede actualizar una instancia de red que esté en el `Updated` estado `Instantiated` o.
- Al actualizar una instancia de red, la `UpdateSolNetworkService` API usa el nuevo paquete de red y los valores de los parámetros para actualizar la topología de la instancia de red.
- AWS TNB comprueba que el número de parámetros NSD y VNFD de la instancia de red no supere los 200. Este límite se aplica para evitar que personas malintencionadas transmitan cargas erróneas o enormes que afecten al servicio.

Parámetros que puede actualizar

Puede actualizar los siguientes parámetros al actualizar una instancia de red instanciada:

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Versión de clúster de Amazon EKS	Puede actualizar el valor del `version` parámetro del plano de control del clúster de Amazon EKS a la siguiente versión secundaria. No puede cambiar la versión a una versión inferior.	`EKSCluster:`
		`  type: tosca.nod`
		`es.AWS.Compute.EKS`
		`  properties:`
		`  version: "1.28"`
			`EKSC`
			`r:`
			
			`typ`
			`tos`
			`es.A`
			`mput`

Parámetro	Description (Descripción)	Ejemplo: antes

Ejemp
Despu

```
 pro
s:

 ver
 "1.
```

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Nodos de trabajo de Amazon EKS	Puede actualizar el valor del `EKSManagedNode kubernetes_version` parámetro para actualizar el grupo de nodos a una versión más reciente de Amazon EKS, o puede actualizar el `ami_id` parámetro para actualizar el grupo de nodos a la última AMI optimizada para EKS.
	
	Puede actualizar el ID de AMI de`EKSSelfManagedNode` . La versión de Amazon EKS de la AMI debe ser igual o tener hasta 2 versiones anteriores a la versión de clúster de Amazon EKS. Por ejemplo, si la versión del clúster de Amazon EKS es 1.31, la versión de la AMI de Amazon EKS debe ser 1.31, 1.30 o 1.29.
		`EKSManagedNodeGroup01:`
		`  ...`
		`  properties:`
		`  kubernetes_version: "  1.28"`
		`  EKSSelfManagedNode01:`
		`  compute:`
		`  compute:`
		`  properties:`
		`  ami_id: "ami-1231230LD "`
			`EKSM`
			`dNod`
			`p01:`
			`...`
			`pro`
			`s:`
			`kub`
			`s_ve`
			`:`
			`"1.`
			`EKS`
			`nage`
			`01:`
			`com`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			com pro s: ami "am 3NEW

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Grupos de nodos de Amazon EKS	Puede añadir o eliminar grupos de nodos según sus necesidades informáticas.
	
	Al eliminar grupos de nodos existentes y añadir otros nuevos, asegúrese de que los nuevos grupos de nodos IDs sean diferentes de los grupos de nodos eliminados; de lo contrario, la operación se considerará una modificación del grupo de nodos en lugar de una eliminación y adición. Tenga en cuenta que, en el caso de los grupos de nodos existentes, solo se puede actualizar un conjunto limitado de parámetros. Desplázate por esta tabla para ver qué parámetros puedes actualizar.
		`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`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			mir 1 max 1 ... Free SNod # New Nod typ tos es.A mput Self edNo ... sca pro s:

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			des ize: 1 mir 1 max 1 ... *Free* *SNod* # New Nod typ tos es.A mput Mana de

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			... sca pro s: des ize: 1 mir 1 max 1

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			...

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Propiedades de escalado	Puede actualizar las propiedades de escalado de los nodos TOSCA `EKSManagedNode` y de los nodos `EKSSelfManagedNode` TOSCA.	`EKSNodeGroup01:`
		`  ...`
		`  scaling:`
		`  properties:`
		`  desired_size: 1`
		`  min_size: 1`
		`  max_size: 1`
			`EKSN`
			`oup0`
			`...`
			`sca`
			`pro`
			`s:`
			`des`
			`ize:`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			mir max

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Propiedades del complemento CSI de Amazon EBS	Puede activar o desactiva r el complemento CSI de Amazon EBS en sus clústeres de Amazon EKS. También puede cambiar la versión del complemento.	`EKSCluster:`
		`  capabilities:`
		`  ...`
		`  ebs_csi:`
		`  properties:`
		`  enabled: false`
			`EKSC`
			`r:`
			`cap`
			`ies:`
			`...`
			`ebs`
			`pro`
			`s:`
			`ena`
			`ver`
			`"v1`
			`e`
			`ksbu`
			`"`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Tamaño del volumen raíz	Puede añadir, eliminar o actualizar la propiedad de tamaño del volumen raíz de los EKSManaged nodos Node y EKSSelf ManagedNode TOSCA.	`Free5GCEKSNode01:`
		`  ...`
		`  capabilities:`
		`  compute:`
		`  properties:`
		`  root_volume_size: 50`
			`Free`
			`SNod`
			`...`
			`cap`
			`ies:`
			`com`
			`pro`
			`s:`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			roo me_s

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
VNF	Puede hacer referencia a ellos VNFs en el NSD e implement arlos en el clúster creado en el NSD mediante el nodo TOSCA. VNFDeployment Como parte de la actualiza ción, podrás añadir, actualizar y eliminar VNFs elementos de la red.	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 : "vr Upd VNF - des r_id "b7 839d -916 a166 " nam : "vr Add VNF .... Sa mple

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			elmD : typ tos es.A ploy VNFD ment req nts: clu EKS r vnf

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemplo: Despu
			- v leVN - v leVN

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Enlaces	Para ejecutar las operaciones del ciclo de vida antes y después de crear una función de red, añada los `post_create` ganchos `pre_create` y al `VNFDeployment` nodo.
	
	En este ejemplo, el `PreCreateHook` gancho se ejecutará antes de crear `vnf3.SampleVNF3` una instancia y se ejecutará después de crearla`vnf3.SampleVNF3` . `PostCreateHook`
		`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`
			`:`
			`"vn`
			`-`
			`des`
			`r_id`
			`"b7`
			`839d`
			`-916`
			`a166`
			`"`
			`nam`
			`:`
			`"vn`
			`....`
			`S`
			`ampl`
			`Helm`
			`y:`
			`typ`
			`tos`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			es.A ploy VNFD ment rec nts: clu EKS r vnf - v leVN No cha to thi fur as the nam and uui rem the sam

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			- v leVN New VNF as the nam , vnf was not pre y pre int s: Hoo pos te: eHoo pre e: Hook

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Enlaces	Para ejecutar las operaciones del ciclo de vida antes y después de actualizar una función de red, puede añadir el `pre_update` gancho y el `post_update` gancho al nodo. `VNFDeployment`
	
	En este ejemplo, `PreUpdate Hook` se ejecutará antes de la actualización y `vnf1.SampleVNF1` se `PostUpdateHook` ejecutará después de que `vnf1.SampleVNF1` se actualice en el `vnf` paquete indicado por la actualización `uuid` para el espacio de nombres vnf1.
		`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`
			vnfd
			
			-
			des
			r_id
			"0e
			bd87
			-
			b8a1
			4666
			"
			
			nam
			:
			"vn
			
			-
			des
			r_id
			"64
			ecd6
			-
			bf94
			4b53
			"
			
			nam
			:
			"vn
			...
			S
			ampl
			Helm
			y:
			
			typ

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			tos es.A ploy VNFD ment rec nts: clu EKS r vnf - v leVN A VNF upd as the uui cha for nam "vr - v

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

```
leVN
 No
 cha
 to
 thi
 fur
 as
 nam
 and
 uui
 rem
 the
 sam

 int
s:


 Hoo


 pre
e:
Hook


 pos
te:
eHoo
```

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Subredes	Puede agregar y eliminar subredes de la red. Antes de eliminar una subred, compruebe que ningún recurso de la red utilice la subred.	`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 }`
			`req`
			`nts:`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			rou le: Fre uteT vpc Fre C

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
Grupos de seguridad	Puede agregar y eliminar grupos de seguridad de la red. Antes de eliminar un grupo de seguridad, compruebe que ningún recurso de la red lo utilice.	*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: security_group: Free5GCSecurityGroup01	*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

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
		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 roup sRul pro s:

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			ip_ ol: "to fro : 800 to_ 900 des on: "Eg Rul for fre clu cid "17 0.1/ req nts: sec grou Fre

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

```
curi
up02


Free
curi
upIr
Rule
 #Ne
 Sec
 Gro
 Ing
 Noc

 typ
 tos
es.A
twor
Secu
roup
ssRu

 pro
s:



 ip_
ol:
 "to



 fro
:
 800


 to_
 900
```

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			des on: "Ir Rul for fre clu cid "17 0.1/ rec nts: sec grou Fre curi up02

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu

Interfaces de red	Puede agregarlo, modificarlo y eliminarlo ENIs de la red.	`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`
			
			`-`
			`Fre`

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			curi up01 Fre e5GC : #Mc ENI typ tos es.A twor ENI pro s: dev dex: 3 sou st_c tru req nts: sub Fre ISub se grou

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			- Fre curi up01 Free I03 #Ne ENI typ tos es.A twor ENI pro s: dev dex: 3 rec nts: sub Fre bnet sec grou

Parámetro	Description (Descripción)	Ejemplo: antes	Ejemp Despu
			- Fre curi up01

Actualizar una instancia de red

Console

Para actualizar una instancia de red mediante la consola

1. Abra la consola AWS TNB en https://console.aws.amazon.com/tnb/.
2. En el panel de navegación, elija Redes.
3. Seleccione la instancia de red. Puede actualizar una instancia de red solo si su estado es `Instantiated` o`Updated`.
4. Seleccione Acciones y Actualizar.

   Aparece la página de actualización de la instancia con los detalles de la red y una lista de parámetros de la infraestructura actual.
5. Elige un paquete de red nuevo.

   Los parámetros del nuevo paquete de red aparecen en la sección Parámetros actualizados.
6. Si lo desea, actualice los valores de los parámetros en la sección Parámetros actualizados. Para ver la lista de valores de parámetros que puede actualizar, consulteParámetros que puede actualizar.
7. Seleccione Actualizar red.

   AWS TNB valida la solicitud e inicia el despliegue. Aparece la página de estado del despliegue.

8. Utilice el icono de actualización para realizar un seguimiento del estado de despliegue de la instancia de red. También puede activar la actualización automática en la sección Tareas de despliegue para realizar un seguimiento del progreso de cada tarea.

   Cuando el estado de la implementación cambia a`Completed`, la instancia de red se actualiza.

9.
   - Si se produce un error en la validación, la instancia de red permanece en el mismo estado en el que estaba antes de solicitar la actualización, ya sea `Instantiated` o no`Updated`.
   - Si se produce un error en la actualización, se muestra el estado de la instancia de red`Update failed`. Elija el enlace de cada tarea fallida para determinar el motivo.
   - Si la actualización se realiza correctamente, se muestra `Updated` el estado de la instancia de red.

AWS CLI

Utilice la CLI para actualizar una instancia de red

Use el update-sol-network-instancecomando con el tipo de `UPDATE_NS` actualización para actualizar una instancia de red.

```
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\"}}"
```

Vea una instancia de red en AWS TNB

Obtenga información sobre cómo ver una instancia de red.

Console

Para ver una instancia de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Instancias de red en el panel de navegación.
3. Utilice el cuadro de búsqueda para encontrar la instancia de red.

AWS CLI

Para ver una instancia de red mediante AWS CLI

1. Utilice el list-sol-network-instancescomando para enumerar las instancias de red.

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

2. Use el get-sol-network-instancecomando para ver los detalles de una instancia de red específica.

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

Termine y elimine una instancia de red de AWS TNB

Para eliminar una instancia de red, la instancia debe tener estado finalizado.

Console

Para finalizar y eliminar una instancia de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. En el panel de navegación, elija Redes.
3. Seleccione el identificador de la instancia de red.
4. Elija Finalizar.
5. Cuando se le solicite confirmación, introduzca el identificador y elija Finalizar.
6. Actualice para realizar un seguimiento del estado de la instancia de red.
7. (Opcional) Seleccione la instancia de red y elija Eliminar.

AWS CLI

Para finalizar y eliminar una instancia de red mediante el AWS CLI

1. Utilice el terminate-sol-network-instancecomando para terminar una instancia de red.

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

2. (Opcional) Utilice el delete-sol-network-instancecomando para eliminar una instancia de red.

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

Operaciones de red para AWS TNB

Una operación de red es cualquier operación que se realiza en la red, como la instanciación o la finalización de una instancia de red.

Tareas

- Vea una operación de AWS red TNB
- Cancela una operación AWS de red TNB

Vea una operación de AWS red TNB

Vea los detalles de una operación de red, incluidas las tareas implicadas en la operación de la red y el estado de las tareas.

Console

Para ver una operación de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. Seleccione Instancias de red en el panel de navegación.
3. Utilice el cuadro de búsqueda para encontrar la instancia de red.
4. En la pestaña Despliegues, seleccione la operación de red.

AWS CLI

Para ver una operación de red mediante el AWS CLI

1. Utilice el list-sol-network-operationscomando para enumerar todas las operaciones de red.

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

2. Utilice el get-sol-network-operationcomando para ver los detalles de una operación de red.

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

Cancela una operación AWS de red TNB

Obtenga información acerca de cómo cancelar una operación de red.

Console

Para cancelar una operación de red con la consola

1. Abra la consola AWS TNB en. https://console.aws.amazon.com/tnb/
2. En el panel de navegación, elija Redes.
3. Seleccione el identificador de la red para abrir su página de detalles.
4. En la pestaña Implementaciones, elija la operación de red.
5. Seleccione Cancelar operación.

AWS CLI

Para cancelar una operación de red mediante el AWS CLI

Utilice el cancel-sol-network-operationcomando para cancelar una operación de red.

