Panduan Pengguna

AWS Pembangun Jaringan Telco

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Pembangun Jaringan Telco: Panduan Pengguna

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan hak milik masing-masing pemiliknya, yang mungkin atau tidak terafiliasi, terkait dengan, atau disponsori oleh Amazon.

Table of Contents

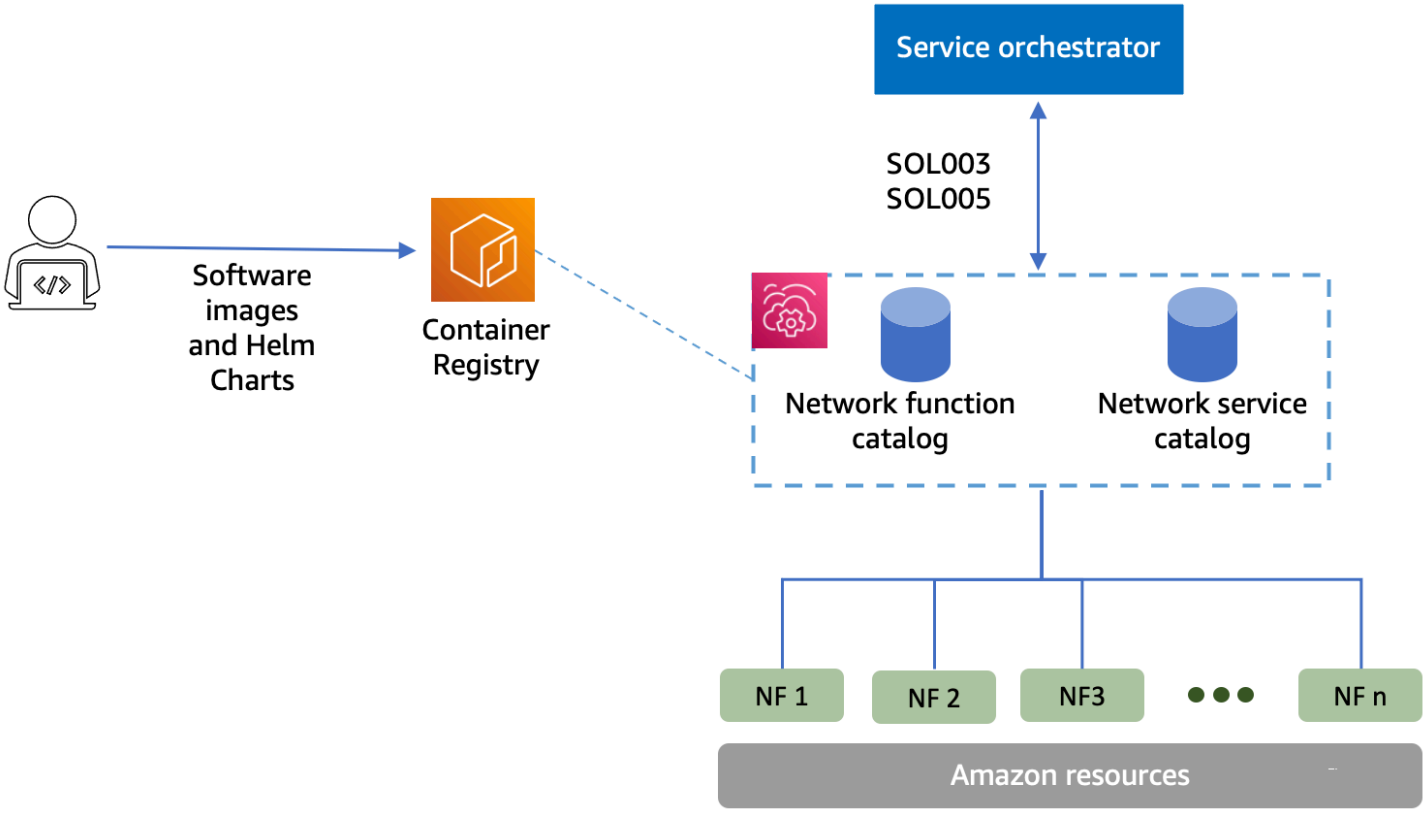
Apa itu Pembuat Jaringan AWS Telco?

AWS Telco Network Builder (AWS TNB) adalah AWS layanan yang menyediakan penyedia layanan komunikasi (CSPs) dengan cara yang efisien untuk menyebarkan, mengelola, dan menskalakan jaringan 5G pada infrastruktur. AWS

Dengan AWS TNB, Anda menerapkan jaringan 5G yang dapat diskalakan dan aman dalam AWS Cloud menggunakan gambar jaringan Anda secara otomatis. Anda tidak perlu mempelajari teknologi baru, memutuskan layanan komputasi mana yang akan digunakan, atau mengetahui cara menyediakan dan mengonfigurasi AWS sumber daya.

Sebagai gantinya, Anda mendeskripsikan infrastruktur jaringan Anda dan memberikan gambar perangkat lunak fungsi jaringan dari mitra vendor perangkat lunak independen (ISV) Anda. AWS TNB terintegrasi dengan orkestra layanan pihak ketiga dan AWS layanan untuk secara otomatis menyediakan AWS infrastruktur yang diperlukan, menyebarkan fungsi jaringan kontainer, dan mengkonfigurasi jaringan dan manajemen akses untuk menciptakan layanan jaringan yang beroperasi penuh.

Diagram berikut menggambarkan integrasi logis antara AWS TNB dan orkestra layanan untuk menyebarkan fungsi jaringan dengan menggunakan antarmuka standar berbasis European Telecommunications Standards Institute (ETSI).



Topik

Baru AWS?

Jika Anda baru mengenal AWS produk dan layanan, mulailah belajar lebih banyak dengan sumber daya berikut:

- Pengantar AWS
- Memulai dengan AWS

Untuk siapa AWS TNB?

AWS TNB adalah untuk CSPs mencari keuntungan dari efisiensi biaya, kelincahan, dan elastisitas AWS Cloud penawaran tanpa menulis dan memelihara skrip dan konfigurasi khusus untuk merancang, menyebarkan, dan mengelola layanan jaringan. AWS TNB secara otomatis menyediakan AWS infrastruktur yang diperlukan, menyebarkan fungsi jaringan kontainer, dan mengkonfigurasi jaringan dan manajemen akses untuk membuat layanan jaringan yang beroperasi penuh berdasarkan deskriptor layanan jaringan yang ditentukan CSP, dan fungsi jaringan yang ingin diterapkan CSP.

AWS Fitur TNB

Berikut ini adalah beberapa alasan mengapa CSP ingin menggunakan AWS TNB:

Membantu menyederhanakan tugas

Memberikan efisiensi lebih untuk operasi jaringan Anda, seperti menyebarkan layanan baru, memperbarui dan meningkatkan fungsi jaringan, dan mengubah topologi infrastruktur jaringan.

Terintegrasi dengan orkestra

AWS TNB terintegrasi dengan orkestra layanan pihak ketiga populer yang sesuai dengan ETSI.

Timbangan

Anda dapat mengonfigurasi AWS TNB untuk menskalakan AWS sumber daya dasar untuk memenuhi permintaan lalu lintas, melakukan pembaruan fungsi jaringan secara lebih efisien, meluncurkan perubahan topologi infrastruktur jaringan, dan mengurangi waktu penyebaran layanan 5G baru dari hari ke jam.

Memeriksa dan memantau sumber daya AWS

AWS TNB memungkinkan Anda Memeriksa dan memantau AWS sumber daya yang mendukung jaringan Anda di satu dasbor, seperti Amazon VPC, Amazon EC2, dan Amazon EKS.

Mendukung template layanan

AWS TNB memungkinkan Anda membuat template layanan untuk semua beban kerja telekomunikasi (RAN, Core, IMS). Anda dapat membuat definisi layanan baru, menggunakan kembali template yang sudah ada, atau mengintegrasikan dengan pipeline continuous integration and continuous delivery (CI/CD) untuk mempublikasikan definisi baru.

Melacak perubahan pada penyebaran jaringan

Saat mengubah konfigurasi dasar penerapan fungsi jaringan, misalnya, mengubah jenis instans dari jenis instans Amazon EC2 , Anda dapat melacak perubahan dengan cara yang dapat diulang dan diskalakan. Melakukannya secara manual akan membutuhkan pengelolaan status jaringan, membuat dan menghapus sumber daya, dan memperhatikan urutan perubahan yang diperlukan. Saat Anda menggunakan AWS TNB untuk mengelola siklus hidup fungsi jaringan Anda, Anda hanya membuat perubahan pada deskriptor layanan jaringan yang menjelaskan fungsi jaringan. AWS TNB kemudian akan secara otomatis melakukan perubahan yang diperlukan dalam urutan yang benar.

Menyederhanakan siklus hidup fungsi jaringan

Anda dapat mengelola versi pertama dan semua versi berikutnya dari fungsi jaringan dan menentukan kapan harus meningkatkan. Anda juga dapat mengelola aplikasi RAN, Core, IMS, dan jaringan Anda dengan cara yang sama.

Mengakses TNB AWS

Anda dapat membuat, mengakses, dan mengelola sumber daya AWS TNB Anda menggunakan salah satu antarmuka berikut:

- AWS Konsol TNB — Menyediakan antarmuka web untuk mengelola jaringan Anda.
- AWS TNB API — Menyediakan RESTful API untuk melakukan tindakan AWS TNB. Untuk informasi selengkapnya lihat AWS Referensi TNB API
- AWS Command Line Interface (AWS CLI) — Menyediakan perintah untuk serangkaian AWS layanan yang luas, termasuk AWS TNB. Ini didukung di Windows, macOS, dan Linux. Untuk informasi selengkapnya, lihat Panduan Pengguna AWS Command Line Interface.
- AWS SDKs— Menyediakan bahasa khusus APIs dan melengkapi banyak detail koneksi. Ini termasuk menghitung tanda tangan, menangani percobaan ulang permintaan, dan penanganan kesalahan. Untuk informasi selengkapnya, lihat AWS SDKs.

Harga untuk AWS TNB

AWS TNB membantu CSPs mengotomatiskan penyebaran dan pengelolaan jaringan telekomunikasi mereka di. AWS Anda membayar untuk dua dimensi berikut saat Anda menggunakan AWS TNB:

- Dengan jam item fungsi jaringan terkelola (MNFI).

- Dengan jumlah permintaan API.

Anda juga dikenakan biaya tambahan saat Anda menggunakan AWS layanan lain bersama dengan AWS TNB. Untuk informasi lebih lanjut, lihat Harga AWS TNB.

Untuk melihat tagihan Anda, pergi ke Dasbor Manajemen Penagihan dan Biaya di AWS Manajemen Penagihan dan Biaya konsol. Tagihan Anda berisi tautan ke laporan penggunaan yang memberikan detail tambahan tentang tagihan Anda. Untuk informasi selengkapnya tentang penagihan AWS akun, lihat Penagihan AWS Akun.

Jika Anda memiliki pertanyaan tentang AWS penagihan, akun, dan acara, hubungi AWS Support.

AWS Trusted Advisor adalah layanan yang dapat Anda gunakan untuk membantu mengoptimalkan biaya, keamanan, dan kinerja AWS lingkungan Anda. Untuk informasi selengkapnya, lihat AWS Trusted Advisor.

Apa selanjutnya

Untuk informasi selengkapnya tentang cara memulai AWS TNB, lihat topik berikut:

- Menyiapkan AWS TNB— Selesaikan langkah-langkah prasyarat.
- Memulai dengan AWS TNB— Terapkan fungsi jaringan pertama Anda, seperti Unit Terpusat (CU), Fungsi Manajemen Akses dan Mobilitas (AMF), Fungsi Pesawat Pengguna (UPF), atau Inti 5G lengkap.

Bagaimana AWS TNB bekerja

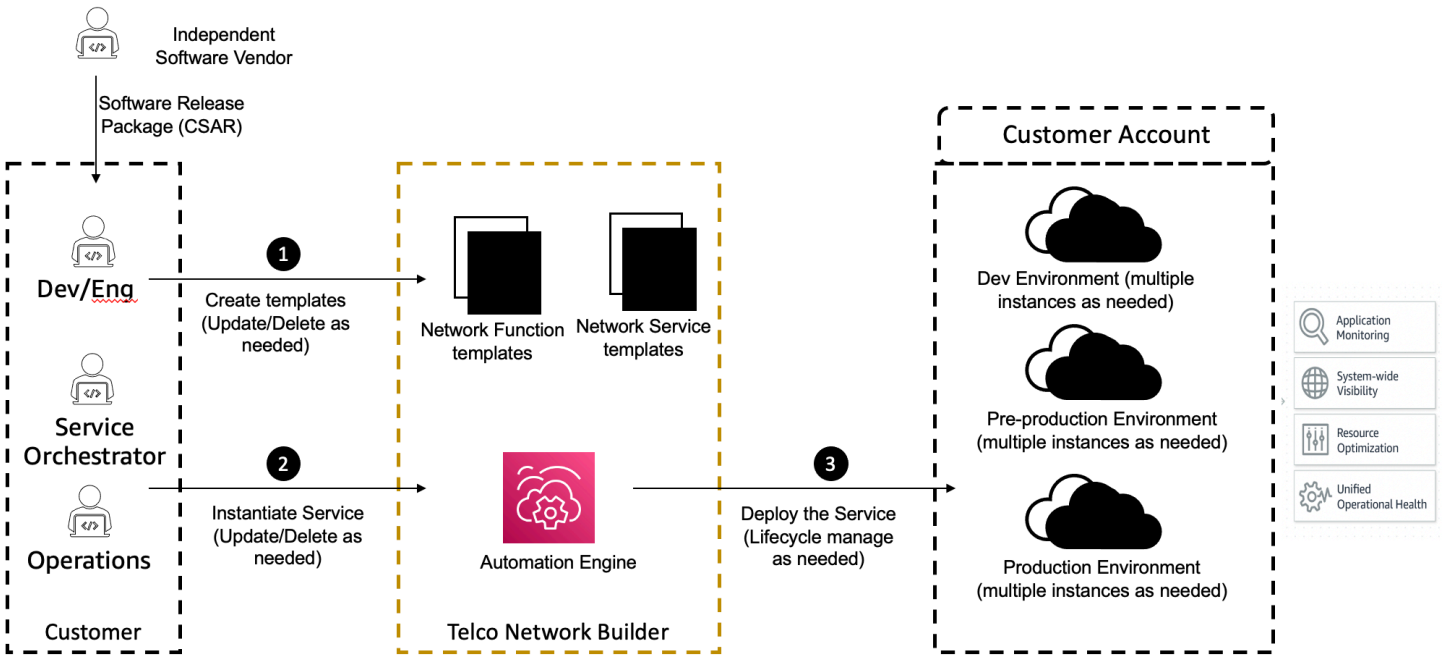
AWS TNB terintegrasi dengan end-to-end orkestra standar dan sumber daya untuk mengoperasikan jaringan 5G penuh. AWS

AWS TNB memungkinkan Anda untuk menelan paket fungsi jaringan dan deskriptor layanan jaringan (NSDs) dan memberi Anda mesin otomatisasi untuk mengoperasikan jaringan Anda. Anda dapat menggunakan end-to-end orkestrator Anda dan berintegrasi dengan AWS TNB APIs, atau menggunakan AWS TNB SDKs untuk membangun alur otomatisasi Anda sendiri. Untuk informasi selengkapnya, lihat AWS Arsitektur TNB .

Topik

AWS Arsitektur TNB

AWS TNB memberi Anda kemampuan untuk melakukan operasi manajemen siklus hidup melalui,, AWS TNB REST AWS CLI API AWS Management Console, dan. SDKs Hal ini memungkinkan persona CSP yang berbeda, seperti anggota tim Engineering, Operations, dan Programmatic System, untuk memanfaatkan TNB. AWS Anda membuat dan mengunggah paket fungsi jaringan sebagai file Cloud Service Archive (CSAR). File CSAR berisi bagan Helm, gambar perangkat lunak, dan Deskriptor Fungsi Jaringan (NFD). Anda dapat menggunakan template untuk berulang kali menerapkan beberapa konfigurasi paket itu. Anda membuat templat layanan jaringan yang mendefinisikan infrastruktur dan fungsi jaringan yang ingin Anda gunakan. Anda dapat menggunakan penggantian parameter untuk menerapkan konfigurasi yang berbeda di lokasi yang berbeda. Anda kemudian dapat membuat instance jaringan, menggunakan template dan menyebarkan fungsi jaringan Anda pada infrastruktur. AWS AWS TNB memberi Anda visibilitas penerapan Anda.



