Manual do usuário

AWS Construtor de rede Telco

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Construtor de rede Telco: Manual do usuário

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens de marcas da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

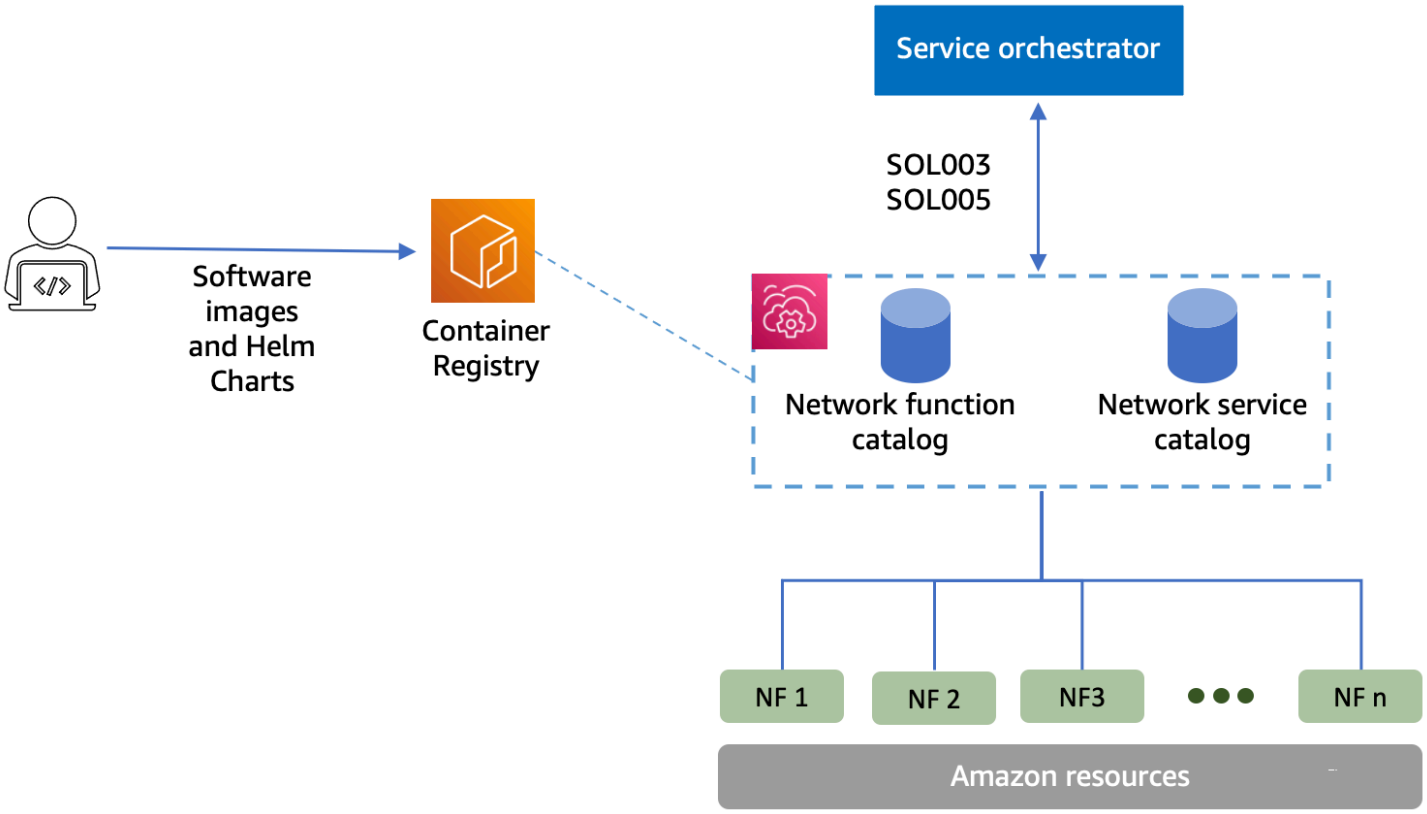
O que é o AWS Telco Network Builder?

AWS O Telco Network Builder (AWS TNB) é um AWS serviço que fornece aos provedores de serviços de comunicação (CSPs) uma maneira eficiente de implantar, gerenciar e escalar redes 5G na AWS infraestrutura.

Com o AWS TNB, você implanta redes 5G escaláveis e seguras Nuvem AWS usando uma imagem da sua rede de forma automatizada. Você não precisa aprender novas tecnologias, decidir qual serviço de computação usar ou saber como provisionar e configurar AWS recursos.

Em vez disso, você descreve a infraestrutura de sua rede e fornece as imagens de software das funções de rede de seus parceiros fornecedores independentes de software (ISV). AWS O TNB se integra a AWS serviços e orquestradores de serviços terceirizados para provisionar automaticamente a AWS infraestrutura necessária, implantar funções de rede em contêineres e configurar o gerenciamento de rede e acesso para criar um serviço de rede totalmente operacional.

O diagrama a seguir ilustra as integrações lógicas entre o AWS TNB e os orquestradores de serviços para implantar funções de rede usando interfaces padrão baseadas no Instituto Europeu de Padrões de Telecomunicações (ETSI).



Tópicos

Novo em AWS?

Se você é iniciante em AWS produtos e serviços, comece a aprender mais com os seguintes recursos:

- Introdução ao AWS
- Começando com AWS

Para quem é o AWS TNB?

AWS A TNB CSPs busca aproveitar a economia, a agilidade e a elasticidade que as Nuvem AWS ofertas oferecem sem escrever e manter scripts e configurações personalizados para projetar, implantar e gerenciar serviços de rede. AWS O TNB provisiona automaticamente a AWS infraestrutura necessária, implanta funções de rede em contêineres e configura o gerenciamento de rede e acesso para criar serviços de rede totalmente operacionais com base nos descritores de serviços de rede definidos pelo CSP e nas funções de rede que o CSP deseja implantar.

AWS Características do TNB

A seguir estão alguns dos motivos pelos quais um CSP gostaria de usar o AWS TNB:

Ajuda a simplificar tarefas

Forneça mais eficiência às suas operações de rede, como implantação de novos serviços, atualização e upgrade de funções de rede e alteração de topologias de infraestrutura de rede.

Integra-se com orquestradores

AWS O TNB se integra a orquestradores de serviços terceirizados populares que são compatíveis com ETSI.

Faz escalonamento

Você pode configurar o AWS TNB para escalar AWS os recursos subjacentes para atender à demanda de tráfego, realizar atualizações de funções de rede com mais eficiência, implementar alterações na topologia da infraestrutura de rede e reduzir o tempo de implantação de novos serviços 5G de dias para horas.

Inspeciona e monitora recursos AWS

AWS O TNB permite que você inspecione e monitore os AWS recursos que dão suporte à sua rede em um único painel, como Amazon VPC EC2, Amazon e Amazon EKS.

Compatibilidade com modelos de serviço

AWS O TNB permite criar modelos de serviço para todas as cargas de trabalho de telecomunicações (RAN, Core, IMS). Você pode criar uma nova definição de serviço, reutilizar um modelo existente ou integrar-se a um pipeline de integração contínua e entrega contínua (CI/CD) para publicar uma nova definição.

Rastreia as alterações nas implantações de rede

Quando você altera a configuração subjacente de uma implantação de função de rede, por exemplo, alterando o tipo de instância de um tipo de EC2 instância da Amazon, você pode acompanhar as alterações de forma repetível e escalável. Fazer isso manualmente exigiria gerenciar o estado da rede, criar e excluir recursos e prestar atenção à ordem das alterações necessárias. Ao usar o AWS TNB para gerenciar o ciclo de vida de sua função de rede, você só faz as alterações nos descritores do serviço de rede que descrevem a função de rede. AWS O TNB então fará automaticamente as alterações necessárias na ordem correta.

Simplifica o ciclo de vida da função de rede

Você pode gerenciar a primeira e todas as versões subsequentes de uma função de rede e especificar quando fazer upgrade. Você também pode gerenciar suas aplicações RAN, Core, IMS e de rede da mesma forma.

Acessando o AWS TNB

Você pode criar, acessar e gerenciar seus recursos AWS do TNB usando qualquer uma das seguintes interfaces:

- AWS Console TNB — Fornece uma interface web para gerenciar sua rede.
- AWS API TNB — Fornece uma RESTful API para realizar ações AWS TNB. Para obter mais informações, consulte Referência de API do AWS TNB
- AWS Command Line Interface (AWS CLI) — Fornece comandos para um amplo conjunto de AWS serviços, incluindo AWS TNB. É compatível com Windows, macOS e Linux. Para obter mais informações, consulte o Guia do usuário do AWS Command Line Interface.
- AWS SDKs— Fornece idiomas específicos APIs e completa muitos dos detalhes da conexão. Por exemplo, cálculo de assinaturas, tratamento de novas tentativas de solicitação e tratamento de erros. Para obter mais informações, consulte AWS SDKs.

Preços do AWS TNB

AWS A TNB ajuda a CSPs automatizar a implantação e o gerenciamento de suas redes de telecomunicações em. AWS Você paga pelas duas dimensões a seguir ao usar o AWS TNB:

- Por horas de item de função de rede gerenciada (MNFI).
- Por número de solicitações de API.

Você também incorre em cobranças adicionais ao usar outros AWS serviços em conjunto com AWS a TNB. Para obter mais informações, consulte Definição de preço do AWS.

Para exibir sua fatura, acesse o Painel do Billing and Cost Management no console do Gerenciamento de Faturamento e Custos da AWS. Sua fatura contém links para relatórios de uso que fornecem mais detalhes da fatura. Para obter mais informações sobre o faturamento AWS da conta, consulte Faturamento AWS da conta.

Se você tiver dúvidas sobre AWS faturamento, contas e eventos, entre em contato com o AWS Support.

AWS Trusted Advisor é um serviço que você pode usar para ajudar a otimizar os custos, a segurança e o desempenho do seu AWS ambiente. Para obter mais informações, consulte AWS Trusted Advisor.

Próximas etapas

Para obter mais informações sobre como começar a usar o AWS TNB, consulte os tópicos a seguir:

- Configurar AWS TNB: concluir as etapas de pré-requisitos.
- Começando com o AWS TNB: implantar sua primeira função de rede, como Unidade Centralizada (UC), Função de Gerenciamento de Acesso e Mobilidade (AMF), Função de Plano de Usuário (UPF) ou um núcleo 5G completo.

Como funciona o AWS TNB

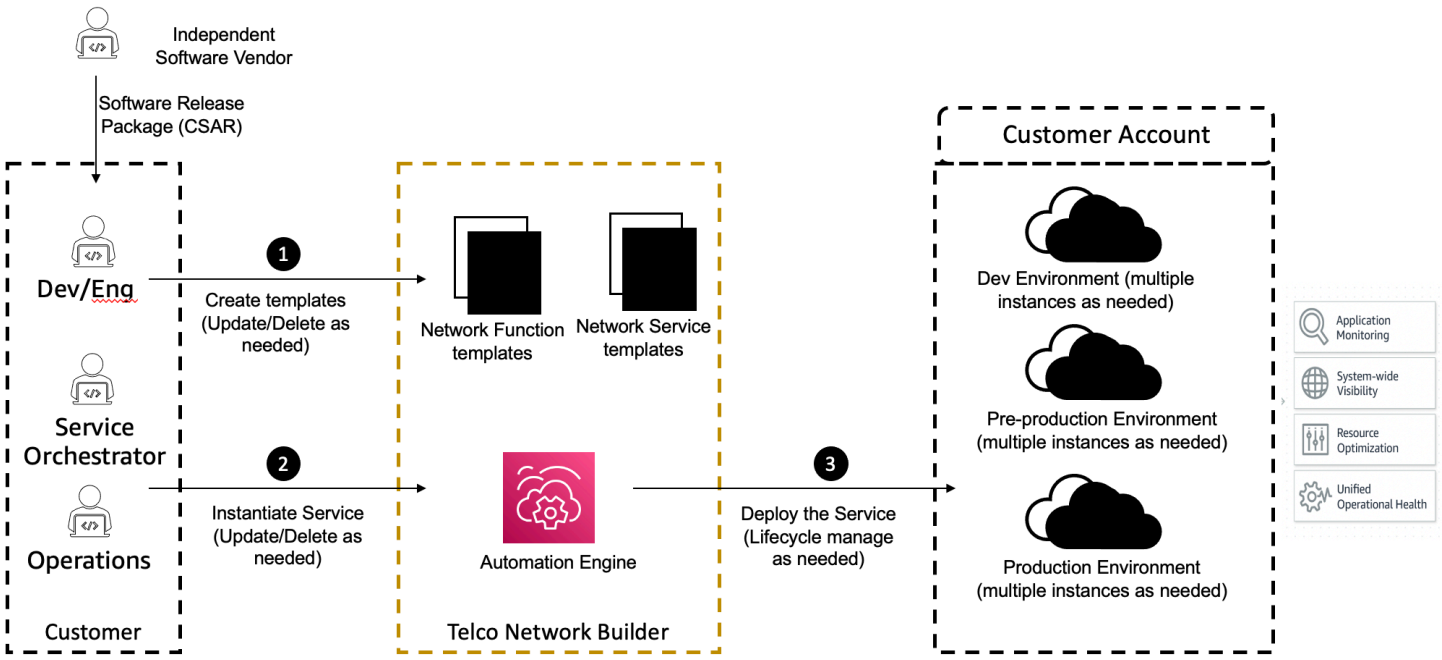
AWS O TNB se integra a end-to-end orquestradores e AWS recursos padronizados para operar redes 5G completas.

AWS O TNB permite ingerir pacotes de funções de rede e descritores de serviços de rede (NSDs) e fornece o mecanismo de automação para operar suas redes. Você pode usar seu end-to-end orquestrador e integrar-se ao AWS TNB APIs, ou usar o AWS TNB SDKs para criar seu próprio fluxo de automação. Para obter mais informações, consulte AWS Arquitetura TNB.

Tópicos

AWS Arquitetura TNB

AWS O TNB fornece a capacidade de realizar operações de gerenciamento do ciclo de vida por meio da Console de gerenciamento da AWS, AWS CLI, API REST AWS TNB e. SDKs Isso permite que as diferentes personalidades do CSP, como membros das equipes de Engenharia, Operações e Sistema Programático, tirem proveito do AWS TNB. Você cria e carrega um pacote de funções de rede como um arquivo Cloud Service Archive (CSAR). O arquivo CSAR contém charts do Helm, imagens de software e um descritor de função de rede (NFD). Você pode usar modelos para implantar repetidamente várias configurações desse pacote. Você cria modelos de serviço de rede definindo a infraestrutura e as funções de rede que você deseja implantar. Você pode usar substituições de parâmetros para implantar configurações diferentes em locais diferentes. Em seguida, você pode instanciar uma rede usando os modelos e implantar suas funções de rede na AWS infraestrutura. AWS O TNB fornece a visibilidade de suas implantações.



Integração com Serviços da AWS

Uma rede 5G é composta por um conjunto de funções de rede em contêineres interconectadas implantadas em milhares de clusters Kubernetes. AWS O TNB se integra ao seguinte Serviços da AWS como específico APIs para telecomunicações para criar um serviço de rede totalmente operacional:

- Amazon Elastic Container Registry (Amazon ECR) para armazenar artefatos de funções de rede de Independent Software Vendors ISVs ().
- Amazon Elastic Kubernetes Service (Amazon EKS) para configurar clusters.
- Amazon VPC para estruturas de rede.
- Grupos de segurança usando CloudFormation.
- AWS CodePipeline para alvos de implantação em Regiões da AWS, AWS Locais Zonas AWS Outposts e.
- IAM para definir perfis.
- AWS Organizations para controlar o acesso ao AWS TNB. APIs
- Health Dashboard e AWS CloudTrail para monitorar a saúde e as métricas de publicação.

AWS Cotas de recursos do TNB

Você Conta da AWS tem cotas padrão, anteriormente chamadas de limites, para cada um. AWS service (Serviço da AWS) Salvo indicação em contrário, cada cota é específica para um Região da AWS. Você pode solicitar aumentos para algumas cotas, mas não para todas as cotas.

Para ver as cotas do AWS TNB, abra o console Service Quotas. No painel de navegação, escolha Serviços da AWS e selecione AWS TNB.

Para solicitar um aumento da cota, consulte Requesting a quota increase no Guia do usuário do Service Quotas.

Você Conta da AWS tem as seguintes cotas relacionadas ao AWS TNB.

Cota de recurso	Description	Valor padrão	Ajustável?
Instâncias do serviço de rede	O número máximo de instâncias do serviço de rede em uma região.	800	Sim
Operações simultâneas de serviços de rede contínuas	O número máximo de operações simultâneas de serviços de rede contínuas em uma região.	40	Sim
Pacotes de rede	O número máximo de pacotes de rede em uma região.	40	Sim
Pacotes de funções	O número máximo de pacotes de funções em uma região.	200	Sim

AWS Conceitos do TNB

Este tópico descreve conceitos essenciais para ajudá-lo a começar a usar o AWS TNB.

Conteúdo

Ciclo de vida de uma função de rede

AWS O TNB ajuda você em todo o ciclo de vida de suas funções de rede. O ciclo de vida da função de rede inclui os seguintes estágios e atividades:

Planejamento

1. Planeje sua rede identificando as funções de rede a serem implantadas.
2. Coloque as imagens do software de função de rede em um repositório de imagens de contêiner.
3. Crie os pacotes CSAR para implantar ou fazer upgrade.
4. Use o AWS TNB para carregar o pacote CSAR que define sua função de rede (por exemplo, CU AMF e UPF) e integre com um pipeline de integração contínua e entrega contínua (CI/CD) que pode ajudá-lo a criar novas versões do seu pacote CSAR à medida que novas imagens de software de função de rede ou scripts de clientes estiverem disponíveis.

Configuração

1. Identifique as informações necessárias para a implantação, como tipo de computação, versão da função de rede, informações de IP e nomes dos recursos.
2. Use as informações para criar seu descritor de serviço de rede (NSD).
3. Ingestão NSDs que define suas funções de rede e os recursos necessários para que a função de rede seja instanciada.

Instanciação

1. Crie a infraestrutura exigida pelas funções de rede.

2. Instancie (ou provisione) a função de rede conforme definida em seu NSD e comece a transportar tráfego.
3. Valide os ativos.

Produção

Durante o ciclo de vida da função de rede, você concluirá as operações de produção, como:

- Atualize a configuração da função de rede, por exemplo, atualize um valor na função da rede implantada.
- Atualize a instância de rede com um novo pacote de rede e valores de parâmetros. Por exemplo, atualize o `version` parâmetro Amazon EKS no pacote de rede.