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

Referencia TOSCA para TNB AWS

La especificación de topología y organización para aplicaciones en la nube (TOSCA) es una sintaxis declarativa que se CSPs utiliza para describir una topología de servicios web basados en la nube, sus componentes, relaciones y los procesos que los gestionan. CSPs describa los puntos de conexión, los enlaces lógicos entre los puntos de conexión y las políticas, como la afinidad y la seguridad, en una plantilla TOSCA. CSPs a continuación, cargue la plantilla en AWS TNB, que sintetiza los recursos necesarios para establecer una red 5G que funcione en todas AWS las zonas de disponibilidad.

Contenido

plantilla VNFD

Define una plantilla de descriptor de funciones de red virtual (VNFD).

Sintaxis

```
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
```

Plantilla de topología

`node_templates`

Los nodos TOSCA. AWS Los posibles nodos son:

- AWS.VNF
- AWS.Artifacts.Helm

AWS.VNF

Define un nodo de función de red AWS virtual (VNF).

Sintaxis

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

Propiedades

`descriptor_id`

El UUID del descriptor.

Obligatorio: sí

Tipo: cadena

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

`descriptor_version`

La versión del VNFD.

Obligatorio: sí

Tipo: cadena

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

`descriptor_name`

El nombre del descriptor.

Obligatorio: sí

Tipo: cadena

`provider`

El autor del VNFD.

Obligatorio: sí

Tipo: cadena

Requisitos

`helm`

El directorio Helm que define los artefactos del contenedor. Esta es una referencia a AWS.Artifacts.Helm.

Obligatorio: sí

Tipo: cadena

Ejemplo

```
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

Define un nodo AWS Helm.

Sintaxis

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

Propiedades

`implementation`

El directorio local que contiene el gráfico de Helm dentro del paquete CSAR.

Obligatorio: sí

Tipo: cadena

Ejemplo

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

Plantilla de descriptor de servicio de red

Define una plantilla de descriptor de servicio de red (NSD).

Sintaxis

```
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
```

Uso de parámetros definidos

Cuando desee transferir dinámicamente un parámetro, como el bloque de CIDR para el nodo de VPC, puede usar la sintaxis `{ get_input: input-parameter-name }` y definir los parámetros en la plantilla de NSD. A continuación, reutilice el parámetro en la misma plantilla de NSD.

En el siguiente ejemplo se muestra cómo definir y utilizar parámetros:

```
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 }
```

Importación de VNFD

`descriptor_id`

El UUID del descriptor.

Obligatorio: sí

Tipo: cadena

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

`namespace`

El nombre único.

Obligatorio: sí

Tipo: cadena

Plantilla de topología

`node_templates`

Los posibles nodos TOSCA son AWS :

- 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

Define un nodo AWS de servicio de red (NS).

Sintaxis

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

Propiedades

`descriptor_id`

El UUID del descriptor.

Obligatorio: sí

Tipo: cadena

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

`descriptor_version`

La versión del NSD.

Obligatorio: sí

Tipo: cadena

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

`descriptor_name`

El nombre del descriptor.

Obligatorio: sí

Tipo: cadena

Ejemplo

```
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

Proporcione el nombre del clúster, la versión de Kubernetes deseada y una función que permita al plano de control de Kubernetes administrar los AWS recursos necesarios para sus NF. Los complementos de la interfaz de red de contenedores (CNI) de Multus están habilitados. Puede conectar varias interfaces de red y aplicar una configuración de red avanzada a las funciones de la red. Kubernetes-based También debe especificar el acceso al punto de conexión del clúster y las subredes del clúster.

Sintaxis

```
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
```

Capacidades

`multus`

Opcional. Propiedades que definen el uso de la interfaz de red de contenedores (CNI) de Multus.

Si incluye `multus`, especifique las propiedades `enabled` y `multus_role`.

`enabled`

Indica si la capacidad de Multus predeterminada está habilitada.

Obligatorio: sí

Tipo: Booleano

`multus_role`

La función de administración de la interfaz de red de Multus.

Obligatorio: sí

Tipo: cadena

`ebs_csi`

Propiedades que definen el controlador de la interfaz de almacenamiento de contenedores (CSI) de Amazon EBS instalado en el clúster de Amazon EKS.

Habilite este complemento para usar nodos autogestionados de Amazon EKS en AWS Outposts, Zonas AWS Locales o Regiones de AWS. Para obtener más información, consulte controlador de CSI de Amazon Elastic Block Store en la Guía del usuario de Amazon EKS.

`enabled`

Indica si el controlador de CSI de Amazon EBS está instalado.

Obligatorio: no

Tipo: booleano

`version`

La versión del complemento de controlador de CSI de Amazon EBS. La versión debe coincidir con una de las versiones devueltas por la DescribeAddonVersionsacción. Para obtener más información, consulte la referencia DescribeAddonVersionsde la API de Amazon EKS

Obligatorio: no

Tipo: cadena

Propiedades

`version`

La versión de Kubernetes para el clúster. AWS Telco Network Builder es compatible con las versiones 1.27 a 1.34 de Kubernetes.

Obligatorio: sí

Tipo: cadena

Valores posibles: 1,27 | 1,28 | 1,29 | 1,30 | 1,31 | 1,32 | 1,33 | 1,34

`access`

El acceso al punto de conexión del clúster.

Obligatorio: sí

Tipo: cadena

Valores posibles: `PRIVATE` | `PUBLIC` | `ALL`

`cluster_role`

El rol de administración de clústeres.

Obligatorio: sí

Tipo: cadena

`tags`

Etiquetas que deben asociarse a este recurso.

Obligatorio: no

Tipo: lista

`ip_family`

Indica la familia de IP de las direcciones de servicio y pod del clúster.

Valor permitido: `IPv4`, `IPv6`

Valor predeterminado: `IPv4`

Obligatorio: no

Tipo: cadena

Requisitos

`subnets`

Y AWS. Networking.Subnetnodo.

Obligatorio: sí

Tipo: lista

Ejemplo

```
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

An AuthRole le permite añadir funciones de IAM al clúster de Amazon EKS para que los usuarios puedan acceder al clúster `aws-auth ConfigMap` de Amazon EKS mediante una función de IAM.

Sintaxis

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

Propiedades

`role_mappings`

Lista de asignaciones que definen los roles de IAM que deben añadirse al clúster de Amazon EKS `aws-auth ConfigMap`.

`arn`

El ARN del rol de IAM.

Obligatorio: sí

Tipo: cadena

`groups`

Grupos de Kubernetes para asignarlos a la función definida en `arn`.

Obligatorio: no

Tipo: lista

Requisitos

`clusters`

Y.AWS Compute.EKSnodo.

Obligatorio: sí

Tipo: lista

Ejemplo

```
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 es compatible con los grupos de nodos gestionados por EKS para automatizar el aprovisionamiento y la administración del ciclo de vida de los nodos (instancias de Amazon EC2) para los clústeres de Amazon EKS Kubernetes. Para crear un grupo de nodos EKS, haga lo siguiente:

- Elija Amazon Machine Images (AMI) para los nodos de trabajo de su clúster proporcionando el ID de la AMI o el tipo de AMI.
- Proporcione un par de claves de Amazon EC2 para el acceso SSH y las propiedades de escalado de su grupo de nodos.
- Asegúrese de que su grupo de nodos esté asociado a un clúster de Amazon EKS.
- Proporcione las subredes de los nodos de trabajo.
- Si lo desea, adjunte grupos de seguridad, etiquetas de nodos y un grupo de ubicación a su grupo de nodos.

Sintaxis

```
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
```

Capacidades

`cálculo`

Propiedades que definen los parámetros de cálculo del grupo de nodos gestionado por Amazon EKS, como los tipos de instancias de Amazon EC2 y las AMI de instancias de Amazon EC2.

`ami_type`

El tipo de Amazon EKS-supported AMI.

Obligatorio: sí

Tipo: cadena

Valores posibles: `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`

Es el ID de la AMI.

Obligatorio: no

Tipo: cadena

> **Note**
>
> Si ambos `ami_type` y `ami_id` se especifican en la plantilla, AWS TNB utilizará solo el `ami_id` valor para crear`EKSManagedNode`.

`instance_types`

El tamaño de la instancia.

Obligatorio: sí

Tipo: lista

`key_pair`

El par de claves EC2 para habilitar el acceso SSH.

Obligatorio: sí

Tipo: cadena

`root_volume_encryption`

Habilita el cifrado de Amazon EBS para el volumen raíz de Amazon EBS. Si no se proporciona esta propiedad, AWS TNB cifra los volúmenes raíz de Amazon EBS de forma predeterminada.

Obligatorio: no

Predeterminado: true

Tipo: Booleano

`root_volume_encryption_key_arn`

El ARN de la AWS KMS clave. AWS TNB admite el ARN de clave normal, el ARN de clave multirregional y el ARN de alias.

Obligatorio: no

Tipo: cadena

> **Note**
>
> - Si `root_volume_encryption` es falso, no lo incluya. `root_volume_encryption_key_arn`
> - AWS TNB admite el cifrado del volumen raíz de las EBS-backed AMI de Amazon.
> - Si el volumen raíz de la AMI ya está cifrado, debe incluirlo `root_volume_encryption_key_arn` para que AWS TNB vuelva a cifrar el volumen raíz.

- Si el volumen raíz de la AMI no está cifrado, AWS TNB lo utiliza `root_volume_encryption_key_arn` para cifrar el volumen raíz.

  Si no la incluye`root_volume_encryption_key_arn`, AWS TNB utilizará la clave predeterminada proporcionada por AWS Key Management Service para cifrar el volumen raíz.
- AWS TNB no descifra una AMI cifrada.

`root_volume_size`

El tamaño del volumen raíz de Amazon Elastic Block Store en GiBs.

Obligatorio: no

Predeterminado: 20

Tipo: entero

Valores posibles: de 1 a 16.384

`escalado`

Propiedades que definen los parámetros de escalado del grupo de nodos gestionado por Amazon EKS, como el número deseado de instancias de Amazon EC2 y el número mínimo y máximo de instancias de Amazon EC2 en el grupo de nodos.

`desired_size`

El número de instancias que contiene. NodeGroup

Obligatorio: sí

Tipo: entero

`min_size`

El número mínimo de instancias que contiene NodeGroup.

Obligatorio: sí

Tipo: entero

`max_size`

El número máximo de instancias que contiene NodeGroup.

Obligatorio: sí

Tipo: entero

Propiedades

`node_role`

El ARN del rol de IAM asociado a la instancia de Amazon EC2.

Obligatorio: sí

Tipo: cadena

`tags`

Las etiquetas que deben asociarse al recurso.

Obligatorio: no

Tipo: lista

`kubernetes_version`

La versión de Kubernetes para el grupo de nodos gestionados. AWS TNB es compatible con las versiones 1.27 a 1.34 de Kubernetes. Considere lo siguiente:

- Especifique la `kubernetes_version` `ami_id` opción o. No especifique ambos.
- `kubernetes_version`Debe ser menor o igual que AWS. Compute.EKSManagedNode versión.
- Puede haber una diferencia de 3 versiones entre AWS. Compute.EKSManagedNode versión y`kubernetes_version`.
- Si no `ami_id` se `kubernetes_version` especifica ni, AWS TNB utilizará la AMI más reciente de la `AWS.Compute.EKSManagedNode` versión para crear `EKSManagedNode`

Obligatorio: no

Tipo: cadena

Valores posibles: 1,27 | 1,28 | 1,29 | 1,30 | 1,31 | 1,32 | 1,33 | 1,34

Requisitos

`cluster`

Y AWS. Compute.EKSnodo.

Obligatorio: sí

Tipo: cadena

`subnets`

Un AWS. Networking.Subnetnodo.

Obligatorio: sí

Tipo: lista

`network_interfaces`

Un AWS. Networking.ENInodo. Asegúrese de que las interfaces de red y las subredes estén configuradas en la misma zona de disponibilidad o se producirá un error en la instanciación.

Cuando se establece`network_interfaces`, AWS TNB obtiene el permiso relacionado con los ENI de la `multus_role` propiedad si se ha incluido la `multus` propiedad en el nodo. AWS.Compute.EKS De lo contrario, AWS TNB obtiene el permiso relacionado con las ENI de la propiedad node_role.

Obligatorio: no

Tipo: lista

`security_groups`

Y AWS. Networking.SecurityGroupnodo.

Obligatorio: no

Tipo: lista

`placement_group`

Un tosca.nodes.AWS. Compute.PlacementGroupnodo.

Obligatorio: no

Tipo: cadena

`user_data`

Un tosca.nodes.AWS. Compute.UserDatareferencia de nodo. Un script de datos de usuario se pasa a las instancias de Amazon EC2 lanzadas por el grupo de nodos administrados. Agregue los permisos necesarios para ejecutar datos de usuario personalizados al node_role pasado al grupo de nodos.