Integrasi dengan Layanan AWS

Jaringan 5G terdiri dari serangkaian fungsi jaringan kontainer yang saling terhubung yang digunakan di ribuan cluster Kubernetes. AWS TNB terintegrasi dengan hal-hal berikut Layanan AWS sebagai telekomunikasi khusus APIs untuk menciptakan layanan jaringan yang beroperasi penuh:

- Amazon Elastic Container Registry (Amazon ECR) untuk menyimpan artefak fungsi jaringan Vendor ISVs Perangkat Lunak Independen ().
- Amazon Elastic Kubernetes Service (Amazon EKS) untuk mengatur cluster.
- Amazon VPC untuk konstruksi jaringan.
- Kelompok keamanan menggunakan AWS CloudFormation.
- AWS CodePipeline untuk target penyebaran di seluruh Wilayah AWS, AWS Local Zones, dan AWS Outposts.
- IAM untuk mendefinisikan peran.
- AWS Organizations untuk mengontrol akses ke AWS TNB APIs.
- AWS Health Dashboard dan AWS CloudTrail untuk memantau kesehatan dan metrik pasca.

AWS Kuota sumber daya TNB

Anda Akun AWS memiliki kuota default, sebelumnya disebut sebagai batas, untuk masing-masing. Layanan AWS Kecuali dinyatakan lain, setiap kuota khusus untuk. Wilayah AWS Anda dapat meminta kenaikan untuk beberapa kuota, tetapi tidak untuk semua kuota.

Untuk melihat kuota AWS TNB, buka konsol Service Quotas. Di panel navigasi, pilih Layanan AWS, dan pilih AWS TNB.

Untuk meminta penambahan kuota, lihat Meminta penambahan kuota di Panduan Pengguna Service Quotas.

Anda Akun AWS memiliki kuota berikut yang terkait dengan AWS TNB.

Kuota sumber daya	Deskripsi	Nilai default	Dapat disesuaikan?
Contoh layanan jaringan	Jumlah maksimum instance layanan jaringan dalam satu Wilayah.	800	Ya
Operasi layanan jaringan yang sedang berlangsung bersamaan	Jumlah maksimum operasi layanan jaringan yang sedang berlangsung bersamaan di satu Wilayah.	40	Ya
Paket jaringan	Jumlah maksimum paket jaringan dalam satu Wilayah.	40	Ya
Paket fungsi	Jumlah maksimum paket fungsi dalam satu Wilayah.	200	Ya

AWS Konsep TNB

Topik ini menjelaskan konsep-konsep penting untuk membantu Anda mulai menggunakan AWS TNB.

Daftar Isi

Siklus hidup fungsi jaringan

AWS TNB membantu Anda sepanjang siklus hidup fungsi jaringan Anda. Siklus hidup fungsi jaringan mencakup tahapan dan aktivitas berikut:

Perencanaan

1. Rencanakan jaringan Anda dengan mengidentifikasi fungsi jaringan yang akan digunakan.
2. Letakkan gambar perangkat lunak fungsi jaringan dalam repositori gambar kontainer.
3. Buat paket CSAR untuk menyebarkan atau meningkatkan.
4. Gunakan AWS TNB untuk mengunggah paket CSAR yang mendefinisikan fungsi jaringan Anda (misalnya, CU AMF, dan UPF), dan berintegrasi dengan pipeline integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) yang dapat membantu Anda membuat versi baru paket CSAR Anda sebagai gambar perangkat lunak fungsi jaringan baru, atau skrip pelanggan, tersedia.

Konfigurasi

1. Identifikasi informasi yang diperlukan untuk penyebaran, seperti jenis komputasi, versi fungsi jaringan, informasi IP, dan nama sumber daya.
2. Gunakan informasi untuk membuat deskriptor layanan jaringan (NSD) Anda.
3. Ingest NSDs yang menentukan fungsi jaringan Anda dan sumber daya yang diperlukan untuk fungsi jaringan untuk membuat instance.

Instantiasi

1. Buat infrastruktur yang dibutuhkan oleh fungsi jaringan.
2. Instantiate (atau ketentuan) fungsi jaringan sebagaimana didefinisikan dalam NSD dan mulai membawa lalu lintas.
3. Validasi aset.

Produksi

Selama siklus hidup fungsi jaringan, Anda akan menyelesaikan operasi produksi, seperti:

- Perbarui konfigurasi fungsi jaringan, misalnya, perbarui nilai dalam fungsi jaringan yang digunakan.
- Perbarui instance jaringan dengan paket jaringan baru dan nilai parameter. Misalnya, perbarui `version` parameter Amazon EKS dalam paket jaringan.

Gunakan antarmuka standar

AWS TNB terintegrasi dengan orkestra layanan yang sesuai dengan European Telecommunications Standards Institute (ETSI) yang memungkinkan Anda menyederhanakan penyebaran layanan jaringan Anda. Service orchestrator dapat menggunakan AWS TNB, SDKs CLI, atau APIs untuk memulai operasi, seperti instantiating atau upgrade fungsi jaringan ke versi baru.

AWS TNB mendukung spesifikasi berikut.

Spesifikasi	Rilis	Deskripsi
ETSI SOL001	v3.6.1	Mendefinisikan standar untuk memungkinkan deskriptor fungsi jaringan berbasis Tosca.
ETSI SOL002	v3.6.1	Mendefinisikan model di sekitar manajemen fungsi jaringan.
ETSI SOL003	v3.6.1	Mendefinisikan standar untuk manajemen siklus hidup fungsi jaringan.
ETSI SOL004	v3.6.1	Mendefinisikan standar CSAR untuk paket fungsi jaringan.

Spesifikasi	Rilis	Deskripsi
ETSI SOL005	v3.6.1	Mendefinisikan standar untuk paket layanan jaringan dan manajemen siklus hidup layanan jaringan.
ETSI SOL007	v3.5.1	Mendefinisikan standar untuk memungkinkan deskriptor layanan jaringan berbasis Tosca.

Paket fungsi jaringan untuk AWS TNB

Dengan AWS TNB, Anda dapat menyimpan paket fungsi jaringan yang sesuai dengan ETSI SOL001/SOL004 ke dalam katalog fungsi. Kemudian, Anda dapat mengunggah paket Cloud Service Archive (CSAR) yang berisi artefak yang menjelaskan fungsi jaringan Anda.

- Deskriptor fungsi jaringan - Mendefinisikan metadata untuk orientasi paket dan manajemen fungsi jaringan
- Gambar Perangkat Lunak — Referensi fungsi jaringan Gambar Kontainer. Amazon Elastic Container Registry (Amazon ECR) dapat bertindak sebagai repositori gambar fungsi jaringan Anda.
- File Tambahan — Gunakan untuk mengelola fungsi jaringan; misalnya, skrip dan bagan Helm.

CSAR adalah paket yang ditentukan oleh standar OASIS TOSCA dan mencakup deskriptor jaringan/layanan yang sesuai dengan spesifikasi YAMAL OASIS TOSCA. Untuk informasi tentang spesifikasi YAMAL yang diperlukan, lihatReferensi TOSCA untuk TNB AWS.

Berikut ini adalah contoh deskriptor fungsi jaringan.

```
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"
```

Deskriptor layanan jaringan untuk TNB AWS

AWS TNB menyimpan deskriptor layanan jaringan (NSDs) tentang fungsi jaringan yang ingin Anda terapkan dan bagaimana Anda ingin menerapkannya ke dalam katalog. Anda dapat mengunggah file YAMM NSD (`vnfd.yaml`), seperti yang dijelaskan oleh ETSI SOL007 untuk menyertakan informasi berikut:

- Fungsi jaringan yang ingin Anda terapkan
- Instruksi jaringan
- Instruksi komputasi
- Kait siklus hidup (skrip khusus)

AWS TNB mendukung standar ETSI untuk pemodelan sumber daya, seperti jaringan, layanan, dan fungsi, dalam bahasa TOSCA. AWS TNB membuatnya lebih efisien untuk Anda gunakan Layanan AWS dengan memodelkannya dengan cara yang dapat dipahami oleh orkestrator layanan yang sesuai dengan ETSI Anda.

Berikut ini adalah cuplikan NSD yang menunjukkan cara memodelkan. Layanan AWS Fungsi jaringan akan digunakan pada cluster Amazon EKS dengan Kubernetes versi 1.27. Subnet untuk aplikasi adalah Subnet01 dan Subnet02. Anda kemudian dapat menentukan NodeGroups untuk aplikasi Anda dengan Amazon Machine Image (AMI), tipe instans, dan konfigurasi penskalaan otomatis.

```
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
```

Manajemen dan operasi untuk AWS TNB

Dengan AWS TNB, Anda dapat mengelola jaringan Anda menggunakan operasi manajemen standar sesuai dengan ETSI SOL003 dan SOL005. Anda dapat menggunakan AWS TNB APIs untuk melakukan operasi siklus hidup seperti:

- Membuat instance fungsi jaringan Anda.
- Mengakhiri fungsi jaringan Anda.

- Memperbarui fungsi jaringan Anda untuk mengganti penerapan Helm.
- Memperbarui instance jaringan yang dipakai atau diperbarui dengan paket jaringan baru dan nilai parameter.
- Mengelola versi paket fungsi jaringan Anda.
- Mengelola versi Anda NSDs.
- Mengambil informasi tentang fungsi jaringan yang Anda gunakan.

Deskriptor layanan jaringan untuk TNB AWS

Deskriptor layanan jaringan (NSD) adalah `.yaml` file dalam paket jaringan yang menggunakan standar TOSCA untuk menggambarkan fungsi jaringan yang ingin Anda gunakan, dan AWS infrastruktur tempat Anda ingin menyebarkan fungsi jaringan. Untuk menentukan NSD Anda dan mengonfigurasi sumber daya dasar dan operasi siklus hidup jaringan Anda, Anda harus memahami Skema TOSCA NSD yang didukung oleh TNB. AWS

File NSD Anda dibagi menjadi beberapa bagian berikut:

1. Versi definisi TOSCA — Ini adalah baris pertama dari file YAMM NSD Anda dan berisi informasi versi, yang ditunjukkan dalam contoh berikut.

```
tosca_definitions_version: tnb_simple_yaml_1_0
```

2. VNFD — NSD berisi definisi fungsi jaringan untuk melakukan operasi siklus hidup. Setiap fungsi jaringan harus diidentifikasi dengan nilai-nilai berikut:
   - ID unik untuk`descriptor_id`. ID harus cocok dengan ID dalam paket CSAR fungsi jaringan.
   - Nama yang unik untuk`namespace`. Nama harus dikaitkan dengan ID unik agar lebih mudah direferensikan di seluruh file YAMM NSD Anda, yang ditunjukkan pada contoh berikut.

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

3. Template topologi - Mendefinisikan sumber daya yang akan digunakan, penyebaran fungsi jaringan, dan skrip apa pun yang disesuaikan, seperti kait siklus hidup. Seperti yang ditunjukkan dalam contoh berikut.

```
topology_template:
```

```
  node_templates:

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

4. Node tambahan — Setiap sumber daya yang dimodelkan memiliki bagian untuk properti dan persyaratan. Properti menjelaskan atribut opsional atau wajib untuk sumber daya, seperti versi. Persyaratan menggambarkan dependensi yang harus disediakan sebagai argumen. Misalnya, untuk membuat Amazon EKS Node Group Resource, itu harus dibuat dalam Amazon EKS Cluster. Seperti yang ditunjukkan dalam contoh berikut.