Use interfaces padronizadas

AWS O TNB se integra aos orquestradores de serviços compatíveis com o Instituto Europeu de Padrões de Telecomunicações (ETSI), permitindo que você simplifique a implantação de seus serviços de rede. Os orquestradores de serviços podem usar SDKs o AWS TNB, a CLI ou o APIs para iniciar operações, como instanciar ou atualizar uma função de rede para uma nova versão.

AWS O TNB suporta as seguintes especificações.

Especificação	Versão	Description
ETSI SOL001	v3.6.1	Define padrões para permitir descritores de funções de rede baseados em TOSCA.
ETSI SOL002	v3.6.1	Define modelos em torno do gerenciamento de funções de rede.
ETSI SOL003	v3.6.1	Define padrões para o gerenciamento do ciclo de vida das funções de rede.
ETSI SOL004	v3.6.1	Define padrões CSAR para pacotes de funções de rede.
ETSI SOL005	v3.6.1	Define padrões para pacotes de serviços de rede e gerenciamento do ciclo de vida do serviço de rede.

Especificação	Versão	Description
ETSI SOL007	v3.5.1	Define padrões para permitir descritores de serviços de rede baseados em TOSCA.

Pacote de funções

Com o AWS TNB, você pode armazenar pacotes de funções que estejam em conformidade com o ETSI SOL001/SOL004 em um catálogo de funções. Em seguida, você pode carregar pacotes do Cloud Service Archive (CSAR) que contêm artefatos que descrevem sua função de rede virtual.

- Descritor de função de rede virtual — Define metadados para integração de pacotes e gerenciamento de funções de rede virtual. Você deve nomear esse arquivo `vnfd.yaml`.
- Imagens de software — Referencia a função de rede virtual Container Images. O Amazon Elastic Container Registry (Amazon ECR) pode atuar como seu repositório de imagens de funções de rede virtual.
- Arquivos adicionais — Use para gerenciar a função de rede virtual; por exemplo, scripts e gráficos do Helm.

O CSAR é um pacote definido pelo padrão OASIS TOSCA e inclui um descritor de rede/serviço que está em conformidade com a especificação OASIS TOSCA YAML. Para obter informações sobre a especificação YAML necessária, consulteReferência TOSCA para TNB AWS.

Veja a seguir um exemplo de descritor de função de rede 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"
```

Pacote de rede

Um pacote de rede é um `.zip` arquivo no formato CSAR (Cloud Service Archive). Ele define os pacotes de funções que você deseja implantar e a AWS infraestrutura na qual deseja implantá-los.

O pacote de rede contém os seguintes arquivos:

- Um arquivo descritor de rede (`nsd.yaml`) no formato TOSCA, conforme descrito pelo ETSI SOL007.

  O `nsd.yaml` arquivo contém referências aos pacotes de funções enviados com seu descritor IDs.
- Scripts de dados do usuário, se houver.
- Scripts de gancho de ciclo de vida, se houver.
- Arquivos de `values.yaml` configuração dos plug-ins, se houver.

AWS O TNB suporta os padrões ETSI para a modelagem de recursos, como rede, serviço e função, na linguagem TOSCA. AWS O TNB torna seu uso mais eficiente, Serviços da AWS modelando-os de uma forma que seu orquestrador de serviços compatível com ETSI possa entender.

Descritores de serviços de rede para TNB AWS

Um descritor de serviço de rede (NSD) é um arquivo `.yaml` em um pacote de rede que usa o padrão TOSCA para descrever as funções de rede que você quer implantar e a infraestrutura da AWS na qual deseja implantá-las. Para definir seu NSD e configurar seus recursos subjacentes e operações do ciclo de vida da rede, você deve entender o esquema NSD TOSCA suportado pelo TNB. AWS

Seu arquivo NSD é dividido nas seguintes partes:

1. Versão da definição TOSCA: é a primeira linha do seu arquivo NSD YAML e contém as informações da versão, mostradas no exemplo a seguir.

```
tosca_definitions_version: tnb_simple_yaml_1_0
```

2. VNFD: o NSD contém a definição da função de rede na qual realizar operações de ciclo de vida. Cada função de rede deve ser identificada pelos seguintes valores:
   - Um ID exclusivo para `descriptor_id`. O ID deve corresponder ao ID no pacote CSAR de funções de rede.
   - Um nome exclusivo para `namespace`. O nome precisa estar associado a um ID exclusivo para ser referenciado com mais facilidade em todo o arquivo NSD YAML, mostrado no exemplo a seguir.

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

3. Modelo de topologia: define os recursos a serem implantados, a implantação da função de rede e quaisquer scripts personalizados, como ganchos do ciclo de vida. Isso é mostrado no exemplo a seguir.

```
topology_template:

  node_templates:

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

4. Nós adicionais: cada recurso modelado tem seções para propriedades e requisitos. As propriedades descrevem atributos opcionais ou obrigatórios de um recurso, como a versão. Os requisitos descrevem dependências que precisam ser fornecidas como argumentos. Por exemplo, para criar um recurso de grupo de nós do Amazon EKS, ele precisa ser criado dentro de um cluster do Amazon EKS. Isso é mostrado no exemplo a seguir.

```
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
```

Exemplo de NSD

A seguir está um trecho de um NSD mostrando como modelar. Serviços da AWS A função de rede será implantada em um cluster do Amazon EKS com o Kubernetes versão 1.27. As sub-redes dos aplicativos são Subnet01 e Subnet02. Em seguida, você pode definir o NodeGroups para seus aplicativos com uma Amazon Machine Image (AMI), tipo de instância e configuração de escalonamento 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
```

Gestão e operações da AWS TNB

Com o AWS TNB, você pode gerenciar sua rede usando operações de gerenciamento padronizadas de acordo com ETSI SOL003 e SOL005. Você pode usar o AWS TNB APIs para realizar operações de ciclo de vida, como:

- Instanciação das suas funções de rede.
- Encerramento das suas funções de rede.
- Atualização das suas funções de rede para substituir as implantações do Helm.
- Atualizar uma instância de rede instanciada ou atualizada com um novo pacote de rede e valores de parâmetros.
- Gerenciamento de versões de seus pacotes de funções de rede.
- Gerenciando versões do seu NSDs.

- Recuperação de informações sobre suas funções de rede implantadas.

Configurar AWS TNB

Configure o AWS TNB concluindo as tarefas descritas neste tópico.

Tarefas

- Inscreva-se para um Conta da AWS
- Escolha um AWS Região
- Observar o endpoint do serviço
- (Opcional) Instale o AWS CLI
- Configurar AWS Funções do TNB

Inscreva-se para um Conta da AWS

Para começar AWS, você precisa de um Conta da AWS. Para obter informações sobre como criar um Conta da AWS, consulte Introdução a um Conta da AWS no Guia de AWS Gerenciamento de contas referência.

Escolha um AWS Região

Para ver a lista de regiões disponíveis para AWS TNB, consulte a Lista de serviços AWS regionais. Para exibir a lista de endpoints para acesso programático, consulte Endpoints do AWS TNB no Referência geral da AWS.

Observar o endpoint do serviço

Para se conectar programaticamente a um AWS serviço, você usa um endpoint. Além dos AWS endpoints padrão, alguns AWS serviços oferecem endpoints FIPS em regiões selecionadas. Para obter mais informações, consulte Endpoints de serviço da AWS.

Nome da região	Região	Endpoint	Protocolo	
Leste dos EUA	us-east-1	tnb.us-east-1.amazonaws.com	HTTPS	

Nome da região	Região	Endpoint	Protocolo	
(Norte da Virgínia)				
Oeste dos EUA (Oregon)	us-west-2	tnb.us-west-2.amazonaws.com	HTTPS	
Ásia-Pacífico (Seul)	ap-northeast-2	tnb.ap-northeast-2.amazonaws.com	HTTPS	
Ásia-Pacífico (Sydney)	ap-southeast-2	tnb.ap-southeast-2.amazonaws.com	HTTPS	
Canadá (Central)	ca-central-1	tnb.ca-central-1.amazonaws.com	HTTPS	
Europa (Frankfurt)	eu-central-1	tnb.eu-central-1.amazonaws.com	HTTPS	
Europa (Paris)	eu-west-3	tnb.eu-west-3.amazonaws.com	HTTPS	
Europa (Espanha)	eu-south-2	tnb.eu-south-2.amazonaws.com	HTTPS	
Europa (Estocolmo)	eu-north-1	tnb.eu-north-1.amazonaws.com	HTTPS	
América do Sul (São Paulo)	sa-east-1	tnb.sa-east-1.amazonaws.com	HTTPS	

(Opcional) Instale o AWS CLI

O AWS Command Line Interface (AWS CLI) fornece comandos para um amplo conjunto de AWS produtos e é compatível com Windows, macOS e Linux. Você pode acessar o AWS TNB usando o. AWS CLI Para começar a usar, consulte o Guia do usuário da AWS Command Line Interface. Para obter mais informações sobre os comandos do AWS TNB, consulte tnb na Referência de AWS CLI Comandos.

Configurar AWS Funções do TNB

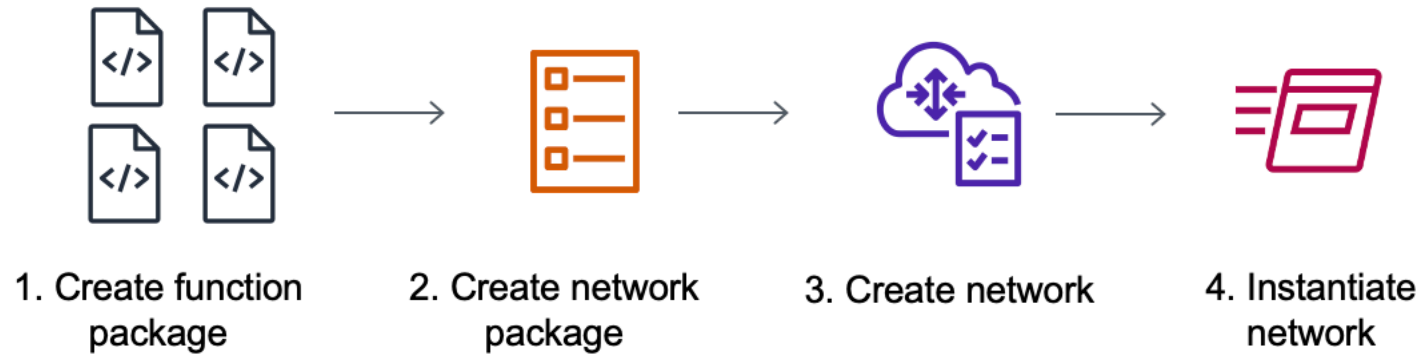
Você deve criar uma função de serviço do IAM para gerenciar diferentes partes da sua solução AWS TNB. AWS As funções de serviço do TNB podem fazer chamadas de API para outros AWS serviços, como AWS CloudFormation AWS CodeBuild, e vários serviços de computação e armazenamento, em seu nome, para instanciar e gerenciar recursos para sua implantação.

Para obter mais informações sobre a função de serviço AWS TNB, consulteGerenciamento de identidade e acesso para AWS TNB.

Começando com o AWS TNB

Este tutorial demonstra como você usa o AWS TNB para implantar uma função de rede, por exemplo, a Unidade Centralizada (UC), a Função de Gerenciamento de Acesso e Mobilidade (AMF) ou a Função de Plano de Usuário (UPF) 5G.

O diagrama a seguir ilustra o processo de implantação:



Tarefas

- Pré-requisitos
- Criar um pacote de funções
- Criar um pacote de rede
- Criar e instanciar uma instância de rede
- Limpeza

Pré-requisitos

Antes de realizar uma implantação bem-sucedida, você deve ter o seguinte:

- Um plano AWS de Business Support.
- Permissões por meio de funções do IAM.
- Um pacote de função de rede (NF) compatível com o ETSI SOL001/SOL004.
- Modelos de Network Service Descriptor (NSD) que estão em conformidade com o ETSI SOL007.

Você pode usar um pacote de funções de amostra ou pacote de rede do GitHub site Pacotes de amostra para AWS TNB.

Criar um pacote de funções

Um pacote de funções de rede é um arquivo Cloud Service Archive (CSAR). O arquivo CSAR contém charts do Helm, imagens de software e um descritor de função de rede (NFD).

Para criar um pacote de funções

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. Selecione Pacotes de funções no painel de navegação.
3. Escolha Criar pacote de funções.
4. Em Carregar pacote de funções, escolha Escolher arquivos e carregue cada pacote CSAR como um `.zip` arquivo. Você pode carregar no máximo 10 arquivos.
5. (Opcional) Em Tags, escolha Adicionar nova tag e insira uma chave e um valor. Você pode usar tags para pesquisar e filtrar seus recursos ou monitorar seus AWS custos.
6. Escolha Próximo.
7. Revise os detalhes do pacote e escolha Criar pacote de funções.

Criar um pacote de rede

Um pacote de rede especifica as funções de rede que você deseja implantar e como deseja implantá-las no catálogo.

Para criar um pacote de rede

1. No painel de navegação, selecione Pacotes de rede.
2. Escolha Criar pacote de rede.
3. Em Carregar pacote de rede, escolha Escolher arquivos e carregue cada NSD como um `.zip` arquivo. Você pode carregar no máximo 10 arquivos.
4. (Opcional) Em Tags, escolha Adicionar nova tag e insira uma chave e um valor. Você pode usar tags para pesquisar e filtrar seus recursos ou monitorar seus AWS custos.
5. Escolha Próximo.
6. Escolha Criar pacote de rede.

Criar e instanciar uma instância de rede

Uma instância de rede é uma rede única criada no AWS TNB que pode ser implantada. Você deve criar uma instância de rede e instanciá-la. Quando você instancia uma instância de rede, o AWS TNB provisiona a AWS infraestrutura necessária, implanta funções de rede em contêineres e configura o gerenciamento de rede e acesso para criar um serviço de rede totalmente operacional.

Para criar e instanciar uma instância de rede

1. No painel de navegação, selecione Redes.
2. Clique em Criar instância de rede.
3. Insira um nome e uma descrição para a rede e escolha Próximo.
4. Escolha um pacote de rede. Verifique os detalhes e escolha Avançar.
5. Clique em Criar instância de rede. O estado inicial é `Created`.

   A página Redes aparece mostrando a nova instância de rede no `Not instantiated` estado.
6. Selecione a instância de rede, escolha Ações e Instanciar.

   A página Instanciar rede é exibida.
7. Revise os detalhes e atualize os valores dos parâmetros. As atualizações nos valores dos parâmetros se aplicam somente a essa instância de rede. Os parâmetros nos pacotes NSD e VNFD não mudam.
8. Escolha Instanciar rede.

   A página de status de implantação é exibida.
9. Use o ícone Atualizar para rastrear o status de implantação da sua instância de rede. Você também pode ativar a atualização automática na seção Tarefas de implantação para acompanhar o progresso de cada tarefa.

Limpeza

Agora você pode excluir os recursos que você criou para este tutorial.

Para limpar recursos

1. No painel de navegação, selecione Redes.
2. Escolha o ID da rede e, em seguida, escolha Encerrar.

3. Quando a confirmação for solicitada, insira o ID da rede e escolha Encerrar.
4. Use o ícone Atualizar para rastrear o status da sua instância de rede.
5. (Opcional) Selecione a rede e escolha Excluir.

Pacotes de funções para AWS TNB

Um pacote de funções é um arquivo .zip no formato CSAR (Cloud Service Archive) que contém uma função de rede (um aplicativo de telecomunicação padrão ETSI) e um descritor de pacote de funções que usa o padrão TOSCA para descrever como as funções de rede devem ser executadas em sua rede.

Tarefas

- Crie um pacote de funções no AWS TNB
- Exibir um pacote de funções no AWS TNB
- Baixe um pacote de funções do AWS TNB
- Excluir um pacote de funções do AWS TNB

Crie um pacote de funções no AWS TNB

Saiba como criar um pacote de funções no catálogo de funções de rede da AWS TNB. Criar um pacote de funções é o primeiro passo para criar uma rede no AWS TNB. Depois de carregar um pacote de funções, você pode criar um pacote de rede.

Console

Para criar um pacote de funções usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. Selecione Pacotes de funções no painel de navegação.
3. Escolha Criar pacote de funções.
4. Escolha Escolher arquivos e carregue cada pacote CSAR como um `.zip` arquivo. Você pode fazer upload de no máximo 10 arquivos.
5. Escolha Próximo.
6. Revise os detalhes do pacote.
7. Escolha Criar pacote de funções.

AWS CLI

Para criar um pacote de funções usando o AWS CLI

1. Use o create-sol-function-packagecomando para criar um novo pacote de funções:

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

2. Use o comando put-sol-function-package-content para carregar o conteúdo do pacote de funções. Por exemplo:

```
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
```

Exibir um pacote de funções no AWS TNB

Saiba como exibir o conteúdo de um pacote de funções.

Console

Para exibir um pacote de funções usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. Selecione Pacotes de funções no painel de navegação.
3. Use a caixa de pesquisa para encontrar o pacote de funções

AWS CLI

Para visualizar um pacote de funções usando o AWS CLI

1. Use o list-sol-function-packagescomando para listar seus pacotes de funções.

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

2. Use o get-sol-function-packagecomando para ver detalhes sobre um pacote de funções.

```
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
```

Baixe um pacote de funções do AWS TNB

Saiba como baixar um pacote de funções do catálogo de funções de rede da AWS TNB.

Console

Para baixar um pacote de funções usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, no lado esquerdo do console, escolha Pacotes de funções.
3. Use a caixa de pesquisa para encontrar o pacote de funções
4. Escolha o pacote de funções
5. Em Ações, escolha Baixar.

AWS CLI

Para baixar um pacote de funções usando o AWS CLI

Use o comando get-sol-function-package-content para baixar um pacote de funções.

```
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
```

Excluir um pacote de funções do AWS TNB

Saiba como excluir um pacote de funções do catálogo de funções de rede do AWS TNB. Para excluir um pacote de funções, é preciso que ele esteja desabilitado.

Console

Para excluir um pacote de funções usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. Selecione Pacotes de funções no painel de navegação.
3. Use a caixa de pesquisa para encontrar o pacote de funções.
4. Escolha um pacote de funções.
5. Escolha Ações, Desabilitar.
6. Escolha Ações, Excluir.

AWS CLI

Para excluir um pacote de funções usando o AWS CLI

1. Use o update-sol-function-packagecomando para desativar um pacote de funções.

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

2. Use o delete-sol-function-packagecomando para excluir um pacote de funções.