Obligatorio: no

Tipo: cadena

`labels`

Una lista de etiquetas de nodos. Una etiqueta de nodo debe tener un nombre y un valor. Cree una etiqueta con los siguientes criterios:

- El nombre y el valor deben estar separados por=.
- El nombre y el valor pueden tener hasta 63 caracteres cada uno.
- La etiqueta puede incluir letras (A-Z, a-z,), números (0-9) y los siguientes caracteres: `[-, _, ., *, ?]`
- El nombre y el valor deben empezar y terminar con un carácter alfanumérico o un carácter?. *

Por ejemplo, `myLabelName1=*NodeLabelValue1`

Obligatorio: no

Tipo: lista

Ejemplo

```
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 es compatible con los nodos autogestionados de Amazon EKS para automatizar el aprovisionamiento y la administración del ciclo de vida de los nodos (instancias de Amazon EC2) para los clústeres de Kubernetes de Amazon EKS. Para crear un grupo de nodos de Amazon EKS, haga lo siguiente:

- Elija Amazon Machine Images (AMI) para los nodos de trabajadores del clúster proporcionando el ID de la AMI.
- Proporcione un par de claves Amazon EC2 para el acceso SSH.
- Asegúrese de que su grupo de nodos esté asociado a un clúster de Amazon EKS.
- Proporcione el tipo de instancia y los tamaños deseado, mínimo y máximo.
- Proporcione las subredes de los nodos de trabajo.
- Si lo desea, adjunte grupos de seguridad, etiquetas de nodos y un grupo de ubicación a su grupo de nodos.

Sintaxis

```
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
```

Capacidades

cálculo

Propiedades que definen los parámetros de cálculo de los nodos autogestionados de Amazon EKS, como los tipos de instancias de Amazon EC2 y las AMI de instancias de Amazon EC2.

`ami_id`

El ID de AMI utilizado para lanzar la instancia. AWS TNB admite instancias que utilizan IMDSv2. Para obtener más información, consulte Versión IMDS.

> **Note**
>
> Puede actualizar el ID de AMI de`EKSSelfManagedNode`. La versión de Amazon EKS de la AMI debe ser igual o tener hasta 2 versiones anteriores a la versión de clúster de Amazon EKS. Por ejemplo, si la versión del clúster de Amazon EKS es 1.31, la versión de la AMI de Amazon EKS debe ser 1.31, 1.30 o 1.29.

Obligatorio: sí

Tipo: cadena

`instance_type`

El tamaño de la instancia.

Obligatorio: sí

Tipo: cadena

`key_pair`

El par de claves Amazon EC2 para habilitar el acceso SSH.

Obligatorio: sí

Tipo: cadena

`root_volume_encryption`

Habilita el cifrado de Amazon EBS para el volumen raíz de Amazon EBS. Si no se proporciona esta propiedad, AWS TNB cifra los volúmenes raíz de Amazon EBS de forma predeterminada.

Obligatorio: no

Predeterminado: true

Tipo: Booleano

`root_volume_encryption_key_arn`

El ARN de la AWS KMS clave. AWS TNB admite el ARN de clave normal, el ARN de clave multirregional y el ARN de alias.

Obligatorio: no

Tipo: cadena

> **Note**
>
> - Si `root_volume_encryption` es falso, no lo incluya. `root_volume_encryption_key_arn`
> - AWS TNB admite el cifrado del volumen raíz de las EBS-backed AMI de Amazon.
> - Si el volumen raíz de la AMI ya está cifrado, debe incluirlo `root_volume_encryption_key_arn` para que AWS TNB vuelva a cifrar el volumen raíz.
> - Si el volumen raíz de la AMI no está cifrado, AWS TNB lo utiliza `root_volume_encryption_key_arn` para cifrar el volumen raíz.
>
>   Si no lo incluye`root_volume_encryption_key_arn`, AWS TNB lo utiliza AWS Managed Services para cifrar el volumen raíz.
> - AWS TNB no descifra una AMI cifrada.

`root_volume_size`

El tamaño del volumen raíz de Amazon Elastic Block Store en GiBs.

Obligatorio: no

Predeterminado: 20

Tipo: entero

Valores posibles: de 1 a 16.384

escalado

Propiedades que definen los parámetros de escalado de los nodos autogestionados de Amazon EKS, como el número deseado de instancias de Amazon EC2 y el número mínimo y máximo de instancias de Amazon EC2 en el grupo de nodos.

`desired_size`

El número de instancias que contiene. NodeGroup

Obligatorio: sí

Tipo: entero

`min_size`

El número mínimo de instancias que contiene NodeGroup.

Obligatorio: sí

Tipo: entero

`max_size`

El número máximo de instancias que contiene NodeGroup.

Obligatorio: sí

Tipo: entero

Propiedades

`node_role`

El ARN del rol de IAM asociado a la instancia de Amazon EC2.

Obligatorio: sí

Tipo: cadena

`tags`

Las etiquetas que deben asociarse al recurso. Las etiquetas se propagarán a las instancias creadas por el recurso.

Obligatorio: no

Tipo: lista

Requisitos

`cluster`

Un AWS. Compute.EKSnodo.

Obligatorio: sí

Tipo: cadena

`subnets`

Un AWS. Networking.Subnetnodo.

Obligatorio: sí

Tipo: lista

`network_interfaces`

Un AWS. Networking.ENInodo. Asegúrese de que las interfaces de red y las subredes estén configuradas en la misma zona de disponibilidad o se producirá un error en la instanciación.

Cuando se establece`network_interfaces`, AWS TNB obtiene el permiso relacionado con los ENI de la `multus_role` propiedad si se ha incluido la `multus` propiedad en el nodo. AWS.Compute.EKS De lo contrario, AWS TNB obtiene el permiso relacionado con las ENI de la propiedad node_role.

Obligatorio: no

Tipo: lista

`security_groups`

Y AWS. Networking.SecurityGroupnodo.

Obligatorio: no

Tipo: lista

`placement_group`

Un tosca.nodes.AWS. Compute.PlacementGroupnodo.

Obligatorio: no

Tipo: cadena

`user_data`

Un tosca.nodes.AWS. Compute.UserDatareferencia de nodo. Un script de datos de usuario se pasa a las instancias de Amazon EC2 lanzadas por el grupo de nodos autoadministrado. Agregue los permisos necesarios para ejecutar datos de usuario personalizados al node_role pasado al grupo de nodos.

Obligatorio: no

Tipo: cadena

`labels`

Una lista de etiquetas de nodos. Una etiqueta de nodo debe tener un nombre y un valor. Cree una etiqueta con los siguientes criterios:

- El nombre y el valor deben estar separados por=.
- El nombre y el valor pueden tener hasta 63 caracteres cada uno.
- La etiqueta puede incluir letras (A-Z, a-z,), números (0-9) y los siguientes caracteres: `[-, _, ., *, ?]`
- El nombre y el valor deben empezar y terminar con un alfanumérico o un carácter?. *

Por ejemplo, `myLabelName1=*NodeLabelValue1`

Obligatorio: no

Tipo: lista

Ejemplo

```
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

Un PlacementGroup nodo admite diferentes estrategias para colocar las instancias de Amazon EC2.

Cuando se lanza una nueva instancia de Amazon EC2, el servicio de Amazon EC2 intenta colocar la instancia de forma que todas las instancias se distribuyan en el hardware subyacente para minimizar los errores correlacionados. Sin embargo, los grupos de ubicación influyen en la ubicación de un grupo de instancias interdependientes para satisfacer las necesidades de la carga de trabajo.

Sintaxis

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

Propiedades

`strategy`

La estrategia que se utilizará para colocar las instancias de Amazon EC2.

Obligatorio: sí

Tipo: cadena

Valores posibles: CLUSTER | PARTITION | SPREAD_HOST | SPREAD_RACK

- CLUSTER: agrupa las instancias unas cerca de otras dentro de una zona de disponibilidad. Esta estrategia le permite que las cargas de trabajo alcancen el rendimiento de red de baja latencia necesario para una comunicación entre nodos estrechamente acoplada, típica de las aplicaciones de computación de alto rendimiento (HPC).
- PARTITION: distribuye las instancias entre las particiones lógicas de modo que los grupos de instancias de una partición no compartan el hardware subyacente con los grupos de instancias de las demás particiones. Esta estrategia suelen utilizarla grandes cargas de trabajo distribuidas y replicadas, como Hadoop, Cassandra y Kafka.
- SPREAD_RACK: coloca estrictamente un pequeño grupo de instancias en distintos equipos de hardware subyacentes para reducir los fallos correlacionados.
- SPREAD_HOST: solo puede usar con grupos de ubicación de Outpost. Coloca un pequeño grupo de instancias en distintos equipos de hardware subyacentes para reducir los fallos correlacionados.

`partition_count`

El número de particiones.

Obligatorio: obligatorio solo cuando `strategy` está establecido en `PARTITION`.

Tipo: entero

Valores posibles: 1 | 2 |3 | 4 | 5 | 6 | 7

`tags`

Las etiquetas que puede adjuntar al recurso de grupo con ubicación.

Obligatorio: no

Tipo: lista

Ejemplo

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

AWS.Compute.UserData

AWS TNB admite el lanzamiento de instancias de Amazon EC2 con datos de usuario personalizados a través UserData del nodo del Network Service Descriptor (NSD). Para obtener más información sobre los datos de usuario personalizados, consulte Datos de usuario y scripts de shell en la Guía del usuario de Amazon EC2.

Durante la instanciación de la red, AWS TNB proporciona el registro de la instancia de Amazon EC2 al clúster mediante un script de datos de usuario. Cuando también se proporcionan datos de usuario personalizados, AWS TNB combina ambos scripts y los pasa como un script multimime a Amazon EC2. El script de datos de usuario personalizado se ejecuta antes que el script de registro de Amazon EKS.

Para utilizar variables personalizadas en el script de datos de usuario, añada un signo de exclamación `!` después de la llave abierta `{`. Por ejemplo, para utilizar `MyVariable` en el script, introduzca: `{!MyVariable}`

> **Note**
>
> - AWS TNB admite scripts de datos de usuario de hasta 7 KB de tamaño.

- Como AWS TNB procesa y renderiza el script de `multimime` datos de usuario, asegúrese de que el script cumpla con todas las reglas. CloudFormation CloudFormation

Sintaxis

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

Propiedades

`implementation`

La ruta relativa a la definición del script de datos de usuario. El formato debe ser: `./scripts/script_name.sh`

Obligatorio: sí

Tipo: cadena

`content_type`

Tipo de contenido del script de datos de usuario.

Obligatorio: sí

Tipo: cadena

Valores posibles: `x-shellscript`

Ejemplo

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

AWS.Networking.SecurityGroup

AWS TNB admite grupos de seguridad para automatizar el aprovisionamiento de los grupos de seguridad de Amazon EC2 que puede adjuntar a los grupos de nodos del clúster de Amazon EKS Kubernetes.

Sintaxis

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

Propiedades

`description`

La descripción del grupo de seguridad. Puede usar hasta 255 caracteres para describir el grupo. Solo puede incluir letras (A-Z y de la a la z), números (del 0 al 9), espacios y los siguientes caracteres especiales: ._-:/() #, @ [] +=&; {}! $*

Obligatorio: sí

Tipo: cadena

`name`

Un nombre para el grupo de seguridad. Puede utilizar hasta 255 caracteres para el nombre. Solo puede incluir letras (A-Z y de la a la z), números (0-9), espacios y los siguientes caracteres especiales: ._-:/() #, @ [] +=&; {}! $*

Obligatorio: sí

Tipo: cadena

`tags`

Las etiquetas que puede adjuntar al recurso de grupo de seguridad.

Obligatorio: no

Tipo: lista

Requisitos

`vpc`

¿AWS Y?. Networking.VPCnodo.

Obligatorio: sí

Tipo: cadena

Ejemplo

```
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 admite reglas de salida de grupos de seguridad para automatizar el aprovisionamiento de las reglas de salida de grupos de seguridad de Amazon EC2 a las que se pueden adjuntar. AWS Networking.SecurityGroup. Tenga en cuenta que debe proporcionar una cidr_ip/ destination_security_group /destination_prefix_list como destino del tráfico de salida.

Sintaxis

```
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
```

Propiedades

`cidr_ip`

El intervalo de direcciones IPv4 en formato CIDR. Debe especificar un rango de CIDR que permita el tráfico de salida.

Obligatorio: no

Tipo: cadena

`cidr_ipv6`

El intervalo de direcciones IPv6, en formato CIDR, para tráfico de salida. Debe especificar un grupo de seguridad de destino (`destination_security_group` o `destination_prefix_list`) o un rango de CIDR (`cidr_ip` o `cidr_ipv6`).

Obligatorio: no

Tipo: cadena

`description`

Descripción de una regla del grupo de seguridad de salida. Puede usar hasta 255 caracteres para describir la regla.

Obligatorio: no

Tipo: cadena

`destination_prefix_list`

El identificador de lista de prefijos de una lista de prefijos gestionada por Amazon VPC existente. Este es el destino de las instancias del grupo de nodos asociado al grupo de seguridad. Para obtener más información sobre listas de prefijos administrados, consulte la Lista de prefijos administrados en la Guía del usuario de Amazon VPC.

Obligatorio: no

Tipo: cadena

`from_port`

Si el protocolo es TCP o UDP, es el inicio del rango de puertos. Si el protocolo es ICMP o ICMPv6, este es el número de tipo. El valor -1 indica todos los tipos. ICMP/ICMPv6 Si especifica todos los ICMP/ICMPv6 tipos, debe especificar todos los ICMP/ICMPv6 códigos.

Obligatorio: no

Tipo: entero

`ip_protocol`

El nombre del protocolo IP (tcp, udp, icmp, icmpv6) o el número de protocolo. Utilice -1 para especificar todos los protocolos. Al autorizar las reglas del grupo de seguridad, si especifica -1 o un número de protocolo que no sea tcp, udp, icmp o icmpv6 permite el tráfico en todos los puertos, independientemente del rango de puertos que especifique. Para tcp, udp e icmp debe especificar un rango de puertos. Para icmpv6, el rango de puertos es opcional; si se omite el rango de puertos, se permite el tráfico para todos los tipos y códigos.

Obligatorio: sí

Tipo: cadena

`to_port`

Si el protocolo es TCP o UDP, es el final del rango de puertos. Si el protocolo es ICMP o ICMPv6, este es el código. El valor -1 indica todos los ICMP/ICMPv6 códigos. Si especifica todos los ICMP/ICMPv6 tipos, debe especificar todos los ICMP/ICMPv6 códigos.

Obligatorio: no

Tipo: entero

Requisitos

`security_group`

El identificador del grupo de seguridad al que se añade esta regla.

Obligatorio: sí

Tipo: cadena

`destination_security_group`

El identificador o la referencia TOSCA del grupo de seguridad de destino al que se permite el tráfico de salida.

Obligatorio: no

Tipo: cadena

Ejemplo

```
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 admite las reglas de entrada de grupos de seguridad para automatizar el aprovisionamiento de las reglas de entrada de grupos de seguridad de Amazon EC2 que se pueden adjuntar a. AWS Networking.SecurityGroup. Tenga en cuenta que debe proporcionar una cidr_ip/ source_security_group /source_prefix_list como fuente del tráfico de entrada.