```
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
```

Menyiapkan AWS TNB

Siapkan AWS TNB dengan menyelesaikan tugas yang dijelaskan dalam topik ini.

Tugas

- Mendaftar untuk Akun AWS
- Buat pengguna dengan akses administratif
- Pilih AWS Wilayah
- Perhatikan titik akhir layanan
- (Opsional) Instal AWS CLI
- Mengatur peran AWS TNB

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimkan email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk https://aws.amazon.comke/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .
2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

  Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Pilih AWS Wilayah

Untuk melihat daftar Wilayah yang tersedia untuk AWS TNB, lihat Daftar Layanan AWS Regional. Untuk melihat daftar titik akhir untuk akses terprogram, lihat titik akhir AWS TNB di. Referensi Umum AWS

Perhatikan titik akhir layanan

Untuk terhubung secara terprogram ke AWS layanan, Anda menggunakan titik akhir. Selain AWS titik akhir standar, beberapa AWS layanan menawarkan titik akhir FIPS di Wilayah tertentu. Untuk informasi selengkapnya, lihat AWS titik akhir layanan.

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
US East (N. Virginia)	us-east-1	tnb.us-east-1.amazonaws.com	HTTPS	
US West (Oregon)	us-west-2	tnb.us-west-2.amazonaws.com	HTTPS	
Asia Pasifik (Seoul)	ap-northeast-2	tnb.ap-northeast-2.amazonaws.com	HTTPS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
Asia Pacific (Sydney)	ap-southeast-2	tnb.ap-southeast-2.amazonaws.com	HTTPS	
Canada (Central)	ca-central-1	tnb.ca-central-1.amazonaws.com	HTTPS	
Eropa (Frankfurt)	eu-central-1	tnb.eu-central-1.amazonaws.com	HTTPS	
Eropa (Paris)	eu-west-3	tnb.eu-west-3.amazonaws.com	HTTPS	
Eropa (Spanyol)	eu-south-2	tnb.eu-south-2.amazonaws.com	HTTPS	
Eropa (Stockholm)	eu-north-1	tnb.eu-north-1.amazonaws.com	HTTPS	
Amerika Selatan (Sao Paulo)	sa-east-1	tnb.sa-east-1.amazonaws.com	HTTPS	

(Opsional) Instal AWS CLI

The AWS Command Line Interface (AWS CLI) menyediakan perintah untuk serangkaian AWS produk yang luas, dan didukung di Windows, macOS, dan Linux. Anda dapat mengakses AWS TNB menggunakan. AWS CLI Untuk memulai, lihat Panduan Pengguna AWS Command Line Interface. Untuk informasi selengkapnya tentang perintah AWS TNB, lihat tnb di Referensi AWS CLI Perintah.

Mengatur peran AWS TNB

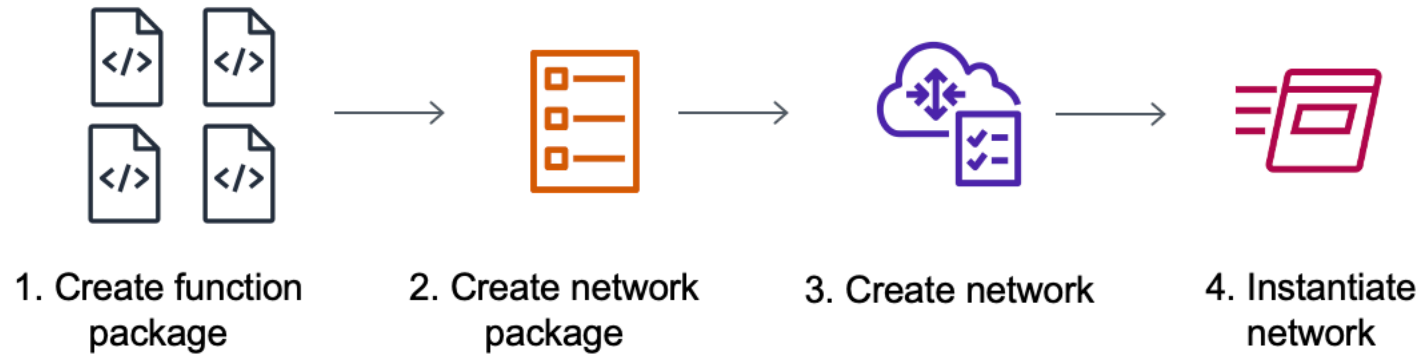
Anda harus membuat peran layanan IAM untuk mengelola berbagai bagian solusi AWS TNB Anda. AWS Peran layanan TNB dapat melakukan panggilan API ke AWS layanan lain, seperti, AWS CloudFormation AWS CodeBuild, dan berbagai layanan komputasi dan penyimpanan, atas nama Anda, untuk membuat instance dan mengelola sumber daya untuk penerapan Anda.

Untuk informasi selengkapnya tentang peran layanan AWS TNB, lihatManajemen identitas dan akses AWS TNB.

Memulai dengan AWS TNB

Tutorial ini menunjukkan bagaimana Anda menggunakan AWS TNB untuk menyebarkan fungsi jaringan, misalnya, Unit Terpusat (CU), Fungsi Manajemen Akses dan Mobilitas (AMF), atau Fungsi Pesawat Pengguna 5G (UPF).

Diagram berikut menggambarkan proses penyebaran:



Tugas

- Prasyarat
- Buat paket fungsi
- Buat paket jaringan
- Membuat dan membuat instance jaringan
- Bersihkan

Prasyarat

Sebelum Anda dapat melakukan penyebaran yang berhasil, Anda harus memiliki yang berikut:

- Rencana Dukungan AWS Bisnis.
- Izin melalui peran IAM.
- Paket Network Function (NF) yang sesuai dengan ETSI SOL001/SOL004.
- Templat Network Service Descriptor (NSD) yang sesuai dengan ETSI SOL007.

Anda dapat menggunakan paket fungsi sampel atau paket jaringan dari paket Sampel untuk GitHub situs AWS TNB.

Buat paket fungsi

Paket fungsi jaringan adalah file Cloud Service Archive (CSAR). File CSAR berisi bagan Helm, gambar perangkat lunak, dan Deskriptor Fungsi Jaringan (NFD).

Untuk membuat paket fungsi

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket fungsi.
3. Pilih Buat paket fungsi.
4. Di bawah Unggah paket fungsi, pilih Pilih file, dan unggah setiap paket CSAR sebagai `.zip` file. Anda dapat mengunggah maksimal 10 file.
5. (Opsional) Di bawah Tag, pilih Tambahkan tag baru dan masukkan kunci dan nilai. Anda dapat menggunakan tag untuk mencari dan memfilter sumber daya Anda atau melacak AWS biaya Anda.
6. Pilih Berikutnya.
7. Tinjau detail paket, lalu pilih Buat paket fungsi.

Buat paket jaringan

Paket jaringan menentukan fungsi jaringan yang ingin Anda gunakan dan bagaimana Anda ingin menerapkannya ke dalam katalog.

Untuk membuat paket jaringan

1. Di panel navigasi, pilih Paket jaringan.
2. Pilih Buat paket jaringan.
3. Di bawah Unggah paket jaringan, pilih Pilih file, dan unggah setiap NSD sebagai `.zip` file. Anda dapat mengunggah maksimal 10 file.
4. (Opsional) Di bawah Tag, pilih Tambahkan tag baru dan masukkan kunci dan nilai. Anda dapat menggunakan tag untuk mencari dan memfilter sumber daya Anda atau melacak AWS biaya Anda.

5. Pilih Berikutnya.
6. Pilih Buat paket jaringan.

Membuat dan membuat instance jaringan

Sebuah instance jaringan adalah jaringan tunggal yang dibuat di AWS TNB yang dapat digunakan. Anda harus membuat instance jaringan dan membuat instance. Saat Anda membuat instance jaringan, AWS TNB menyediakan AWS infrastruktur yang diperlukan, menyebarkan fungsi jaringan kontainer, dan mengonfigurasi jaringan dan manajemen akses untuk membuat layanan jaringan yang beroperasi penuh.

Untuk membuat dan membuat instance jaringan

1. Di panel navigasi, pilih Jaringan.
2. Pilih Buat instance jaringan.
3. Masukkan nama dan deskripsi untuk jaringan, lalu pilih Berikutnya.
4. Pilih paket jaringan. Verifikasi detailnya dan pilih Berikutnya.
5. Pilih Buat instance jaringan. Keadaan awal adalah`Created`.

   Halaman Networks muncul menampilkan instance jaringan baru di `Not instantiated` negara bagian.
6. Pilih instance jaringan, pilih Actions dan Instantiate.

   Halaman instantiate Jaringan muncul.
7. Tinjau detail dan perbarui nilai parameter. Pembaruan pada nilai parameter hanya berlaku untuk contoh jaringan ini. Parameter dalam paket NSD dan VNFD tidak berubah.
8. Pilih jaringan Instantiate.

   Halaman status Deployment muncul.
9. Gunakan ikon Refresh untuk melacak status penerapan instance jaringan Anda. Anda juga dapat mengaktifkan Auto refresh di bagian tugas Deployment untuk melacak kemajuan setiap tugas.

Bersihkan

Anda sekarang dapat menghapus sumber daya yang Anda buat untuk tutorial ini.

Membersihkan sumber daya Anda

1. Di panel navigasi, pilih Jaringan.
2. Pilih ID jaringan, lalu pilih Terminate.
3. Saat diminta konfirmasi, masukkan ID jaringan, lalu pilih Hentikan.
4. Gunakan ikon Refresh untuk melacak status instance jaringan Anda.
5. (Opsional) Pilih jaringan, dan pilih Hapus.

Paket fungsi untuk AWS TNB

Paket fungsi adalah file.zip dalam format CSAR (Cloud Service Archive) yang berisi fungsi jaringan (aplikasi telekomunikasi standar ETSI) dan deskriptor paket fungsi yang menggunakan standar TOSCA untuk menggambarkan bagaimana fungsi jaringan harus berjalan di jaringan Anda.

Tugas

- Buat paket fungsi di AWS TNB
- Lihat paket fungsi di AWS TNB
- Unduh paket fungsi dari AWS TNB
- Hapus paket fungsi dari AWS TNB

Buat paket fungsi di AWS TNB

Pelajari cara membuat paket fungsi di katalog fungsi jaringan AWS TNB. Membuat paket fungsi adalah langkah pertama untuk membuat jaringan di AWS TNB. Setelah Anda mengunggah paket fungsi, Anda dapat membuat paket jaringan.

Console

Untuk membuat paket fungsi menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket fungsi.
3. Pilih Buat paket fungsi.
4. Pilih Pilih file dan unggah setiap paket CSAR sebagai `.zip` file. Anda dapat mengunggah maksimal 10 file.
5. Pilih Berikutnya.
6. Tinjau detail paket.
7. Pilih Buat paket fungsi.

AWS CLI

Untuk membuat paket fungsi menggunakan AWS CLI

1. Gunakan create-sol-function-packageperintah untuk membuat paket fungsi baru:

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

2. Gunakan perintah put-sol-function-package-content untuk mengunggah konten paket fungsi. Sebagai contoh:

```
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
```

Lihat paket fungsi di AWS TNB

Pelajari cara melihat konten paket fungsi.

Console

Untuk melihat paket fungsi menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket fungsi.
3. Gunakan kotak pencarian untuk menemukan paket fungsi

AWS CLI

Untuk melihat paket fungsi menggunakan AWS CLI

1. Gunakan list-sol-function-packagesperintah untuk daftar paket fungsi Anda.

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

2. Gunakan get-sol-function-packageperintah untuk melihat detail tentang paket fungsi.

```
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
```

Unduh paket fungsi dari AWS TNB

Pelajari cara mengunduh paket fungsi dari katalog fungsi jaringan AWS TNB.

Console

Untuk mengunduh paket fungsi menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi di sisi kiri konsol, pilih Paket fungsi.
3. Gunakan kotak pencarian untuk menemukan paket fungsi
4. Pilih paket fungsi
5. Pilih Tindakan, Unduh.

AWS CLI

Untuk mengunduh paket fungsi menggunakan AWS CLI

Gunakan perintah get-sol-function-package-content untuk mengunduh paket fungsi.

```
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
```

Hapus paket fungsi dari AWS TNB

Pelajari cara menghapus paket fungsi dari katalog fungsi jaringan AWS TNB. Untuk menghapus paket fungsi, paket harus dalam keadaan dinonaktifkan.

Console

Untuk menghapus paket fungsi menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket fungsi.
3. Gunakan kotak pencarian untuk menemukan paket fungsi.
4. Pilih paket fungsi.
5. Pilih Tindakan, Nonaktifkan .
6. Pilih Tindakan, Hapus.

AWS CLI

Untuk menghapus paket fungsi menggunakan AWS CLI

1. Gunakan update-sol-function-packageperintah untuk menonaktifkan paket fungsi.

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

2. Gunakan delete-sol-function-packageperintah untuk menghapus paket fungsi.

```
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
```

Paket jaringan untuk AWS TNB

Paket jaringan adalah file.zip dalam format CSAR (Cloud Service Archive) mendefinisikan paket fungsi yang ingin Anda terapkan dan AWS infrastruktur yang ingin Anda gunakan.

Tugas

- Buat paket jaringan di AWS TNB
- Lihat paket jaringan di AWS TNB
- Unduh paket jaringan dari AWS TNB
- Hapus paket jaringan dari AWS TNB

Buat paket jaringan di AWS TNB

Paket jaringan terdiri dari file deskriptor layanan jaringan (NSD) (wajib) dan file tambahan (opsional), seperti skrip khusus untuk kebutuhan Anda. Misalnya, jika Anda memiliki beberapa paket fungsi dalam paket jaringan Anda, Anda dapat menggunakan NSD untuk menentukan fungsi jaringan mana yang harus dijalankan di klaster tertentu VPCs, subnet, atau Amazon EKS.

Buat paket jaringan setelah membuat paket fungsi. Setelah Anda membuat paket jaringan, Anda perlu membuat instance jaringan.

Console

Untuk membuat paket jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket jaringan.
3. Pilih Buat paket jaringan.
4. Pilih Pilih file dan unggah setiap NSD sebagai `.zip` file. Anda dapat mengunggah maksimal 10 file.
5. Pilih Berikutnya.
6. Tinjau detail paket.
7. Pilih Buat paket jaringan.

AWS CLI

Untuk membuat paket jaringan menggunakan AWS CLI

1. Gunakan create-sol-network-packageperintah untuk membuat paket jaringan.

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

2. Gunakan perintah put-sol-network-package-content untuk mengunggah konten paket jaringan. Sebagai contoh:

```
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
```

Lihat paket jaringan di AWS TNB

Pelajari cara melihat konten paket jaringan.

Console

Untuk melihat paket jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket jaringan.
3. Gunakan kotak pencarian untuk menemukan paket jaringan.

AWS CLI

Untuk melihat paket jaringan menggunakan AWS CLI

1. Gunakan list-sol-network-packagesperintah untuk membuat daftar paket jaringan Anda.

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

2. Gunakan get-sol-network-packageperintah untuk melihat detail tentang paket jaringan.

```
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
```

Unduh paket jaringan dari AWS TNB

Pelajari cara mengunduh paket jaringan dari katalog layanan jaringan AWS TNB.

Console

Untuk mengunduh paket jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket jaringan.
3. Gunakan kotak pencarian untuk menemukan paket jaringan
4. Pilih paket jaringan.
5. Pilih Tindakan, Unduh.

AWS CLI

Untuk mengunduh paket jaringan menggunakan AWS CLI

- Gunakan perintah get-sol-network-package-content untuk mengunduh paket jaringan.

```
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
```

Hapus paket jaringan dari AWS TNB

Pelajari cara menghapus paket jaringan dari katalog layanan jaringan AWS TNB. Untuk menghapus paket jaringan, paket harus dalam keadaan nonaktif.

Console

Untuk menghapus paket jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Paket jaringan.
3. Gunakan kotak pencarian untuk menemukan paket jaringan
4. Pilih paket jaringan
5. Pilih Tindakan, Nonaktifkan .
6. Pilih Tindakan, Hapus.

AWS CLI

Untuk menghapus paket jaringan menggunakan AWS CLI

1. Gunakan update-sol-network-packageperintah untuk menonaktifkan paket jaringan.

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

2. Gunakan delete-sol-network-packageperintah untuk menghapus paket jaringan.