```
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
```

Pacotes de rede para AWS TNB

Um pacote de rede é um arquivo.zip no formato CSAR (Cloud Service Archive). Ele define os pacotes de funções que você deseja implantar e a AWS infraestrutura na qual deseja implantá-los.

O pacote de rede contém os seguintes arquivos:

- Um arquivo descritor de rede (`nsd.yaml`) no formato TOSCA, conforme descrito pelo ETSI. SOL007

  O `nsd.yaml` arquivo contém referências aos pacotes de funções enviados com seu descritor IDs.
- Scripts de dados do usuário, se houver.
- Scripts de gancho de ciclo de vida, se houver.
- Arquivos de `values.yaml` configuração dos plug-ins, se houver.

Tarefas

- Crie um pacote de rede no AWS TNB
- Exibir um pacote de rede no AWS TNB
- Baixe um pacote de rede do AWS TNB
- Exclua um pacote de rede do AWS TNB

Crie um pacote de rede no AWS TNB

Um pacote de rede consiste em um arquivo descritor de serviço de rede (NSD) (obrigatório) e quaisquer arquivos adicionais (opcionais), como scripts específicos às suas necessidades. Por exemplo, se você tiver vários pacotes de funções em seu pacote de rede, poderá usar o NSD para definir quais funções de rede devem ser executadas em determinadas VPCs sub-redes ou clusters do Amazon EKS.

Crie um pacote de rede depois de criar pacotes de funções. Depois de criar um pacote de rede, você precisa criar uma instância de rede.

Console

Para criar um pacote de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Pacotes de rede.
3. Escolha Criar pacote de rede.
4. Escolha Escolher arquivos e carregue cada NSD como um `.zip` arquivo. Você pode fazer upload de no máximo 10 arquivos.
5. Escolha Próximo.
6. Revise os detalhes do pacote.
7. Escolha Criar pacote de rede.

AWS CLI

Para criar um pacote de rede usando o AWS CLI

1. Use o create-sol-network-packagecomando para criar um pacote de rede.

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

2. Use o comando put-sol-network-package-content para carregar o conteúdo do pacote de rede. Por exemplo:

```
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
```

Exibir um pacote de rede no AWS TNB

Saiba como exibir o conteúdo de um pacote de rede.

Console

Para exibir um pacote de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Pacotes de rede.
3. Use a caixa de pesquisa para encontrar o pacote de rede.

AWS CLI

Para visualizar um pacote de rede usando o AWS CLI

1. Use o list-sol-network-packagescomando para listar seus pacotes de rede.

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

2. Use o get-sol-network-packagecomando para ver detalhes sobre um pacote de rede.

```
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
```

Baixe um pacote de rede do AWS TNB

Saiba como baixar um pacote de rede do catálogo de serviços de rede da AWS TNB.

Console

Para baixar um pacote de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Pacotes de rede.
3. Use a caixa de pesquisa para encontrar o pacote de rede
4. Escolha o pacote de rede.
5. Em Ações, escolha Baixar.

AWS CLI

Para baixar um pacote de rede usando o AWS CLI

- Use o comando get-sol-network-package-content para baixar um pacote de rede.

```
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
```

Exclua um pacote de rede do AWS TNB

Saiba como excluir um pacote de rede do catálogo de serviços de rede AWS TNB. Para excluir um pacote de rede, é preciso que ele esteja desabilitado.

Console

Para excluir um pacote de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Pacotes de rede.
3. Use a caixa de pesquisa para encontrar o pacote de rede
4. Escolher o pacote de rede
5. Escolha Ações, Desabilitar.
6. Escolha Ações, Excluir.

AWS CLI

Para excluir um pacote de rede usando o AWS CLI

1. Use o update-sol-network-packagecomando para desativar um pacote de rede.

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

2. Use o delete-sol-network-packagecomando para excluir um pacote de rede.

```
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
```

Instâncias de rede para AWS TNB

Uma instância de rede é uma rede única criada no AWS TNB que pode ser implantada.

Tarefas

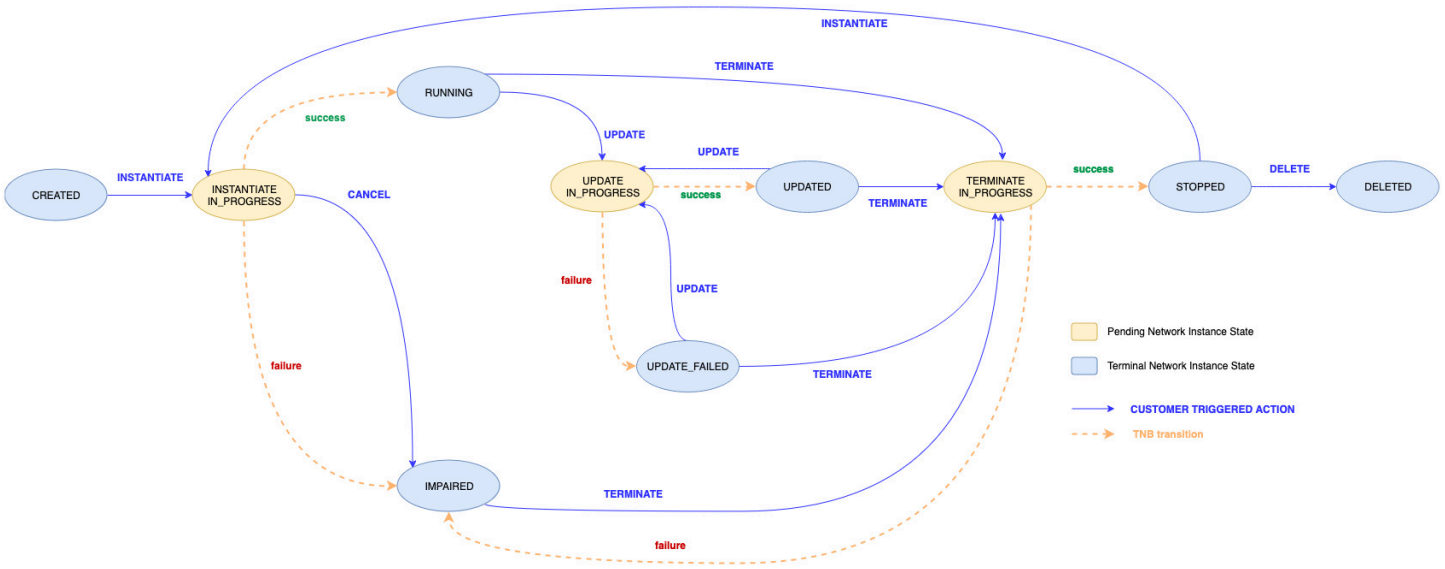
- Operações do ciclo de vida de uma instância de rede
- Crie uma instância de rede usando AWS TNB
- Instancie uma instância de rede usando TNB AWS
- Atualizar uma instância de função no AWS TNB
- Atualizar uma instância de rede no AWS TNB
- Veja uma instância de rede no AWS TNB
- Encerrar e excluir uma instância de rede do TNB AWS

Operações do ciclo de vida de uma instância de rede

AWS O TNB permite que você gerencie facilmente sua rede usando operações de gerenciamento padronizadas alinhadas com o ETSI e. SOL003 SOL005 Você pode realizar as seguintes operações de ciclo de vida:

- Crie a rede
- Instancie a rede
- Atualize a função de rede
- Atualizar a instância de rede
- Exibir detalhes e status da rede
- Encerrar a rede

A imagem a seguir mostra as operações de gerenciamento de rede:



Crie uma instância de rede usando AWS TNB

Você cria uma instância de rede depois de criar um pacote de rede. Depois de criar uma instância de rede, instancie-a.

Console

Para criar uma instância de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Redes.
3. Clique em Criar instância de rede.
4. Insira um nome e uma descrição para a instância e, em seguida, escolha Próximo.
5. Selecione o pacote de rede, verifique os detalhes e escolha Avançar.
6. Clique em Criar instância de rede.

   A nova instância de rede aparece na página Redes. Em seguida, instancie essa instância de rede.

AWS CLI

Para criar uma instância de rede usando o AWS CLI

- Use o create-sol-network-instancecomando para criar uma instância de rede.

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

Em seguida, instancie essa instância de rede.

Instancie uma instância de rede usando TNB AWS

Depois de criar uma instância de rede, você deve instanciá-la. Quando você instancia uma instância de rede, o AWS TNB provisiona a AWS infraestrutura necessária, implanta funções de rede em contêineres e configura o gerenciamento de rede e acesso para criar um serviço de rede totalmente operacional.

Console

Para instanciar uma instância de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Redes.
3. Selecione a instância de rede que você deseja instanciar.
4. Escolha Ações e, em seguida, Instanciar.
5. Na página Instanciar rede, revise os detalhes e, opcionalmente, atualize os valores dos parâmetros.

   As atualizações nos valores dos parâmetros se aplicam somente a essa instância de rede. Os parâmetros nos pacotes NSD e VNFD não são alterados.
6. Escolha Instanciar rede.

   A página de status de implantação é exibida.
7. Use o ícone Atualizar para rastrear o status de implantação da sua instância de rede. Você também pode ativar a atualização automática na seção Tarefas de implantação para acompanhar o progresso de cada tarefa.

   Quando o status de implantação muda para`Completed`, a instância de rede é instanciada.

AWS CLI

Para instanciar uma instância de rede usando o AWS CLI

1. Use o instantiate-sol-network-instancecomando para instanciar a instância de rede.

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

2. Em seguida, visualize o status da operação da rede.

Atualizar uma instância de função no AWS TNB

Depois que uma instância de rede é instanciada, você pode atualizar um pacote de funções na instância de rede.

Console

Para atualizar uma instância de função usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Redes.
3. Selecione a instância de rede. Você pode atualizar uma instância de rede somente se seu estado for`Instantiated`.

   A página da instância de rede é exibida.
4. Na guia Funções, selecione a instância da função a ser atualizada.
5. Selecione Atualizar.
6. Insira suas substituições de atualização.
7. Selecione Atualizar.

AWS CLI

Use a CLI para atualizar uma instância de função

Use o update-sol-network-instancecomando com o tipo de `MODIFY_VNF_INFORMATION` atualização para atualizar uma instância de função em uma instância de rede.

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

Atualizar uma instância de rede no AWS TNB

Depois que uma instância de rede for instanciada, talvez seja necessário atualizar a infraestrutura ou o aplicativo. Para fazer isso, você atualiza o pacote de rede e os valores dos parâmetros da instância de rede e implanta a operação de atualização para aplicar as alterações.

Considerações

- Você pode atualizar uma instância de rede que esteja no `Updated` estado `Instantiated` ou.
- Quando você atualiza uma instância de rede, a `UpdateSolNetworkService` API usa o novo pacote de rede e os valores dos parâmetros para atualizar a topologia da instância de rede.
- AWS O TNB verifica se o número de parâmetros NSD e VNFD na instância de rede não excede 200. Esse limite é aplicado para evitar que agentes mal-intencionados transmitam cargas errôneas ou enormes que afetam o serviço.

Parâmetros que você pode atualizar

Você pode atualizar os seguintes parâmetros ao atualizar uma instância de rede instanciada:

Parâmetro	Description	Exemplo: Antes	Exem Depo

Versão do cluster Amazon EKS	Você pode atualizar o valor do `version` parâmetro do plano de controle do cluster Amazon EKS para a próxima versão secundária. Você não pode fazer o downgrade da versão.	`EKSCluster:`
		`  type: tosca.nod`
		`es.AWS.Compute.EKS`
		`  properties:`
		`  version: "1.28"`
			`EKSC`
			`r:`
			
			`typ`
			`tos`
			`es.A`
			`mput`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			pro s: ver "1.

Parâmetro	Description	Exemplo: Antes	Exem Depo

Nós de processamento do Amazon EKS	Você pode atualizar o valor do `EKSManagedNode` `kubernetes_version` parâmetro para atualizar seu grupo de nós para uma versão mais recente do Amazon EKS ou pode atualizar o `ami_id` parâmetro para atualizar seu grupo de nós para a AMI otimizada para EKS mais recente.
	
	Você pode atualizar o ID da AMI para`EKSSelfMa` `nagedNode` . A versão do Amazon EKS da AMI deve ser igual ou até duas versões inferior à versão do cluster do Amazon EKS. Por exemplo, se a versão do cluster Amazon EKS for 1.31, a versão da Amazon EKS AMI deverá ser 1.31, 1.30 ou 1.29.
		`EKSManagedNodeGroup01:`
		`...`
		`properties:`
		`kubernete s_version: " 1.28"`
		`EKSSelfManagedNode 01:`
		`compute:`
		`compute:`
		`properties:`
		`ami_id: "ami-1231230LD "`
			`EKSM dNod p01:`
			`...`
			`pro s:`
			`kub s_ve : "1.`
			`EKS nage 01:`
			`com`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			com pro s: ami "am 3NEW

Parâmetro	Description	Exemplo: Antes	Exem Depo

Grupos de nós do Amazon EKS	Você pode adicionar ou remover grupos de nós de acordo com suas necessidades computacionais.
	
	Ao excluir grupos de nós existentes e adicionar novos, certifique-se de que os novos grupos de nós sejam diferentes IDs dos grupos de nós excluídos, caso contrário, a operação será tratada como uma modificação do grupo de nós em vez de uma exclusão e adição. Observe que, para grupos de nós existentes, somente um conjunto limitado de parâmetros pode ser atualizado. Percorra esta tabela para ver quais parâmetros você pode atualizar.
		`Free5GCEKSNode01:`
		`  type: tosca.nodes.AWS.Compute.EKSManagedNode`
		`  ...`
		`  scaling:`
		`  properties:`
		`  desired_size: 1`
		`  min_size: 1`
		`  max_size: 1`
		`  ...`
		`Free5GCEKSNode02` `: # Deleted Nodegroup`
		`  type: tosca.nodes.AWS.Compute.EKSManagedNode`
		`  ...`
		`  scaling:`
		`  properties:`
		`  desired_size: 1`
		`  min_size: 1`
		`  max_size: 1`
		`  ...`
		`Free5GCEKSNode03` `: # Deleted Nodegroup`
		`  type: tosca.nodes.AWS.Compute.EKSSelfManagedNode`
		`  ...`
		`  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`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			mir 1 max 1 ... Free SNod # New Nod typ tos es.A mput Self edNo ... sca pro s:

Parâmetro	Description	Exemplo: Antes	Exem Depo
			des ize: 1 mir 1 max 1 ... *Free* *SNod* # New Nod typ tos es.A mput Mana de

Parâmetro	Description	Exemplo: Antes	Exem Depo
			... sca pro s: des ize: 1 mir 1 max 1

Parâmetro	Description	Exemplo: Antes	Exem Depo
			...

Parâmetro	Description	Exemplo: Antes	Exem Depo

Propriedades de escala	Você pode atualizar as propriedades de dimension amento dos nós `EKSManage dNode` e `EKSSelfMa nagedNode` TOSCA.	`EKSNodeGroup01:`
		`  ...`
		`  scaling:`
		`  properties:`
		`  desired_s ize: 1`
		`  min_size: 1`
		`  max_size: 1`
			`EKSN oup0`
			`...`
			`sca`
			`pro s:`
			`des ize:`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			mir max

Parâmetro	Description	Exemplo: Antes	Exem Depo

Propriedades do plug-in Amazon EBS CSI	Você pode ativar ou desativar o plug-in Amazon EBS CSI em seus clusters do Amazon EKS. Você também pode alterar a versão do plugin.	`EKSCluster:`
		`  capabilities:`
		`  ...`
		`  ebs_csi:`
		`  properties:`
		`  enabled: false`
			`EKSC`
			`r:`
			`cap`
			`ies:`
			`...`
			`ebs`
			`pro`
			`s:`
			`ena`
			`ver`
			`"v1`
			`e`
			`ksbu`
			`"`

Parâmetro	Description	Exemplo: Antes	Exem Depo

Tamanho do volume raiz	Você pode adicionar, remover ou atualizar a propriedade do tamanho do volume raiz dos nós EKSManaged Node e EKSSelf ManagedNode TOSCA.	`Free5GCEKSNode01:`
		`  ...`
		`  capabilities:`
		`  compute:`
		`  properties:`
		`  root_volume_size: 50`
			Free SNod
			...
			cap ies:
			com
			pro s:

Parâmetro	Description	Exemplo: Antes	Exem Depo
			roo me_s

Parâmetro	Description	Exemplo: Antes	Exem Depo

VNF	Você pode referenciá-los VNFs no NSD e implantá-los no cluster criado no NSD usando o nó VNFDeployment TOSCA. Como parte da atualização, você poderá adicionar, atualizar e excluir VNFs na rede.	`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`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			elmD :
			typ tos es.A ploy VNFD ment
			rec nts:
			clu EKS r
			vnf

Parâmetro	Description	Exemplo: Antes	Exem Depo
			- v leVN - v leVN

Parâmetro	Description	Exemplo: Antes	Exem Depo

Hooks	Para executar operações de ciclo de vida antes e depois de criar uma função de rede, adicione os `post_create` ganchos `pre_create` e ao `VNFDeployment` nó.
	
	Neste exemplo, o `PreCreate Hook` gancho será executado antes de ser `vnf3.Samp leVNF3` instanciado e o `PostCreateHook` gancho será executado depois de ser `vnf3.SampleVNF3` instanciado.
		`vnfds:`
		`  - descriptor_id:`
		`"43c012fa-2616-41a8-a833-0dfd4c5a049e "`
		`    namespace: " vnf1"`
		`  - descriptor_id:`
		`"64222f98-ecd6-4871-bf94-7354b53f3ee5 "`
		`    namespace: " vnf2"`
		`...`
		`SampleVNF1HelmDeploy:`
		`  type: tosca.nod es.AWS.Deployment. VNFDeployment`
		`  requirements:`
		`    cluster: EKSCluster`
		`    vnfs:`
		`      - vnf1.SampleVNF1`
		`      - vnf2.Samp leVNF2 // 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`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			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

Parâmetro	Description	Exemplo: Antes	Exem Depo
			- v leVN New VNF as the nam , vnf was not pre y pre int s: Hoo pos te: eHoo pre e: Hook

Parâmetro	Description	Exemplo: Antes	Exem Depo

Hooks	Para executar operações de ciclo de vida antes e depois de atualizar uma função de rede, você pode adicionar o `pre_update` gancho e o `post_update` gancho ao `VNFDeployment` nó.
	
	Neste exemplo, `PreUpdate Hook` será executado antes da `vnf1.SampleVNF1` atualização e `PostUpdat eHook` será executado após `vnf1.SampleVNF1` a atualização para o `vnf` pacote indicado pela atualização `uuid` para o namespace vnf1.
		`vnfds:`
		`  - descriptor_id:`
		`  "43c012fa-2616-41a8-a833-0dfd4c5a049e  "`
		`  namespace: " vnf1"`
		`  - descriptor_id:`
		`  "64222f98-ecd6-4871-bf94-7354b53f3ee5  "`
		`  namespace: " vnf2"`
		`...`
		
		`SampleVNF1HelmDeploy:`
		`  type: tosca.nod es.AWS.Deployment. VNFDeployment`
		`  requirements:`
		`  cluster: EKSCluster`
		`  vnfs:`
		`  - vnf1.SampleVNF1`
		`  - vnf2.Samp leVNF2`
			`vnfd`
			
			`  -`
			`  des r_id`
			`  "0e bd87 -`
			`  b8a1 4666`
			`  "`
			
			`  nam :`
			`  "vn`
			
			`  -`
			`  des r_id`
			`  "64 ecd6 -`
			`  bf94 4b53`
			`  "`
			
			`  nam :`
			`  "vn`
			`...`
			`S ampl Helm y:`
			
			`  typ`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			tos es.A ploy VNFD ment rec nts: clu EKS r vnf - v leVN A VNF upo as the uui cha for nam "vr - v

Parâmetro	Description	Exemplo: Antes	Exem Depo
			leVN No cha to thi fur as nam and uui rem the sam int s: Hoo pre e: Hook pos te: eHoo

Parâmetro	Description	Exemplo: Antes	Exem Depo

Sub-redes	Você pode adicionar e excluir sub-redes da rede. Antes de excluir uma sub-rede, verifique se a sub-rede não é usada por nenhum recurso na rede.	`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:`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			rou le: Fre uteT vpc Fre C

Parâmetro	Description	Exemplo: Antes	Exem Depo
Grupos de segurança	Você pode adicionar e excluir grupos de segurança da rede. Antes de excluir um grupo de segurança, verifique se o grupo de segurança não é usado por nenhum recurso na rede.	*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

Parâmetro	Description	Exemplo: Antes	Exem Depo
		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:

Parâmetro	Description	Exemplo: Antes	Exem Depo
			ip_ ol: "to fro : 800 to_ 900 des on: "Eg Rul for fre clu cid "17 0.1/ req nts: sec grou Fre

Parâmetro	Description	Exemplo: Antes	Exem Depo