Sintaxis

```
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
```

Propiedades

`cidr_ip`

El intervalo de direcciones IPv4 en formato CIDR. Debe especificar un rango de CIDR que permita el tráfico de entrada.

Obligatorio: no

Tipo: cadena

`cidr_ipv6`

El intervalo de direcciones IPv6, en formato CIDR, para tráfico de entrada. Debe especificar un grupo de seguridad de origen (`source_security_group` o `source_prefix_list`) o un rango de CIDR (`cidr_ip` o `cidr_ipv6`).

Obligatorio: no

Tipo: cadena

`description`

La descripción de una regla del grupo de seguridad de entrada. Puede usar hasta 255 caracteres para describir la regla.

Obligatorio: no

Tipo: cadena

`source_prefix_list`

El identificador de lista de prefijos de una lista de prefijos gestionada por Amazon VPC existente. Esta es la fuente desde la que se permitirá recibir tráfico a las instancias del grupo de nodos asociadas al grupo de seguridad. Para obtener más información sobre listas de prefijos administrados, consulte la Lista de prefijos administrados en la Guía del usuario de Amazon VPC.

Obligatorio: no

Tipo: cadena

`from_port`

Si el protocolo es TCP o UDP, es el inicio del rango de puertos. Si el protocolo es ICMP o ICMPv6, este es el número de tipo. Un valor de -1 indica todos los tipos. ICMP/ICMPv6 Si especifica todos los ICMP/ICMPv6 tipos, debe especificar todos los ICMP/ICMPv6 códigos.

Obligatorio: no

Tipo: entero

`ip_protocol`

El nombre del protocolo IP (tcp, udp, icmp, icmpv6) o el número de protocolo. Utilice -1 para especificar todos los protocolos. Al autorizar las reglas del grupo de seguridad, si especifica -1 o un número de protocolo que no sea tcp, udp, icmp o icmpv6 permite el tráfico en todos los puertos, independientemente del rango de puertos que especifique. Para tcp, udp e icmp debe especificar un rango de puertos. Para icmpv6, el rango de puertos es opcional; si se omite el rango de puertos, se permite el tráfico para todos los tipos y códigos.

Obligatorio: sí

Tipo: cadena

`to_port`

Si el protocolo es TCP o UDP, es el final del rango de puertos. Si el protocolo es ICMP o ICMPv6, este es el código. El valor -1 indica todos los ICMP/ICMPv6 códigos. Si especifica todos los ICMP/ICMPv6 tipos, debe especificar todos los ICMP/ICMPv6 códigos.

Obligatorio: no

Tipo: entero

Requisitos

`security_group`

El identificador del grupo de seguridad al que se añade esta regla.

Obligatorio: sí

Tipo: cadena

`source_security_group`

El identificador o la referencia TOSCA del grupo de seguridad de origen desde el que se va a permitir el tráfico de entrada.

Obligatorio: no

Tipo: cadena

Ejemplo

```
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

Puede importar los siguientes AWS recursos a AWS TNB:

- VPC
- Subred
- Tabla de enrutamiento
- Puerta de enlace de Internet
- Security Group

Sintaxis

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

Propiedades

`resource_type`

El tipo de recurso que se importa a AWS TNB.

Obligatorio: no

Tipo: lista

`resource_id`

El identificador del recurso que se importa a AWS TNB.

Obligatorio: no

Tipo: lista

Ejemplo

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

AWS.Networking.ENI

Una interfaz de red es un componente de red lógico en una VPC que representa una tarjeta de red virtual. A una interfaz de red se le asigna una dirección IP automática o manualmente en función de su subred. Tras implementar una instancia de Amazon EC2 en una subred, puede adjuntarle una interfaz de red o desconectar una interfaz de red de esa instancia de Amazon EC2 y volver a conectarla a otra instancia de Amazon EC2 de esa subred. El índice del dispositivo identifica la posición en el orden en que se adjunta.

Sintaxis

```
tosca.nodes.AWS.Networking.ENI:
  properties:
    device_index: Integer
    source_dest_check: Boolean
    tags: List
```

```
  requirements:
    subnet: String
    security_groups: List
```

Propiedades

`device_index`

El índice del dispositivo debe ser mayor que cero.

Obligatorio: sí

Tipo: entero

`source_dest_check`

Indica si la interfaz de red realiza la source/destination comprobación. Un valor de `true` significa que la comprobación está habilitada y un valor de `false` significa que la comprobación está deshabilitada.

Valor permitido: verdadero, falso

Predeterminado: true

Obligatorio: no

Tipo: booleano

`tags`

Las etiquetas que deben asociarse al recurso.

Obligatorio: no

Tipo: lista

Requisitos

`subnet`

Y AWS. Networking.Subnetnodo.

Obligatorio: sí

Tipo: cadena

`security_groups`

Un AWS. Networking.SecurityGroupnodo.

Obligatorio: no

Tipo: cadena

Ejemplo

```
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

Un enlace de ciclo de vida le permite ejecutar sus propios scripts como parte de la instanciación de su infraestructura y red.

Sintaxis

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

Capacidades

`ejecución`

Propiedades del motor de ejecución de enlaces que ejecuta los guiones de enlace.

`type`

Tipo de motor de ejecución de enlaces.

Obligatorio: no

Tipo: cadena

Valores posibles: `CODE_BUILD`

Requisitos

`definition`

Un AWS. HookDefinition.Bashnodo.

Obligatorio: sí

Tipo: cadena

`vpc`

Un AWS. Networking.VPCnodo.

Obligatorio: sí

Tipo: cadena

Ejemplo

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

AWS.Networking.InternetGateway

Define un nodo AWS de Internet Gateway.

Sintaxis

```
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
```

Capacidades

`enrutamiento`

Propiedades que definen la conexión de enrutamiento dentro de la VPC. Debe incluir la propiedad `dest_cidr` o `ipv6_dest_cidr`.

`dest_cidr`

El bloque de CIDR IPv4 utilizado para la coincidencia del destino. Esta propiedad se utiliza para crear una ruta en `RouteTable` y su valor se utiliza como `DestinationCidrBlock`.

Obligatorio: no si se ha incluido la propiedad `ipv6_dest_cidr`.

Tipo: cadena

`ipv6_dest_cidr`

El bloque de CIDR IPv6 utilizado para la coincidencia del destino.

Obligatorio: no si se ha incluido la propiedad `dest_cidr`.

Tipo: cadena

Propiedades

`tags`

Las etiquetas que deben asociarse al recurso.

Obligatorio: no

Tipo: lista

`egress_only`

Una IPv6-specific propiedad. Indica si la puerta de enlace de Internet es solo para la comunicación de salida o no. Si `egress_only` es verdadero, debe definir la propiedad `ipv6_dest_cidr`.

Obligatorio: no

Tipo: booleano

Requisitos

`vpc`

¿Una? AWS. Networking.VPCnodo.

Obligatorio: sí

Tipo: cadena

`route_table`

Un AWS. Networking.RouteTablenodo.

Obligatorio: sí

Tipo: cadena

Ejemplo

```
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

Las tablas de enrutamiento contienen conjuntos de reglas, denominadas rutas, que determinan hacia dónde se dirige el tráfico de red desde las subredes de la VPC o puerta de enlace. Debe asociar una tabla de enrutamiento a una VPC.

Sintaxis

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

Propiedades

`tags`

Etiquetas que deben asociarse a este recurso.

Obligatorio: no

Tipo: lista

Requisitos

`vpc`

Un AWS. Networking.VPCnodo.

Obligatorio: sí

Tipo: cadena

Ejemplo

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

AWS.Networking.Subnet

Una subred es un rango de direcciones IP en su VPC, y debe residir completamente en una zona de disponibilidad. Debe especificar una VPC, un bloque de CIDR, una zona de disponibilidad y una tabla de enrutamiento para la subred. También debe definir si su subred es privada o pública.

Sintaxis

```
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
```

Propiedades

`type`

Indica si las instancias lanzadas en esta subred reciben una dirección IPv4 pública.

Obligatorio: sí

Tipo: cadena

Los valores posibles son: `PUBLIC` | `PRIVATE`

`availability_zone`

La zona de disponibilidad de la subred. Este campo admite las zonas de AWS disponibilidad de una AWS región, por ejemplo `us-west-2` (EE.UU. Oeste (Oregón)). También es compatible con las Zonas AWS Locales dentro de la Zona de Disponibilidad, por ejemplo`us-west-2-lax-1a`.

Obligatorio: sí

Tipo: cadena

`cidr_block`

El bloque de CIDR de la subred.

Obligatorio: no

Tipo: cadena

`ipv6_cidr_block`

El bloque de CIDR que se utiliza para crear la subred IPv6. Si se incluye esta propiedad, no incluya `ipv6_cidr_block_suffix`.

Obligatorio: no

Tipo: cadena

`ipv6_cidr_block_suffix`

El sufijo hexadecimal de 2 dígitos del bloque de CIDR IPv6 para la subred creada a través de Amazon VPC. Use el siguiente formato *`2-digit hexadecimal`*`::/`*`subnetMask`*

Si se incluye esta propiedad, no incluya `ipv6_cidr_block`.

Obligatorio: no

Tipo: cadena

`outpost_arn`

El ARN en el AWS Outposts que se creará la subred. Añada esta propiedad a la plantilla de NSD si desea lanzar nodos autogestionados de Amazon EKS en AWS Outposts. Para obtener más información, consulte Amazon EKS en AWS Outposts en la Guía del usuario de Amazon EKS.

Si añade esta propiedad a la plantilla de NSD, debe establecer el valor de la propiedad `availability_zone` en la Zona de disponibilidad de AWS Outposts.

Obligatorio: no

Tipo: cadena

`tags`

Las etiquetas que deben asociarse al recurso.

Obligatorio: no

Tipo: lista

Requisitos

`vpc`

Un.AWS Networking.VPCnodo.

Obligatorio: sí

Tipo: cadena

`route_table`

Un AWS. Networking.RouteTablenodo.

Obligatorio: sí

Tipo: cadena

Ejemplo

```
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

Las implementaciones de NF se modelan proporcionando la infraestructura y la aplicación asociadas a ellas. El atributo de clúster especifica el clúster de EKS que alojará sus NF. El atributo vnfs especifica las funciones de red de su implementación. También puede proporcionar operaciones opcionales de enlace de ciclo de vida del tipo pre_create y post_create para ejecutar instrucciones específicas de su implementación, como llamar a la API del sistema de gestión de inventario.

Sintaxis

```
tosca.nodes.AWS.Deployment.VNFDeployment:
  requirements:
    deployment: String
    cluster: String
```

```
    vnfs: List
  interfaces:
    Hook:
      pre_create: String
      post_create: String
```

Requisitos

`deployment`

Un AWS. Deployment.VNFDeploymentnodo.

Obligatorio: no

Tipo: cadena

`cluster`

Un AWS. Compute.EKSnodo.

Obligatorio: sí

Tipo: cadena

`vnfs`

Un nodo AWS.VNF.

Obligatorio: sí

Tipo: cadena

Interfaces

Enlaces

Define la etapa en la que se ejecutan los enlaces del ciclo de vida.

`pre_create`

Un AWS. HookExecutionnodo. Este enlace se ejecuta antes de que se implemente el nodo `VNFDeployment`.

Obligatorio: no

Tipo: cadena

`post_create`

Un AWS. HookExecutionnodo. Este enlace se ejecuta después de la implementación del nodo `VNFDeployment`.

Obligatorio: no

Tipo: cadena

Ejemplo

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

AWS.Networking.VPC

Debe especificar un bloque de CIDR para su nube privada virtual (VPC).

Sintaxis

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

Propiedades

`cidr_block`

El rango de red IPv4 para la VPC en la notación de CIDR.

Obligatorio: sí

Tipo: cadena

`ipv6_cidr_block`

El bloque de IPv6 CIDR que se utiliza para crear la VPC.

Valor permitido: `AMAZON_PROVIDED`

Obligatorio: no

Tipo: cadena

`dns_support`

Especifica si las instancias lanzadas en la VPC obtienen nombres de host de DNS.

Obligatorio: no

Tipo: booleano

Valor predeterminado: `false`

`tags`

Etiquetas que deben asociarse a este recurso.

Obligatorio: no

Tipo: lista

Ejemplo

```
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

Puede definir un nodo de puerta de enlace de NAT público o privado a través de una subred. En el caso de una puerta de enlace pública, si no proporciona una ID de asignación de IP elástica, AWS TNB asignará una IP elástica a su cuenta y la asociará a la puerta de enlace.

Sintaxis

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

Propiedades

`subnet`

El AWS. Networking.Subnetreferencia de nodo.

Obligatorio: sí

Tipo: cadena

`internet_gateway`

El AWS. Networking.InternetGatewayreferencia de nodo.

Obligatorio: sí

Tipo: cadena

Propiedades

`type`

Indica si la puerta de enlace es pública o privada.

Valor permitido: `PUBLIC`, `PRIVATE`

Obligatorio: sí

Tipo: cadena

`eip_allocation_id`

El ID que representa la asignación de la dirección IP elástica.

Obligatorio: no

Tipo: cadena

`tags`

Etiquetas que deben asociarse a este recurso.

Obligatorio: no

Tipo: lista

Ejemplo

```
Free5GCNatGateway01:
  type: tosca.nodes.AWS.Networking.NATGateway
    requirements:
       subnet: Free5GCSubnet01
       internet_gateway: Free5GCIGW
    properties:
       type: PUBLIC
       eip_allocation_id: eipalloc-12345
```

AWS.Networking.Route

Puede definir un nodo de ruta que asocie la ruta de destino a la puerta de enlace NAT como recurso de destino y agregue la ruta a la tabla de rutas asociada.

Sintaxis

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

```
    route_table: String
```

Propiedades

`dest_cidr_blocks`

La lista de rutas IPv4 de destino al recurso de destino.

Obligatorio: sí

Tipo: lista

Tipo de miembro: cadena

Requisitos

`nat_gateway`

El AWS. Networking.NATGatewayreferencia de nodo.

Obligatorio: sí

Tipo: cadena

`route_table`

El AWS. Networking.RouteTablereferencia de nodo.

Obligatorio: sí

Tipo: cadena

Ejemplo

```
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

Puede crear parámetros SSM a través de TNB. AWS Los parámetros SSM que cree se crean en SSM y llevan como prefijo el ID de la AWS instancia de red TNB. Esto evita que los valores de los parámetros se anulen cuando se instancian y actualizan varias instancias con la misma plantilla NSD.