```
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
```

Instans jaringan untuk TNB AWS

Sebuah instance jaringan adalah jaringan tunggal yang dibuat di AWS TNB yang dapat digunakan.

Tugas

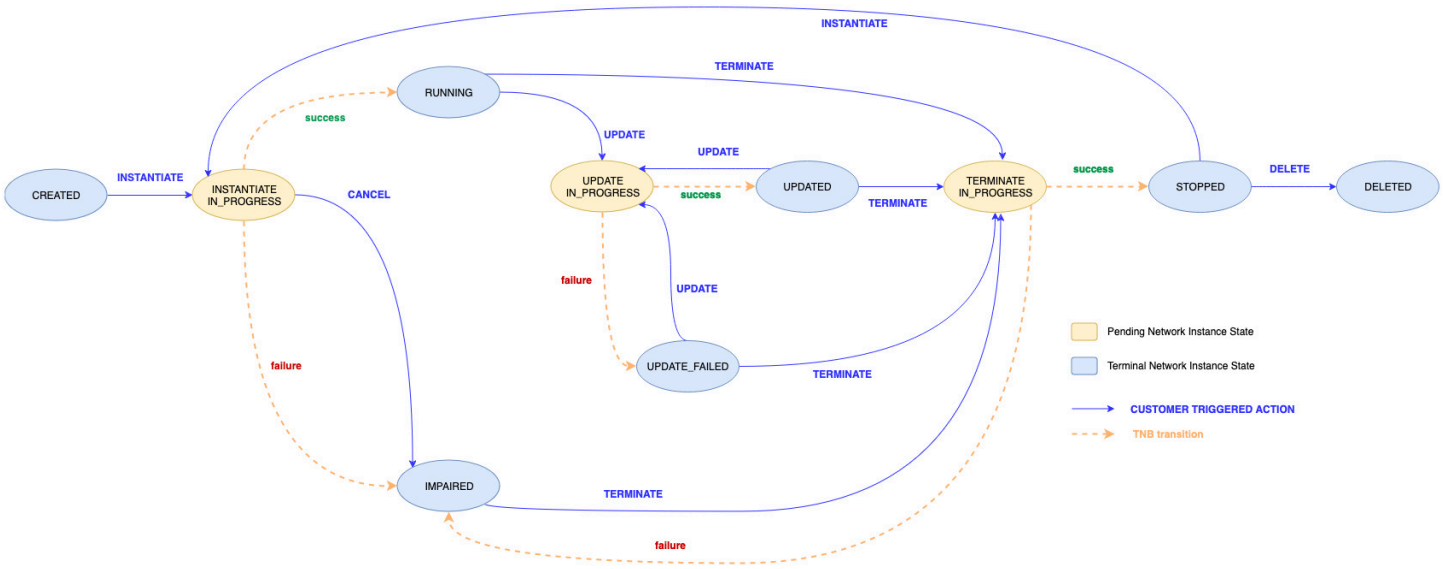
- Operasi siklus hidup dari instance jaringan
- Buat instance jaringan menggunakan AWS TNB
- Membuat instance jaringan menggunakan TNB AWS
- Perbarui instance fungsi di AWS TNB
- Perbarui instance jaringan di AWS TNB
- Lihat contoh jaringan di AWS TNB
- Mengakhiri dan menghapus instance jaringan dari TNB AWS

Operasi siklus hidup dari instance jaringan

AWS TNB memungkinkan Anda untuk dengan mudah mengelola jaringan Anda menggunakan operasi manajemen standar sejalan dengan ETSI SOL003 dan SOL005. Anda dapat melakukan operasi siklus hidup berikut:

- Buat jaringan
- Instantiate jaringan
- Perbarui fungsi jaringan
- Perbarui instance jaringan
- Lihat detail dan status jaringan
- Mengakhiri jaringan

Gambar berikut menunjukkan operasi manajemen jaringan:



Buat instance jaringan menggunakan AWS TNB

Anda membuat instance jaringan setelah membuat paket jaringan. Setelah Anda membuat instance jaringan, buat instance.

Console

Untuk membuat instance jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Jaringan.
3. Pilih Buat instance jaringan.
4. Masukkan nama dan deskripsi untuk instance dan kemudian pilih Berikutnya.
5. Pilih paket jaringan, verifikasi detailnya, dan pilih Berikutnya.
6. Pilih Buat instance jaringan.

   Contoh jaringan baru muncul di halaman Jaringan. Selanjutnya, buat instance jaringan ini.

AWS CLI

Untuk membuat instance jaringan menggunakan AWS CLI

- Gunakan create-sol-network-instanceperintah untuk membuat instance jaringan.

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

Selanjutnya, buat instance jaringan ini.

Membuat instance jaringan menggunakan TNB AWS

Setelah Anda membuat instance jaringan, Anda harus membuat instance. Saat Anda membuat instance jaringan, AWS TNB menyediakan AWS infrastruktur yang diperlukan, menyebarkan fungsi jaringan kontainer, dan mengonfigurasi jaringan dan manajemen akses untuk membuat layanan jaringan yang beroperasi penuh.

Console

Untuk membuat instance jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Jaringan.
3. Pilih instance jaringan yang ingin Anda instantiate.
4. Pilih Actions dan kemudian Instantiate.
5. Pada halaman jaringan Instantiate, tinjau detail dan opsional, perbarui nilai parameter.

   Pembaruan pada nilai parameter hanya berlaku untuk contoh jaringan ini. Parameter dalam paket NSD dan VNFD tidak berubah.
6. Pilih jaringan Instantiate.

   Halaman status Deployment muncul.
7. Gunakan ikon Refresh untuk melacak status penerapan instance jaringan Anda. Anda juga dapat mengaktifkan Auto refresh di bagian tugas Deployment untuk melacak kemajuan setiap tugas.

   Ketika status penerapan berubah menjadi`Completed`, instance jaringan akan dipakai.

AWS CLI

Untuk membuat instance jaringan menggunakan AWS CLI

1. Gunakan instantiate-sol-network-instanceperintah untuk membuat instance jaringan.

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

2. Selanjutnya, lihat status operasi jaringan.

Perbarui instance fungsi di AWS TNB

Setelah instance jaringan dipakai, Anda dapat memperbarui paket fungsi dalam instance jaringan.

Console

Untuk memperbarui instance fungsi menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Jaringan.
3. Pilih contoh jaringan. Anda dapat memperbarui instance jaringan hanya jika statusnya`Instantiated`.

   Halaman instance jaringan muncul.
4. Dari tab Functions, pilih instance fungsi yang akan diperbarui.
5. Pilih Perbarui.
6. Masukkan penggantian pembaruan Anda.
7. Pilih Perbarui.

AWS CLI

Gunakan CLI untuk memperbarui instance fungsi

Gunakan update-sol-network-instanceperintah dengan jenis `MODIFY_VNF_INFORMATION` pembaruan untuk memperbarui instance fungsi dalam instance jaringan.

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

Perbarui instance jaringan di AWS TNB

Setelah instance jaringan dipakai, Anda mungkin perlu memperbarui infrastruktur atau aplikasi. Untuk melakukannya, Anda memperbarui paket jaringan dan nilai parameter untuk instance jaringan dan menerapkan operasi pembaruan untuk menerapkan perubahan.

Pertimbangan

- Anda dapat memperbarui instance jaringan yang ada di `Updated` negara bagian `Instantiated` atau.
- Saat Anda memperbarui instance jaringan, `UpdateSolNetworkService` API menggunakan paket jaringan baru dan nilai parameter untuk memperbarui topologi instance jaringan.
- AWS TNB memverifikasi bahwa jumlah parameter NSD dan VNFD dalam instance jaringan tidak melebihi 200. Batas ini diberlakukan untuk melindungi dari pelaku jahat yang melewati muatan yang salah atau besar yang memengaruhi layanan.

Parameter yang dapat Anda perbarui

Anda dapat memperbarui parameter berikut saat memperbarui instance jaringan yang dipakai:

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela

Versi kluster Amazon EKS	Anda dapat memperbarui nilai untuk `version` parameter bidang kontrol cluster Amazon EKS ke versi minor berikutnya. Anda tidak dapat menurunkan versi.	`EKSCluster:`
		`  type: tosca.nodes.AWS.Compute.EKS`
		`  properties:`
		`  version: "1.28"`
			`EKSC`
			`r:`
			
			`typ`
			`tos`
			`es.A`
			`mput`

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			pro s: ver "1.

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
Node pekerja Amazon EKS	Anda dapat memperbarui nilai `EKSManagedNode kubernetes_version` parameter untuk memutakhirkan grup node Anda ke versi Amazon EKS yang lebih baru, atau Anda dapat memperbarui `ami_id` parameter untuk memutakhirkan grup node Anda ke AMI terbaru yang dioptimalkan EKS. Anda dapat memperbarui ID AMI untuk`EKSSelfManagedNode` . Versi Amazon EKS dari AMI harus sama dengan atau hingga 2 versi lebih rendah dari versi cluster Amazon EKS. Misalnya jika versi cluster Amazon EKS adalah 1.31, maka versi Amazon EKS AMI harus 1.31, 1.30, atau 1.29.	`EKSManagedNodeGroup01: ... properties: kubernetes_version: " 1.28" EKSSelfManagedNode01: compute: compute: properties: ami_id: "ami-1231230LD "`	`EKSM dNod p01: ... pro s: kub s_ve : "1. EKS nage 01: com`

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			com pro s: ami "am 3NEW

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela

Grup simpul Amazon EKS	Anda dapat menambah atau menghapus grup node sesuai kebutuhan komputasi Anda.
	
	Saat menghapus grup node yang ada dan menambahkan yang baru, pastikan bahwa grup node baru berbeda IDs dari grup node yang dihapus, jika tidak operasi akan diperlakukan sebagai modifikasi grup node alih-alih penghapusan dan penambahan. Perhatikan bahwa untuk grup node yang ada, hanya satu set parameter terbatas yang dapat diperbarui.
	Gulir tabel ini untuk melihat parameter mana yang dapat Anda perbarui.
		`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`

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			mir 1 max 1 ... *Free* *SNod* # New Nod typ tos es.A mput Self edNo ... sca pro s:

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			des ize: 1 mir 1 max 1 ... *Free* *SNod* # New Nod typ tos es.A mput Mana de

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			... sca pro s: des ize: 1 mir 1 max 1

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			...

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela

Properti penskalaan	Anda dapat memperbarui properti penskalaan node `EKSManagedNode` dan `EKSSelfManagedNode` TOSCA.	`EKSNodeGroup01:`
		`  ...`
		`  scaling:`
		`  properties:`
		`  desired_size: 1`
		`  min_size: 1`
		`  max_size: 1`
			`EKSN`
			`oup0`
			`...`
			`sca`
			`pro`
			`s:`
			`des`
			`ize:`

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			mir max

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela

Properti plugin Amazon EBS CSI	Anda dapat mengaktifkan atau menonaktifkan plugin Amazon EBS CSI di kluster Amazon EKS Anda. Anda juga dapat mengubah versi plugin.	`EKSCluster:`
		`  capabilities:`
		`  ...`
		`  ebs_csi:`
		`  properties:`
		`  enabled: false`
			`EKSC`
			`r:`
			`cap`
			`ies:`
			`...`
			`ebs`
			`pro`
			`s:`
			`ena`
			`ver`
			`"v1`
			`e`
			`ksbu`
			`"`

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela

Ukuran volume akar	Anda dapat menambahkan, menghapus, atau memperbarui properti ukuran volume root dari EKSManaged node Node dan EKSSelf ManagedNode TOSCA.	`Free5GCEKSNode01:`
		`  ...`
		`  capabilities:`
		`  compute:`
		`  properties:`
		`  root_volume_size: 50`
			Free SNod
			...
			cap ies:
			com
			pro s:

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			roo me_s

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
VNF	Anda dapat mereferensikan VNFs di NSD dan menerapkannya ke cluster yang dibuat di NSD menggunakan VNFDeployment node TOSCA. Sebagai bagian dari pembaruan, Anda akan dapat menambahkan, memperbarui, dan menghapus VNFs ke jaringan.	`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`

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			elmD : typ tos es.A ploy VNFD ment req nts: clu EKS r vnf

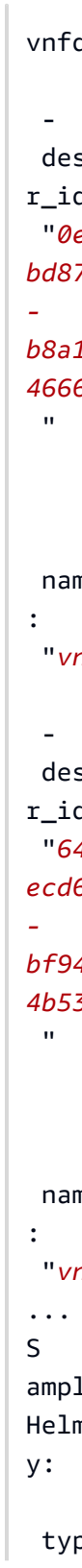
Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			- v leVN - v leVN

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela

Kait	Untuk menjalankan operasi siklus hidup sebelum dan sesudah Anda membuat fungsi jaringan, tambahkan `pre_create` dan `post_create` kait ke `VNFDeployment` node.
	
	Dalam contoh ini, `PreCreate Hook` hook akan berjalan sebelum `vnf3.SampleVNF3` dipakai dan `PostCreat eHook` hook akan berjalan setelah `vnf3.SampleVNF3` dipakai.
		`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`

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			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

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			- v leVN New VNF as the nam , vnf was not pre y pre int s: Hoo pos te: eHoo pre e: Hook

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
Kait	Untuk menjalankan operasi siklus hidup sebelum dan sesudah Anda memperbarui fungsi jaringan, Anda dapat menambahkan `pre_update` hook dan `post_update` hook ke `VNFDeployment` node. Dalam contoh ini, `PreUpdateHook` akan berjalan sebelum `vnf1.SampleVNF1` diperbarui dan `PostUpdateHook` akan berjalan setelah `vnf1.SampleVNF1` diperbarui ke `vnf` paket yang ditunjukkan oleh diperbarui `uuid` untuk namespace vnf1.		

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			tos es.A ploy VNFD ment rec nts: clu EKS r vnf - v leVN A VNF upd as the uui cha for nam "vr - v

Parameter	Deskripsi	Contoh: Sebelum	Conto Setela
			leVN No cha to thi fur as nam and uui rem the sam int s: Hoo pre e: Hook pos te: eHoo

Memperbarui instance jaringan

Console

Untuk memperbarui instance jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Jaringan.
3. Pilih contoh jaringan. Anda dapat memperbarui instance jaringan hanya jika statusnya adalah `Instantiated` atau`Updated`.
4. Pilih Tindakan dan Perbarui.

   Halaman instans Update muncul dengan rincian jaringan dan daftar parameter dalam infrastruktur saat ini.
5. Pilih paket jaringan baru.

   Parameter dalam paket jaringan baru muncul di bagian Parameter yang diperbarui.
6. Secara opsional, perbarui nilai parameter di bagian Parameter yang diperbarui. Untuk daftar nilai parameter yang dapat Anda perbarui, lihatParameter yang dapat Anda perbarui.
7. Pilih Perbarui jaringan.

   AWS TNB memvalidasi permintaan dan memulai penerapan. Halaman status Deployment muncul.
8. Gunakan ikon Refresh untuk melacak status penerapan instance jaringan Anda. Anda juga dapat mengaktifkan Auto refresh di bagian tugas Deployment untuk melacak kemajuan setiap tugas.

   Ketika status penerapan berubah`Completed`, instance jaringan diperbarui.
9. - Jika validasi gagal, instance jaringan tetap dalam keadaan yang sama seperti sebelum Anda meminta pembaruan - baik `Instantiated` atau`Updated`.
   - Jika pembaruan gagal, status instance jaringan akan ditampilkan`Update failed`. Pilih tautan untuk setiap tugas yang gagal untuk menentukan alasannya.
   - Jika pembaruan berhasil, status instance jaringan akan ditampilkan`Updated`.

AWS CLI

Gunakan CLI untuk memperbarui instance jaringan

Gunakan update-sol-network-instanceperintah dengan jenis UPDATE_NS pembaruan untuk memperbarui instance jaringan.

```
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\"}}"
```

Lihat contoh jaringan di AWS TNB

Pelajari cara melihat instance jaringan.

Console

Untuk melihat instance jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Instans jaringan.
3. Gunakan kotak pencarian untuk menemukan contoh jaringan.

AWS CLI

Untuk melihat instance jaringan menggunakan AWS CLI

1. Gunakan list-sol-network-instancesperintah untuk membuat daftar instance jaringan Anda.

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

2. Gunakan get-sol-network-instanceperintah untuk melihat detail tentang instance jaringan tertentu.

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

Mengakhiri dan menghapus instance jaringan dari TNB AWS

Untuk menghapus instance jaringan, instance harus dalam keadaan dihentikan.

Console

Untuk mengakhiri dan menghapus instance jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Jaringan.
3. Pilih ID dari instance jaringan.
4. Pilih Akhiri.
5. Saat diminta konfirmasi, masukkan ID dan pilih Hentikan.
6. Refresh untuk melacak status instans jaringan Anda.
7. (Opsional) Pilih instance jaringan dan pilih Hapus.

AWS CLI

Untuk mengakhiri dan menghapus instance jaringan menggunakan AWS CLI

1. Gunakan terminate-sol-network-instanceperintah untuk mengakhiri instance jaringan.

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

2. (Opsional) Gunakan delete-sol-network-instanceperintah untuk menghapus instance jaringan.

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

Operasi jaringan untuk AWS TNB

Operasi jaringan adalah setiap operasi yang dilakukan untuk jaringan Anda, seperti instantiasi instance jaringan atau terminasi.

Tugas

- Lihat operasi jaringan AWS TNB
- Membatalkan AWS operasi jaringan TNB

Lihat operasi jaringan AWS TNB

Lihat rincian operasi jaringan, termasuk tugas-tugas yang terlibat dalam operasi jaringan dan status tugas.

Console

Untuk melihat operasi jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Instans jaringan.
3. Gunakan kotak pencarian untuk menemukan contoh jaringan.
4. Pada tab Deployment, pilih operasi jaringan.

AWS CLI

Untuk melihat operasi jaringan menggunakan AWS CLI

1. Gunakan list-sol-network-operationsperintah untuk membuat daftar semua operasi jaringan.

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

2. Gunakan get-sol-network-operationperintah untuk melihat detail tentang operasi jaringan.

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

Membatalkan AWS operasi jaringan TNB

Pelajari cara membatalkan operasi jaringan.

Console

Untuk membatalkan operasi jaringan menggunakan konsol

1. Buka konsol AWS TNB di https://console.aws.amazon.com/tnb/.
2. Di panel navigasi, pilih Jaringan.
3. Pilih ID jaringan untuk membuka halaman detailnya.
4. Pada tab Deployments, pilih Operasi Jaringan.
5. Pilih Batalkan operasi.

AWS CLI

Untuk membatalkan operasi jaringan menggunakan AWS CLI

Gunakan cancel-sol-network-operationperintah untuk membatalkan operasi jaringan.

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

Referensi TOSCA untuk TNB AWS

Spesifikasi Topologi dan Orkestrasi untuk Aplikasi Cloud (TOSCA) adalah sintaks deklaratif yang CSPs digunakan untuk menggambarkan topologi layanan web berbasis cloud, komponennya, hubungan, dan proses yang mengelolanya. CSPs jelaskan titik koneksi, tautan logis antara titik koneksi, dan kebijakan seperti afinitas dan keamanan dalam templat TOSCA. CSPs kemudian unggah template ke AWS TNB yang mensintesis sumber daya yang dibutuhkan untuk membangun jaringan 5G yang berfungsi di seluruh AWS Availability Zones.

Konten

Templat VNFD

Mendefinisikan template deskriptor fungsi jaringan virtual (VNFD).

Sintaks

```
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
```

Templat topologi

`node_templates`

Node TOSCA. AWS Node yang mungkin adalah:

- AWS.VNF
- AWS.Artefacts.Helm

AWS.VNF

Mendefinisikan node fungsi jaringan AWS virtual (VNF).

Sintaks

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

Properti

`descriptor_id`

UUID deskriptor.

Wajib: Ya

Tipe: String

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

`descriptor_version`

Versi VNFD.

Wajib: Ya

Tipe: String

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

`descriptor_name`

Nama deskriptor.

Wajib: Ya

Tipe: String

`provider`

Penulis VNFD.

Wajib: Ya

Tipe: String

Persyaratan

`helm`

Direktori Helm mendefinisikan artefak kontainer. Ini adalah referensi ke AWS.Artifacts.Helm.

Wajib: Ya

Tipe: String

Contoh

```
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

Mendefinisikan AWS Helm Node.

Sintaks

```
tosca.nodes.AWS.Artifacts.Helm:
```

```
  properties:
    implementation: String
```

Properti

`implementation`

Direktori lokal yang berisi bagan Helm dalam paket CSAR.

Wajib: Ya

Tipe: String

Contoh

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

Templat deskriptor layanan jaringan

Mendefinisikan template deskriptor layanan jaringan (NSD).

Sintaksis

```
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
```

Menggunakan parameter yang ditentukan

Bila Anda ingin meneruskan parameter secara dinamis, seperti blok CIDR untuk node VPC, Anda dapat menggunakan `{ get_input: input-parameter-name }` sintaks dan menentukan parameter dalam template NSD. Kemudian gunakan kembali parameter di template NSD yang sama.

Contoh berikut menunjukkan bagaimana mendefinisikan dan menggunakan parameter:

```
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 }
```

Impor VNFD

`descriptor_id`

UUID deskriptor.

Wajib: Ya

Tipe: String

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

`namespace`

Nama yang unik.

Wajib: Ya

Tipe: String

Templat topologi

`node_templates`

AWS Node TOSCA yang mungkin adalah:

- AWS.NS
- AWS.compute.eks
- AWS.compute.eks. AuthRole
- AWS.Menghitung. EKSManagedNode
- AWS.Menghitung. EKSSelfManagedNode
- AWS.Menghitung. PlacementGroup
- AWS.Menghitung. UserData
- AWS.Jaringan. SecurityGroup
- AWS.Jaringan. SecurityGroupEgressRule
- AWS.Jaringan. SecurityGroupIngressRule
- AWS.Resource.Impor
- AWS.networking.eni
- AWS.HookExecution
- AWS.Jaringan. InternetGateway
- AWS.Jaringan. RouteTable
- AWS.Networking.Subnet
- AWS.Penerapan. VNFDeployment

- AWS.networking.vpc
- AWS.Jaringan. NATGateway
- AWS.Networking.Route

AWS.NS

Mendefinisikan node layanan AWS jaringan (NS).