```
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
```

Parâmetro	Description	Exemplo: Antes	Exem Depo
			des on: "Ir Rul for fre clu cid "17 0.1/ rec nts: sec grou Fre curi up02

Parâmetro	Description	Exemplo: Antes	Exem Depo

Interfaces de rede	Você pode adicionar, modificar e excluir ENIs da rede.	`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`

Parâmetro	Description	Exemplo: Antes	Exem Depo
			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

Parâmetro	Description	Exemplo: Antes	Exem Depo
			- 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

Parâmetro	Description	Exemplo: Antes	Exem Depo
			- Fre curi up01

Atualização de uma instância de rede

Console

Para atualizar uma instância de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Redes.
3. Selecione a instância de rede. Você pode atualizar uma instância de rede somente se seu estado for `Instantiated` ou`Updated`.
4. Escolha Ações e atualização.

   A página Atualizar instância aparece com os detalhes da rede e uma lista de parâmetros na infraestrutura atual.
5. Escolha um novo pacote de rede.

   Os parâmetros no novo pacote de rede aparecem na seção Parâmetros atualizados.
6. Opcionalmente, atualize os valores dos parâmetros na seção Parâmetros atualizados. Para obter a lista de valores de parâmetros que você pode atualizar, consulteParâmetros que você pode atualizar.
7. Escolha Atualizar rede.

   AWS O TNB valida a solicitação e inicia a implantação. A página de status de implantação é exibida.

8. Use o ícone Atualizar para rastrear o status de implantação da sua instância de rede. Você também pode ativar a atualização automática na seção Tarefas de implantação para acompanhar o progresso de cada tarefa.

   Quando o status de implantação muda para`Completed`, a instância de rede é atualizada.

9.
   - Se a validação falhar, a instância de rede permanecerá no mesmo estado em que estava antes de você solicitar a atualização - `Instantiated` ou`Updated`.
   - Se a atualização falhar, o estado da instância da rede será exibido`Update failed`. Escolha o link para cada tarefa que falhou para determinar o motivo.
   - Se a atualização for bem-sucedida, o estado da instância da rede será exibido`Updated`.

AWS CLI

Usar a CLI para atualizar uma instância de rede

Use o update-sol-network-instancecomando com o tipo de `UPDATE_NS` atualização para atualizar uma instância de rede.

```
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\"}}"
```

Veja uma instância de rede no AWS TNB

Saiba como exibir uma instância de rede.

Console

Para exibir uma instância de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, escolha Interfaces de rede.
3. Use a caixa de pesquisa para encontrar a instância de rede.

AWS CLI

Para visualizar uma instância de rede usando o AWS CLI

1. Use o list-sol-network-instancescomando para listar suas instâncias de rede.

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

2. Use o get-sol-network-instancecomando para ver detalhes sobre uma instância de rede específica.

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

Encerrar e excluir uma instância de rede do TNB AWS

Para excluir uma instância de rede, é preciso que ela esteja encerrada.

Console

Para encerrar e excluir uma instância de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Redes.
3. Selecione o ID da instância de rede.
4. Escolha Encerrar.
5. Quando receber a solicitação de confirmação, insira o ID e escolha Encerrar.
6. Atualize para rastrear o status da instância de rede.
7. (Opcional) Selecione a instância de rede e escolha Excluir.

AWS CLI

Para encerrar e excluir uma instância de rede usando o AWS CLI

1. Use o terminate-sol-network-instancecomando para encerrar uma instância de rede.

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

2. (Opcional) Use o delete-sol-network-instancecomando para excluir uma instância de rede.

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

Operações de rede para AWS TNB

Uma operação de rede é qualquer operação feita em sua rede, como instanciação ou encerramento de instância de rede.

Tarefas

- Exibir uma operação de rede AWS TNB
- Cancelar uma operação de rede AWS TNB

Exibir uma operação de rede AWS TNB

Exiba os detalhes de uma operação de rede, incluindo as tarefas envolvidas na operação de rede e o status das tarefas.

Console

Para exibir uma operação de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, escolha Interfaces de rede.
3. Use a caixa de pesquisa para encontrar a instância de rede.
4. Na guia Implantações, escolha a operação de rede.

AWS CLI

Para visualizar uma operação de rede usando o AWS CLI

1. Use o list-sol-network-operationscomando para listar todas as operações de rede.

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

2. Use o get-sol-network-operationcomando para ver detalhes sobre uma operação de rede.

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

Cancelar uma operação de rede AWS TNB

Saiba como cancelar uma operação de rede.

Console

Para cancelar uma operação de rede usando o console

1. Abra o console AWS TNB em. https://console.aws.amazon.com/tnb/
2. No painel de navegação, selecione Redes.
3. Selecione o ID da rede para abrir sua página de detalhes.
4. Na guia Implantações, escolha a operação de rede.
5. Escolha Cancelar operação.

AWS CLI

Para cancelar uma operação de rede usando o AWS CLI

Use o cancel-sol-network-operationcomando para cancelar uma operação de rede.

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

Referência TOSCA para TNB AWS

A Especificação de Topologia e Orquestração para Aplicativos em Nuvem (TOSCA) é uma sintaxe declarativa CSPs usada para descrever uma topologia de serviços web baseados em nuvem, seus componentes, relacionamentos e os processos que os gerenciam. CSPs descreva os pontos de conexão, os links lógicos entre os pontos de conexão e as políticas, como afinidade e segurança, em um modelo TOSCA. CSPs em seguida, faça o upload do modelo para o AWS TNB, que sintetiza os recursos necessários para estabelecer uma rede 5G funcional em todas AWS as zonas de disponibilidade.

Conteúdo

Modelo de VNFD

Define um modelo de descritor de função de rede virtual (VNFD).

Sintaxe

```
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
```

Modelo de topologia

`node_templates`

Os AWS nós TOSCA. Os nós possíveis são:

- AWS.VNF
- AWS.Artifacts.Helm

AWS.VNF

Define um AWS nó de função de rede virtual (VNF).

Sintaxe

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

Propriedades

`descriptor_id`

O UUID do descritor.

Obrigatório: sim

Tipo: string

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

`descriptor_version`

A versão do VNFD.

Obrigatório: Sim

Tipo: string

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

`descriptor_name`

O nome do descritor.

Obrigatório: Sim

Tipo: String

`provider`

O autor do VNFD.

Obrigatório: Sim

Tipo: String

Requisitos

`helm`

O diretório Helm que define artefatos de contêiner. Essa é uma referência a AWS.Artifacts.Helm.

Obrigatório: Sim

Tipo: String

Exemplo

```
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 um AWS Helm Node.

Sintaxe

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

Propriedades

`implementation`

O diretório local que contém o chart do Helm no pacote CSAR.

Obrigatório: Sim

Tipo: String

Exemplo

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

Modelo de descritor de serviço de rede

Define um modelo de descritor de serviço de rede (NSD).

Sintaxe

```
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 parâmetros definidos

Quando quiser passar um parâmetro dinamicamente, como o bloco CIDR para o nó VPC, você pode usar a sintaxe `{ get_input: input-parameter-name }` e definir os parâmetros no modelo de NSD. Em seguida, reutilize o parâmetro no mesmo modelo de NSD.

O exemplo a seguir mostra como definir e usar parâmetros:

```
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 }
```

Importação de VNFD

`descriptor_id`

O UUID do descritor.

Obrigatório: sim

Tipo: sequência

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

`namespace`

O nome exclusivo.

Obrigatório: Sim

Tipo: string

Modelo de topologia

`node_templates`

Os possíveis AWS nós do TOSCA são:

- 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 um nó de serviço de AWS rede (NS).

Sintaxe

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

Propriedades

`descriptor_id`

O UUID do descritor.

Obrigatório: sim

Tipo: sequência

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

`descriptor_version`

A versão do NSD.

Obrigatório: Sim

Tipo: sequência

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

`descriptor_name`

O nome do descritor.

Obrigatório: Sim

Tipo: string

Exemplo

```
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

Forneça o nome do cluster, a versão desejada do Kubernetes e uma função que permita que o plano de controle do Kubernetes gerencie os recursos necessários para seus NFs. AWS Os plug-ins da interface de rede de contêineres (CNI) Multus estão habilitados. Você pode conectar várias interfaces de rede e aplicar configurações avançadas de rede às funções Kubernetes-based de rede. Você também especifica o acesso ao endpoint do cluster e as sub-redes do seu cluster.

Sintaxe

```
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. Propriedades que definem o uso da interface de rede de contêineres (CNI) Multus.

Se você incluir `multus`, especifique as propriedades `enabled` e `multus_role`.

`enabled`

Indica se o recurso Multus padrão está habilitado.

Obrigatório: Sim

Tipo: booliano

`multus_role`

O perfil do gerenciamento da interface de rede Multus.

Obrigatório: Sim

Tipo: string

`ebs_csi`

Propriedades que definem o driver da CSI (Container Storage Interface) do Amazon EBS instalado no cluster do Amazon EKS.

Habilite esse plug-in para usar os nós autogerenciados do Amazon EKS em AWS Outposts AWS Locais Zones ou Regiões da AWS. Para obter mais informações, consulte Driver da CSI do Amazon EBS no Guia do usuário do Amazon EKS.

`enabled`

Indica se o driver padrão da CSI do Amazon EBS está instalado.

Obrigatório: não

Tipo: booliano

`version`

A versão do complemento do driver da CSI do Amazon EBS. A versão deve corresponder a uma das versões retornadas pela DescribeAddonVersionsação. Para obter mais informações, consulte DescribeAddonVersionsa Referência da API Amazon EKS

Obrigatório: não

Tipo: string

Propriedades

`version`

A versão do Kubernetes para o cluster. AWS O Telco Network Builder oferece suporte às versões 1.27 a 1.34 do Kubernetes.

Obrigatório: Sim

Tipo: string

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

`access`

Acesso ao endpoint do cluster.

Obrigatório: Sim

Tipo: string

Valores possíveis: `PRIVATE` | `PUBLIC` | `ALL`

`cluster_role`

O perfil do gerenciamento de clusters.

Obrigatório: Sim

Tipo: string

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

`ip_family`

Indica a família de IP para endereços de serviço e pod no cluster.

Valor permitido: `IPv4`, `IPv6`

Valor padrão: `IPv4`

Obrigatório: não

Tipo: string

Requisitos

`subnets`

Um AWS. Networking.Subnetnodo.

Obrigatório: Sim

Tipo: lista

Exemplo

```
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

Um AuthRole permite que você adicione funções do IAM ao cluster `aws-auth ConfigMap` do Amazon EKS para que os usuários possam acessar o cluster do Amazon EKS usando uma função do IAM.

Sintaxe

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

Propriedades

`role_mappings`

Lista de mapeamentos que definem perfis do IAM que precisam ser adicionadas ao cluster `aws-auth ConfigMap` do Amazon EKS.

`arn`

O ARN do perfil do IAM.

Obrigatório: Sim

Tipo: string

`groups`

Grupos do Kubernetes a serem atribuídos ao perfil definido em `arn`.

Obrigatório: não

Tipo: lista

Requisitos

`clusters`

Um AWS. Compute.EKSnodo.

Obrigatório: Sim

Tipo: lista

Exemplo

```
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 O TNB oferece suporte a grupos de nós gerenciados do EKS para automatizar o provisionamento e o gerenciamento do ciclo de vida dos nós (instâncias do Amazon EC2) para clusters do Amazon EKS Kubernetes. Para criar um grupo de nós EKS, faça o seguinte:

- Escolha as Amazon Machine Images (AMI) para seus nós de trabalho de cluster fornecendo a ID da AMI ou o tipo de AMI.

- Forneça um par de chaves do Amazon EC2 para acesso SSH e as propriedades de escalabilidade para seu grupo de nós.
- Certifique-se de que seu grupo de nós esteja associado a um cluster do Amazon EKS.
- Forneça as sub-redes para os nós de trabalho.
- Opcionalmente, anexe grupos de segurança, rótulos de nós e um grupo de posicionamento ao seu grupo de nós.

Sintaxe

```
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

computação

Propriedades que definem os parâmetros de computação para o grupo de nós gerenciados do Amazon EKS, como tipos de instância do Amazon EC2 e AMIs de instância do Amazon EC2.

`ami_type`

O tipo Amazon EKS-supported AMI.

Obrigatório: Sim

Tipo: string

Valores possíveis: `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`

O ID da AMI.

Obrigatório: não

Tipo: string

> **Note**
>
> Se ambos `ami_type` `ami_id` forem especificados no modelo, o AWS TNB usará somente o `ami_id` valor para criar`EKSManagedNode`.

`instance_types`

O tamanho da instância.

Obrigatório: Sim

Tipo: lista

`key_pair`

O par de chaves do EC2 para habilitar o acesso SSH.

Obrigatório: Sim

Tipo: string

`root_volume_encryption`

Ativa a criptografia do Amazon EBS para o volume raiz do Amazon EBS. Se essa propriedade não for fornecida, o AWS TNB criptografará os volumes raiz do Amazon EBS por padrão.

Obrigatório: não

Padrão: verdadeiro

Tipo: booliano

`root_volume_encryption_key_arn`

O ARN da chave. AWS KMS AWS O TNB suporta ARN de chave regular, ARN de chave multirregional e ARN de alias.

Obrigatório: não

Tipo: string

> **Note**
>
> - Se `root_volume_encryption` for falso, não inclua`root_volume_encryption_key_arn`.
> - AWS O TNB oferece suporte à criptografia do volume raiz das EBS-backed AMIs da Amazon.
> - Se o volume raiz da AMI já estiver criptografado, você deverá incluir o `root_volume_encryption_key_arn` para que o AWS TNB recriptografe o volume raiz.
> - Se o volume raiz da AMI não estiver criptografado, o AWS TNB usará o `root_volume_encryption_key_arn` para criptografar o volume raiz.
>
>   Se você não incluir`root_volume_encryption_key_arn`, o AWS TNB usa a chave padrão fornecida por AWS Key Management Service para criptografar o volume raiz.
> - AWS O TNB não decifra uma AMI criptografada.

`root_volume_size`

O tamanho do volume raiz do Amazon Elastic Block Store em GiBs.

Obrigatório: não

Padrão: 20

Tipo: inteiro

Valores possíveis: 1 a 16.384

`Em escala`

Propriedades que definem os parâmetros de escalabilidade para o grupo de nós gerenciados do Amazon EKS, como o número desejado de instâncias do Amazon EC2 e os números mínimo e máximo de instâncias do Amazon EC2 no grupo de nós.

`desired_size`

O número de instâncias neste NodeGroup.

Obrigatório: Sim

Tipo: inteiro

`min_size`

O número mínimo de instâncias neste NodeGroup.

Obrigatório: Sim

Tipo: inteiro

`max_size`

O número máximo de instâncias neste NodeGroup.

Obrigatório: Sim

Tipo: inteiro

Propriedades

`node_role`

O ARN do perfil do IAM anexado à instância do Amazon EC2.

Obrigatório: Sim

Tipo: string

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

`kubernetes_version`

A versão do Kubernetes para o grupo de nós gerenciados. AWS O TNB é compatível com as versões 1.27 a 1.34 do Kubernetes. Considere o seguinte:

- Especifique o `kubernetes_version` ou`ami_id`. Não especifique ambos.
- O `kubernetes_version` deve ser menor ou igual ao AWS. Compute.EKSManagedNode versão.
- Pode haver uma diferença de 3 versões entre AWS o. Compute.EKSManagedNode versão `kubernetes_version` e.
- Se nenhum `kubernetes_version` ou `ami_id` for especificado, o AWS TNB usará a AMI mais recente da `AWS.Compute.EKSManagedNode` versão para criar `EKSManagedNode`

Obrigatório: não

Tipo: string

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

Requisitos

`cluster`

Um AWS. Compute.EKSnodo.

Obrigatório: Sim

Tipo: string

`subnets`

Um AWS. Networking.Subnetnodo.

Obrigatório: Sim

Tipo: lista

`network_interfaces`

Um AWS. Networking.ENInodo. Certifique-se de que as interfaces de rede e sub-redes estejam definidas com a mesma zona de disponibilidade, senão a instanciação falhará.

Quando você define`network_interfaces`, o AWS TNB obtém a permissão relacionada aos ENIs da `multus_role` propriedade se você incluiu a `multus` propriedade no nó. AWS.Compute.EKS Caso contrário, o AWS TNB recebe a permissão relacionada a ENIs da propriedade node_role.

Obrigatório: não

Tipo: lista

`security_groups`

Um AWS. Networking.SecurityGroupnodo.

Obrigatório: não

Tipo: lista

`placement_group`

Um tosca.nodes.AWS. Compute.PlacementGroupnodo.

Obrigatório: não

Tipo: string

`user_data`

Um tosca.nodes.AWS. Compute.UserDatareferência de nó. Um script de dados de usuário é transmitido às instâncias do Amazon EC2 iniciadas pelo grupo de nós gerenciados. Adicione as permissões necessárias para executar dados de usuário personalizados no node_role transmitido ao grupo de nós.