Sintaxis

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

Propiedades

Parameters

`name`

El nombre de la propiedad ssm. Use el siguiente formato `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`

El nombre de cada parámetro debe tener menos de 256 caracteres.

Obligatorio: sí

Tipo: cadena

`value`

El valor de la propiedad ssm. Utilice uno de los siguientes formatos:

- Para valores sin referencias: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`
- Para referencias estáticas: `^\$\{[a-zA-Z0-9]+\.(properties|capabilities|requirements)(\.([a-zA-Z0-9\-_]+))+\}$`
- Para referencias dinámicas: `^\$\{[a-zA-Z0-9]+\.(name|id|arn)\}$`

El valor de cada parámetro debe ser inferior a 4 KB.

Obligatorio: sí

Tipo: cadena

`tags`

Las etiquetas que puede adjuntar a una propiedad de SSM.

Obligatorio: no

Tipo: lista

Ejemplo

```
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"
```

Nodos comunes

Defina los nodos para el NSD y el VNFD.

- AWS. HookDefinition.Bash

AWS.HookDefinition.Bash

Define una entrada AWS HookDefinition . `bash`

Sintaxis

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

Propiedades

`implementation`

La ruta relativa a la definición del enlace. El formato debe ser: `./hooks/`*`script_name`*`.sh`

Obligatorio: sí

Tipo: cadena

`environment_variables`

Las variables de entorno del guion bash de enlace. Utilice el siguiente formato: **`envName`**`=`**`envValue`** con los siguientes patrones de expresiones regulares:

- Para valores sin referencias: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`
- Para referencias estáticas: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=\$\{[a-zA-Z0-9]+\.(properties|capabilities|requirements)(\.([a-zA-Z0-9\-_]+))+\}$`
- Para referencias dinámicas: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=\$\{[a-zA-Z0-9]+\.(name|id|arn)\}$`

Asegúrese de que el valor **`envName`**`=`**`envValue`** cumpla los siguientes criterios:

- No utilice espacios.
- Comience **`envName`** con una letra (A-Z o a-z) o un número (0-9).
- No inicie el nombre de la variable de entorno con las siguientes palabras clave reservadas de AWS TNB (no distingue entre mayúsculas y minúsculas):
  - CODEBUILD
  - TNB
  - INICIO

  - AWS
- Puede utilizar cualquier número de letras (A-Z o a-z), números (0-9) y caracteres especiales - y _ para **envName** y **envValue**.
- Cada variable de entorno (cada una **envName** =**envValue**) debe tener menos de 128 caracteres.

Ejemplo: `A123-45xYz=Example_789`

Obligatorio: no

Tipo: lista

`execution_role`

El rol de ejecución de enlaces.

Obligatorio: sí

Tipo: cadena

Ejemplo

```
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"
```

Seguridad en AWS TNB

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, usted se beneficia de los centros de datos y las arquitecturas de red diseñados para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre AWS usted y usted. El https://aws.amazon.com/compliance/shared-responsibility-model/ describe estos conceptos como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la Nube de AWS. AWS también le proporciona servicios que puede utilizar de forma segura. Third-party los auditores prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los AWS programas de de . Para obtener más información sobre los programas de cumplimiento que se aplican a AWS Telco Network Builder, consulte AWS Servicios incluidos en el ámbito de aplicación del programa AWS Servicios incluidos .
- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. También es responsable de otros factores, incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y la normativa aplicables.

Esta documentación le ayuda a comprender cómo aplicar el modelo de responsabilidad compartida al utilizar AWS TNB. Los siguientes temas muestran cómo configurar AWS TNB para cumplir sus objetivos de seguridad y conformidad. También aprenderá a utilizar otros AWS servicios que le ayudan a supervisar y proteger sus recursos de AWS TNB.

Contenido

Protección de datos en AWS TNB

El modelo de se aplica a la protección de datos en AWS Telco Network Builder. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los Nube de AWS. Eres responsable de mantener el control sobre el contenido alojado en esta infraestructura. También eres responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte las Preguntas frecuentes sobre privacidad de datos y los . Para obtener más información sobre la protección de datos en Europa, consulte el Centro del Reglamento General de Protección de Datos (RGPD).

Para fines de protección de datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utiliza la autenticación multifactor (MFA) en cada cuenta.
- Se utiliza SSL/TLS para comunicarse con AWS los recursos. Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad de los usuarios con AWS CloudTrail. Para obtener información sobre el uso de CloudTrail senderos para capturar AWS actividades, consulte Cómo trabajar con CloudTrail senderos en la Guía del AWS CloudTrail usuario.
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utiliza servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger la información confidencial almacenada en Amazon S3.
- Si necesita módulos criptográficos validados por FIPS 140-3 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un punto final FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-3.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye cuando trabaja con AWS TNB u otro tipo Servicios de AWS mediante la consola, la API o los SDK. AWS CLI AWS Cualquier dato que introduzca

en etiquetas o campos de formato libre utilizados para los nombres se pueden emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Gestión de datos

Cuando cierras tu AWS cuenta, AWS TNB marca tus datos para eliminarlos y los elimina para que no se puedan usar. Si reactivas tu AWS cuenta en un plazo de 90 días, AWS TNB restaurará tus datos. Transcurridos 120 días, AWS TNB borra tus datos de forma permanente. AWS TNB también cierra sus redes y elimina sus paquetes de funciones y sus paquetes de red.

Cifrado en reposo

AWS TNB siempre cifra todos los datos almacenados en el servicio en reposo sin requerir ninguna configuración adicional. Este cifrado es totalmente automático. AWS Key Management Service

Cifrado en tránsito

AWS TNB protege todos los datos en tránsito mediante Transport Layer Security (TLS) 1.2.

Es su responsabilidad cifrar los datos entre sus agentes de simulación y sus clientes.

Inter-network privacidad del tráfico

AWS Los recursos de cómputo de TNB residen en una nube privada virtual (VPC) compartida por todos los clientes. Todo el tráfico interno de AWS TNB permanece dentro de la AWS red y no atraviesa Internet. Las conexiones entre sus agentes de simulación y sus clientes se enrutan a través de Internet.

Administración de identidad y acceso para AWS TNB

AWS Identity and Access Management (IAM) es una Servicio de AWS que ayuda al administrador a controlar de forma segura el acceso a los AWS recursos. Los administradores de IAM controlan quién puede autenticarse (iniciar sesión) y quién puede autorizarse (tener permisos) para usar AWS los recursos de TNB. La IAM es una opción Servicio de AWS que puede utilizar sin coste adicional.

Contenido

- Autenticación con identidades
- Administración del acceso con políticas
- Cómo AWS TNB trabaja con IAM
- Identity-based ejemplos de políticas para AWS Creador de redes de telecomunicaciones
- Resolución de problemas AWS Identidad y acceso a Telco Network Builder

Público

La forma de usar AWS Identity and Access Management (IAM) varía según la función que desempeñes:

- Usuario del servicio: solicite permisos al administrador si no puede acceder a las características (consulte Resolución de problemas AWS Identidad y acceso a Telco Network Builder).
- Administrador del servicio: determine el acceso de los usuarios y envíe las solicitudes de permiso (consulte Cómo AWS TNB trabaja con IAM).
- Administrador de IAM: escribe las políticas para administrar el acceso (consulte Identity-based ejemplos de políticas para AWS Creador de redes de telecomunicaciones).

Autenticación con identidades

La autenticación es la forma en que inicias sesión AWS con tus credenciales de identidad. Debe autenticarse como usuario de Usuario raíz de la cuenta de AWS IAM o asumir una función de IAM.

Puede iniciar sesión como una identidad federada con las credenciales de una fuente de identidad, como AWS IAM Identity Center (IAM Identity Center), la autenticación de inicio de sesión único o las credenciales. Google/Facebook Para obtener más información sobre el inicio de sesión, consulte Cómo iniciar sesión en la Cuenta de AWS en la Guía del usuario de AWS Sign-In .

Para el acceso programático, AWS proporciona un SDK y una CLI para firmar criptográficamente las solicitudes. Para obtener más información, consulte AWS Signature Version 4 para solicitudes de API en la Guía del usuario de IAM.

Cuenta de AWS usuario raíz

Al crear una Cuenta de AWS, se comienza con una identidad de inicio de sesión denominada usuario Cuenta de AWS raíz, que tiene acceso completo a todos los Servicios de AWS recursos. Se

recomiendaencarecidamente que no utilice el usuario raíz para las tareas diarias. Para ver las tareas que requieren credenciales de usuario raíz, consulte Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Identidad federada

Como práctica recomendada, exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder Servicios de AWS mediante credenciales temporales.

Una identidad federada es un usuario del directorio empresarial, del proveedor de identidades web o al Directory Service que se accede Servicios de AWS mediante credenciales de una fuente de identidad. Las identidades federadas asumen roles que proporcionan credenciales temporales.

Para una administración de acceso centralizada, se recomienda AWS IAM Identity Center. Para obtener más información, consulte ¿Qué es el Centro de identidades de IAM? en la Guía del usuario de AWS IAM Identity Center .

Usuarios y grupos de IAM

Un usuario de IAM es una identidad con permisos específicos para una sola persona o aplicación. Recomendamos el uso de credenciales temporales en lugar de usuarios de IAM con credenciales de larga duración. Para obtener más información, consulte Exigir a los usuarios humanos que utilicen la federación con un proveedor de identidad para acceder AWS mediante credenciales temporales en la Guía del usuario de IAM.

Un grupo de IAM especifica un conjunto de usuarios de IAM y facilita la administración de los permisos para grupos grandes de usuarios. Para obtener más información, consulte Casos de uso para usuarios de IAM en la Guía del usuario de IAM.

Roles de IAM

Un Rol de IAM es una identidad con permisos específicos que proporciona credenciales temporales. Puede asumir un rol cambiando de un rol de usuario a uno de IAM (consola) o llamando a una AWS CLI operación de AWS API. Para obtener más información, consulte Métodos para asumir un rol en la Guía del usuario de IAM.

Los roles de IAM son útiles para el acceso de usuario federado, los permisos de usuario de IAM temporales, el acceso entre cuentas, el acceso entre servicios y las aplicaciones que se ejecutan en Amazon EC2. Para obtener más información, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Administración del acceso con políticas

El acceso se controla AWS creando políticas y adjuntándolas a AWS identidades o recursos. Una política define los permisos cuando están asociados a una identidad o un recurso. AWS evalúa estas políticas cuando un director hace una solicitud. La mayoría de las políticas se almacenan AWS como documentos JSON. Para obtener más información sobre los documentos de políticas de JSON, consulte Información general de políticas de JSON en la Guía del usuario de IAM.

Mediante las políticas, los administradores especifican quién tiene acceso a qué, definiendo qué entidad principal puede realizar acciones sobre qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Un administrador de IAM crea políticas de IAM y las agrega a roles, que los usuarios pueden asumir posteriormente. Las políticas de IAM definen permisos independientemente del método que se utilice para realizar la operación.

Identity-based políticas

Identity-based las políticas son documentos de política de permisos de JSON que se adjuntan a una identidad (usuario, grupo o rol). Estas políticas controlan qué acciones pueden realizar las identidades, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en la identidad, consulte Definición de permisos de IAM personalizados con políticas administradas por el cliente en la Guía del usuario de IAM.

Identity-based las políticas pueden ser políticas integradas (integradas directamente en una sola identidad) o políticas administradas (políticas independientes asociadas a varias identidades). Para obtener información sobre cómo elegir entre políticas administradas e insertadas, consulte Selección entre políticas administradas y políticas insertadas en la Guía del usuario de IAM.

Resource-based políticas

Resource-based las políticas son documentos de políticas de JSON que se adjuntan a un recurso. Los ejemplos incluyen las Políticas de confianza de roles de IAM y las Políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Debe especificar una entidad principal en una política basada en recursos.

Resource-based las políticas son políticas en línea que se encuentran en ese servicio. No puedes usar políticas AWS gestionadas de IAM en una política basada en recursos.

Otros tipos de políticas

AWS admite tipos de políticas adicionales que pueden establecer los permisos máximos que conceden los tipos de políticas más comunes:

- Límites de permisos: establecen los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM. Para obtener más información, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.
- Políticas de control de servicios (SCP): especifican los permisos máximos para una organización o unidad organizativa en AWS Organizations. Para obtener más información, consulte Políticas de control de servicios en la Guía del usuario de AWS Organizations .
- Políticas de control de recursos (RCP): definen los permisos máximos disponibles para los recursos de las cuentas. Para obtener más información, consulte Políticas de control de recursos (RCP) en la Guía del usuario de AWS Organizations .
- Políticas de sesión: políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal para un rol o un usuario federado. Para obtener más información, consulte Políticas de sesión en la Guía del usuario de IAM.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para saber cómo se AWS determina si se debe permitir una solicitud cuando se trata de varios tipos de políticas, consulte la lógica de evaluación de políticas en la Guía del usuario de IAM.