Sintaksis

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

Properti

`descriptor_id`

UUID deskriptor.

Wajib: Ya

Tipe: String

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

`descriptor_version`

Versi NSD.

Wajib: Ya

Tipe: String

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

`descriptor_name`

Nama deskriptor.

Wajib: Ya

Tipe: String

Contoh

```
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

Berikan nama klaster, versi Kubernetes yang diinginkan, dan peran yang memungkinkan bidang kontrol Kubernetes mengelola sumber daya yang diperlukan untuk Anda. AWS NFs Plugin Multus Container Network Interface (CNI) diaktifkan. Anda dapat melampirkan beberapa antarmuka jaringan dan menerapkan konfigurasi jaringan lanjutan ke fungsi jaringan berbasis Kubernetes. Anda juga menentukan akses endpoint cluster dan subnet untuk cluster Anda.

Sintaksis

```
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
```

Kemampuan

`multus`

Opsional. Properti yang mendefinisikan penggunaan Multus Container Network Interface (CNI).

Jika Anda menyertakan`multus`, tentukan `enabled` dan `multus_role` properti.

`enabled`

Menunjukkan apakah kemampuan Multus default diaktifkan.

Wajib: Ya

Jenis: Boolean

`multus_role`

Peran untuk manajemen antarmuka jaringan Multus.

Wajib: Ya

Tipe: String

`ebs_csi`

Properti yang menentukan driver Amazon EBS Container Storage Interface (CSI) yang diinstal di cluster Amazon EKS.

Aktifkan plugin ini untuk menggunakan node yang dikelola sendiri Amazon EKS di AWS Outposts, AWS Local Zones, atau Wilayah AWS. Untuk informasi selengkapnya, lihat driver Amazon Elastic Block Store CSI di Panduan Pengguna Amazon EKS.

`enabled`

Menunjukkan apakah driver Amazon EBS CSI default diinstal.

Wajib: Tidak

Jenis: Boolean

`version`

Versi add-on driver Amazon EBS CSI. Versi harus cocok dengan salah satu versi yang dikembalikan oleh DescribeAddonVersionstindakan. Untuk informasi selengkapnya, lihat DescribeAddonVersionsdi Referensi API Amazon EKS

Wajib: Tidak

Tipe: String

Properti

`version`

Versi Kubernetes untuk cluster. AWS Telco Network Builder mendukung Kubernetes versi 1.25 hingga 1.32.

Wajib: Ya

Tipe: String

Nilai yang mungkin: 1,25 | 1,26 | 1,27 | 1,28 | 1,29 | 1,30 | 1,31 | 1,32

`access`

Akses endpoint cluster.

Wajib: Ya

Tipe: String

Nilai yang mungkin: `PRIVATE` | `PUBLIC` | `ALL`

`cluster_role`

Peran manajemen cluster.

Wajib: Ya

Tipe: String

`tags`

Tag untuk dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

`ip_family`

Menunjukkan keluarga IP untuk alamat layanan dan pod di cluster.

Nilai yang diizinkan:`IPv4`, `IPv6`

Nilai default: `IPv4`

Wajib: Tidak

Tipe: String

Persyaratan

`subnets`

Sebuah AWS simpul.Networking.Subnet.

Wajib: Ya

Tipe: Daftar

Contoh

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

```
        enabled: true
        version: "v1.16.0-eksbuild.1"
  requirements:
    subnets:
    - SampleSubnet01
    - SampleSubnet02
```

AWS.compute.eks. AuthRole

An AuthRole memungkinkan Anda menambahkan peran IAM ke kluster Amazon EKS `aws-auth ConfigMap` sehingga pengguna dapat mengakses kluster Amazon EKS menggunakan peran IAM.

Sintaksis

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

Properti

`role_mappings`

Daftar pemetaan yang menentukan peran IAM yang perlu ditambahkan ke kluster Amazon EKS. `aws-auth ConfigMap`

`arn`

ARN dari IAM role.

Wajib: Ya

Tipe: String

`groups`

Grup Kubernetes untuk menetapkan peran yang ditentukan dalam. `arn`

Wajib: Tidak

Tipe: Daftar

Persyaratan

`clusters`

Sebuah AWS simpul.compute.eks.

Wajib: Ya

Tipe: Daftar

Contoh

```
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.Menghitung. EKSManagedNode

AWS TNB mendukung grup Node Terkelola EKS untuk mengotomatiskan penyediaan dan pengelolaan siklus hidup node (instans Amazon EC2) untuk klaster Amazon EKS Kubernetes. Untuk membuat grup EKS Node, lakukan hal berikut:

- Pilih Amazon Machine Images (AMI) untuk node pekerja klaster Anda dengan memberikan ID AMI atau tipe AMI.
- Menyediakan EC2 key pair Amazon untuk akses SSH dan properti penskalaan untuk grup node Anda.
- Pastikan grup node Anda dikaitkan dengan kluster Amazon EKS.

- Berikan subnet untuk node pekerja.
- Secara opsional, lampirkan grup keamanan, label node, dan grup penempatan ke grup node Anda.

Sintaksis

```
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
```

Kemampuan

`compute`

Properti yang menentukan parameter komputasi untuk grup node terkelola Amazon EKS, seperti, jenis EC2 instans Amazon dan EC2 instans Amazon AMIs.

`ami_type`

Jenis AMI yang didukung Amazon EKS.

Wajib: Ya

Tipe: String

Nilai yang mungkin: `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`

ID AMI.

Wajib: Tidak

Tipe: String

> **Note**
>
> Jika keduanya `ami_type` dan `ami_id` ditentukan dalam template, AWS TNB hanya akan menggunakan `ami_id` nilai untuk membuat`EKSManagedNode`.

`instance_types`

Ukuran instance.

Wajib: Ya

Tipe: Daftar

`key_pair`

Pasangan EC2 kunci untuk mengaktifkan akses SSH.

Wajib: Ya

Tipe: String

`root_volume_encryption`

Mengaktifkan enkripsi Amazon EBS untuk volume root Amazon EBS. Jika properti ini tidak disediakan, AWS TNB mengenkripsi volume root Amazon EBS secara default.

Wajib: Tidak

Default: betul

Jenis: Boolean

`root_volume_encryption_key_arn`

ARN dari kuncinya. AWS KMS AWS TNB mendukung ARN kunci reguler, ARN kunci multi-wilayah dan alias ARN.

Wajib: Tidak

Tipe: String

> **Note**
>
> - Jika `root_volume_encryption` salah, jangan sertakan`root_volume_encryption_key_arn`.
> - AWS TNB mendukung enkripsi volume root dari AMI yang didukung Amazon EBS.
> - Jika volume root AMI sudah dienkripsi, Anda harus menyertakan AWS TNB `root_volume_encryption_key_arn` untuk mengenkripsi ulang volume root.
> - Jika volume root AMI tidak dienkripsi, AWS TNB menggunakan `root_volume_encryption_key_arn` untuk mengenkripsi volume root.
>
> Jika Anda tidak menyertakan`root_volume_encryption_key_arn`, AWS TNB menggunakan kunci default yang disediakan oleh AWS Key Management Service untuk mengenkripsi volume root.
> - AWS TNB tidak mendekripsi AMI terenkripsi.

`root_volume_size`

Ukuran volume root Amazon Elastic Block Store di GiBs.

Wajib: Tidak

Default: 20

Jenis: Integer

Nilai yang mungkin: 1 hingga 16.384

`scaling`

Properti yang menentukan parameter penskalaan untuk grup node terkelola Amazon EKS, seperti, jumlah EC2 instans Amazon yang diinginkan, dan jumlah EC2 instans Amazon minimum dan maksimum dalam grup node.

`desired_size`

Jumlah contoh dalam hal ini NodeGroup.

Wajib: Ya

Jenis: Integer

`min_size`

Jumlah minimum contoh dalam hal ini NodeGroup.

Wajib: Ya

Jenis: Integer

`max_size`

Jumlah maksimum contoh dalam hal ini NodeGroup.

Wajib: Ya

Jenis: Integer

Properti

`node_role`

ARN dari peran IAM yang melekat pada instans Amazon. EC2

Wajib: Ya

Tipe: String

`tags`

Tag yang akan dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

`kubernetes_version`

Versi Kubernetes untuk grup Managed Node. AWS TNB mendukung Kubernetes versi 1.25 hingga 1.32. Pertimbangkan hal berikut:

- Tentukan salah satu `kubernetes_version` atau`ami_id`. Jangan menentukan keduanya.
- `kubernetes_version`Harus kurang dari atau sama dengan AWS.Compute. EKSManagedVersi simpul.
- Mungkin ada perbedaan 3 versi antara AWS.Compute. EKSManagedVersi node dan`kubernetes_version`.
- Jika tidak `ami_id` ada `kubernetes_version` atau yang ditentukan, AWS TNB akan menggunakan AMI terbaru dari `AWS.Compute.EKSManagedNode` versi tersebut untuk membuat `EKSManagedNode`

Wajib: Tidak

Tipe: String

Nilai yang mungkin: 1,25 | 1,26 | 1,27 | 1,28 | 1,29 | 1,30 | 1,31 | 1,32

Persyaratan

`cluster`

Sebuah AWS simpul.compute.eks.

Wajib: Ya

Tipe: String

`subnets`

Sebuah AWS simpul.Networking.Subnet.

Wajib: Ya

Tipe: Daftar

`network_interfaces`

Sebuah AWS simpul.networking.ENI. Pastikan antarmuka jaringan dan subnet disetel ke Availability Zone yang sama atau instantiasi akan gagal.

Saat Anda AWS menyetel`network_interfaces`, TNB memperoleh izin yang terkait dengan ENIs dari `multus_role` properti jika Anda menyertakan `multus` properti di simpul AWS.compute.eks. Jika tidak, AWS TNB memperoleh izin yang terkait dengan ENIs dari properti node_role.

Wajib: Tidak

Tipe: Daftar

`security_groups`

Sebuah AWS.Networking. SecurityGroupsimpul.

Wajib: Tidak

Tipe: Daftar

`placement_group`

Sebuah tosca.nodes.AWS.Menghitung. PlacementGroupsimpul.

Wajib: Tidak

Tipe: String

`user_data`

Sebuah tosca.nodes.AWS.Menghitung. UserDatareferensi simpul. Skrip data pengguna diteruskan ke EC2 instans Amazon yang diluncurkan oleh grup node terkelola. Tambahkan izin yang diperlukan untuk menjalankan data pengguna kustom ke node_role yang diteruskan ke grup node.

Wajib: Tidak

Tipe: String

`labels`

Daftar label node. Label node harus memiliki nama dan nilai. Buat label menggunakan kriteria berikut:

- Nama dan nilai harus dipisahkan oleh=.
- Nama dan nilai masing-masing dapat mencapai 63 karakter panjangnya.
- Label dapat mencakup huruf (A-Z, a-z,), angka (0-9) dan karakter berikut: `[-, _, ., *, ?]`
- Nama dan nilai harus dimulai dan diakhiri dengan alfanumerik,?, atau karakter. *

Sebagai contoh, `myLabelName1=*NodeLabelValue1`.

Wajib: Tidak

Tipe: Daftar

Contoh

```
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.Menghitung. EKSSelfManagedNode

AWS TNB mendukung node yang dikelola sendiri Amazon EKS untuk mengotomatiskan penyediaan dan pengelolaan siklus hidup node (EC2 instans Amazon) untuk klaster Amazon EKS Kubernetes. Untuk membuat grup node Amazon EKS, lakukan hal berikut:

- Pilih Amazon Machine Images (AMI) untuk node pekerja klaster Anda dengan memberikan salah satu ID AMI.
- Menyediakan EC2 key pair Amazon untuk akses SSH.
- Pastikan grup node Anda dikaitkan dengan kluster Amazon EKS.
- Berikan jenis instans dan ukuran yang diinginkan, minimum, dan maksimum.
- Berikan subnet untuk node pekerja.
- Secara opsional, lampirkan grup keamanan, label node, dan grup penempatan ke grup node Anda.

Sintaksis

```
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
```

Kemampuan

compute

Properti yang menentukan parameter komputasi untuk node yang dikelola sendiri Amazon EKS, seperti, jenis EC2 instans Amazon dan EC2 instans AMIs Amazon.

`ami_id`

ID AMI digunakan untuk meluncurkan instance. AWS TNB mendukung instans yang memanfaatkan. IMDSv2 Untuk informasi selengkapnya, lihat Versi IMDS.

> **Note**
>
> Anda dapat memperbarui ID AMI untuk`EKSSelfManagedNode`. Versi Amazon EKS dari AMI harus sama dengan atau hingga 2 versi lebih rendah dari versi cluster Amazon EKS. Misalnya jika versi cluster Amazon EKS adalah 1.31, maka versi Amazon EKS AMI harus 1.31, 1.30, atau 1.29.

Wajib: Ya

Tipe: String

`instance_type`

Ukuran instance.

Wajib: Ya

Tipe: String

`key_pair`

Amazon EC2 key pair untuk mengaktifkan akses SSH.

Wajib: Ya

Tipe: String

`root_volume_encryption`

Mengaktifkan enkripsi Amazon EBS untuk volume root Amazon EBS. Jika properti ini tidak disediakan, AWS TNB mengenkripsi volume root Amazon EBS secara default.

Wajib: Tidak

Default: betul

Jenis: Boolean

`root_volume_encryption_key_arn`

ARN dari kuncinya. AWS KMS AWS TNB mendukung ARN kunci reguler, ARN kunci multi-wilayah dan alias ARN.

Wajib: Tidak

Tipe: String

> **Note**
>
> - Jika `root_volume_encryption` salah, jangan sertakan`root_volume_encryption_key_arn`.

- AWS TNB mendukung enkripsi volume root dari AMI yang didukung Amazon EBS.
- Jika volume root AMI sudah dienkripsi, Anda harus menyertakan AWS TNB `root_volume_encryption_key_arn` untuk mengenkripsi ulang volume root.
- Jika volume root AMI tidak dienkripsi, AWS TNB menggunakan `root_volume_encryption_key_arn` untuk mengenkripsi volume root.

  Jika Anda tidak menyertakan`root_volume_encryption_key_arn`, AWS TNB menggunakan AWS Managed Services untuk mengenkripsi volume root.
- AWS TNB tidak mendekripsi AMI terenkripsi.

`root_volume_size`

Ukuran volume root Amazon Elastic Block Store di GiBs.

Wajib: Tidak

Default: 20

Jenis: Integer

Nilai yang mungkin: 1 hingga 16.384

`scaling`

Properti yang menentukan parameter penskalaan untuk node yang dikelola sendiri Amazon EKS, seperti, jumlah instans Amazon yang diinginkan, dan jumlah EC2 instans Amazon minimum dan maksimum dalam EC2 grup node.

`desired_size`

Jumlah contoh dalam hal ini NodeGroup.

Wajib: Ya

Jenis: Integer

`min_size`

Jumlah minimum contoh dalam hal ini NodeGroup.

Wajib: Ya

Jenis: Integer

`max_size`

Jumlah maksimum contoh dalam hal ini NodeGroup.

Wajib: Ya

Jenis: Integer

Properti

`node_role`

ARN dari peran IAM yang melekat pada instans Amazon. EC2

Wajib: Ya

Tipe: String

`tags`

Tag yang akan dilampirkan ke sumber daya. Tag akan disebarkan ke instance yang dibuat oleh sumber daya.

Wajib: Tidak

Tipe: Daftar

Persyaratan

`cluster`

Sebuah AWS simpul.compute.eks.

Wajib: Ya

Tipe: String

`subnets`

Sebuah AWS simpul.Networking.Subnet.

Wajib: Ya

Tipe: Daftar

`network_interfaces`

Sebuah AWS simpul.networking.ENI. Pastikan antarmuka jaringan dan subnet disetel ke Availability Zone yang sama atau instantiasi akan gagal.

Saat Anda AWS menyetel`network_interfaces`, TNB memperoleh izin yang terkait dengan ENIs dari `multus_role` properti jika Anda menyertakan `multus` properti di simpul AWS.compute.eks. Jika tidak, AWS TNB memperoleh izin yang terkait dengan ENIs dari properti node_role.

Wajib: Tidak

Tipe: Daftar

`security_groups`

Sebuah AWS.Networking. SecurityGroupsimpul.

Wajib: Tidak

Tipe: Daftar

`placement_group`

Sebuah tosca.nodes.AWS.Menghitung. PlacementGroupsimpul.

Wajib: Tidak

Tipe: String

`user_data`

Sebuah tosca.nodes.AWS.Menghitung. UserDatareferensi simpul. Skrip data pengguna diteruskan ke EC2 instans Amazon yang diluncurkan oleh grup node yang dikelola sendiri. Tambahkan izin yang diperlukan untuk mengeksekusi data pengguna kustom ke node_role yang diteruskan ke grup node.

Wajib: Tidak

Tipe: String

`labels`

Daftar label node. Label node harus memiliki nama dan nilai. Buat label menggunakan kriteria berikut:

- Nama dan nilai harus dipisahkan oleh=.
- Nama dan nilai masing-masing dapat mencapai 63 karakter panjangnya.
- Label dapat mencakup huruf (A-Z, a-z,), angka (0-9), dan karakter berikut: `[-, _, ., *, ?]`
- Nama dan nilai harus dimulai dan diakhiri dengan alfanumerik,?, atau karakter. *

Sebagai contoh, `myLabelName1=*NodeLabelValue1`.

Wajib: Tidak

Tipe: Daftar

Contoh

```
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.Menghitung. PlacementGroup

Sebuah PlacementGroup node mendukung berbagai strategi untuk menempatkan EC2 instans Amazon.