Obrigatório: não

Tipo: string

`labels`

Uma lista de rótulos de nós. Um rótulo de nó deve ter um nome e um valor. Crie um rótulo usando os seguintes critérios:

- O nome e o valor devem ser separados por=.
- O nome e o valor podem ter, cada um, até 63 caracteres.
- O rótulo pode incluir letras (A-Z, a-z,), números (0-9) e os seguintes caracteres: `[-, _, ., *, ?]`
- O nome e o valor devem começar e terminar com um caractere alfanumérico ou * caractere. ?

Por exemplo, `myLabelName1=*NodeLabelValue1`.

Obrigatório: não

Tipo: lista

Exemplo

```
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 O TNB oferece suporte aos nós autogerenciados do Amazon EKS para automatizar o provisionamento e o gerenciamento do ciclo de vida dos nós (instâncias do Amazon EC2) para clusters do Amazon EKS Kubernetes. Para criar um grupo de nós do Amazon EKS, faça o seguinte:

- Escolha as Amazon Machine Images (AMI) para seus nós de trabalho de cluster fornecendo o ID da AMI.
- Forneça um par de chaves do Amazon EC2 para acesso SSH.
- Certifique-se de que seu grupo de nós esteja associado a um cluster do Amazon EKS.
- Forneça o tipo de instância e os tamanhos desejados, mínimos e máximos.
- Forneça as sub-redes para os nós de trabalho.
- Opcionalmente, anexe grupos de segurança, rótulos de nós e um grupo de posicionamento ao seu grupo de nós.

Sintaxe

```
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

computação

Propriedades que definem os parâmetros de computação para os nós autogerenciados do Amazon EKS, como tipos de instância do Amazon EC2 e AMIs de instância do Amazon EC2.

`ami_id`

O ID da AMI usado para iniciar a instância. AWS O TNB oferece suporte a instâncias que utilizam o IMDSv2. Para obter mais informações, consulte Versão do IMDS.

> **Note**
>
> Você pode atualizar o ID da AMI para`EKSSelfManagedNode`. A versão do Amazon EKS da AMI deve ser igual ou até duas versões inferior à versão do cluster do Amazon EKS. Por exemplo, se a versão do cluster Amazon EKS for 1.31, a versão da Amazon EKS AMI deverá ser 1.31, 1.30 ou 1.29.

Obrigatório: Sim

Tipo: string

`instance_type`

O tamanho da instância.

Obrigatório: Sim

Tipo: string

`key_pair`

O par de chaves do Amazon EC2 para permitir o acesso SSH.

Obrigatório: Sim

Tipo: string

`root_volume_encryption`

Ativa a criptografia do Amazon EBS para o volume raiz do Amazon EBS. Se essa propriedade não for fornecida, o AWS TNB criptografará os volumes raiz do Amazon EBS por padrão.

Obrigatório: não

Padrão: verdadeiro

Tipo: booliano

`root_volume_encryption_key_arn`

O ARN da chave. AWS KMS AWS O TNB suporta ARN de chave regular, ARN de chave multirregional e ARN de alias.

Obrigatório: não

Tipo: string

> **Note**
>
> - Se `root_volume_encryption` for falso, não inclua`root_volume_encryption_key_arn`.

- AWS O TNB oferece suporte à criptografia do volume raiz das EBS-backed AMIs da Amazon.
- Se o volume raiz da AMI já estiver criptografado, você deverá incluir o `root_volume_encryption_key_arn` para que o AWS TNB recriptografe o volume raiz.
- Se o volume raiz da AMI não estiver criptografado, o AWS TNB usará o `root_volume_encryption_key_arn` para criptografar o volume raiz.

  Se você não incluir`root_volume_encryption_key_arn`, o AWS TNB usa AWS Managed Services para criptografar o volume raiz.
- AWS O TNB não decifra uma AMI criptografada.

`root_volume_size`

O tamanho do volume raiz do Amazon Elastic Block Store em GiBs.

Obrigatório: não

Padrão: 20

Tipo: inteiro

Valores possíveis: 1 a 16.384

Em escala

Propriedades que definem os parâmetros de escalabilidade para os nós autogerenciados do Amazon EKS, como o número desejado de instâncias do Amazon EC2 e os números mínimo e máximo de instâncias do Amazon EC2 no grupo de nós.

`desired_size`

O número de instâncias neste NodeGroup.

Obrigatório: Sim

Tipo: inteiro

`min_size`

O número mínimo de instâncias neste NodeGroup.

Obrigatório: Sim

Tipo: inteiro

`max_size`

O número máximo de instâncias neste NodeGroup.

Obrigatório: Sim

Tipo: inteiro

Propriedades

`node_role`

O ARN do perfil do IAM anexado à instância do Amazon EC2.

Obrigatório: Sim

Tipo: string

`tags`

As tags a serem anexadas ao recurso. As tags serão propagadas para as instâncias criadas pelo recurso.

Obrigatório: não

Tipo: lista

Requisitos

`cluster`

Um AWS. Compute.EKSnodo.

Obrigatório: Sim

Tipo: string

`subnets`

Um AWS. Networking.Subnetnodo.

Obrigatório: Sim

Tipo: lista

`network_interfaces`

Um AWS. Networking.ENInodo. Certifique-se de que as interfaces de rede e sub-redes estejam definidas com a mesma zona de disponibilidade, senão a instanciação falhará.

Quando você define`network_interfaces`, o AWS TNB obtém a permissão relacionada aos ENIs da `multus_role` propriedade se você incluiu a `multus` propriedade no nó. AWS.Compute.EKS Caso contrário, o AWS TNB recebe a permissão relacionada a ENIs da propriedade node_role.

Obrigatório: não

Tipo: lista

`security_groups`

Um AWS. Networking.SecurityGroupnodo.

Obrigatório: não

Tipo: lista

`placement_group`

Um tosca.nodes.AWS. Compute.PlacementGroupnodo.

Obrigatório: não

Tipo: string

`user_data`

Um tosca.nodes.AWS. Compute.UserDatareferência de nó. Um script de dados de usuário é transmitido às instâncias do Amazon EC2 iniciadas pelo grupo de nós autogerenciados. Adicione

as permissões necessárias para executar dados de usuário personalizados no node_role transmitido ao grupo de nós.

Obrigatório: não

Tipo: string

`labels`

Uma lista de rótulos de nós. Um rótulo de nó deve ter um nome e um valor. Crie um rótulo usando os seguintes critérios:

- O nome e o valor devem ser separados por=.
- O nome e o valor podem ter, cada um, até 63 caracteres.
- O rótulo pode incluir letras (A-Z, a-z,), números (0-9) e os seguintes caracteres: `[-, _, ., *, ?]`
- O nome e o valor devem começar e terminar com um caractere alfanumérico ou * caractere. ?

Por exemplo, `myLabelName1=*NodeLabelValue1`.

Obrigatório: não

Tipo: lista

Exemplo

```
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

Um PlacementGroup nó oferece suporte a diferentes estratégias para colocar instâncias do Amazon EC2.

Ao executar uma nova instância do Amazon EC2, o serviço do Amazon EC2 tenta posicionar a instância de forma que todas as instâncias sejam distribuídas pelo hardware subjacente para minimizar falhas correlacionadas. É possível usar grupos de posicionamento para influenciar o posicionamento de um grupo de instâncias interdependentes para atender às necessidades de sua workload.

Sintaxe

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

Propriedades

`strategy`

A estratégia a ser usada para posicionar instâncias do Amazon EC2.

Obrigatório: Sim

Tipo: string

Valores possíveis: CLUSTER | PARTITION | SPREAD_HOST | SPREAD_RACK

- CLUSTER: agrupa as instâncias em uma zona de disponibilidade. Essa estratégia permite que as workloads atinjam a performance de rede de baixa latência necessária para a comunicação de nó a nó totalmente acoplada que é típica das aplicações de computação de alta performance (HPC).
- PARTITION: distribui as instâncias entre partições lógicas, de tal modo que as instâncias em uma partição não compartilhem o hardware subjacente com os grupos de instâncias em outras partições. Essa estratégia é normalmente usada por grandes workloads distribuídas e replicadas, como Hadoop, Cassandra e Kafka.
- SPREAD_RACK: posiciona um pequeno grupo de instâncias no hardware subjacente distinto para reduzir falhas correlacionadas.
- SPREAD_HOST: usado somente com grupos de posicionamento do Outpost. Posiciona um pequeno grupo de instâncias no hardware subjacente distinto para reduzir falhas correlacionadas.

`partition_count`

O número de partições.

Obrigatório: obrigatório somente quando `strategy` é definido como `PARTITION`.

Tipo: inteiro

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

`tags`

As tags que você pode anexar ao recurso de grupo de posicionamento.

Obrigatório: não

Tipo: lista

Exemplo

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

AWS.Compute.UserData

AWS O TNB oferece suporte ao lançamento de instâncias do Amazon EC2 com dados personalizados do usuário, por meio do nó UserData no Network Service Descriptor (NSD). Para obter mais informações sobre dados personalizados do usuário, consulte Dados do usuário e scripts de shell no Guia do usuário do Amazon EC2.

Durante a instanciação da rede, o AWS TNB fornece o registro da instância do Amazon EC2 para o cluster por meio de um script de dados do usuário. Quando dados personalizados do usuário também são fornecidos, o AWS TNB mescla os dois scripts e os transmite como um script multimime para o Amazon EC2. O script de dados de usuário personalizado é executado antes do script de registro do Amazon EKS.

Para usar variáveis personalizadas no script de dados de usuário, adicione um ponto de exclamação `!` após o colchete aberto `{`. Por exemplo, para usar `MyVariable` no script, insira: `{!MyVariable}`

> **Note**
>
> - AWS O TNB suporta scripts de dados do usuário de até 7 KB de tamanho.
> - Como o AWS TNB usa CloudFormation para processar e renderizar o script de `multimime` dados do usuário, certifique-se de que o script cumpra todas as regras. CloudFormation

Sintaxe

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

```
    content_type: String
```

Propriedades

`implementation`

O caminho relativo para a definição do script de dados de usuário. O formato precisa ser: `./scripts/script_name.sh`

Obrigatório: Sim

Tipo: string

`content_type`

Tipo de conteúdo do script de dados de usuário.

Obrigatório: Sim

Tipo: string

Valores possíveis: `x-shellscript`

Exemplo

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

AWS.Networking.SecurityGroup

AWS O TNB oferece suporte a grupos de segurança para automatizar o provisionamento de grupos de segurança do Amazon EC2, que você pode anexar aos grupos de nós do cluster Amazon EKS Kubernetes.

Sintaxe

```
tosca.nodes.AWS.Networking.SecurityGroup
  properties:
    description: String
```

```
    name: String
    tags: List
  requirements:
    vpc: String
```

Propriedades

`description`

Descrição do grupo de segurança. Podem ser usados até 255 caracteres para descrever o grupo. Você pode incluir somente letras (A-Z e a-z), números (0-9), espaços e os seguintes caracteres especiais: ._-:/() #, @ [] +=&; {}! $*

Obrigatório: Sim

Tipo: string

`name`

Um nome para o grupo de segurança. Você pode usar até 255 caracteres para o nome. Você pode incluir somente letras (A-Z e a-z), números (0-9), espaços e os seguintes caracteres especiais: ._-:/() #, @ [] +=&; {}! $*

Obrigatório: Sim

Tipo: string

`tags`

As tags que você pode anexar ao recurso de grupo de segurança.

Obrigatório: não

Tipo: lista

Requisitos

`vpc`

Um AWS. Networking.VPCnodo.

Obrigatório: Sim

Tipo: string

Exemplo

```
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 O TNB suporta regras de saída de grupos de segurança para automatizar o provisionamento das regras de saída de grupos de segurança do Amazon EC2, às quais podem ser anexadas. AWS Networking.SecurityGroup. Observe que você deve fornecer um cidr_ip/destination_security_group / destination_prefix_list como destino para o tráfego de saída.

Sintaxe

```
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
```

Propriedades

`cidr_ip`

O intervalo de endereços IPv4, no formato CIDR. Você precisa especificar um intervalo CIDR que permita o tráfego de saída.

Obrigatório: não

Tipo: string

`cidr_ipv6`

O intervalo de endereços IPv6 no formato CIDR, para tráfego de saída. Você deve especificar um grupo de segurança de destino (`destination_security_group` ou `destination_prefix_list`) ou um intervalo de CIDR (`cidr_ip` ou `cidr_ipv6`).

Obrigatório: não

Tipo: string

`description`

A descrição de uma regra de saída de grupos de segurança. Podem ser usados até 255 caracteres para descrever a regra.

Obrigatório: não

Tipo: string

`destination_prefix_list`

O ID da lista de prefixos de uma lista de prefixos gerenciada do Amazon VPC. Esse é o destino das instâncias do grupo de nós associadas ao grupo de segurança. Para obter mais informações sobre listas de prefixos gerenciadas, consulte Listas de prefixos gerenciados no Guia do usuário da Amazon VPC.

Obrigatório: não

Tipo: string

`from_port`

Se o protocolo for TCP ou UDP, esse será o início do intervalo de portas. Se o protocolo for ICMP ou ICMPv6, esse será o número do tipo. Um valor de -1 indica todos os ICMP/ICMPv6 tipos. Se você especificar todos os ICMP/ICMPv6 tipos, deverá especificar todos os ICMP/ICMPv6 códigos.

Obrigatório: não

Tipo: inteiro

`ip_protocol`

O nome do protocolo IP (tcp, udp, icmp, icmpv6) ou o número do protocolo. Use -1 para especificar todos os protocolos. Ao autorizar regras de grupo de segurança, especificar -1 ou um número de protocolo diferente de tcp, udp, icmp ou icmpv6 permitirá o tráfego em todas as portas, seja qual for o intervalo de portas especificado. Para tcp, udp e icmp, você precisa especificar um intervalo de portas. Para icmpv6, o intervalo de portas é opcional. Se você omiti-lo, o tráfego de todos os tipos e códigos será permitido.

Obrigatório: Sim

Tipo: string

`to_port`

Se o protocolo for TCP ou UDP, esse será o fim do intervalo de portas. Se o protocolo for ICMP ou ICMPv6, esse será o código. Um valor de -1 indica todos os ICMP/ICMPv6 códigos. Se você especificar todos os ICMP/ICMPv6 tipos, deverá especificar todos os ICMP/ICMPv6 códigos.

Obrigatório: não

Tipo: inteiro

Requisitos

`security_group`

O ID do grupo de segurança ao qual essa regra deve ser adicionada.

Obrigatório: Sim

Tipo: string

`destination_security_group`

O ID ou referência TOSCA do grupo de segurança de destino para o qual o tráfego de saída é permitido.

Obrigatório: não

Tipo: string

Exemplo

```
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 O TNB suporta regras de entrada de grupos de segurança para automatizar o provisionamento das regras de entrada de grupos de segurança do Amazon EC2 que podem ser anexadas. AWS Networking.SecurityGroup. Observe que você deve fornecer um cidr_ip/source_security_group / source_prefix_list como fonte para o tráfego de entrada.

Sintaxe

```
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
```

Propriedades

`cidr_ip`

O intervalo de endereços IPv4, no formato CIDR. Você precisa especificar um intervalo CIDR que permita o tráfego de entrada.

Obrigatório: não

Tipo: string

`cidr_ipv6`

O intervalo de endereços IPv6 no formato CIDR, para tráfego de entrada. Você deve especificar um grupo de segurança de origem (`source_security_group` ou `source_prefix_list`) ou um intervalo de CIDR (`cidr_ip` ou `cidr_ipv6`).

Obrigatório: não

Tipo: string

`description`

A descrição de uma regra de entrada de grupo de segurança. Podem ser usados até 255 caracteres para descrever a regra.

Obrigatório: não

Tipo: string

`source_prefix_list`

O ID da lista de prefixos de uma lista de prefixos gerenciada do Amazon VPC. Essa é a fonte da qual as instâncias do grupo de nós associadas ao grupo de segurança poderão receber tráfego. Para obter mais informações sobre listas de prefixos gerenciadas, consulte Listas de prefixos gerenciados no Guia do usuário da Amazon VPC.

Obrigatório: não

Tipo: string

`from_port`

Se o protocolo for TCP ou UDP, esse será o início do intervalo de portas. Se o protocolo for ICMP ou ICMPv6, esse será o número do tipo. Um valor de -1 indica todos os ICMP/ICMPv6 tipos.

Se você especificar todos os ICMP/ICMPv6 tipos, deverá especificar todos os ICMP/ICMPv6 códigos.

Obrigatório: não

Tipo: inteiro

`ip_protocol`

O nome do protocolo IP (tcp, udp, icmp, icmpv6) ou o número do protocolo. Use -1 para especificar todos os protocolos. Ao autorizar regras de grupo de segurança, especificar -1 ou um número de protocolo diferente de tcp, udp, icmp ou icmpv6 permitirá o tráfego em todas as portas, seja qual for o intervalo de portas especificado. Para tcp, udp e icmp, você precisa especificar um intervalo de portas. Para icmpv6, o intervalo de portas é opcional. Se você omiti-lo, o tráfego de todos os tipos e códigos será permitido.

Obrigatório: Sim

Tipo: string

`to_port`

Se o protocolo for TCP ou UDP, esse será o fim do intervalo de portas. Se o protocolo for ICMP ou ICMPv6, esse será o código. Um valor de -1 indica todos os ICMP/ICMPv6 códigos. Se você especificar todos os ICMP/ICMPv6 tipos, deverá especificar todos os ICMP/ICMPv6 códigos.

Obrigatório: não

Tipo: inteiro

Requisitos

`security_group`

O ID do grupo de segurança ao qual essa regra deve ser adicionada.

Obrigatório: Sim

Tipo: string

`source_security_group`

O ID ou referência TOSCA do grupo de segurança de origem do qual o tráfego de entrada deve ser permitido.

Obrigatório: não

Tipo: string

Exemplo

```
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

Você pode importar os seguintes AWS recursos para o AWS TNB:

- VPC
- Sub-rede
- Tabela de rotas
- Gateway da Internet
- Grupo de segurança

Sintaxe

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

Propriedades

`resource_type`

O tipo de recurso que é importado para o AWS TNB.

Obrigatório: não

Tipo: lista

`resource_id`

O ID do recurso que é importado para o AWS TNB.

Obrigatório: não

Tipo: lista

Exemplo

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

AWS.Networking.ENI

Uma interface de rede é um componente lógico de redes em uma VPC que representa uma cartão de rede virtual. Um endereço IP é atribuído a uma interface de rede de forma automática ou manual, com base em sua sub-rede. Depois de implantar uma instância do Amazon EC2 em uma sub-rede, você pode anexar uma interface de rede a ela ou separar uma interface de rede dessa instância do Amazon EC2 e reconectar-se a outra instância do Amazon EC2 nessa sub-rede. O índice do dispositivo identifica a posição na ordem do anexo.