Cómo AWS TNB trabaja con IAM

Antes de utilizar IAM para gestionar el acceso a la AWS TNB, infórmese sobre las funciones de IAM disponibles para su uso con la TNB. AWS

Funciones de IAM que puede utilizar con AWS Telco Network Builder

Característica de IAM	AWS Soporte TNB
Identity-based políticas	Sí
Resource-based políticas	No

Característica de IAM	AWS Soporte TNB
Acciones de políticas	Sí
Recursos de políticas	Sí
Claves de condición de política	Sí
ACL	No
ABAC (etiquetas en políticas)	Sí
Credenciales temporales	Sí
Permisos de entidades principales	Sí
Roles de servicio	No
Service-linked roles	No

Para obtener una visión general de cómo funcionan AWS TNB y otros AWS servicios con la mayoría de las funciones de IAM, consulte AWS los servicios que funcionan con IAM en la Guía del usuario de IAM.

Identity-based políticas para AWS TNB

Compatibilidad con las políticas basadas en identidad: sí

Identity-based las políticas son documentos de política de permisos de JSON que puedes adjuntar a una identidad, como un usuario, un grupo de usuarios o un rol de IAM. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en la identidad, consulte Definición de permisos de IAM personalizados con políticas administradas por el cliente en la Guía del usuario de IAM.

Con las políticas basadas en identidades de IAM, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. Para obtener más información sobre los elementos que puede utilizar en una política de JSON, consulte Referencia de los elementos de la política de JSON de IAM en la Guía del usuario de IAM.

Identity-based ejemplos de políticas para AWS TNB

Para ver ejemplos de políticas de AWS TNB basadas en la identidad, consulte. Identity-based ejemplos de políticas para AWS Creador de redes de telecomunicaciones

Resource-based políticas dentro AWS TNB

Admite políticas basadas en recursos: no

Resource-based las políticas son documentos de políticas de JSON que se adjuntan a un recurso. Los ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política basada en recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Para habilitar el acceso entre cuentas, puede especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política en función de recursos. Para obtener más información, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Acciones políticas para AWS TNB

Compatibilidad con las acciones de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Para ver una lista de las acciones de AWS TNB, consulte las acciones definidas por AWS Telco Network Builder en la Referencia de autorización de servicios.

Las acciones políticas en AWS TNB utilizan el siguiente prefijo antes de la acción:

```
tnb
```

Para especificar varias acciones en una única instrucción, sepárelas con comas.

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

Puede utilizar caracteres comodín (*) para especificar varias acciones . Por ejemplo, para especificar todas las acciones que comiencen con la palabra `List`, incluya la siguiente acción:

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

Para ver ejemplos de políticas de AWS TNB basadas en la identidad, consulte. Identity-based ejemplos de políticas para AWS Creador de redes de telecomunicaciones

Recursos de políticas para AWS TNB

Compatibilidad con los recursos de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). En el caso de las acciones que no admiten permisos por recurso, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

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

Para ver una lista de los tipos de recursos de AWS TNB y sus ARN, consulte los recursos definidos por AWS Telco Network Builder en la Referencia de autorización de servicios. Para saber con qué acciones puede especificar el ARN de cada recurso, consulte Acciones definidas por AWS Telco Network Builder.

Para ver ejemplos de políticas de AWS TNB basadas en la identidad, consulte. Identity-based ejemplos de políticas para AWS Creador de redes de telecomunicaciones

Claves de condición de la política para AWS TNB

Compatibilidad con claves de condición de políticas específicas del servicio: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Condition` especifica cuándo se ejecutan las instrucciones en función de criterios definidos. Puede crear expresiones condicionales que utilizan operadores de condición, tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del usuario de IAM.

Para ver una lista de las claves de condición AWS TNB, consulte las claves de condición de AWS Telco Network Builder en la Referencia de autorización de servicios. Para saber con qué acciones y recursos puede utilizar una clave de condición, consulte Acciones definidas por AWS Telco Network Builder.

Para ver ejemplos de políticas de AWS TNB basadas en la identidad, consulte. Identity-based ejemplos de políticas para AWS Creador de redes de telecomunicaciones

ACL en AWS TNB

Compatibilidad con ACL: no

Las listas de control de acceso (ACL) controlan qué entidades principales (miembros de cuentas, usuarios o roles) tienen permisos para acceder a un recurso. Las ACL son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

ABAC con AWS TNB

Admite ABAC (etiquetas en las políticas): sí

Attribute-based el control de acceso (ABAC) es una estrategia de autorización que define los permisos en función de unos atributos denominados etiquetas. Puede adjuntar etiquetas a las entidades y AWS los recursos de IAM y, a continuación, diseñar políticas de ABAC para permitir las operaciones cuando la etiqueta del director coincida con la etiqueta del recurso.

Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`.

Si un servicio admite las tres claves de condición para cada tipo de recurso, el valor es Sí para el servicio. Si un servicio admite las tres claves de condición solo para algunos tipos de recursos, el valor es Parcial.

Para obtener más información sobre ABAC, consulte Definición de permisos con la autorización de ABAC en la Guía del usuario de IAM. Para ver un tutorial con los pasos para configurar ABAC, consulte Uso del control de acceso basado en atributos (ABAC) en la Guía del usuario de IAM.

Utilizar credenciales temporales con AWS TNB

Compatibilidad con credenciales temporales: sí

Las credenciales temporales proporcionan acceso a AWS los recursos a corto plazo y se crean automáticamente al utilizar la federación o al cambiar de función. AWS recomienda generar credenciales temporales de forma dinámica en lugar de utilizar claves de acceso a largo plazo. Para obtener más información, consulte Credenciales de seguridad temporales en IAM y Servicios de AWS que funcionan con IAM en la Guía del usuario de IAM.

Cross-service permisos principales para AWS TNB

Admite sesiones de acceso directo (FAS): sí

Las sesiones de acceso directo (FAS) utilizan los permisos de la persona principal que llama y la que solicita Servicio de AWS para realizar solicitudes a los servicios descendentes. Servicio de AWS Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Sesiones de acceso directo.

Funciones de servicio para AWS TNB

Compatible con roles de servicio: No

Un rol de servicio es un rol de IAM que asume un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Crear un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.

Service-linked funciones para AWS TNB

Compatibilidad con roles vinculados al servicio: no

Un rol vinculado a un servicio es un tipo de rol de servicio que está vinculado a un. Servicio de AWS El servicio puede asumir la función de realizar una acción en su nombre. Service-linked las funciones aparecen en su nombre Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

Identity-based ejemplos de políticas para AWS Creador de redes de telecomunicaciones

De forma predeterminada, los usuarios y los roles no tienen permiso para crear o modificar los recursos de AWS TNB. Un administrador de IAM puede crear políticas de IAM para conceder permisos a los usuarios para realizar acciones en los recursos que necesitan.

Para obtener información acerca de cómo crear una política basada en identidades de IAM mediante el uso de estos documentos de políticas JSON de ejemplo, consulte Creación de políticas de IAM (consola) en la Guía del usuario de IAM.

Para obtener más información sobre las acciones y los tipos de recursos definidos por AWS TNB, incluido el formato de los ARN de cada uno de los tipos de recursos, consulte Acciones, recursos y claves de condición de AWS Telco Network Builder en la Referencia de autorización de servicios.

Contenido

Prácticas recomendadas sobre las políticas

Identity-based las políticas determinan si alguien puede crear, acceder o eliminar los recursos de AWS TNB de su cuenta. Estas acciones pueden generar costos adicionales para su Cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comience con las políticas AWS administradas y avance hacia los permisos con privilegios mínimos: para empezar a conceder permisos a sus usuarios y cargas de trabajo, utilice las políticas AWS administradas que otorgan permisos para muchos casos de uso comunes. Están disponibles en su. Cuenta de AWS Le recomendamos que reduzca aún más los permisos definiendo políticas administradas por el AWS cliente que sean específicas para sus casos de uso. Con el fin de obtener más información, consulte las políticas administradas por AWS o las políticas administradas por AWS para funciones de tarea en la Guía de usuario de IAM.
- Aplique permisos de privilegio mínimo: cuando establezca permisos con políticas de IAM, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se

pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Con el fin de obtener más información sobre el uso de IAM para aplicar permisos, consulte Políticas y permisos en IAM en la Guía del usuario de IAM.

- Utilice condiciones en las políticas de IAM para restringir aún más el acceso: puede agregar una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puede escribir una condición de políticas para especificar que todas las solicitudes deben enviarse utilizando SSL. También puedes usar condiciones para conceder el acceso a las acciones del servicio si se utilizan a través de una acción específica Servicio de AWS, por ejemplo CloudFormation. Para obtener más información, consulte Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.
- Utiliza el analizador de acceso de IAM para validar las políticas de IAM con el fin de garantizar la seguridad y funcionalidad de los permisos: el analizador de acceso de IAM valida políticas nuevas y existentes para que respeten el lenguaje (JSON) de las políticas de IAM y las prácticas recomendadas de IAM. El analizador de acceso de IAM proporciona más de 100 verificaciones de políticas y recomendaciones procesables para ayudar a crear políticas seguras y funcionales. Para más información, consulte Validación de políticas con el Analizador de acceso de IAM en la Guía del usuario de IAM.
- Requerir autenticación multifactor (MFA): si tiene un escenario que requiere usuarios de IAM o un usuario raíz en Cuenta de AWS su cuenta, active la MFA para mayor seguridad. Para exigir la MFA cuando se invoquen las operaciones de la API, añada condiciones de MFA a sus políticas. Para más información, consulte Acceso seguro a la API con MFA en la Guía del usuario de IAM.

Para obtener más información sobre las prácticas recomendadas de IAM, consulte Prácticas recomendadas de seguridad en IAM en la Guía del usuario de IAM.

Uso de AWS Consola TNB

Para acceder a la consola de AWS Telco Network Builder, debe tener un conjunto mínimo de permisos. Estos permisos deben permitirle enumerar y ver detalles sobre los recursos de AWS TNB de su propiedad. Cuenta de AWS Si crea una política basada en identidades que sea más restrictiva que el mínimo de permisos necesarios, la consola no funcionará del modo esperado para las entidades (usuarios o roles) que tengan esa política.

No es necesario que concedas permisos mínimos de consola a los usuarios que solo realicen llamadas a la API AWS CLI o a la AWS API. En su lugar, permita el acceso únicamente a las acciones que coincidan con la operación de API que intentan realizar.

Ejemplos de políticas de roles de servicio

Como administrador, usted es propietario y administra los recursos que AWS TNB crea, tal como se definen en las plantillas de entorno y servicio. Debe asociar las funciones de servicio de IAM a su cuenta para que AWS TNB pueda crear recursos para la administración del ciclo de vida de la red.

Una función de servicio de IAM permite a AWS TNB realizar llamadas a los recursos en su nombre para crear instancias de sus redes y administrarlas. Si especificas un rol de servicio, AWS TNB usa la credencial de ese rol.

Puede crear el rol de servicio y su política de permisos con el servicio de IAM. Para obtener más información sobre la creación de un rol de servicio, consulte Crear un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM.

AWS Función de servicio TNB

Como miembro del equipo de la plataforma, como administrador puede crear un rol de servicio de AWS TNB y asignárselo a AWS TNB. Esta función permite a AWS TNB realizar llamadas a otros servicios, como Amazon Elastic Kubernetes CloudFormation Service, y aprovisionar la infraestructura necesaria para su red y aprovisionar las funciones de red tal como se definen en su NSD.

Se recomienda utilizar el siguiente rol de IAM y la política de confianza para el rol de servicio de AWS TNB. Al determinar el alcance de los permisos de esta política, tenga en cuenta que AWS TNB puede fallar y provocar errores de acceso denegado en relación con los recursos no incluidos en su política.

El siguiente código muestra una política de funciones de servicio de AWS TNB:

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"
        }
    ]
}
```

El siguiente código muestra la política de confianza del servicio de AWS TNB:

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 Función de servicio TNB para el clúster Amazon EKS

Al crear un recurso de Amazon EKS en su NSD, proporciona el atributo `cluster_role` para especificar qué rol se utilizará para crear su clúster de Amazon EKS.

El siguiente ejemplo muestra una AWS CloudFormation plantilla que crea un rol de servicio AWS TNB para la política de clústeres de Amazon EKS.

```
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"
```

Para obtener más información sobre los roles de IAM que utilizan AWS CloudFormation plantillas, consulte las siguientes secciones de la Guía del AWS CloudFormation usuario:

- AWS::IAM::Role
- Selección de una plantilla de pila

AWS Función de servicio TNB para el grupo de nodos Amazon EKS

Al crear recursos de un grupo de nodos de Amazon EKS en su NSD, proporciona el atributo `node_role` para especificar qué rol se utilizará para crear su grupo de nodos de Amazon EKS.

El siguiente ejemplo muestra una CloudFormation plantilla que crea un rol de servicio AWS TNB para la política de grupo de nodos de Amazon EKS.

```
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/*"
```

Para obtener más información sobre los roles de IAM que utilizan AWS CloudFormation plantillas, consulte las siguientes secciones de la Guía del AWS CloudFormation usuario:

- AWS::IAM::Role
- Selección de una plantilla de pila

AWS Función de servicio TNB para Multus

Cuando cree un recurso de Amazon EKS en su NSD y desee administrar Multus como parte de su plantilla de implementación, debe proporcionar el atributo `multus_role` para especificar qué rol se utilizará para administrar Multus.

El siguiente ejemplo muestra una CloudFormation plantilla que crea un rol de servicio de AWS TNB para una política de Multus.

```
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: "*"
```

Para obtener más información sobre las funciones de IAM que utilizan AWS CloudFormation plantillas, consulte las siguientes secciones de la Guía del AWS CloudFormation usuario:

- AWS::IAM::Role
- Selección de una plantilla de pila

AWS Función de servicio de TNB para una política vinculada al ciclo de vida

Cuando su NSD o paquete de funciones de red utiliza un enlace de ciclo de vida, necesita un rol de servicio que le permita crear un entorno para la ejecución de sus enlaces de ciclo de vida.

> **Note**
>
> Su política de enlace de ciclo de vida debe basarse en lo que intente hacer su enlace de ciclo de vida.

El siguiente ejemplo muestra una CloudFormation plantilla que crea un rol de servicio de AWS TNB para una política de enlace de ciclo de vida.