Saat Anda meluncurkan Amazon baru EC2instance, EC2 layanan Amazon mencoba menempatkan instance sedemikian rupa sehingga semua instans Anda tersebar di perangkat keras yang mendasarinya untuk meminimalkan kegagalan yang berkorelasi. Anda dapat menggunakan grup penempatan untuk mempengaruhi penempatan grup instans interdependen guna memenuhi kebutuhan beban kerja Anda.

Sintaksis

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

Properti

`strategy`

Strategi yang digunakan untuk menempatkan EC2 instance Amazon.

Wajib: Ya

Tipe: String

Nilai yang mungkin: CLUSTER | PARTISI | SPREAD_HOST | SPREAD_RACK

- CLUSTER — mengemas instance berdekatan di dalam Availability Zone. Strategi ini memungkinkan beban kerja untuk mencapai kinerja jaringan latensi rendah yang diperlukan untuk node-to-node komunikasi yang digabungkan secara ketat yang khas dari aplikasi komputasi kinerja tinggi (HPC).
- PARTISI — menyebarkan instance Anda di seluruh partisi logis sehingga grup instance dalam satu partisi tidak berbagi perangkat keras yang mendasarinya dengan grup instance di partisi yang berbeda. Strategi ini biasanya digunakan oleh beban kerja yang terdistribusi dan direplikasi besar, seperti Hadoop, Cassandra, dan Kafka.
- SPREAD_RACK — menempatkan sekelompok kecil instance di perangkat keras dasar yang berbeda untuk mengurangi kegagalan yang berkorelasi.
- SPREAD_HOST - digunakan hanya dengan kelompok penempatan Outpost. Menempatkan sekelompok kecil instance di perangkat keras dasar yang berbeda untuk mengurangi kegagalan yang berkorelasi.

`partition_count`

Jumlah partisi.

Wajib: Diperlukan hanya ketika `strategy` diatur ke`PARTITION`.

Jenis: Integer

Nilai yang mungkin: 1 | 2 |3 | 4 | 5 | 6 | 7

`tags`

Tag yang dapat Anda lampirkan ke sumber daya grup penempatan.

Wajib: Tidak

Tipe: Daftar

Contoh

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

```
      - tag_key=tag_value
```

AWS.Menghitung. UserData

AWS TNB mendukung peluncuran EC2 instans Amazon dengan data pengguna khusus, melalui UserData node di Network Service Descriptor (NSD). Untuk informasi selengkapnya tentang data pengguna kustom, lihat Data pengguna dan skrip shell di Panduan EC2 Pengguna Amazon.

Selama instantiasi jaringan, AWS TNB menyediakan pendaftaran EC2 instans Amazon ke cluster melalui skrip data pengguna. Ketika data pengguna khusus juga disediakan, AWS TNB menggabungkan kedua skrip dan meneruskannya sebagai skrip multimime ke Amazon. EC2 Skrip data pengguna khusus dijalankan sebelum skrip pendaftaran Amazon EKS.

Untuk menggunakan variabel kustom dalam skrip data pengguna, tambahkan tanda seru `!` setelah kurawal kurawal terbuka. `{` Misalnya, untuk digunakan `MyVariable` dalam skrip, masukkan: `{! MyVariable}`

> **Note**
>
> - AWS TNB mendukung skrip data pengguna hingga berukuran 7 KB.
> - Karena AWS TNB menggunakan AWS CloudFormation untuk memproses dan merender skrip `multimime` data pengguna, pastikan bahwa skrip mematuhi semua aturan. AWS CloudFormation

Sintaksis

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

Properti

`implementation`

Jalur relatif ke definisi skrip data pengguna. Formatnya harus: `./scripts/script_name.sh`

Wajib: Ya

Tipe: String

`content_type`

Jenis konten skrip data pengguna.

Wajib: Ya

Tipe: String

Nilai yang mungkin: `x-shellscript`

Contoh

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

AWS.Jaringan. SecurityGroup

AWS TNB mendukung grup keamanan untuk mengotomatiskan penyediaan Grup Keamanan EC2 Amazon yang dapat Anda lampirkan ke grup node cluster Amazon EKS Kubernetes.

Sintaksis

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

Properti

`description`

Deskripsi kelompok keamanan. Anda dapat menggunakan hingga 255 karakter untuk menggambarkan grup. Anda hanya dapat menyertakan huruf (A-Z dan a-z), angka (0-9), spasi, dan karakter khusus berikut: ._-:/() #, @ [] +=&; {}! $*

Wajib: Ya

Tipe: String

`name`

Nama untuk kelompok keamanan. Anda dapat menggunakan hingga 255 karakter untuk nama tersebut. Anda hanya dapat menyertakan huruf (A-Z dan a-z), angka (0-9), spasi, dan karakter khusus berikut: ._-:/() #, @ [] +=&; {}! $*

Wajib: Ya

Tipe: String

`tags`

Tag yang dapat Anda lampirkan ke sumber daya grup keamanan.

Wajib: Tidak

Tipe: Daftar

Persyaratan

`vpc`

Sebuah AWS simpul.networking.vpc.

Wajib: Ya

Tipe: String

Contoh

```
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.Jaringan. SecurityGroupEgressRule

AWS TNB mendukung aturan keluar grup keamanan untuk mengotomatiskan penyediaan Aturan Keluar Grup EC2 Keamanan Amazon yang dapat dilampirkan ke .Networking. AWS SecurityGroup. Perhatikan bahwa Anda harus memberikan cidr_ ip/destination_security_group/destination _prefix_list sebagai tujuan untuk lalu lintas keluar.

Sintaksis

```
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
```

Properti

`cidr_ip`

Rentang IPv4 alamat dalam format CIDR. Anda harus menentukan rentang CIDR yang memungkinkan lalu lintas keluar.

Wajib: Tidak

Tipe: String

`cidr_ipv6`

Rentang IPv6 alamat dalam format CIDR, untuk lalu lintas keluar. Anda harus menentukan grup keamanan tujuan (`destination_security_group` atau `destination_prefix_list`) atau rentang CIDR (`cidr_ip` atau `cidr_ipv6`).

Wajib: Tidak

Tipe: String

`description`

Deskripsi aturan grup keamanan jalan keluar (keluar). Anda dapat menggunakan hingga 255 karakter untuk menggambarkan aturan.

Wajib: Tidak

Tipe: String

`destination_prefix_list`

ID daftar awalan dari daftar awalan terkelola Amazon VPC yang ada. Ini adalah tujuan dari instance grup node yang terkait dengan grup keamanan. Untuk informasi selengkapnya tentang daftar awalan terkelola, lihat Daftar awalan terkelola di Panduan Pengguna Amazon VPC.

Wajib: Tidak

Tipe: String

`from_port`

Jika protokolnya adalah TCP atau UDP, ini adalah awal dari rentang port. Jika protokolnya adalah ICMP atau ICMPv6, ini adalah nomor jenisnya. Nilai -1 menunjukkan semua ICMP/ICMPv6 types. If you specify all ICMP/ICMPv6 types, you must specify all ICMP/ICMPv 6 kode.

Wajib: Tidak

Jenis: Integer

`ip_protocol`

Nama protokol IP (tcp, udp, icmp, icmpv6) atau nomor protokol. Gunakan -1 untuk menentukan semua protokol. Saat mengotorisasi aturan grup keamanan, menentukan -1 atau nomor protokol selain tcp, udp, icmp, atau icmpv6 memungkinkan lalu lintas di semua port, terlepas dari rentang port apa pun yang Anda tentukan. Untuk tcp, udp, dan icmp, Anda harus menentukan rentang port. Untuk icmpv6, rentang port adalah opsional; jika Anda menghilangkan rentang port, lalu lintas untuk semua jenis dan kode diperbolehkan.

Wajib: Ya

Tipe: String

`to_port`

Jika protokolnya adalah TCP atau UDP, ini adalah akhir dari rentang port. Jika protokolnya adalah ICMP atau ICMPv6, ini adalah kodenya. Nilai -1 menunjukkan semua ICMP/ICMPv6 codes. If you specify all ICMP/ICMPv6 types, you must specify all ICMP/ICMPv 6 kode.

Wajib: Tidak

Jenis: Integer

Persyaratan

`security_group`

ID grup keamanan tempat aturan ini akan ditambahkan.

Wajib: Ya

Tipe: String

`destination_security_group`

Referensi ID atau TOSCA dari grup keamanan tujuan yang diizinkan lalu lintas keluar.

Wajib: Tidak

Tipe: String

Contoh

```
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.Jaringan. SecurityGroupIngressRule

AWS TNB mendukung aturan masuknya grup keamanan untuk mengotomatiskan penyediaan Aturan Ingress Grup Keamanan EC2 Amazon yang dapat dilampirkan ke .Networking. AWS SecurityGroup. Perhatikan bahwa Anda harus memberikan cidr_ ip/source_security_group/source _prefix_list sebagai sumber untuk lalu lintas masuk.

Sintaksis

```
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
```

Properti

`cidr_ip`

Rentang IPv4 alamat dalam format CIDR. Anda harus menentukan rentang CIDR yang memungkinkan lalu lintas masuk.

Wajib: Tidak

Tipe: String

`cidr_ipv6`

Rentang IPv6 alamat dalam format CIDR, untuk lalu lintas masuk. Anda harus menentukan grup keamanan sumber (`source_security_groupatausource_prefix_list`) atau rentang CIDR (`cidr_ipataucidr_ipv6`).

Wajib: Tidak

Tipe: String

`description`

Deskripsi aturan grup keamanan ingress (inbound). Anda dapat menggunakan hingga 255 karakter untuk menggambarkan aturan.

Wajib: Tidak

Tipe: String

`source_prefix_list`

ID daftar awalan dari daftar awalan terkelola Amazon VPC yang ada. Ini adalah sumber dari mana instance grup node yang terkait dengan grup keamanan akan diizinkan untuk menerima lalu lintas dari. Untuk informasi selengkapnya tentang daftar awalan terkelola, lihat Daftar awalan terkelola di Panduan Pengguna Amazon VPC.

Wajib: Tidak

Tipe: String

`from_port`

Jika protokolnya adalah TCP atau UDP, ini adalah awal dari rentang port. Jika protokolnya adalah ICMP atau ICMPv6, ini adalah nomor jenisnya. Nilai -1 menunjukkan semua ICMP/ICMPv6 types. If you specify all ICMP/ICMPv6 types, you must specify all ICMP/ICMPv 6 kode.

Wajib: Tidak

Jenis: Integer

`ip_protocol`

Nama protokol IP (tcp, udp, icmp, icmpv6) atau nomor protokol. Gunakan -1 untuk menentukan semua protokol. Saat mengotorisasi aturan grup keamanan, menentukan -1 atau nomor protokol selain tcp, udp, icmp, atau icmpv6 memungkinkan lalu lintas di semua port, terlepas dari rentang port apa pun yang Anda tentukan. Untuk tcp, udp, dan icmp, Anda harus menentukan rentang port. Untuk icmpv6, rentang port adalah opsional; jika Anda menghilangkan rentang port, lalu lintas untuk semua jenis dan kode diperbolehkan.

Wajib: Ya

Tipe: String

`to_port`

Jika protokolnya adalah TCP atau UDP, ini adalah akhir dari rentang port. Jika protokolnya adalah ICMP atau ICMPv6, ini adalah kodenya. Nilai -1 menunjukkan semua ICMP/ICMPv6 codes. If you specify all ICMP/ICMPv6 types, you must specify all ICMP/ICMPv 6 kode.

Wajib: Tidak

Jenis: Integer

Persyaratan

`security_group`

ID grup keamanan tempat aturan ini akan ditambahkan.

Wajib: Ya

Tipe: String

`source_security_group`

Referensi ID atau TOSCA dari grup keamanan sumber tempat lalu lintas masuk diizinkan.

Wajib: Tidak

Tipe: String

Contoh

```
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.Impor

Anda dapat mengimpor AWS sumber daya berikut ke AWS TNB:

- VPC
- Subnet
- Tabel Rute
- Internet Gateway
- Grup Keamanan

Sintaksis

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

Properti

`resource_type`

Jenis sumber daya yang diimpor ke AWS TNB.

Wajib: Tidak

Tipe: Daftar

`resource_id`

ID sumber daya yang diimpor ke AWS TNB.

Wajib: Tidak

Tipe: Daftar

Contoh

```
SampleImportedVPC:
  type: tosca.nodes.AWS.Resource.Import
```

```
  properties:
    resource_type: "tosca.nodes.AWS.Networking.VPC"
    resource_id: "vpc-123456"
```

AWS.networking.eni

Antarmuka jaringan adalah komponen jaringan logis dalam VPC yang mewakili kartu jaringan virtual. Antarmuka jaringan diberi alamat IP baik secara otomatis atau manual berdasarkan subnetnya. Setelah menerapkan EC2 instans Amazon di subnet, Anda dapat melampirkan antarmuka jaringan ke subnet, atau melepaskan antarmuka jaringan dari instans Amazon tersebut dan menyambung kembali ke EC2 instans Amazon EC2 lain di subnet tersebut. Indeks perangkat mengidentifikasi posisi dalam urutan lampiran.

Sintaksis

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

Properti

`device_index`

Indeks perangkat harus lebih besar dari nol.

Wajib: Ya

Jenis: Integer

`source_dest_check`

Menunjukkan apakah antarmuka jaringan melakukan pemeriksaan sumber/tujuan. Nilai `true` berarti bahwa pemeriksaan diaktifkan, dan `false` berarti pemeriksaan dinonaktifkan.

Nilai yang diizinkan: true, false

Default: betul

Wajib: Tidak

Jenis: Boolean

`tags`

Tag yang akan dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

Persyaratan

`subnet`

Sebuah AWS simpul.Networking.Subnet.

Wajib: Ya

Tipe: String

`security_groups`

Sebuah AWS.Networking. SecurityGroupsimpul.

Wajib: Tidak

Tipe: String

Contoh

```
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

Pengait siklus hidup memberi Anda kemampuan untuk menjalankan skrip Anda sendiri sebagai bagian dari infrastruktur dan instantiasi jaringan Anda.

Sintaksis

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

Kemampuan

`execution`

Properti untuk mesin eksekusi hook yang menjalankan skrip hook.

`type`

Jenis mesin eksekusi hook.

Wajib: Tidak

Tipe: String

Nilai yang mungkin: `CODE_BUILD`

Persyaratan

`definition`

Sebuah AWS. HookDefinition.Bash simpul.

Wajib: Ya

Tipe: String

`vpc`

Sebuah AWS simpul.networking.vpc.

Wajib: Ya

Tipe: String

Contoh

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

AWS.Jaringan. InternetGateway

Mendefinisikan Node Gateway AWS Internet.

Sintaksis

```
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
```

Kemampuan

`routing`

Properti yang menentukan koneksi routing dalam VPC. Anda harus menyertakan `ipv6_dest_cidr` properti `dest_cidr` atau properti.

`dest_cidr`

Blok IPv4 CIDR yang digunakan untuk pencocokan tujuan. Properti ini digunakan untuk membuat rute masuk `RouteTable` dan nilainya digunakan sebagai`DestinationCidrBlock`.