Sintaxe

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

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

Propriedades

`device_index`

O índice do dispositivo precisa ser maior que zero.

Obrigatório: Sim

Tipo: inteiro

`source_dest_check`

Indica se a interface de rede executa a source/destination verificação. O valor `true` significa que a verificação está habilitada e `false` significa que a verificação está desabilitada.

Valor permitido: verdadeiro, falso

Padrão: verdadeiro

Obrigatório: não

Tipo: booliano

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

Requisitos

`subnet`

Um AWS. Networking.Subnetnodo.

Obrigatório: Sim

Tipo: string

`security_groups`

Um AWS. Networking.SecurityGroupnodo.

Obrigatório: não

Tipo: string

Exemplo

```
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

Um gancho do ciclo de vida fornece a capacidade de executar seus próprios scripts como parte de sua infraestrutura e instanciação de rede.

Sintaxe

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

Capacidades

`execution`

Propriedades do mecanismo de execução de hook que executa os scripts de hook.

`type`

O tipo de mecanismo de execução de hook.

Obrigatório: não

Tipo: string

Valores possíveis: `CODE_BUILD`

Requisitos

`definition`

Um AWS. HookDefinition.Bashnodo.

Obrigatório: Sim

Tipo: string

`vpc`

Um AWS. Networking.VPCnodo.

Obrigatório: Sim

Tipo: string

Exemplo

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

AWS.Networking.InternetGateway

Define um nó do AWS Internet Gateway.

Sintaxe

```
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

`roteamento`

Propriedades que definem a conexão de roteamento dentro da VPC. Você deve incluir a propriedade `dest_cidr` ou `ipv6_dest_cidr`.

`dest_cidr`

O bloco CIDR IPv4 usado para a correspondência do destino. Essa propriedade é usada para criar uma rota em `RouteTable` e seu valor é usado como `DestinationCidrBlock`.

Obrigatório: não se você incluiu a propriedade `ipv6_dest_cidr`.

Tipo: string

`ipv6_dest_cidr`

O bloco CIDR IPv6 usado para a correspondência do destino.

Obrigatório: não se você incluiu a propriedade `dest_cidr`.

Tipo: string

Propriedades

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

`egress_only`

Uma IPv6-specific propriedade. Indica se o gateway da Internet serve apenas para comunicação de saída ou não. Quando `egress_only` é verdadeiro, você deve definir a propriedade `ipv6_dest_cidr`.

Obrigatório: não

Tipo: booliano

Requisitos

`vpc`

Um AWS. Networking.VPCnodo.

Obrigatório: Sim

Tipo: string

`route_table`

Um AWS. Networking.RouteTablenodo.

Obrigatório: Sim

Tipo: string

Exemplo

```
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

Uma tabela de rotas contém um conjunto de regras, chamado de rotas, que determina para onde o tráfego de rede de sua sub-rede ou gateway é direcionado. Você precisa associar uma tabela de rotas a uma VPC.

Sintaxe

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

Propriedades

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

Requisitos

`vpc`

Um AWS. Networking.VPCnodo.

Obrigatório: Sim

Tipo: string

Exemplo

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

AWS.Networking.Subnet

Uma sub-rede é um intervalo de endereços IP na VPC que precisa residir inteiramente em uma zona de disponibilidade. Você precisa especificar uma VPC, um bloco CIDR, uma zona de disponibilidade e uma tabela de rotas para sua sub-rede. Você também precisa definir se sua sub-rede é privada ou pública.

Sintaxe

```
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
```

Propriedades

`type`

Indica se as instâncias executadas nessa sub-rede recebem um endereço IPv4 público.

Obrigatório: Sim

Tipo: string

Valores possíveis: `PUBLIC` | `PRIVATE`

`availability_zone`

A zona de disponibilidade da sub-rede. Esse campo é compatível com zonas de AWS disponibilidade em uma AWS região, por exemplo `us-west-2` (Oeste dos EUA (Oregon)). Ele também suporta Zonas AWS Locais dentro da Zona de Disponibilidade, por exemplo`us-west-2-lax-1a`.

Obrigatório: Sim

Tipo: string

`cidr_block`

O bloco CIDR da sub-rede.

Obrigatório: não

Tipo: string

`ipv6_cidr_block`

O bloco CIDR usado para criar a sub-rede IPv6. Se você incluir essa propriedade, não inclua `ipv6_cidr_block_suffix`.

Obrigatório: não

Tipo: string

`ipv6_cidr_block_suffix`

O sufixo hexadecimal de dois dígitos do bloco CIDR IPv6 para a sub-rede criada na Amazon VPC. Use o seguinte formato: *`2-digit hexadecimal`*`::/`*`subnetMask`*.

Se você incluir essa propriedade, não inclua `ipv6_cidr_block`.

Obrigatório: não

Tipo: string

`outpost_arn`

O ARN do AWS Outposts qual a sub-rede será criada. Adicione essa propriedade ao modelo de NSD se quiser executar nós autogerenciados do Amazon EKS no AWS Outposts. Para obter mais informações, consulte Amazon EKS no AWS Outposts no Guia do usuário do Amazon EKS.

Se você adicionar essa propriedade ao modelo de NSD, precisará definir o valor da propriedade `availability_zone` como a zona de disponibilidade do AWS Outposts.

Obrigatório: não

Tipo: string

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

Requisitos

`vpc`

Um AWS. Networking.VPCnodo.

Obrigatório: Sim

Tipo: string

`route_table`

Um AWS. Networking.RouteTablenodo.

Obrigatório: Sim

Tipo: string

Exemplo

```
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

As implantações de NF são modeladas fornecendo a infraestrutura e o aplicativo associado a ele. O atributo cluster especifica o cluster do EKS que vai hospedar seus NFs. O atributo vnfs especifica as funções de rede da sua implantação. Você também pode fornecer operações opcionais de ganchos do ciclo de vida do tipo pre_create e post_create para executar instruções específicas da sua implantação, como chamar uma API do sistema de gerenciamento de inventário.

Sintaxe

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

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

Requisitos

`deployment`

Um AWS. Deployment.VNFDeploymentnodo.

Obrigatório: não

Tipo: string

`cluster`

Um AWS. Compute.EKSnodo.

Obrigatório: Sim

Tipo: string

`vnfs`

Um nó AWS.VNF.

Obrigatório: Sim

Tipo: string

Interfaces

Hooks

Define o estágio em que os ganchos do ciclo de vida são executados.

`pre_create`

Um AWS. HookExecutionnodo. Esse hook é executado antes da implantação do nó `VNFDeployment`.

Obrigatório: não

Tipo: string

`post_create`

Um AWS. HookExecutionnodo. Esse hook é executado após a implantação do nó `VNFDeployment`.

Obrigatório: não

Tipo: string

Exemplo

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

AWS.Networking.VPC

Você precisa especificar um bloco CIDR para sua nuvem privada virtual (VPC).

Sintaxe

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

Propriedades

`cidr_block`

O intervalo de rede do IPv4 para a VPC, na notação CIDR.

Obrigatório: Sim

Tipo: string

`ipv6_cidr_block`

O bloco CIDR IPv6 usado para criar a VPC.

Valor permitido: `AMAZON_PROVIDED`

Obrigatório: não

Tipo: string

`dns_support`

Indica se as instâncias executadas na VPC obtêm nomes de host DNS.

Obrigatório: não

Tipo: booliano

Padrão: `false`

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

Exemplo

```
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

É possível definir um nó público ou privado do NAT Gateway em uma sub-rede. Para um gateway público, se você não fornecer um ID de alocação de IP elástico, o AWS TNB alocará um IP elástico para sua conta e o associará ao gateway.

Sintaxe

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

Propriedades

`subnet`

AWS A. Networking.Subnetreferência de nó.

Obrigatório: Sim

Tipo: string

`internet_gateway`

AWS A. Networking.InternetGatewayreferência de nó.

Obrigatório: Sim

Tipo: string

Propriedades

`type`

Indica se o gateway é público ou privado.

Valor permitido: `PUBLIC`, `PRIVATE`

Obrigatório: Sim

Tipo: string

`eip_allocation_id`

O ID que representa a alocação do endereço IP elástico.

Obrigatório: não

Tipo: string

`tags`

As tags a serem anexadas ao recurso.

Obrigatório: não

Tipo: lista

Exemplo

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

AWS.Networking.Route

Você pode definir um nó de rota que associe a rota de destino ao NAT Gateway como o recurso de destino e adicione a rota à tabela de rotas associada.

Sintaxe

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

```
    route_table: String
```

Propriedades

`dest_cidr_blocks`

A lista de rotas IPv4 de destino para o recurso de destino.

Obrigatório: Sim

Tipo: lista

Tipo de membro: string

Requisitos

`nat_gateway`

AWS A. Networking.NATGatewayreferência de nó.

Obrigatório: Sim

Tipo: string

`route_table`

AWS A. Networking.RouteTablereferência de nó.

Obrigatório: Sim

Tipo: string

Exemplo

```
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

Você pode criar parâmetros SSM por meio do AWS TNB. Os parâmetros SSM que você cria são criados no SSM e prefixados pelo ID da instância de rede AWS TNB. Isso evita que os valores dos parâmetros sejam substituídos quando várias instâncias são instanciadas e atualizadas usando o mesmo modelo NSD.

Sintaxe

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

Propriedades

Parâmetros

`name`

O nome da propriedade ssm. Use o seguinte formato: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`.

O nome de cada parâmetro deve ter menos de 256 caracteres.

Obrigatório: Sim

Tipo: string

`value`

O valor da propriedade ssm. Use um dos seguintes formatos:

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

O valor de cada parâmetro deve ser menor que 4 KB.

Obrigatório: Sim

Tipo: string

`tags`

As tags que você pode anexar a uma propriedade SSM.

Obrigatório: não

Tipo: lista

Exemplo

```
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"
```

Nós comuns

Defina nós para o NSD e o VNFD.

- AWS. HookDefinition.Bash

AWS.HookDefinition.Bash

Define uma AWS HookDefinition entrada`bash`.

Sintaxe

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

Propriedades

`implementation`

O caminho relativo para a definição do hook. O formato precisa ser: `./hooks/`*`script_name`*`.sh`

Obrigatório: sim

Tipo: String

`environment_variables`

As variáveis de ambiente para o script bash do hook. Use o seguinte formato: **`envName`**`=`**`envValue`** com os seguintes padrões de regex:

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

Certifique-se de que o valor **`envName`**`=`**`envValue`** atenda aos seguintes critérios:

- Não use espaços.
- Comece **`envName`** com uma letra (A-Z ou a-z) ou número (0-9).
- Não inicie o nome da variável de ambiente com as seguintes palavras-chave reservadas do AWS TNB (sem distinção entre maiúsculas e minúsculas):
  - CODEBUILD
  - TNB
  - HOME

  - AWS
- Você pode usar qualquer número de letras (A-Z ou a-z), números (0-9) e os caracteres especiais - e _ para **`envName`** e **`envValue`**.
- Cada variável de ambiente (each **`envName`** =**`envValue`**) deve ter menos de 128 caracteres.

Example: `A123-45xYz=Example_789`

Obrigatório: Não

Tipo: lista

`execution_role`

O perfil da execução do hook.

Obrigatório: sim

Tipo: String

Exemplo

```
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"
```

Segurança em AWS TNB

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de data centers e arquiteturas de rede criados para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços no Nuvem AWS. AWS também fornece serviços que você pode usar com segurança. Third-party auditores testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS de . Para saber mais sobre os programas de conformidade que se aplicam ao AWS Telco Network Builder, consulte AWS Serviços no escopo do programa de conformidade AWS .
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da empresa e as leis e regulamentos aplicáveis.

Essa documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar o AWS TNB. Os tópicos a seguir mostram como configurar o AWS TNB para atender aos seus objetivos de segurança e conformidade. Você também aprende a usar outros AWS serviços que ajudam a monitorar e proteger seus recursos AWS do TNB.

Conteúdo

Proteção de dados em AWS TNB

O modelo de responsabilidade AWS compartilhada O modelo de se aplica à proteção de dados no AWS Telco Network Builder. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre privacidade de dados, consulte Perguntas frequentes sobre privacidade de dados . Para obter informações sobre proteção de dados na Europa, consulte o Centro de Regulamento Geral sobre a Proteção de Dados (RGPD).

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com Centro de Identidade do AWS IAM ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com AWS os recursos. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail. Para obter informações sobre o uso de CloudTrail trilhas para capturar AWS atividades, consulte Como trabalhar com CloudTrail trilhas no Guia AWS CloudTrail do usuário.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sensíveis armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para saber mais sobre os endpoints FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-3.

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sensíveis, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com o AWS TNB ou outro Serviços da AWS usando o console, a API ou os AWS SDKs. AWS CLI Quaisquer dados inseridos em tags ou em campos de texto de

formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse servidor.

Tratamento de dados

Quando você fecha sua AWS conta, o AWS TNB marca seus dados para exclusão e os remove de qualquer uso. Se você reativar sua AWS conta dentro de 90 dias, o AWS TNB restaurará seus dados. Após 120 dias, o AWS TNB exclui permanentemente seus dados. AWS O TNB também encerra suas redes e exclui seus pacotes de funções e seus pacotes de rede.

Criptografia em repouso

AWS O TNB sempre criptografa todos os dados armazenados no serviço em repouso sem exigir nenhuma configuração adicional. Essa criptografia é automática por meio de AWS Key Management Service.

Criptografia em trânsito

AWS O TNB protege todos os dados em trânsito usando o Transport Layer Security (TLS) 1.2.

É sua responsabilidade criptografar os dados entre seus agentes de simulação e os clientes deles.

Inter-network privacidade no trânsito

AWS Os recursos computacionais da TNB residem em uma nuvem privada virtual (VPC) compartilhada por todos os clientes. Todo o tráfego interno AWS do TNB permaneceu na AWS rede e não atravessa a Internet. As conexões entre seus agentes de simulação e os clientes deles são roteadas pela Internet.

Gerenciamento de identidade e acesso para AWS TNB

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (tem permissões) para usar os recursos AWS do TNB. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Conteúdo

- Autenticação com identidades
- Gerenciar o acesso usando políticas
- Como AWS O TNB trabalha com o IAM
- Identity-based exemplos de políticas para AWS Construtor de rede Telco
- Solução de problemas AWS Identidade e acesso do Telco Network Builder

Público

A forma como você usa AWS Identity and Access Management (IAM) difere com base na sua função:

- Usuário do serviço: solicite permissões ao seu administrador se você não conseguir acessar os atributos (consulte Solução de problemas AWS Identidade e acesso do Telco Network Builder).
- Administrador do serviço: determine o acesso do usuário e envie solicitações de permissão (consulte Como AWS O TNB trabalha com o IAM)
- Administrador do IAM: escreva políticas para gerenciar o acesso (consulte Identity-based exemplos de políticas para AWS Construtor de rede Telco)

Autenticação com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado como usuário do IAM ou assumindo uma função do IAM. Usuário raiz da conta da AWS

Você pode fazer login como uma identidade federada usando credenciais de uma fonte de identidade como Centro de Identidade do AWS IAM (IAM Identity Center), autenticação de login único ou credenciais. Google/Facebook Para ter mais informações sobre como fazer login, consulte Como fazer login em sua Conta da AWS no Guia do usuário do Início de Sessão da AWS .

Para acesso programático, AWS fornece um SDK e uma CLI para assinar solicitações criptograficamente. Para ter mais informações, consulte AWS Signature Version 4 para solicitações de API no Guia do usuário do IAM.

Conta da AWS usuário-raiz

Ao criar um Conta da AWS, você começa com uma identidade de login chamada usuário Conta da AWS raiz que tem acesso completo a todos Serviços da AWS os recursos. É altamente

recomendável não usar o usuário-raiz em tarefas diárias. Consulte as tarefas que exigem credenciais de usuário-raiz em Tarefas que exigem credenciais de usuário-raiz no Guia do usuário do IAM.

Identidade federada

Como prática recomendada, exija que os usuários humanos usem a federação com um provedor de identidade para acessar Serviços da AWS usando credenciais temporárias.

Uma identidade federada é um usuário do seu diretório corporativo, provedor de identidade da web ou Directory Service que acessa Serviços da AWS usando credenciais de uma fonte de identidade. As identidades federadas assumem funções que oferecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos Centro de Identidade do AWS IAM. Para saber mais, consulte O que é o IAM Identity Center? no Guia do usuário do Centro de Identidade do AWS IAM .

Usuários e grupos do IAM

Um usuário do IAM é uma identidade com permissões específicas para uma única pessoa ou aplicação. É recomendável usar credenciais temporárias, em vez de usuários do IAM com credenciais de longo prazo. Para obter mais informações, consulte Exigir que usuários humanos usem a federação com um provedor de identidade para acessar AWS usando credenciais temporárias no Guia do usuário do IAM.

Um grupo do IAM especifica um conjunto de usuários do IAM e facilita o gerenciamento de permissões para grandes conjuntos de usuários. Para ter mais informações, consulte Casos de uso de usuários do IAM no Guia do usuário do IAM.

Perfis do IAM

Uma perfil do IAM é uma identidade com permissões específicas que oferece credenciais temporárias. Você pode assumir uma função mudando de um usuário para uma função do IAM (console) ou chamando uma operação de AWS API AWS CLI ou. Para saber mais, consulte Métodos para assumir um perfil no Manual do usuário do IAM.

Os perfis do IAM são úteis para acesso de usuário federado, permissões de usuário do IAM temporárias, acesso entre contas, acesso entre serviços e aplicações em execução no Amazon EC2. Consulte mais informações em Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Gerenciar o acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política define permissões quando associada a uma identidade ou recurso. AWS avalia essas políticas quando um diretor faz uma solicitação. A maioria das políticas é armazenada AWS como documentos JSON. Para ter mais informações sobre documentos de política JSON, consulte Visão geral das políticas JSON no Guia do usuário do IAM.

Por meio de políticas, os administradores especificam quem tem acesso a que, definindo qual entidade principal pode realizar ações em quais recursos e sob quais condições.

Por padrão, usuários e perfis não têm permissões. Um administrador do IAM cria políticas do IAM e as adiciona aos perfis, os quais os usuários podem então assumir. As políticas do IAM definem permissões, independentemente do método usado para realizar a operação.

Identity-based políticas

Identity-based políticas são documentos de políticas de permissões JSON que você anexa a uma identidade (usuário, grupo ou função). Essas políticas controlam quais ações as identidades podem realizar, em quais recursos e sob quais condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

Identity-based as políticas podem ser políticas em linha (incorporadas diretamente em uma única identidade) ou políticas gerenciadas (políticas autônomas anexadas a várias identidades). Para saber como escolher entre uma política gerenciada e políticas em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do usuário do IAM.