```
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"
```

Para obtener más información sobre las funciones de IAM que utilizan AWS CloudFormation plantillas, consulte las siguientes secciones de la Guía del AWS CloudFormation usuario:

- AWS::IAM::Role
- Selección de una plantilla de pila

Cómo permitir a los usuarios consultar sus propios permisos

En este ejemplo, se muestra cómo podría crear una política que permita a los usuarios de IAM ver las políticas administradas e insertadas que se asocian a la identidad de sus usuarios. Esta política incluye permisos para completar esta acción en la consola o mediante programación mediante la AWS CLI API o. AWS

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

Resolución de problemas AWS Identidad y acceso a Telco Network Builder

Utilice la siguiente información como ayuda para diagnosticar y solucionar problemas comunes que pueden surgir al trabajar con AWS TNB e IAM.

Problemas

- No estoy autorizado a realizar ninguna acción en AWS TNB
- No estoy autorizado a realizar tareas como: PassRole
- Quiero permitir que personas ajenas a mi Cuenta de AWS para acceder a mi AWS recursos de TNB

No estoy autorizado a realizar ninguna acción en AWS TNB

Si recibe un error que indica que no tiene autorización para realizar una acción, las políticas se deben actualizar para permitirle realizar la acción.

En el siguiente ejemplo, el error se produce cuando el usuario de IAM `mateojackson` intenta utilizar la consola para consultar los detalles acerca de un recurso ficticio *`my-example-widget`*, pero no tiene los permisos ficticios `tnb:`*`GetWidget`*.

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

En este caso, la política de Mateo se debe actualizar para permitirle acceder al recurso *`my-example-widget`* mediante la acción `tnb:`*`GetWidget`*.

Si necesita ayuda, póngase en contacto con su AWS administrador. El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

No estoy autorizado a realizar tareas como: PassRole

Si recibes un mensaje de error que indica que no estás autorizado a realizar la `iam:PassRole` acción, debes actualizar tus políticas para que puedas transferir una función a AWS TNB.

Algunas Servicios de AWS permiten transferir una función existente a ese servicio en lugar de crear una nueva función de servicio o una función vinculada al servicio. Para ello, debe tener permisos para transferir la función al servicio.

En el siguiente ejemplo, el error se produce cuando un usuario de IAM denominado `marymajor` intenta utilizar la consola para realizar una acción en AWS TNB. Sin embargo, la acción requiere que el servicio cuente con permisos que otorguen un rol de servicio. Mary no tiene permisos para transferir el rol al servicio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

En este caso, las políticas de Mary se deben actualizar para permitirle realizar la acción `iam:PassRole`.

Si necesita ayuda, póngase en contacto con su AWS administrador. El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

Quiero permitir que personas ajenas a mi Cuenta de AWS para acceder a mi AWS recursos de TNB

Se puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Se puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que admitan las políticas basadas en recursos o las listas de control de acceso (ACL), puede utilizar dichas políticas para conceder a las personas acceso a sus recursos.

Para obtener más información, consulte lo siguiente:

- Para saber si AWS TNB admite estas funciones, consulte. Cómo AWS TNB trabaja con IAM
- Para obtener información sobre cómo proporcionar acceso a los recursos de su Cuentas de AWS propiedad, consulte Proporcionar acceso a un usuario de IAM en otro usuario de su propiedad Cuenta de AWS en la Guía del usuario de IAM.

- Para obtener información sobre cómo proporcionar acceso a tus recursos a terceros Cuentas de AWS, consulta Cómo proporcionar acceso a recursos que Cuentas de AWS son propiedad de terceros en la Guía del usuario de IAM.
- Para obtener información sobre cómo proporcionar acceso mediante una federación de identidades, consulte Proporcionar acceso a usuarios autenticados externamente (identidad federada) en la Guía del usuario de IAM.
- Para conocer sobre la diferencia entre las políticas basadas en roles y en recursos para el acceso entre cuentas, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Validación de conformidad para AWS TNB

Para saber si un programa de cumplimiento Servicio de AWS está dentro del ámbito de aplicación de programas de cumplimiento específicos, consulte Servicios de AWS Alcance por programa de cumplimiento Servicios de AWS y elija el programa de cumplimiento que le interese. Para obtener información general, consulte Programas de AWS cumplimiento > Programas AWS .

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact .

Su responsabilidad de cumplimiento al Servicios de AWS utilizarlos viene determinada por la confidencialidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y reglamentos aplicables. Para obtener más información sobre su responsabilidad de conformidad al utilizarlos Servicios de AWS, consulte AWS la documentación de seguridad.

Resiliencia en AWS TNB

La infraestructura AWS global se basa en zonas Regiones de AWS de disponibilidad. Regiones de AWS proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre las zonas sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de uno o varios centros de datos.

Para obtener más información sobre las zonas de disponibilidad Regiones de AWS y las zonas de disponibilidad, consulte Infraestructura global.AWS

AWS TNB ejecuta su servicio de red en clústeres de EKS en una nube privada virtual (VPC) en AWS la región que elija.

Seguridad de la infraestructura en AWS TNB

Como servicio gestionado, AWS Telco Network Builder está protegido por la seguridad de la red AWS global. Para obtener información sobre los servicios AWS de seguridad y cómo se AWS protege la infraestructura, consulte Seguridad AWS en la nube. Para diseñar su AWS entorno utilizando las mejores prácticas de seguridad de la infraestructura, consulte Protección de infraestructuras en un marco de buena AWS arquitectura basado en el pilar de la seguridad.

Utiliza las llamadas a la API AWS publicadas para acceder a AWS TNB a través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Paquetes de cifrado con perfecto secreto directo (PFS), como el DHE (Ephemeral) o el ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

A continuación, se muestran algunos ejemplos de responsabilidades compartidas:

- AWS es responsable de proteger los componentes compatibles con TNB, entre los que se incluyen: AWS
  - Instancias de cómputo (también conocidas como trabajadores)
  - Base de datos interna
  - Comunicaciones de red entre componentes internos
  - La interfaz de programación de aplicaciones (API) de AWS TNB
  - AWS Kits de desarrollo de software (SDK)
- Usted es responsable de proteger el acceso a sus AWS recursos y a los componentes de la carga de trabajo, incluidos (entre otros):
  - Usuarios, grupos, roles y políticas de IAM
  - Depósitos de S3 que utiliza para almacenar sus datos para TNB AWS
  - Otros Servicios de AWS recursos que utiliza para respaldar el servicio de red que aprovisionó a través de TNB AWS
  - Su código de la aplicación

- Conexiones entre el servicio de red que aprovisionó a través de AWS TNB y sus clientes

> **Important**
>
> Usted es responsable de implementar un plan de recuperación ante desastres que pueda recuperar de manera efectiva un servicio de red que haya aprovisionado a través de TNB. AWS

Modelo de seguridad de la conectividad de red

Los servicios de red que aprovisiona a través de AWS TNB se ejecutan en instancias de procesamiento dentro de una nube privada virtual (VPC) ubicada en AWS la región que seleccione. Una VPC es una red virtual en la AWS nube que aísla la infraestructura por carga de trabajo o entidad organizativa. La comunicación entre las instancias informáticas de la VPC permanecen dentro de la red de AWS y no circulan por Internet. Algunas comunicaciones de los servicios internos se transmiten por Internet y están cifradas. Los servicios de red aprovisionados a través de AWS TNB para todos los clientes que se ejecutan en la misma región comparten la misma VPC. Los servicios de red aprovisionados a través de AWS TNB para diferentes clientes utilizan instancias informáticas independientes dentro de la misma VPC.

Las comunicaciones entre sus clientes de servicio de red y su servicio de red en AWS TNB atraviesan Internet. AWS TNB no gestiona estas conexiones. Es su responsabilidad proteger las conexiones de sus clientes.

Sus conexiones a AWS TNB a través de los AWS SDK Consola de administración de AWS, AWS Command Line Interface (AWS CLI) y están cifradas.

Versión IMDS

AWS TNB admite instancias que utilizan la versión 2 del Servicio de metadatos de instancias (IMDSv2), un método orientado a la sesión. IMDSv2 incluye una mayor seguridad que IMDSV1. Para obtener más información, consulte Agregar defensa en profundidad contra firewalls abiertos, proxies inversos y vulnerabilidades SSRF con mejoras en el servicio de metadatos de instancias EC2 de Amazon.

Al lanzar la instancia, debe usar IMDSv2. Para obtener más información sobre IMDSv2, consulte Utilizar IMDSv2 en la Guía del usuario de Amazon EC2.

Monitorización de AWS TNB

El monitoreo es una parte importante del mantenimiento de la confiabilidad, la disponibilidad y el rendimiento de AWS TNB y sus demás AWS soluciones. AWS permite AWS CloudTrail vigilar AWS TNB, informar cuando algo va mal y tomar medidas automáticas cuando sea apropiado.

Se utiliza CloudTrail para capturar información detallada sobre las llamadas realizadas a AWS APIs. Puede almacenar estas llamadas como archivos de registro en Amazon S3. Puede usar estos CloudTrail registros para determinar información como qué llamada se realizó, la dirección IP de origen de la llamada, quién hizo la llamada y cuándo se realizó la llamada.

Los CloudTrail registros contienen información sobre las llamadas a las acciones de la API para AWS TNB. También contienen información sobre las llamadas a las acciones de la API desde servicios como Amazon EC2 y Amazon EBS.

Registro de llamadas a la API de AWS Telco Network Builder mediante AWS CloudTrail

AWS Telco Network Builder está integrado con AWS CloudTrailun servicio que proporciona un registro de las acciones realizadas por un usuario, rol o un. Servicio de AWS CloudTrail captura todas las llamadas a la API de AWS TNB como eventos. Las llamadas capturadas incluyen llamadas desde la consola de AWS TNB y llamadas en código a las operaciones de la API de AWS TNB. Con la información recopilada por TNB CloudTrail, puede determinar la solicitud que se realizó a AWS TNB, la dirección IP desde la que se realizó la solicitud, cuándo se realizó y detalles adicionales.

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con las credenciales del usuario raíz o del usuario.
- Si la solicitud se realizó en nombre de un usuario de IAM Identity Center.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro Servicio de AWS.

CloudTrail está activa en tu cuenta Cuenta de AWS cuando creas la cuenta y tienes acceso automáticamente al historial de CloudTrail eventos. El historial de CloudTrail eventos proporciona un

registro visible, consultable, descargable e inmutable de los últimos 90 días de eventos de gestión registrados en un. Región de AWSPara obtener más información, consulte Uso del historial de CloudTrail eventos en la Guía del usuario.AWS CloudTrail La visualización del historial de eventos no conlleva ningún CloudTrail cargo.

Para tener un registro continuo de los eventos de Cuenta de AWS los últimos 90 días, crea un almacén de datos de eventos de senderos o CloudTraillagos.

CloudTrail senderos

Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. Todos los senderos creados con él Consola de administración de AWS son multirregionales. Puede crear un registro de seguimiento de una sola región o multirregionales mediante la AWS CLI. Se recomienda crear un sendero multirregional, ya que puedes capturar toda la actividad de tu Regiones de AWS cuenta. Si crea un registro de seguimiento de una sola región, solo podrá ver los eventos registrados en la Región de AWS del registro de seguimiento. Para obtener más información acerca de los registros de seguimiento, consulte Creación de un registro de seguimiento para su Cuenta de AWS y Creación de un registro de seguimiento para una organización en la Guía del usuario de AWS CloudTrail .

Puede enviar una copia de sus eventos de administración en curso a su bucket de Amazon S3 sin coste alguno CloudTrail mediante la creación de una ruta; sin embargo, hay cargos por almacenamiento en Amazon S3. Para obtener más información sobre CloudTrail los precios, consulte AWS CloudTrail Precios. Para obtener información acerca de los precios de Amazon S3, consulte Precios de Amazon S3.

CloudTrail Almacenes de datos de eventos en Lake

CloudTrail Lake le permite ejecutar consultas basadas en SQL en sus eventos. CloudTrail Lake convierte los eventos existentes en formato JSON basado en filas al formato Apache ORC. ORC es un formato de almacenamiento en columnas optimizado para una recuperación rápida de datos. Los eventos se agregan en almacenes de datos de eventos, que son recopilaciones inmutables de eventos en función de criterios que se seleccionan aplicando selectores de eventos avanzados. Los selectores que se aplican a un almacén de datos de eventos controlan los eventos que perduran y están disponibles para la consulta. Para obtener más información sobre CloudTrail Lake, consulte Cómo trabajar con AWS CloudTrail Lake en la Guía del AWS CloudTrail usuario.

CloudTrail Los almacenes de datos y las consultas sobre eventos de Lake conllevan costes. Cuando crea un almacén de datos de eventos, debe elegir la opción de precios que desee utilizar

para él. La opción de precios determina el costo de la incorporación y el almacenamiento de los eventos, así como el período de retención predeterminado y máximo del almacén de datos de eventos. Para obtener más información sobre CloudTrail los precios, consulte AWS CloudTrail Precios.

AWS Ejemplos de eventos de TNB

Un evento representa una solicitud única de cualquier fuente e incluye información sobre la operación de API solicitada, la fecha y la hora de la operación, los parámetros de la solicitud, etc. CloudTrail Los archivos de registro no son un registro ordenado de las llamadas a la API pública, por lo que los eventos no aparecen en ningún orden específico.

En el siguiente ejemplo, se muestra un CloudTrail evento que demuestra la `CreateSolFunctionPackage` operación.