Wajib: Tidak jika Anda menyertakan `ipv6_dest_cidr` properti.

Tipe: String

`ipv6_dest_cidr`

Blok IPv6 CIDR yang digunakan untuk pencocokan tujuan.

Wajib: Tidak jika Anda menyertakan `dest_cidr` properti.

Tipe: String

Properti

`tags`

Tag yang akan dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

`egress_only`

Properti IPv6 -spesifik. Menunjukkan apakah gateway internet hanya untuk komunikasi jalan keluar atau tidak. Kapan `egress_only` benar, Anda harus mendefinisikan `ipv6_dest_cidr` properti.

Wajib: Tidak

Jenis: Boolean

Persyaratan

`vpc`

Sebuah AWS simpul.networking.vpc.

Wajib: Ya

Tipe: String

`route_table`

Sebuah AWS.Networking. RouteTablesimpul.

Wajib: Ya

Tipe: String

Contoh

```
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.Jaringan. RouteTable

Tabel rute berisi seperangkat aturan, yang disebut rute, yang menentukan ke mana lalu lintas jaringan dari subnet dalam VPC atau gateway Anda diarahkan. Anda harus mengaitkan tabel rute dengan VPC.

Sintaksis

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

Properti

`tags`

Tag untuk dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

Persyaratan

`vpc`

Sebuah AWS simpul.networking.vpc.

Wajib: Ya

Tipe: String

Contoh

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

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

AWS.Networking.Subnet

Subnet adalah berbagai alamat IP di VPC Anda, dan harus berada sepenuhnya dalam satu Availability Zone. Anda harus menentukan VPC, blok CIDR, Availability Zone, dan tabel rute untuk subnet Anda. Anda juga harus menentukan apakah subnet Anda pribadi atau publik.

Sintaksis

```
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
```

Properti

`type`

Menunjukkan apakah instance yang diluncurkan di subnet ini menerima alamat publik IPv4.

Wajib: Ya

Tipe: String

Nilai yang mungkin: `PUBLIC` | `PRIVATE`

`availability_zone`

Availability Zone untuk subnet. Bidang ini mendukung AWS Availability Zone dalam suatu AWS Wilayah, misalnya `us-west-2` (US West (Oregon)). Ini juga mendukung AWS Local Zones dalam Availability Zone, misalnya`us-west-2-lax-1a`.

Wajib: Ya

Tipe: String

`cidr_block`

Blok CIDR untuk subnet.

Wajib: Tidak

Tipe: String

`ipv6_cidr_block`

Blok CIDR digunakan untuk membuat IPv6 subnet. Jika Anda menyertakan properti ini, jangan sertakan`ipv6_cidr_block_suffix`.

Wajib: Tidak

Tipe: String

`ipv6_cidr_block_suffix`

Sufiks heksadesimal 2 digit dari blok IPv6 CIDR untuk subnet yang dibuat melalui Amazon VPC. Gunakan format berikut: *`2-digit hexadecimal`*`::/`*`subnetMask`*

Jika Anda menyertakan properti ini, jangan sertakan`ipv6_cidr_block`.

Wajib: Tidak

Tipe: String

`outpost_arn`

ARN dari subnet AWS Outposts itu akan dibuat di. Tambahkan properti ini ke template NSD jika Anda ingin meluncurkan node yang dikelola sendiri Amazon EKS. AWS Outposts Untuk informasi selengkapnya, lihat Amazon EKS AWS Outposts di Panduan Pengguna Amazon EKS.

Jika Anda menambahkan properti ini ke template NSD, Anda harus menetapkan nilai untuk `availability_zone` properti ke Availability Zone dari AWS Outposts.

Wajib: Tidak

Tipe: String

`tags`

Tag yang akan dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

Persyaratan

`vpc`

Sebuah AWS simpul.networking.vpc.

Wajib: Ya

Tipe: String

`route_table`

Sebuah AWS.Networking. RouteTablesimpul.

Wajib: Ya

Tipe: String

Contoh

```
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.Penerapan. VNFDeployment

Penerapan NF dimodelkan dengan menyediakan infrastruktur dan aplikasi yang terkait dengannya. Atribut cluster menentukan kluster EKS untuk meng-host Anda NFs. Atribut vnfs menentukan fungsi jaringan untuk penyebaran Anda. Anda juga dapat menyediakan operasi kait siklus hidup opsional tipe pre_create dan post_create untuk menjalankan instruksi khusus untuk penerapan Anda, seperti memanggil API sistem Manajemen Inventaris.

Sintaksis

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

Persyaratan

`deployment`

Sebuah AWS.Deployment. VNFDeploymentsimpul.

Wajib: Tidak

Tipe: String

`cluster`

Sebuah AWS simpul.compute.eks.

Wajib: Ya

Tipe: String

`vnfs`

Sebuah AWS simpul.VNF.

Wajib: Ya

Tipe: String

Antarmuka

Kait

Mendefinisikan tahap saat kait siklus hidup dijalankan.

`pre_create`

Sebuah AWS. HookExecutionsimpul. Hook ini dijalankan sebelum `VNFDeployment` node menyebar.

Wajib: Tidak

Tipe: String

`post_create`

Sebuah AWS. HookExecutionsimpul. Hook ini dijalankan setelah `VNFDeployment` node menyebar.

Wajib: Tidak

Tipe: String

Contoh

```
SampleHelmDeploy:
```

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

AWS.networking.vpc

Anda harus menentukan blok CIDR untuk virtual private cloud (VPC) Anda.

Sintaksis

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

Properti

`cidr_block`

Rentang IPv4 jaringan untuk VPC, dalam notasi CIDR.

Wajib: Ya

Tipe: String

`ipv6_cidr_block`

Blok IPv6 CIDR digunakan untuk membuat VPC.

Nilai yang diizinkan: `AMAZON_PROVIDED`

Wajib: Tidak

Tipe: String

`dns_support`

Menunjukkan apakah instans yang diluncurkan di VPC mendapatkan nama host DNS.

Wajib: Tidak

Jenis: Boolean

Default: `false`

`tags`

Tag untuk dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

Contoh

```
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.Jaringan. NATGateway

Anda dapat menentukan node NAT Gateway publik atau pribadi melalui subnet. Untuk gateway publik, jika Anda tidak memberikan id alokasi IP Elastis, AWS TNB akan mengalokasikan IP Elastis untuk akun Anda dan mengaitkannya ke gateway.

Sintaksis

```
tosca.nodes.AWS.Networking.NATGateway:
  requirements:
    subnet: String
    internet_gateway: String
  properties:
```

```
    type: String
    eip_allocation_id: String
    tags: List
```

Properti

`subnet`

AWS Referensi simpul.Networking.Subnet.

Wajib: Ya

Tipe: String

`internet_gateway`

AWS.Networking. InternetGatewayreferensi simpul.

Wajib: Ya

Tipe: String

Properti

`type`

Menunjukkan apakah gateway bersifat publik atau pribadi.

Nilai yang diizinkan:`PUBLIC`, `PRIVATE`

Wajib: Ya

Tipe: String

`eip_allocation_id`

ID yang mewakili alokasi alamat IP Elastis.

Wajib: Tidak

Tipe: String

`tags`

Tag untuk dilampirkan ke sumber daya.

Wajib: Tidak

Tipe: Daftar

Contoh

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

AWS.Networking.Route

Anda dapat menentukan node rute yang mengaitkan rute tujuan ke NAT Gateway sebagai sumber daya target, dan menambahkan rute ke tabel rute terkait.

Sintaksis

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

Properti

`dest_cidr_blocks`

Daftar IPv4 rute tujuan ke sumber daya target.

Wajib: Ya

Tipe: Daftar

Jenis anggota: String

Persyaratan

`nat_gateway`

AWS.Networking. NATGatewayreferensi simpul.

Wajib: Ya

Tipe: String

`route_table`

AWS.Networking. RouteTablereferensi simpul.

Wajib: Ya

Tipe: String

Contoh

```
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.Toko. SSMParameters

Anda dapat membuat parameter SSM melalui AWS TNB. Parameter SSM yang Anda buat dibuat di SSM dan diawali dengan ID instance jaringan AWS TNB. Hal ini mencegah nilai parameter dari penggantian ketika beberapa instance dipakai dan ditingkatkan menggunakan template NSD yang sama.

Sintaksis

```
tosca.nodes.AWS.Store.SSMParameters
  properties:
```

```
    parameters:
      name: String
      value: String
    tags: List
```

Properti

Parameter

`name`

Nama properti ssm. Gunakan format berikut: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`

Setiap nama parameter harus kurang dari 256 karakter.

Wajib: Ya

Tipe: String

`value`

Nilai properti ssm. Gunakan salah satu format berikut:

- Untuk nilai tanpa referensi: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`
- Untuk referensi statis: `^\$\{[a-zA-Z0-9]+\.(properties|capabilities|requirements)(\.([a-zA-Z0-9\-_]+))+\}$`
- Untuk referensi dinamis: `^\$\{[a-zA-Z0-9]+\.(name|id|arn)\}$`

Nilai setiap parameter harus kurang dari 4 KB.

Wajib: Ya

Tipe: String

`tags`

Tag yang dapat Anda lampirkan ke properti SSM.

Wajib: Tidak

Tipe: Daftar

Contoh

```
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"
```

Node umum

Tentukan node untuk NSD dan VNFD.

- AWS. HookDefinition.Bash

AWS.HookDefinition.Bash

Mendefinisikan sebuah AWS HookDefinition in`bash`.

Sintaksis

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

Properti

`implementation`

Jalur relatif ke definisi hook. Formatnya harus: `./hooks/`*`script_name`*`.sh`

Wajib: Ya

Tipe: String

`environment_variables`

Variabel lingkungan untuk skrip hook bash. Gunakan format berikut: **`envName`**=**`envValue`** dengan pola regex berikut:

- Untuk nilai tanpa referensi: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+$`
- Untuk referensi statis: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=\$\{[a-zA-Z0-9]+\.(properties|capabilities|requirements)(\.([a-zA-Z0-9\-_]+))+\}$`
- Untuk referensi dinamis: `^[a-zA-Z0-9]+[a-zA-Z0-9\-\_]*[a-zA-Z0-9]+=\$\{[a-zA-Z0-9]+\.(name|id|arn)\}$`

Pastikan **`envName`**=**`envValue`** nilainya memenuhi kriteria berikut:

- Jangan gunakan spasi.
- Mulailah **`envName`** dengan huruf (A-Z atau a-z) atau angka (0-9).
- Jangan memulai nama variabel lingkungan dengan kata kunci yang dicadangkan AWS TNB berikut (case insensitive):
  - CODEBUILD
  - TNB
  - RUMAH
  - AWS
- Anda dapat menggunakan sejumlah huruf (A-Z atau a-z), angka (0-9), dan karakter khusus dan untuk - dan_. **`envName envValue`**
- Setiap variabel lingkungan (masing-masing **`envName`** =**`envValue`**) harus kurang dari 128 karakter.

Contoh: `A123-45xYz=Example_789`

Wajib: Tidak

Tipe: Daftar

`execution_role`

Peran untuk eksekusi hook.

Wajib: Ya

Tipe: String

Contoh

```
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"
```

Keamanan di AWS TNB

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan Program AWS Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku untuk Pembuat Jaringan AWS Telco, lihat AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS TNB. Topik berikut menunjukkan cara mengonfigurasi AWS TNB untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya AWS TNB Anda.

Daftar Isi

Perlindungan data di AWS TNB

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di AWS Telco Network Builder. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensil dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk ketika Anda bekerja dengan AWS TNB atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan

atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Penanganan data

Saat Anda menutup AWS akun, AWS TNB menandai data Anda untuk dihapus dan menghapusnya dari penggunaan apa pun. Jika Anda mengaktifkan kembali AWS akun Anda dalam waktu 90 hari, AWS TNB mengembalikan data Anda. Setelah 120 hari, AWS TNB menghapus data Anda secara permanen. AWS TNB juga menghentikan jaringan Anda dan menghapus paket fungsi dan paket jaringan Anda.

Enkripsi diam

AWS TNB selalu mengenkripsi semua data yang disimpan dalam layanan saat istirahat tanpa memerlukan konfigurasi tambahan. Enkripsi ini otomatis melalui AWS Key Management Service.

Enkripsi bergerak

AWS TNB mengamankan semua data dalam perjalanan menggunakan Transport Layer Security (TLS) 1.2.

Merupakan tanggung jawab Anda untuk mengenkripsi data antara agen simulasi Anda dan klien mereka.

Privasi lalu lintas antar jaringan

AWS Sumber daya komputasi TNB berada di virtual private cloud (VPC) yang dibagikan oleh semua pelanggan. Semua lalu lintas AWS TNB internal tetap berada dalam AWS jaringan dan tidak melintasi internet. Koneksi antara agen simulasi Anda dan klien mereka diarahkan melalui internet.

Manajemen identitas dan akses AWS TNB

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya TNB. AWS IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Daftar Isi

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di AWS TNB.

Pengguna layanan — Jika Anda menggunakan layanan AWS TNB untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak fitur AWS TNB untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di AWS TNB, lihatMemecahkan masalah identitas dan AWS akses Telco Network Builder.

Administrator layanan — Jika Anda bertanggung jawab atas sumber daya AWS TNB di perusahaan Anda, Anda mungkin memiliki akses penuh ke AWS TNB. Tugas Anda adalah menentukan fitur dan sumber daya AWS TNB mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasar IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM dengan AWS TNB, lihat. Bagaimana AWS TNB bekerja dengan IAM

Administrator IAM — Jika Anda seorang administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses ke AWS TNB. Untuk melihat contoh kebijakan berbasis identitas AWS TNB yang dapat Anda gunakan di IAM, lihat. Contoh kebijakan berbasis identitas untuk AWS Telco Network Builder

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensyal identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensyal yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensil Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Guna mengetahui informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat AWS Signature Version 4 untuk permintaan API dalam Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Autentikasi multi-faktor AWS di IAM dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensil yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensyal sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apakah itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat meminta kelompok untuk menyebutkan IAMAdmins dan memberikan izin kepada grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat Kasus penggunaan untuk pengguna IAM dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Untuk mengambil peran IAM sementara AWS Management Console, Anda dapat beralih dari pengguna ke peran IAM (konsol). Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat Metode untuk mengambil peran dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Buat peran untuk penyedia identitas pihak ketiga dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan

hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

- Peran layanan – Peran layanan adalah peran IAM yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke peran layanan. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS permintaan API. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensi sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon di Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat Pilih antara kebijakan yang dikelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnya ACLs, lihat Ringkasan daftar kontrol akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCPs) — SCPs adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organizations dan SCPs, lihat Kebijakan kontrol layanan di Panduan AWS Organizations Pengguna.
- Kebijakan kontrol sumber daya (RCPs) — RCPs adalah kebijakan JSON yang dapat Anda gunakan untuk menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda tanpa memperbarui kebijakan IAM yang dilampirkan ke setiap sumber daya yang Anda miliki. RCP membatasi izin untuk sumber daya di akun anggota dan dapat memengaruhi izin efektif untuk identitas, termasuk Pengguna root akun AWS, terlepas dari apakah itu milik organisasi Anda. Untuk informasi selengkapnya tentang Organizations dan RCPs, termasuk daftar dukungan

Layanan AWS tersebut RCPs, lihat Kebijakan kontrol sumber daya (RCPs) di Panduan AWS Organizations Pengguna.

- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

Bagaimana AWS TNB bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke AWS TNB, pelajari fitur IAM apa yang tersedia untuk digunakan dengan TNB. AWS

Fitur IAM yang dapat Anda gunakan dengan AWS Telco Network Builder

Fitur IAM	AWS Dukungan TNB
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
Kunci kondisi kebijakan	Ya
ACLs	Tidak
ABAC (tanda dalam kebijakan)	Ya
Kredensial sementara	Ya

Fitur IAM	AWS Dukungan TNB
Izin principal	Ya
Peran layanan	Tidak
Peran terkait layanan	Tidak

Untuk mendapatkan pandangan tingkat tinggi tentang bagaimana AWS TNB dan AWS layanan lainnya bekerja dengan sebagian besar fitur IAM, lihat AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk TNB AWS

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk TNB AWS

Untuk melihat contoh kebijakan berbasis identitas AWS TNB, lihat. Contoh kebijakan berbasis identitas untuk AWS Telco Network Builder

Kebijakan berbasis sumber daya dalam TNB AWS

Mendukung kebijakan berbasis sumber daya: Tidak

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke principal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

Tindakan kebijakan untuk AWS TNB

Mendukung tindakan kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar tindakan AWS TNB, lihat Tindakan yang ditentukan oleh Pembuat Jaringan AWS Telco di Referensi Otorisasi Layanan.