Resource-based políticas

Resource-based políticas são documentos de política JSON que você anexa a um recurso. Entre os exemplos estão políticas de confiança de perfil do IAM e políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. É necessário especificar uma entidade principal em uma política baseada em recursos.

Resource-based políticas são políticas embutidas que estão localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais que podem definir o máximo de permissões concedidas por tipos de políticas mais comuns:

- Limites de permissões: definem o número máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM. Para saber mais sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do usuário do IAM.
- Políticas de Controle de Serviços (SCPs): as SCPs especificam o número máximo de permissões para uma organização ou uma unidade organizacional no AWS Organizations. Para saber mais, consulte Políticas de controle de serviço no Guia do usuário do AWS Organizations .
- Políticas de controle de recursos (RCPs): definem o número máximo de permissões disponíveis para recursos em suas contas. Consulte mais informações em Resource control policies (RCPs) no Guia do usuário do AWS Organizations .
- Políticas de sessão: políticas avançadas transmitidas como um parâmetro durante a criação de uma sessão temporária para um perfil ou um usuário federado. Para saber mais, consulte Políticas de sessão no Guia do usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte Lógica de avaliação de políticas no Guia do usuário do IAM.

Como AWS O TNB trabalha com o IAM

Antes de usar o IAM para gerenciar o acesso ao AWS TNB, saiba quais recursos do IAM estão disponíveis para uso com o AWS TNB.

Recursos do IAM que você pode usar com AWS Construtor de rede Telco

Recurso do IAM	AWS Suporte TNB
Identity-based políticas	Sim
Resource-based políticas	Não

Recurso do IAM	AWS Suporte TNB
Ações de políticas	Sim
Recursos de políticas	Sim
Chaves de condição de políticas	Sim
ACLs	Não
ABAC (tags em políticas)	Sim
Credenciais temporárias	Sim
Permissões de entidade principal	Sim
Perfis de serviço	Não
Service-linked funções	Não

Para ter uma visão de alto nível de como o AWS TNB e outros AWS serviços funcionam com a maioria dos recursos do IAM, consulte AWS os serviços que funcionam com o IAM no Guia do usuário do IAM.

Identity-based políticas para AWS TNB

Compatível com políticas baseadas em identidade: sim

Identity-based políticas são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como um usuário do IAM, um grupo de usuários ou uma função. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais atributos e em que condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações e recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte Referência de elemento de política JSON do IAM no Guia do usuário do IAM.

Identity-based exemplos de políticas para AWS TNB

Para ver exemplos de políticas baseadas em identidade do AWS TNB, consulte. Identity-based exemplos de políticas para AWS Construtor de rede Telco

Resource-based políticas dentro AWS TNB

Compatibilidade com políticas baseadas em recursos: não

Resource-based políticas são documentos de política JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o atributo ao qual a política está anexada, a política define quais ações uma entidade principal especificado pode executar nesse atributo e em que condições. É necessário especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Para permitir o acesso entre contas, é possível especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em recursos. Consulte mais informações em Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Ações políticas para AWS TNB

Compatível com ações de políticas: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Action` de uma política JSON descreve as ações que podem ser usadas para permitir ou negar acesso em uma política. Incluem ações em uma política para conceder permissões para executar a operação associada.

Para ver uma lista de ações do AWS TNB, consulte Ações definidas pelo AWS Telco Network Builder na Referência de Autorização de Serviço.

As ações de política no AWS TNB usam o seguinte prefixo antes da ação:

```
tnb
```

Para especificar várias ações em uma única instrução, separe-as com vírgulas.

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

Você também pode especificar várias ações usando caracteres-curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `List`, inclua a seguinte ação:

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

Para ver exemplos de políticas baseadas em identidade do AWS TNB, consulte. Identity-based exemplos de políticas para AWS Construtor de rede Telco

Recursos políticos para AWS TNB

Compatível com recursos de políticas: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Para ações que não oferecem compatibilidade com permissões em nível de recurso, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

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

Para ver uma lista dos tipos de recursos AWS TNB e seus ARNs, consulte Recursos definidos pelo AWS Telco Network Builder na Referência de Autorização de Serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pelo AWS Telco Network Builder.

Para ver exemplos de políticas baseadas em identidade do AWS TNB, consulte. Identity-based exemplos de políticas para AWS Construtor de rede Telco

Chaves de condição de política para AWS TNB

Compatível com chaves de condição de política específicas de serviço: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Condition` especifica quando as instruções são executadas com base em critérios definidos. É possível criar expressões condicionais que usem agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

Para ver uma lista das chaves de condição AWS TNB, consulte Chaves de condição para o AWS Telco Network Builder na Referência de Autorização de Serviço. Para saber com quais ações e recursos você pode usar uma chave de condição, consulte Ações definidas pelo AWS Telco Network Builder.

Para ver exemplos de políticas baseadas em identidade do AWS TNB, consulte. Identity-based exemplos de políticas para AWS Construtor de rede Telco

ACLs em AWS TNB

Compatível com ACLs: não

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

ABAC com AWS TNB

Compatível com ABAC (tags em políticas): sim

Attribute-based controle de acesso (ABAC) é uma estratégia de autorização que define permissões com base em atributos chamados de tags. Você pode anexar tags a entidades e AWS recursos do IAM e, em seguida, criar políticas ABAC para permitir operações quando a tag do diretor corresponder à tag no recurso.

Para controlar o acesso baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou chaves de condição `aws:TagKeys`.

Se um serviço for compatível com as três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço for compatível com as três chaves de condição somente para alguns tipos de recursos, o valor será Parcial

Para saber mais sobre o ABAC, consulte Definir permissões com autorização do ABAC no Guia do usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Usar controle de acesso por atributo (ABAC) no Guia do usuário do IAM.

Usando credenciais temporárias com AWS TNB

Compatível com credenciais temporárias: sim

As credenciais temporárias fornecem acesso de curto prazo aos AWS recursos e são criadas automaticamente quando você usa a federação ou troca de funções. AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para ter mais informações, consulte Credenciais de segurança temporárias no IAM e Serviços da Serviços da AWS que funcionam com o IAM no Guia do usuário do IAM.

Cross-service permissões principais para AWS TNB

Compatibilidade com o recurso de encaminhamento de sessões de acesso (FAS): sim

As sessões de acesso direto (FAS) usam as permissões do principal chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) de fazer solicitações aos serviços posteriores. Para obter detalhes da política ao fazer solicitações de FAS, consulte Sessões de acesso direto.

Funções de serviço para AWS TNB

Compatível com perfis de serviço: não

O perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para saber mais, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.

Service-linked funções para AWS TNB

Compatível com perfis vinculados ao serviço: Não

Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode assumir a função de realizar uma ação em seu nome. Service-linked as funções aparecem no seu Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados ao serviço.

Identity-based exemplos de políticas para AWS Construtor de rede Telco

Por padrão, usuários e funções não têm permissão para criar ou modificar recursos AWS do TNB. Para conceder permissão aos usuários para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM.

Para aprender a criar uma política baseada em identidade do IAM ao usar esses documentos de política em JSON de exemplo, consulte Criar políticas do IAM (console) no Guia do usuário do IAM.

Para obter detalhes sobre ações e tipos de recursos definidos pelo AWS TNB, incluindo o formato dos ARNs para cada um dos tipos de recursos, consulte Ações, recursos e chaves de condição para o AWS Telco Network Builder na Referência de Autorização de Serviço.

Conteúdo

Práticas recomendadas de política

Identity-based as políticas determinam se alguém pode criar, acessar ou excluir recursos AWS do TNB em sua conta. Essas ações podem incorrer em custos para sua Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para saber mais, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do usuário do IAM.
- Aplique permissões de privilégio mínimo: ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para saber mais sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do usuário do IAM.

- Use condições nas políticas do IAM para restringir ainda mais o acesso: é possível adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, é possível escrever uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como CloudFormation. Para saber mais, consulte Elementos da política JSON do IAM: condição no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de cem verificações de política e recomendações práticas para ajudar a criar políticas seguras e funcionais. Para saber mais, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.
- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para saber mais, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para saber mais sobre as práticas recomendadas do IAM, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

Usar o AWS Consola TNB

Para acessar o console do AWS Telco Network Builder, você deve ter um conjunto mínimo de permissões. Essas permissões devem permitir que você liste e visualize detalhes sobre os recursos do AWS TNB em seu Conta da AWS. Caso crie uma política baseada em identidade mais restritiva que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis) com essa política.

Você não precisa permitir permissões mínimas do console para usuários que estão fazendo chamadas somente para a API AWS CLI ou para a AWS API. Em vez disso, permita o acesso somente a ações que correspondam à operação de API que estiverem tentando executar.

Exemplos de política de perfil de serviço

Como administrador, você possui e gerencia os recursos que o AWS TNB cria, conforme definido pelos modelos de ambiente e serviço. Você deve anexar funções de serviço do IAM à sua conta para permitir que o AWS TNB crie recursos para o gerenciamento do ciclo de vida da sua rede.

Uma função de serviço do IAM permite que o AWS TNB faça chamadas para recursos em seu nome para instanciar e gerenciar suas redes. Se você especificar uma função de serviço, o AWS TNB usará a credencial dessa função.

Você cria o perfil de serviço e a respectiva política de permissão com o serviço do IAM. Para obter mais informações sobre a criação de uma função de serviço, consulte Criação de uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

AWS Função de serviço da TNB

Como membro da equipe da plataforma, você pode, como administrador, criar uma função de serviço da AWS TNB e fornecê-la à AWS TNB. Essa função permite que o AWS TNB faça chamadas para outros serviços, como o Amazon Elastic Kubernetes CloudFormation Service, provisione a infraestrutura necessária para sua rede e provisione funções de rede conforme definido em seu NSD.

Recomendamos que você use o seguinte perfil do IAM e a política de confiança para seu perfil de serviço do AWS TNB. Ao definir o escopo da permissão nesta política, lembre-se de que o AWS TNB pode falhar com erros de acesso negado em relação a recursos decodificados de sua política.

O código a seguir mostra uma política de função de serviço do 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"
        }
    ]
}
```

O código a seguir mostra a política de confiança do serviço 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 Função de serviço TNB para o cluster Amazon EKS

Ao criar um recurso do Amazon EKS em seu NSD, você fornece o atributo `cluster_role` para especificar qual perfil será usado para criar seu cluster do Amazon EKS.

O exemplo a seguir mostra um AWS CloudFormation modelo que cria uma função de serviço AWS TNB para a política de cluster do 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 obter mais informações sobre as funções do IAM usando o AWS CloudFormation modelo, consulte as seções a seguir no Guia AWS CloudFormation do usuário:

- AWS::IAM::Perfil
- Seleção de um modelo de pilha

AWS Função de serviço TNB para o grupo de nós Amazon EKS

Ao criar recursos de um grupo de nós do Amazon EKS em seu NSD, você fornece o atributo `node_role` para especificar qual perfil será usado para criar seu grupo de nós do Amazon EKS.

O exemplo a seguir mostra um CloudFormation modelo que cria uma função de serviço AWS TNB para a política de grupo de nós do 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 obter mais informações sobre as funções do IAM usando o AWS CloudFormation modelo, consulte as seções a seguir no Guia AWS CloudFormation do usuário:

- AWS::IAM::Perfil
- Seleção de um modelo de pilha

AWS Função de serviço da TNB para Multus

Ao criar um recurso do Amazon EKS em seu NSD, se você quiser gerenciar o Multus como parte do seu modelo de implantação, deverá fornecer o atributo `multus_role` para especificar qual perfil será usado para gerenciar o Multus.

O exemplo a seguir mostra um CloudFormation modelo que cria uma função de serviço AWS TNB para uma política 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 obter mais informações sobre as funções do IAM usando o AWS CloudFormation modelo, consulte as seções a seguir no Guia AWS CloudFormation do usuário:

- AWS::IAM::Perfil
- Seleção de um modelo de pilha

AWS Função de serviço da TNB para uma política de gancho de ciclo de vida

Quando seu pacote de perfis de rede ou NSD usa um hook de ciclo de vida, você precisa de um perfil de serviço que permita criar um ambiente para a execução de seus hooks de ciclo de vida.

> **Note**
>
> Sua política de gancho do ciclo de vida deve ser baseada no que seu gancho de ciclo de vida está tentando fazer.

O exemplo a seguir mostra um CloudFormation modelo que cria uma função de serviço AWS TNB para uma política de gancho 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 obter mais informações sobre as funções do IAM usando o AWS CloudFormation modelo, consulte as seções a seguir no Guia AWS CloudFormation do usuário:

- AWS::IAM::Perfil
- Seleção de um modelo de pilha

Permitir que os usuários exibam as próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou 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": "*"
        }
    ]
}
```

Solução de problemas AWS Identidade e acesso do Telco Network Builder

Use as informações a seguir para ajudá-lo a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com o AWS TNB e o IAM.

Problemas

- Não estou autorizado a realizar uma ação em AWS TNB
- Não estou autorizado a realizar iam: PassRole
- Quero permitir que pessoas fora da minha Conta da AWS para acessar meu AWS Recursos do TNB

Não estou autorizado a realizar uma ação em AWS TNB

Se você receber uma mensagem de erro informando que não tem autorização para executar uma ação, é preciso atualizar suas políticas para permitir que você realize a ação.

O erro do exemplo a seguir ocorre quando o usuário do IAM `mateojackson` tenta usar o console para exibir detalhes sobre um recurso do *`my-example-widget`* fictício, mas não tem as permissões fictícias do `tnb:`*`GetWidget`*.

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

Nesse caso, a política de Mateo deve ser atualizada para permitir que ele tenha acesso ao recurso *`my-example-widget`* usando a ação `tnb:`*`GetWidget`*.

Se precisar de ajuda, entre em contato com seu AWS administrador. Seu administrador é a pessoa que forneceu suas credenciais de login.

Não estou autorizado a realizar iam: PassRole

Se você receber um erro informando que não está autorizado a realizar a `iam:PassRole` ação, suas políticas devem ser atualizadas para permitir que você passe uma função para a AWS TNB.

Alguns Serviços da AWS permitem que você passe uma função existente para esse serviço em vez de criar uma nova função de serviço ou uma função vinculada ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O erro de exemplo a seguir ocorre quando uma usuária do IAM chamada `marymajor` tenta usar o console para executar uma ação no AWS TNB. No entanto, a ação exige que o serviço tenha permissões concedidas por um perfil de serviço. Mary não tem permissões para passar o perfil para o serviço.

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

Nesse caso, as políticas de Mary devem ser atualizadas para permitir que ela realize a ação `iam:PassRole`.

Se precisar de ajuda, entre em contato com seu AWS administrador. Seu administrador é a pessoa que forneceu suas credenciais de login.

Quero permitir que pessoas fora da minha Conta da AWS para acessar meu AWS Recursos do TNB

É possível criar um perfil que os usuários de outras contas ou pessoas fora da organização podem usar para acessar seus recursos. É possível especificar quem é confiável para assumir o perfil. Para serviços que oferecem compatibilidade com políticas baseadas em recursos ou listas de controle de acesso (ACLs), é possível usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se o AWS TNB oferece suporte a esses recursos, consulteComo AWS O TNB trabalha com o IAM.
- Para saber como fornecer acesso aos seus recursos em todos os Contas da AWS que você possui, consulte Como fornecer acesso a um usuário do IAM em outro Conta da AWS que você possui no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos a terceiros Contas da AWS, consulte Como fornecer acesso Contas da AWS a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.

- Para saber a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Validação de conformidade AWS TNB

Para saber se um AWS service (Serviço da AWS) está dentro do escopo de programas de conformidade específicos, consulte Serviços da AWS Escopo por Programa de Conformidade Serviços da AWS e escolha o programa de conformidade em que você está interessado. Para obter informações gerais, consulte Programas de AWS conformidade Programas AWS de .

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact .

Sua responsabilidade de conformidade ao usar Serviços da AWS é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade de sua empresa e pelas leis e regulamentações aplicáveis. Para obter mais informações sobre sua responsabilidade de conformidade ao usar Serviços da AWS, consulte a Documentação AWS de segurança.

Resiliência em AWS TNB

A infraestrutura AWS global é construída em torno Regiões da AWS de zonas de disponibilidade. Regiões da AWS fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, conectadas a redes de baixa latência, alta taxa de transferência e alta redundância. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre zonas de disponibilidade Regiões da AWS e zonas de disponibilidade, consulte Infraestrutura AWS global.

AWS O TNB executa seu serviço de rede em clusters EKS em uma nuvem privada virtual (VPC) AWS na região que você escolher.

Segurança da infraestrutura em AWS TNB

Como um serviço gerenciado, o AWS Telco Network Builder é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a

infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar o AWS TNB pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Suítes de criptografia com sigilo direto perfeito (PFS), como DHE (Ephemeral) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Diffie-Hellman A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Estes são alguns exemplos de responsabilidades compartilhadas:

- AWS é responsável por proteger os componentes que suportam o AWS TNB, incluindo:
  - Instâncias de computação (também conhecidas como trabalhadores)
  - Bancos de dados internos
  - Comunicações de rede entre componentes internos
  - A interface de programação de aplicativos (API) AWS TNB
  - AWS Kits de desenvolvimento de software (SDK)
- Você é responsável por proteger seu acesso aos seus AWS recursos e aos componentes da carga de trabalho, incluindo (mas não se limitando a):
  - Usuários, grupos, perfis e políticas do IAM
  - Buckets S3 que você usa para armazenar seus dados para TNB AWS
  - Outros recursos Serviços da AWS e recursos que você usa para oferecer suporte ao serviço de rede que você provisionou por meio do TNB AWS
  - Código da sua aplicação
  - Conexões entre o serviço de rede que você provisionou por meio do AWS TNB e seus clientes

> **Important**
>
> Você é responsável por implementar um plano de recuperação de desastres que possa efetivamente recuperar um serviço de rede que você provisionou por meio do AWS TNB.

Modelo de segurança de conectividade de rede

Os serviços de rede que você provisiona por meio AWS do TNB são executados em instâncias de computação em uma nuvem privada virtual (VPC) localizada em uma AWS região selecionada por você. Uma VPC é uma rede virtual na AWS nuvem, que isola a infraestrutura por carga de trabalho ou entidade organizacional. A comunicação entre instâncias de computação nas VPCs permanece dentro da rede AWS e não trafega pela Internet. Algumas comunicações internas de serviços cruzam a Internet e são criptografadas. Os serviços de rede provisionados por meio AWS do TNB para todos os clientes que operam na mesma região compartilham a mesma VPC. Os serviços de rede provisionados por meio AWS do TNB para diferentes clientes usam instâncias de computação separadas na mesma VPC.

As comunicações entre seus clientes de serviços de rede e seu serviço de rede no AWS TNB atravessam a Internet. AWS O TNB não gerencia essas conexões. É sua responsabilidade proteger as conexões de seus clientes.