```
{
    "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"
}
```


Para obtener información sobre el contenido de los CloudTrail registros, consulte el contenido de los CloudTrail registros en la Guía del AWS CloudTrail usuario.

AWS Tareas de despliegue de TNB

Comprenda las tareas de implementación para supervisar las implementaciones de manera efectiva y tomar medidas más rápido.

En la siguiente tabla se enumeran las tareas de despliegue de AWS TNB:

Nombre de la tarea para las implement aciones iniciadas antes del 7 de marzo de 2024	Nombre de la tarea para las implement aciones que se iniciaron a partir del 7 de marzo de 2024	Task description
AppInstallation	ClusterPluginInstall	Instala el complemento Multus en el clúster de Amazon EKS.

Nombre de la tarea para las implement aciones iniciadas antes del 7 de marzo de 2024	Nombre de la tarea para las implement aciones que se iniciaron a partir del 7 de marzo de 2024	Task description
AppUpdate	sin cambio de nombre	Actualiza las funciones de red que ya están instaladas en una instancia de red.
-	ClusterPluginUninstall	Desinstala los complementos en el clúster de Amazon EKS.
ClusterStorageClas sConfiguration	sin cambios de nombre	Configura la clase de almacenamiento (controla dor de CSI) en un clúster de Amazon EKS.
FunctionDeletion	sin cambio de nombre	Elimina las funciones de red de los recursos de AWS TNB.
FunctionInstantiation	FunctionInstall	Implementa funciones de red mediante HELM.
FunctionUninstallation	FunctionUninstall	Desinstala la función de red de un clúster de Amazon EKS.
HookExecution	sin cambios de nombre	Ejecuta los enlaces de ciclo de vida tal como se define en el NSD.
InfrastructureCanc ellation	sin cambio de nombre	Cancela un servicio de red.
InfrastructureInst antation	sin cambio de nombre	Aprovisiona AWS recursos en nombre del usuario.
InfrastructureTerm ination	sin cambios de nombre	Desaprovisiona AWS los recursos invocados a través de AWS TNB.
-	InfrastructureUpdate	Actualiza los AWS recursos aprovisionados en nombre del usuario.

Nombre de la tarea para las implement aciones iniciadas antes del 7 de marzo de 2024	Nombre de la tarea para las implement aciones que se iniciaron a partir del 7 de marzo de 2024	Task description
InventoryDeregistr ation	sin cambios de nombre	Anula el registro de los AWS recursos de TNB AWS .
-	InventoryRegistration	Registra los AWS recursos en TNB. AWS
KubernetesClusterC onfiguration	ClusterConfiguration	Configura el clúster de Kubernetes y añade funciones de IAM adicionales a Amazon EKS AuthMap tal y como se define en la NSD.
NetworkServiceFina lization	sin cambios de nombre	Finaliza el servicio de red y proporciona una actualización del estado de éxito o error.
NetworkServiceInst antiation	sin cambio de nombre	Inicia el servicio de red.
SelfManag edNodesConfiguration	sin cambio de nombre	Arranca los nodos autogestionados con Amazon EKS y el plano de control de Kubernetes.
-	ValidateNetworkSer viceUpdate	Ejecuta las validaciones antes de actualizar una instancia de red.

Cuotas de servicio para AWS TNB

Las cuotas de servicio, también denominadas límites, son la cantidad máxima de recursos u operaciones de servicio para su AWS cuenta. Para obtener más información, consulte el artículo sobre AWS Service Quotas en la Referencia general de Amazon Web Services.

Las siguientes son las cuotas de servicio de AWS TNB.

Name	Valor predeterminado	Ajusta	Description (Descripción)
Operaciones simultáneas y continuas de servicios de red	Cada región admitida: 40	Sí	Número máximo de operaciones simultáneas de servicios de red en curso en una región.
Paquetes de funciones	Cada región admitida: 200	Sí	Número máximo de paquetes de funciones en una región.
Paquetes de red	Cada región admitida: 40	Sí	Número máximo de paquetes de red en una región.
Instancias de servicios de red	Cada región admitida: 800	Sí	Número máximo de instancias de servicios de red en una región.

Historial de documentos de la guía de usuario de AWS TNB

En la siguiente tabla se describen las versiones de la documentación de AWS TNB.

Cambio	Descripción	Fecha
Actualizaciones de la configuración de red del grupo de nodos Amazon EKS	Agregue y elimine subredes y grupos de seguridad. Agregue, modifique y elimine ENIs de la red. Para obtener más información, consulte Parámetros que puede actualizar.	10 de septiembre de 2025
Adición y eliminación de grupos de nodos de Amazon EKS en clústeres existentes	AWS TNB ahora admite la adición de nuevos grupos de nodos y la eliminación de los grupos de nodos existentes de los clústeres de Amazon EKS. Para obtener más informaci ón, consulte Parámetros que puede actualizar.	4 de junio de 2025
Tamaño del volumen raíz	Puede especificar el tamaño del volumen raíz de Amazon EBS subyacente de sus nodos de trabajo de Amazon EKS a través del `root_volu me_size` campo de AWS.Compute. EKSManage dNode y .Compute.AWS EKSSelfManagedNode Nodos TOSCA.	19 de mayo de 2025
Recursos de referencia en scripts	Puede hacer referencia a los recursos creados por AWS TNB para configurarlos en	2 de mayo de 2025

	sus scripts de Lifecycle Hook y en los scripts de datos de usuario.	
La versión 1.32 de Kubernetes ahora es compatible con los nodos y grupos de nodos gestionados de Amazon EKS.	AWS TNB es compatible con la versión 1.32 de Kubernetes para .Compute.EKS y .Compute.AWSAWSEKSManagedNodo.	24 de abril de 2025
La versión 1.24 de Kubernetes ya no es compatible con los nodos y grupos de nodos gestionados de Amazon EKS	AWS TNB ya no es compatible con la versión 1.24 de Kubernetes para .Compute.EKS y .Compute.AWSAWSEKSManagedNodo.	17 de abril de 2025
AL2023 Compatibilidad con AMI para nodos gestionados por Amazon EKS	AWS TNB admite los tipos de AL2023 AMI para AWS.Compute. EKSManagedNodo.	17 de abril de 2025
La versión 1.23 de Kubernetes ya no es compatible con los nodos y grupos de nodos gestionados de Amazon EKS	AWS TNB ya no es compatible con la versión 1.23 de Kubernetes para .Compute.EKS y .Compute.AWSAWSEKSManagedNodo.	4 de abril de 2025
El ID de AMI se puede actualizar	Ahora puede actualizar el campo ami_id durante una llamada a la `UpdateSolNetworkService` API.	31 de marzo de 2025
La versión 1.31 de Kubernetes ahora es compatible con los nodos y grupos de nodos gestionados de Amazon EKS.	AWS TNB es compatible con la versión 1.31 de Kubernetes para .Compute.EKS y .Compute.AWSAWSEKSManagedNodo.	18 de febrero de 2025

Versión de Kubernetes para .Compute. AWS EKSManagedNode	AWS TNB admite las versiones 1.23 a 1.30 de Kubernetes para crear un grupo de nodos gestionado por Amazon EKS.	28 de enero de 2025
Versión de Kubernetes para clúster	AWS TNB ahora es compatible con la versión 1.30 de Kubernetes para crear clústeres de Amazon EKS.	19 de agosto de 2024

AWS TNB admite una operación adicional para administrar el ciclo de vida de la red.	Puede actualizar una instancia de red instancia da o previamente actualiza da con un nuevo paquete de red y valores de parámetros. Consulte:
	• Operaciones del ciclo de vida
	• Actualizar una instancia de red
	• AWS Ejemplo de rol de servicio TNB:
	  • Añada estas acciones de Amazon EKS: `eks:UpdateAddon` `eks:UpdateClusterVersion` ,`eks:UpdateNodegroupConfig` ,`eks:UpdateNodegroupVersion` , `eks:DescribeUpdate`
	  • Añada esta CloudFormation acción: `cloudformation:UpdateStack`
	• Nuevas tareas de despliegue:`InfrastructureUpdate` ,`Inventory Registration` , `ValidateNetworkServiceUpdate`
	• Actualizaciones de API: GetSolNetworkOpera
		30 de julio de 2024

	tionListSolNetworkOperations , y UpdateSolNetworkInstance	
Nueva tarea y nuevos nombres de tareas para las tareas existentes	Hay una nueva tarea disponible. A partir del 7 de marzo de 2024, algunas tareas existentes tienen nuevos nombres para mayor claridad.	7 de mayo de 2024
Versión de Kubernetes para clúster	AWS TNB ahora es compatible con la versión 1.29 de Kubernetes para crear clústeres de Amazon EKS.	10 de abril de 2024
Support para interfaz de red `security_groups`	Puede adjuntar grupos de seguridad al nodo AWS.Networking.ENI.	2 de abril de 2024
Support para el cifrado de volúmenes raíz de Amazon EBS	Puede habilitar el cifrado de Amazon EBS para el volumen raíz de Amazon EBS. Para habilitarlo, añada las propiedades en AWS.Compute. EKSManagedNode o AWS.Compute. EKSSelfManagedNodenodo.	2 de abril de 2024
Support for node `labels`	Puede adjuntar etiquetas de nodo a su grupo de nodos en AWS.Compute. EKSManagedNode o AWS.Compute. EKSSelfManagedNodenodo.	19 de marzo de 2024

Support para interfaz de red `source_dest_check`	Puede indicar si desea activar o desactivar la source/destination comprobación de la interfaz de red a través del nodo AWS.Networking.ENI.	25 de enero de 2024
Compatibilidad con instancias de Amazon EC2 con datos de usuario personalizados	Puede lanzar instancias de Amazon EC2 con datos de usuario personalizados a través de .Compute. AWS UserData nodo.	16 de enero de 2024
Compatibilidad con grupo de seguridad	AWS TNB le permite importar el AWS recurso del grupo de seguridad.	8 de enero de 2024
Descripción actualizada de `network_interfaces`	Cuando la `network_interfaces` propiedad está incluida en AWS.Compute. EKSManagedNode o AWS.Compute. EKSSelfManagedNodenodo, AWS TNB obtiene el permiso correspondiente ENIs de la `multus_role` propiedad, si está disponible, o de la propiedad. `node_role`	18 de diciembre de 2023
Compatibilidad con clúster privado	AWS TNB ahora admite clústeres privados. Para indicar un clúster privado, establezca la propiedad `access` en `PRIVATE`.	11 de diciembre de 2023

Versión de Kubernetes para clúster	AWS TNB ahora es compatible con la versión 1.28 de Kubernetes para crear clústeres de Amazon EKS.	11 de diciembre de 2023
AWS TNB admite grupos de ubicación	Se agregó un grupo de ubicación para las definiciones de nodo `AWS.Compute.EKSManagedNode` y `AWS.Compute.EKSSelfManagedNode`.	11 de diciembre de 2023

AWS TNB añade soporte para IPv6	AWS TNB ahora admite la creación de instancias de red con IPv6 infraestructura. Compruebe los nodos AWS.Networking.VPC, .Networking.Subnet, .Networking.AWSAWS https://docs.aws.amazon.com/tnb/latest/ug/node-internet-gateway.htmlInternetGatewayAWS, .Redes. SecurityGroupIngressRule, AWS.Redes. SecurityGroupEgressRuley AWS.compute.EKS para las configuraciones. IPv6 También hemos añadido los nodos .Networking.AWS NATGatewayy AWS.Networking.Route para la configuración. NAT64 Hemos actualizado el rol de servicio AWS TNB y el rol de servicio AWS TNB para el grupo de nodos EKS de Amazon para obtener IPv6 permisos. Consulte Ejemplos de políticas de roles de servicio.	16 de noviembre de 2023
Se agregaron permisos a la política de roles de AWS servicio de TNB	Hemos añadido permisos a la política de funciones de servicio de AWS TNB para Amazon S3 y CloudFormation para permitir la instanciación de la infraestructura.	23 de octubre de 2023

AWS TNB se lanzó en más regiones	AWS TNB ya está disponible en las regiones de Asia Pacífico (Seúl), Canadá (Central), Europa (España), Europa (Estocolmo) y Sudamérica (São Paulo).	27 de septiembre de 2023
Etiquetas para AWS.Compute. EKSSelfManagedNode	AWS TNB ahora admite etiquetas para la definición del `AWS.Compute.EKSSelfManagedNode` nodo.	22 de agosto de 2023
AWS TNB admite instancias que aprovechan IMDSv2	Al lanzar su instancia, debe usar IMDSv2.	14 de agosto de 2023
Permisos actualizados para MultusRoleInlinePolicy	MultusRoleInlinePolicy Ahora incluye el `ec2:DeleteNetworkInterface` permiso.	7 de agosto de 2023
Versión de Kubernetes para clúster	AWS TNB ahora es compatible con las versiones 1.27 de Kubernetes para crear clústeres de Amazon EKS.	25 de julio de 2023
AWS.compute.eks. AuthRole	AWS TNB admite AuthRole que le permite añadir funciones de IAM al clúster de Amazon EKS para que los usuarios puedan acceder al clúster `aws-auth ConfigMap` de Amazon EKS mediante una función de IAM.	19 de julio de 2023

AWS TNB admite grupos de seguridad.	Se agregó el archivo AWS.Networking. SecurityGroup, AWS.Networking. SecurityGroupEgressRuley AWS.Networking. SecurityGroupIngressRulea la plantilla NSD.	18 de julio de 2023
Versión de Kubernetes para clúster	AWS TNB admite las versiones 1.22 a 1.26 de Kubernetes para crear clústeres de Amazon EKS. AWS TNB ya no es compatible con las versiones 1.21 de Kubernetes.	11 de mayo de 2023
AWS.Compute. EKSSelfManagedNode	Puede crear nodos de trabajo autogestionados en la región, en las Zonas AWS Locales y. AWS Outposts	29 de marzo de 2023
Versión inicial	Esta es la primera versión de la Guía del usuario de AWS TNB.	21 de febrero de 2023

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.