Suas conexões com o AWS TNB por meio de Console de gerenciamento da AWS, AWS Command Line Interface (AWS CLI) e AWS SDKs são criptografadas.

Versão do IMDS

AWS O TNB oferece suporte a instâncias que utilizam o Instance Metadata Service versão 2 (IMDSv2), um método orientado a sessões. O IMDSv2 inclui maior segurança do que o IMDSV1. Para obter mais informações, consulte Adicionar defesa profunda contra firewalls abertos, proxies reversos e vulnerabilidades SSRF com melhorias no Serviço de metadados da instância do Amazon EC2.

Ao executar sua instância, você precisa usar o IMDSv2. Para mais informações sobre o IMDSv2, consulte Usar IMDSv2 no Guia do usuário do Amazon EC2.

Monitoramento AWS TNB

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho do AWS TNB e de suas outras AWS soluções. AWS permite assistir AWS CloudTrail ao AWS TNB, relatar quando algo está errado e realizar ações automáticas quando apropriado.

Use CloudTrail para capturar informações detalhadas sobre as chamadas feitas para AWS APIs o. Você pode armazenar essas chamadas como arquivos de log no Amazon S3. Você pode usar esses CloudTrail registros para determinar informações como qual chamada foi feita, o endereço IP de origem de onde veio a chamada, quem fez a chamada e quando a chamada foi feita.

Os CloudTrail registros contêm informações sobre as chamadas para ações de API para AWS TNB. Eles também contêm informações para chamadas para ações de API de serviços como Amazon EC2 e Amazon EBS.

Registrando chamadas da API do AWS Telco Network Builder usando AWS CloudTrail

AWS O Telco Network Builder é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou um AWS service (Serviço da AWS). CloudTrail captura todas as chamadas de API para AWS TNB como eventos. As chamadas capturadas incluem chamadas do console do AWS TNB e chamadas de código para as operações da API do AWS TNB. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita à AWS TNB, o endereço IP do qual a solicitação foi feita, quando foi feita e detalhes adicionais.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar o seguinte:

- Se a solicitação foi feita com credenciais de usuário raiz ou credenciais de usuário.
- Se a solicitação foi feita em nome de um usuário do Centro de Identidade do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS service (Serviço da AWS).

CloudTrail está ativo Conta da AWS quando você cria a conta e você tem acesso automático ao histórico de CloudTrail eventos. O histórico de CloudTrail eventos fornece um registro visível,

pesquisável, baixável e imutável dos últimos 90 dias de eventos de gerenciamento registrados em um. Região da AWS Para obter mais informações, consulte Trabalhando com o histórico de CloudTrail eventos no Guia AWS CloudTrail do usuário. Não há CloudTrail cobrança pela visualização do histórico de eventos.

Para um registro contínuo dos eventos dos Conta da AWS últimos 90 dias, crie uma trilha ou um armazenamento de dados de eventos do CloudTrailLake.

CloudTrail trilhas

Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Todas as trilhas criadas usando o Console de gerenciamento da AWS são multirregionais. Só é possível criar uma trilha de região única ou de várias regiões usando a AWS CLI. É recomendável criar uma trilha multirregional porque você captura todas as atividades Regiões da AWS em sua conta. Ao criar uma trilha de região única, é possível visualizar somente os eventos registrados na Região da AWS da trilha. Para obter mais informações sobre trilhas, consulte Criar uma trilha para a Conta da AWS e Criar uma trilha para uma organização no Guia do usuário do AWS CloudTrail .

Você pode entregar uma cópia dos seus eventos de gerenciamento em andamento para o bucket do Amazon S3 sem nenhum custo CloudTrail criando uma trilha. No entanto, existem taxas de armazenamento do Amazon S3. Para obter mais informações sobre CloudTrail preços, consulte AWS CloudTrail Preços. Para receber informações sobre a definição de preços do Amazon S3, consulte Definição de preços do Amazon S3.

CloudTrail Armazenamentos de dados de eventos em Lake

CloudTrail O Lake permite que você execute consultas baseadas em SQL em seus eventos. CloudTrail O Lake converte eventos existentes no formato JSON baseado em linhas para o formato Apache ORC. O ORC é um formato colunar de armazenamento otimizado para recuperação rápida de dados. Os eventos são agregados em armazenamentos de dados de eventos, que são coleções imutáveis de eventos baseados nos critérios selecionados com a aplicação de seletores de eventos avançados. Os seletores que aplicados a um armazenamento de dados de eventos controlam quais eventos persistem e estão disponíveis para consulta. Para obter mais informações sobre o CloudTrail Lake, consulte Trabalhando com o AWS CloudTrail Lake no Guia AWS CloudTrail do Usuário.

CloudTrail Os armazenamentos e consultas de dados de eventos em Lake incorrem em custos. Ao criar um armazenamento de dados de eventos, você escolhe a opção de preço que deseja usar para ele. A opção de preço determina o custo para a ingestão e para o armazenamento de

eventos, e o período de retenção padrão e máximo para o armazenamento de dados de eventos. Para obter mais informações sobre CloudTrail preços, consulte AWS CloudTrail Preços.

AWS Exemplos de eventos TNB

Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a operação de API solicitada, a data e a hora da operação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, os eventos não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra um CloudTrail evento que demonstra a `CreateSolFunctionPackage` operação.


```
{
    "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 obter informações sobre o conteúdo do CloudTrail registro, consulte o conteúdo do CloudTrail registro no Guia AWS CloudTrail do usuário.

AWS Tarefas de implantação do TNB

Entenda as tarefas de implantação para monitorar efetivamente as implantações e agir com mais rapidez.

A tabela a seguir lista as tarefas de implantação AWS do TNB:

Nome da tarefa para implantações iniciadas antes de 7 de março de 2024	Nome da tarefa para implantações iniciadas em e após 7 de março de 2024	Task description
AppInstallation	ClusterPluginInstall	Instala o plug-in Multus no cluster do Amazon EKS.
AppUpdate	nenhuma mudança no nome	Atualiza as funções de rede que já estão instaladas em uma instância de rede.
-	ClusterPluginUninstall	Desinstala os plug-ins no cluster Amazon EKS.

Nome da tarefa para implantações iniciadas antes de 7 de março de 2024	Nome da tarefa para implantações iniciadas em e após 7 de março de 2024	Task description
ClusterStorageClassConfiguration	nenhuma mudança no nome	Configura a classe de armazenamento (driver CSI) em um cluster do Amazon EKS.
FunctionDeletion	nenhuma mudança no nome	Exclui funções de rede dos recursos do AWS TNB.
FunctionInstantiation	FunctionInstall	Implanta funções de rede usando o HELM.
FunctionUninstallation	FunctionUninstall	Desinstala a função de rede de um cluster do Amazon EKS.
HookExecution	nenhuma mudança no nome	Executa ganchos do ciclo de vida conforme definido no NSD.
InfrastructureCancellation	nenhuma mudança no nome	Cancela um serviço de rede.
InfrastructureInstantation	nenhuma mudança no nome	AWS Provisiona recursos em nome do usuário.
InfrastructureTermination	nenhuma mudança no nome	Desprovisiona AWS recursos invocados por meio do TNB AWS .
-	InfrastructureUpdate	Atualiza os AWS recursos provisionados em nome do usuário.
InventoryDeregistration	nenhuma mudança no nome	Cancela o registro de AWS recursos do TNB. AWS
-	InventoryRegistration	Registra os AWS recursos no AWS TNB.
KubernetesClusterConfiguration	ClusterConfiguration	Configura o cluster Kubernetes e adiciona funções adicionais do IAM ao Amazon EKS, AuthMap conforme definido no NSD.

Nome da tarefa para implantações iniciadas antes de 7 de março de 2024	Nome da tarefa para implantações iniciadas em e após 7 de março de 2024	Task description
NetworkServiceFinalization	nenhuma mudança no nome	Finaliza o serviço de rede e fornece uma atualização do status de sucesso ou falha.
NetworkServiceInstantiation	nenhuma mudança no nome	Inicializa o serviço de rede.
SelfManagedNodesConfiguration	nenhuma mudança no nome	Inicializa nós autogerenciados com o Amazon EKS e o ambiente de gerenciamento do Kubernetes.
-	ValidateNetworkServiceUpdate	Executa as validações antes de atualizar uma instância de rede.

Cotas de serviço para TNB AWS

As cotas de serviço, também chamadas de limites, são o número máximo de recursos ou operações de serviço da sua AWS conta. Para obter mais informações, consulte Service Quotas do AWS em Referência geral da Amazon Web Services.

A seguir estão as cotas de serviço para AWS TNB.

Name	Padrão	Ajustá	Description
Operações simultâneas de serviços de rede contínuos	Cada região compatível: 40	Sim	Define o número máximo de operações de serviços de rede simultâneas em uma região.
Pacotes de funções	Cada região compatível: 200	Sim	Define o número máximo de pacotes de funções em uma região.
Pacotes de rede	Cada região compatível: 40	Sim	Número máximo de pacotes de funções em uma região.
Instâncias do serviço de rede	Cada região compatível: 800	Sim	O número máximo de instâncias de serviço de rede em uma região.

Histórico de documentos do guia do usuário do AWS TNB

A tabela a seguir descreve os lançamentos da documentação AWS do TNB.

Alteração	Descrição	Data
Atualizações na configuração de rede do grupo de nós do Amazon EKS	Adicione e exclua sub-redes e grupos de segurança. Adicione, modifique e ENIs exclua da rede. Para obter mais informações, consulte Parâmetros que você pode atualizar.	10 de setembro de 2025
Adição e exclusão de grupos de nós do Amazon EKS em clusters existentes	AWS O TNB agora suporta a adição de novos grupos de nós e a remoção de grupos de nós existentes dos clusters do Amazon EKS. Para obter mais informações, consulte Parâmetros que você pode atualizar.	4 de junho de 2025
Tamanho do volume da raiz	Você pode especificar o tamanho do volume raiz subjacente do Amazon EBS dos seus nós de trabalho do Amazon EKS por meio do `root_volume_size` campo em AWS.Compute. EKSManagedNode e AWS.Compute. EKSSelfManagedNode Nódulos TOSCA.	19 de maio de 2025
Recursos de referência em scripts	Você pode referenciar recursos criados pelo AWS TNB para configurá-los em	2 de maio de 2025

	seus scripts Lifecycle Hook e scripts de dados do usuário.	
A versão 1.32 do Kubernetes agora é compatível com nós do Amazon EKS e grupos de nós gerenciados.	AWS O TNB é compatível com o Kubernetes versão 1.32 para .compute.EKS e .Compute.AWSAWS EKSManagedNodo.	24 de abril de 2025
A versão 1.24 do Kubernetes não é mais compatível com os nós e grupos de nós gerenciados do Amazon EKS	AWS O TNB não é mais compatível com o Kubernetes versão 1.24 para .compute. EKS e .Compute.AWSAWS EKSManagedNodo.	17 de abril de 2025
AL2023 Suporte de AMI para nós gerenciados do Amazon EKS	AWS O TNB é compatível com tipos de AL2023 AMI para AWS.Compute. EKSManagedNodo.	17 de abril de 2025
A versão 1.23 do Kubernetes não é mais compatível com os nós e grupos de nós gerenciados do Amazon EKS	AWS O TNB não é mais compatível com o Kubernetes versão 1.23 para .compute. EKS e .Compute.AWSAWS EKSManagedNodo.	04 de abril de 2025
A ID da AMI pode ser atualizada	Agora você pode atualizar o campo ami_id durante uma chamada de `UpdateSolNetworkService` API.	31 de março de 2025
A versão 1.31 do Kubernetes agora é compatível com nós do Amazon EKS e grupos de nós gerenciados.	AWS O TNB é compatível com o Kubernetes versão 1.31 para .compute.EKS e .Compute.AWSAWS EKSManagedNodo.	18 de fevereiro de 2025

Versão Kubernetes para .Compute. AWS EKSManagedNodo	AWS O TNB oferece suporte às versões 1.23 a 1.30 do Kubernetes para criar um grupo de nós gerenciados pelo Amazon EKS.	28 de janeiro de 2025
Versão do Kubernetes para cluster	AWS O TNB agora oferece suporte ao Kubernetes versão 1.30 para criar clusters do Amazon EKS.	19 de agosto de 2024

AWS O TNB suporta uma operação adicional para gerenciar o ciclo de vida da rede.	Você pode atualizar uma instância de rede instanciada ou atualizada anteriormente com um novo pacote de rede e valores de parâmetros. Consulte:
	• Operações do ciclo de vida
	• Atualizar uma instância de rede
	• AWS Exemplo de função de serviço TNB:
	  • Adicione essas ações do Amazon EKS: `eks:UpdateAddon`, `eks:UpdateClusterVersion`, `eks:UpdateNodegroupConfig`, `eks:UpdateNodegroupVersion`, `eks:DescribeUpdate`
	  • Adicione esta CloudFormation ação: `cloudformation:UpdateStack`
	• Novas tarefas de implantação: `InfrastructureUpdate`, `Inventory Registration`, `ValidateNetworkServiceUpdate`
	• Atualizações da API: GetSolNetworkOperation, ListSolNetworkOperations
		30 de julho de 2024

	, e UpdateSolNetworkInstance	
Nova tarefa e novos nomes de tarefas para tarefas existentes	Uma nova tarefa está disponível. Em 7 de março de 2024, algumas tarefas existentes têm novos nomes para maior clareza.	7 de maio de 2024
Versão do Kubernetes para cluster	AWS O TNB agora oferece suporte ao Kubernetes versão 1.29 para criar clusters do Amazon EKS.	10 de abril de 2024
Support para interface de rede `security_groups`	Você pode anexar grupos de segurança ao nó AWS.networking.ENI.	2 de abril de 2024
Support para criptografia de volume raiz do Amazon EBS	Você pode habilitar a criptografia do Amazon EBS para o volume raiz do Amazon EBS. Para habilitar, adicione as propriedades no AWS.Compute. EKSManagedNode ou AWS.Compute. EKSSelfManagedNodenodo.	2 de abril de 2024
Support para node `labels`	Você pode anexar rótulos de nós ao seu grupo de nós no AWS.Compute. EKSManagedNode ou AWS.Compute. EKSSelfManagedNodenodo.	19 de março de 2024
Support para interface de rede `source_dest_check`	Você pode indicar se deseja ativar ou desativar a source/destination verificação da interface de rede por meio do nó AWS.Networking.ENI.	25 de janeiro de 2024

Suporte a instâncias do Amazon EC2 com dados de usuário personalizados	Você pode iniciar instâncias do Amazon EC2 com dados personalizados do usuário por meio do AWS.Compute. UserData nodo.	16 de janeiro de 2024
Suporte a grupo de segurança	AWS O TNB permite que você importe o AWS recurso Security Group.	8 de janeiro de 2024
Descrição de `network_interfaces` atualizada	Quando a `network_interfaces` propriedade é incluída no AWS.Compute. EKSManagedNode ou AWS.Compute. EKSSelfManagedNodenode, o AWS TNB obtém a permissão relacionada à ENIs `multus_role` propriedade, se disponível, ou à `node_role` propriedade.	18 de dezembro de 2023
Suporte a cluster privado	AWS O TNB agora oferece suporte a clusters privados. Para indicar um cluster privado, defina a propriedade `access` como `PRIVATE`.	11 de dezembro de 2023
Versão do Kubernetes para cluster	AWS O TNB agora oferece suporte ao Kubernetes versão 1.28 para criar clusters do Amazon EKS.	11 de dezembro de 2023

AWS TNB apoia grupo de colocação	Grupo de posicionamento adicionado para as definições do nó `AWS.Compute.EKSManagedNode` e `AWS.Compute.EKSSelfManagedNode`.	11 de dezembro de 2023
AWS TNB adiciona suporte para IPv6	AWS O TNB agora oferece suporte à criação de instâncias de rede com IPv6 infraestrutura. Verifique os nós AWS.Networking.VPC, .Networking.Subnet, .Networking.AWSAWS InternetGateway, AWS.Rede. SecurityGroupIngressRule, AWS.Rede. SecurityGroupEgressRulee AWS.compute.eks para configurações. IPv6 Também adicionam os os nós AWS.Networking. NATGatewaye AWS.Networking.Route para configuração. NAT64 Atualizamos a função de serviço AWS TNB e a função de serviço AWS TNB para o grupo de nós Amazon EKS para obter IPv6 permissões. Consulte Service role policy examples.	16 de novembro de 2023

Permissões adicionadas à política de função de serviço do AWS TNB	Adicionamos permissões à política de função de serviço do AWS TNB para o Amazon S3 CloudFormation e para permitir a instanciação da infraestrutura.	23 de outubro de 2023
AWS TNB lançado em mais regiões	AWS O TNB agora está disponível nas regiões Ásia-Pacífico (Seul), Canadá (Central), Europa (Espanha), Europa (Estocolmo) e América do Sul (São Paulo).	27 de setembro de 2023
Etiquetas para AWS.Compute. EKSSelfManagedNode	AWS O TNB agora suporta tags para a definição do `AWS.Compute.EKSSelfManagedNode` nó.	22 de agosto de 2023
AWS O TNB oferece suporte a instâncias que aproveitam IMDSv2	Ao iniciar sua instância, você deve usar IMDSv2.	14 de agosto de 2023
Permissões atualizadas para o MultusRoleInlinePolicy	O MultusRoleInlinePolicy agora inclui a `ec2:DeleteNetworkInterface` permissão.	7 de agosto de 2023
Versão do Kubernetes para cluster	AWS O TNB agora oferece suporte às versões 1.27 do Kubernetes para criar clusters do Amazon EKS.	25 de julho de 2023

AWS.compute.eks. AuthRole	AWS O TNB oferece suporte para AuthRole que você adicione funções do IAM ao cluster `aws-auth` `ConfigMap` do Amazon EKS para que os usuários possam acessar o cluster do Amazon EKS usando uma função do IAM.	19 de julho de 2023
AWS O TNB oferece suporte a grupos de segurança.	Adicionou o AWS.Networking. SecurityGroup, AWS.Rede. SecurityGroupEgressRule, e AWS.Networking. SecurityG roupIngressRulepara o modelo NSD.	18 de julho de 2023
Versão do Kubernetes para cluster	AWS O TNB oferece suporte às versões 1.22 a 1.26 do Kubernetes para criar clusters do Amazon EKS. AWS O TNB não é mais compatível com as versões 1.21 do Kubernetes.	11 de maio de 2023
AWS.Computação. EKSSelfManagedNode	Você pode criar nós de trabalho autogerenciados na região, nas Zonas AWS Locais e. AWS Outposts	29 de março de 2023
Lançamento inicial	Esta é a primeira versão do Guia do Usuário do AWS TNB.	21 de fevereiro de 2023

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.