Tindakan kebijakan di AWS TNB menggunakan awalan berikut sebelum tindakan:

```
tnb
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

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

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard (*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata `List`, sertakan tindakan berikut:

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

Untuk melihat contoh kebijakan berbasis identitas AWS TNB, lihat. Contoh kebijakan berbasis identitas untuk AWS Telco Network Builder

Sumber daya kebijakan untuk AWS TNB

Mendukung sumber daya kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

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

Untuk melihat daftar jenis sumber daya AWS TNB dan jenisnya ARNs, lihat Sumber daya yang ditentukan oleh Pembuat Jaringan AWS Telco di Referensi Otorisasi Layanan. Untuk mempelajari

tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang ditentukan oleh Pembuat Jaringan AWS Telco.

Untuk melihat contoh kebijakan berbasis identitas AWS TNB, lihat. Contoh kebijakan berbasis identitas untuk AWS Telco Network Builder

Kunci kondisi kebijakan untuk AWS TNB

Mendukung kunci kondisi kebijakan khusus layanan: Yes

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi `AND` logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan `OR` operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tanda yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tanda dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Untuk melihat daftar kunci kondisi AWS TNB, lihat Kunci kondisi untuk Pembuat Jaringan AWS Telco di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Tindakan yang ditentukan oleh Pembuat Jaringan AWS Telco.

Untuk melihat contoh kebijakan berbasis identitas AWS TNB, lihat. Contoh kebijakan berbasis identitas untuk AWS Telco Network Builder

ACLs di AWS TNB

Mendukung ACLs: Tidak

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

ABAC dengan TNB AWS

Mendukung ABAC (tanda dalam kebijakan): Ya

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tanda milik prinsipal cocok dengan tanda yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tanda, berikan informasi tentang tanda di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Tentukan izin dengan otorisasi ABAC dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

Menggunakan kredensil sementara dengan TNB AWS

Mendukung kredensial sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensil sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensi sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis

membuat kredensil sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Beralih dari pengguna ke peran IAM (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensyal sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensil sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensyal sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

Izin utama lintas layanan untuk TNB AWS

Mendukung sesi akses maju (FAS): Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

Peran layanan untuk AWS TNB

Mendukung peran layanan: Tidak

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

Peran terkait layanan untuk TNB AWS

Mendukung peran terkait layanan: Tidak

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul

di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Contoh kebijakan berbasis identitas untuk AWS Telco Network Builder

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi sumber daya AWS TNB. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM (konsol) di Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh AWS TNB, termasuk format ARNs untuk setiap jenis sumber daya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Pembuat Jaringan AWS Telco di Referensi Otorisasi Layanan.

Daftar Isi

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya AWS TNB di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi

selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.

- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Amankan akses API dengan MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM.

Menggunakan konsol AWS TNB

Untuk mengakses konsol Pembuat Jaringan AWS Telco, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang sumber daya AWS TNB di Anda. Akun AWS Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada

izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang sesuai dengan operasi API yang coba mereka lakukan.

Contoh kebijakan peran layanan

Sebagai administrator, Anda memiliki dan mengelola sumber daya yang dibuat AWS TNB seperti yang ditentukan oleh template lingkungan dan layanan. Anda harus melampirkan peran layanan IAM ke akun Anda untuk mengizinkan AWS TNB membuat sumber daya untuk manajemen siklus hidup jaringan Anda.

Peran layanan IAM memungkinkan AWS TNB melakukan panggilan ke sumber daya atas nama Anda untuk membuat instance dan mengelola jaringan Anda. Jika Anda menentukan peran layanan, AWS TNB menggunakan kredensi peran tersebut.

Anda membuat peran layanan dan kebijakan izinnya dengan layanan IAM. Untuk informasi selengkapnya tentang membuat peran layanan, lihat Membuat peran untuk mendelegasikan izin ke AWS layanan di Panduan Pengguna IAM.

AWS Peran layanan TNB

Sebagai anggota tim platform, Anda dapat sebagai administrator membuat peran layanan AWS TNB dan memberikannya kepada AWS TNB. Peran ini memungkinkan AWS TNB untuk melakukan panggilan ke layanan lain seperti Amazon Elastic Kubernetes AWS CloudFormation Service dan untuk menyediakan infrastruktur yang diperlukan untuk jaringan dan fungsi jaringan penyediaan Anda sebagaimana didefinisikan dalam NSD Anda.

Kami menyarankan Anda menggunakan peran IAM berikut dan kebijakan kepercayaan untuk peran layanan AWS TNB Anda. Saat mencentang izin pada kebijakan ini, ingatlah bahwa AWS TNB mungkin gagal dengan kesalahan Akses Ditolak terhadap sumber daya yang dihapus dari kebijakan Anda.

Kode berikut menunjukkan kebijakan peran layanan AWS TNB:

```
{
    "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",
                "iam:PassRole"
            ],
            "Resource": "*",
            "Effect": "Allow",
```

```
            "Sid": "TNBAccessComputePerms"
        },
        {
            "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: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"
        }
    ]
}
```

Kode berikut menunjukkan kebijakan kepercayaan layanan AWS TNB:

```
{
  "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 Peran layanan TNB untuk klaster Amazon EKS

Saat membuat resource Amazon EKS di NSD, Anda memberikan `cluster_role` atribut untuk menentukan peran mana yang akan digunakan untuk membuat klaster Amazon EKS Anda.

Contoh berikut menunjukkan AWS CloudFormation template yang membuat peran layanan AWS TNB untuk kebijakan klaster 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"
```

Untuk informasi selengkapnya tentang peran IAM menggunakan AWS CloudFormation templat, lihat bagian berikut di Panduan AWS CloudFormation Pengguna:

- AWS::IAM::Role
- Memilih template tumpukan

AWS Peran layanan TNB untuk grup node Amazon EKS

Saat membuat resource grup node Amazon EKS di NSD, Anda memberikan `node_role` atribut untuk menentukan peran mana yang akan digunakan untuk membuat grup node Amazon EKS Anda.

Contoh berikut menunjukkan AWS CloudFormation template yang membuat peran layanan AWS TNB untuk kebijakan grup node 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/*"
```

Untuk informasi selengkapnya tentang peran IAM menggunakan AWS CloudFormation templat, lihat bagian berikut di Panduan AWS CloudFormation Pengguna:

- AWS::IAM::Role
- Memilih template tumpukan

AWS Peran layanan TNB untuk Multus

Saat Anda membuat resource Amazon EKS di NSD Anda dan Anda ingin mengelola Multus sebagai bagian dari template penerapan Anda, Anda harus memberikan `multus_role` atribut untuk menentukan peran mana yang akan digunakan untuk mengelola Multus.

Contoh berikut menunjukkan AWS CloudFormation template yang membuat peran layanan AWS TNB untuk kebijakan 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: "*"
```

Untuk informasi selengkapnya tentang peran IAM menggunakan AWS CloudFormation templat, lihat bagian berikut di Panduan AWS CloudFormation Pengguna:

- AWS::IAM::Role
- Memilih template tumpukan

AWS Peran layanan TNB untuk kebijakan hook siklus hidup

Ketika NSD atau paket fungsi jaringan Anda menggunakan hook siklus hidup, Anda memerlukan peran layanan untuk memungkinkan Anda menciptakan lingkungan untuk eksekusi kait siklus hidup Anda.

> **Note**
>
> Kebijakan pengait siklus hidup Anda harus didasarkan pada apa yang coba dilakukan oleh pengait siklus hidup Anda.

Contoh berikut menunjukkan AWS CloudFormation template yang membuat peran layanan AWS TNB untuk kebijakan hook siklus hidup.

```
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"
```

Untuk informasi selengkapnya tentang peran IAM menggunakan AWS CloudFormation templat, lihat bagian berikut di Panduan AWS CloudFormation Pengguna:

- AWS::IAM::Role
- Memilih template tumpukan

Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI 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": "*"
        }
    ]
}
```

Memecahkan masalah identitas dan AWS akses Telco Network Builder

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan AWS TNB dan IAM.

Masalah

- Saya tidak berwenang untuk melakukan tindakan di AWS TNB
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang-orang di luar saya Akun AWS untuk mengakses sumber daya AWS TNB saya

Saya tidak berwenang untuk melakukan tindakan di AWS TNB

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya fiktif *`my-example-widget`*, tetapi tidak memiliki izin fiktif `tnb:`*`GetWidget`*.

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

Dalam hal ini, kebijakan Mateo harus diperbarui untuk memungkinkannya mengakses *`my-example-widget`* sumber daya menggunakan `tnb:`*`GetWidget`* tindakan tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan bahwa Anda tidak berwenang untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke AWS TNB.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di AWS TNB. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

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

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang-orang di luar saya Akun AWS untuk mengakses sumber daya AWS TNB saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACLs), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mengetahui apakah AWS TNB mendukung fitur-fitur ini, lihatBagaimana AWS TNB bekerja dengan IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

Validasi kepatuhan untuk TNB AWS

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Kepatuhan dan Tata Kelola Keamanan – Panduan implementasi solusi ini membahas pertimbangan arsitektur serta memberikan langkah-langkah untuk menerapkan fitur keamanan dan kepatuhan.
- Referensi Layanan yang Memenuhi Syarat HIPAA — Daftar layanan yang memenuhi syarat HIPAA. Tidak semua memenuhi Layanan AWS syarat HIPAA.
- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan,

seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.

- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Ketahanan di TNB AWS

Infrastruktur AWS global dibangun di sekitar Wilayah AWS dan Availability Zones. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur AWS Global.

AWS TNB menjalankan Layanan Jaringan Anda di kluster EKS di cloud pribadi virtual (VPC) di AWS Wilayah yang Anda pilih.

Keamanan infrastruktur di AWS TNB

Sebagai layanan terkelola, AWS Telco Network Builder dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses AWS TNB melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

Berikut adalah beberapa contoh tanggung jawab bersama:

- AWS bertanggung jawab untuk mengamankan komponen yang mendukung AWS TNB, termasuk:
  - Contoh komputasi (juga dikenal sebagai pekerja)
  - Database internal
  - Komunikasi jaringan antar komponen internal
  - Antarmuka pemrograman aplikasi AWS TNB (API)
  - AWS Kit Pengembangan Perangkat Lunak (SDK)
- Anda bertanggung jawab untuk mengamankan akses ke AWS sumber daya dan komponen beban kerja Anda, termasuk (namun tidak terbatas pada):
  - Pengguna, grup, peran, dan kebijakan IAM
  - Bucket S3 yang Anda gunakan untuk menyimpan data Anda untuk TNB AWS
  - Sumber lain Layanan AWS dan sumber daya yang Anda gunakan untuk mendukung layanan jaringan yang Anda berikan melalui TNB AWS
  - Kode aplikasi Anda
  - Koneksi antara layanan jaringan yang Anda berikan melalui AWS TNB dan kliennya

> **Important**
>
> Anda bertanggung jawab untuk menerapkan rencana pemulihan bencana yang secara efektif dapat memulihkan layanan jaringan yang Anda berikan melalui AWS TNB.

Model keamanan konektivitas jaringan

Layanan jaringan yang Anda sediakan melalui AWS TNB, berjalan pada instans komputasi dalam virtual private cloud (VPC) yang terletak di Wilayah yang AWS Anda pilih. VPC adalah jaringan virtual di AWS Cloud, yang mengisolasi infrastruktur berdasarkan beban kerja atau entitas organisasi. Komunikasi antara instance komputasi dalam VPCs tetap berada dalam AWS jaringan dan tidak melakukan perjalanan melalui internet. Beberapa komunikasi layanan internal melintasi internet,

dan dienkripsi. Layanan jaringan yang disediakan melalui AWS TNB untuk semua pelanggan yang berjalan di Wilayah yang sama berbagi VPC yang sama. Layanan jaringan yang disediakan melalui AWS TNB untuk pelanggan yang berbeda menggunakan instans komputasi terpisah dalam VPC yang sama.

Komunikasi antara klien layanan jaringan Anda dan layanan jaringan Anda di AWS TNB melintasi internet. AWS TNB tidak mengelola koneksi ini. Adalah tanggung jawab Anda untuk mengamankan koneksi klien Anda.

Koneksi Anda ke AWS TNB melalui AWS Management Console, AWS Command Line Interface (AWS CLI), dan AWS SDKs dienkripsi.

Versi IMDS

AWS TNB mendukung instans yang memanfaatkan Layanan Metadata Instans versi 2 (IMDSv2), metode berorientasi sesi. IMDSv2 termasuk keamanan yang lebih tinggi dari IMDSV1. Untuk informasi selengkapnya, lihat Menambahkan pertahanan secara mendalam terhadap firewall terbuka, proxy terbalik, dan kerentanan SSRF dengan penyempurnaan pada Layanan Metadata Instans Amazon. EC2

Saat meluncurkan instance Anda, Anda harus menggunakan IMDSv2. Untuk informasi selengkapnya IMDSv2, lihat Menggunakan IMDSv2 di Panduan EC2 Pengguna Amazon.

Pemantauan AWS TNB

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja AWS TNB dan AWS solusi Anda yang lain. AWS menyediakan AWS CloudTrail untuk menonton AWS TNB, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu.

Gunakan CloudTrail untuk menangkap informasi terperinci tentang panggilan yang dilakukan AWS APIs. Anda dapat menyimpan panggilan ini sebagai file log di Amazon S3. Anda dapat menggunakan CloudTrail log ini untuk menentukan informasi seperti panggilan mana yang dibuat, alamat IP sumber dari mana panggilan itu berasal, siapa yang melakukan panggilan, dan kapan panggilan dilakukan.

CloudTrail Log berisi informasi tentang panggilan ke tindakan API untuk AWS TNB. Mereka juga berisi informasi untuk panggilan ke tindakan API dari layanan seperti Amazon EC2 dan Amazon EBS.

AWS Pencatatan panggilan API Pembuat Jaringan Telco menggunakan AWS CloudTrail

AWS Telco Network Builder terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau. Layanan AWS CloudTrail menangkap semua panggilan API untuk AWS TNB sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari konsol AWS TNB dan panggilan kode ke operasi AWS TNB API. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke AWS TNB, alamat IP dari mana permintaan dibuat, kapan dibuat, dan detail tambahan.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut hal ini:

- Baik permintaan tersebut dibuat dengan kredensial pengguna root atau pengguna.
- Apakah permintaan dibuat atas nama pengguna Pusat Identitas IAM.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan tersebut dibuat oleh Layanan AWS lain.

CloudTrail aktif di Anda Akun AWS ketika Anda membuat akun dan Anda secara otomatis memiliki akses ke riwayat CloudTrail Acara. Riwayat CloudTrail Acara menyediakan catatan yang dapat

dilihat, dapat dicari, dapat diunduh, dan tidak dapat diubah dari 90 hari terakhir dari peristiwa manajemen yang direkam dalam file. Wilayah AWS Untuk informasi selengkapnya, lihat Bekerja dengan riwayat CloudTrail Acara di Panduan AWS CloudTrail Pengguna. Tidak ada CloudTrail biaya untuk melihat riwayat Acara.

Untuk catatan acara yang sedang berlangsung dalam 90 hari Akun AWS terakhir Anda, buat jejak atau penyimpanan data acara CloudTrailDanau.

CloudTrail jalan setapak

Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Semua jalur yang dibuat menggunakan AWS Management Console Multi-region. Anda dapat membuat jalur Single-region atau Multi-region dengan menggunakan. AWS CLI Membuat jejak Multi-wilayah disarankan karena Anda menangkap aktivitas Wilayah AWS di semua akun Anda. Jika Anda membuat jejak wilayah Tunggal, Anda hanya dapat melihat peristiwa yang dicatat di jejak. Wilayah AWS Untuk informasi selengkapnya tentang jejak, lihat Membuat jejak untuk Anda Akun AWS dan Membuat jejak untuk organisasi di Panduan AWS CloudTrail Pengguna.

Anda dapat mengirimkan satu salinan acara manajemen yang sedang berlangsung ke bucket Amazon S3 Anda tanpa biaya CloudTrail dengan membuat jejak, namun, ada biaya penyimpanan Amazon S3. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga. Untuk informasi tentang harga Amazon S3, lihat Harga Amazon S3.

CloudTrail Menyimpan data acara danau

CloudTrail Lake memungkinkan Anda menjalankan kueri berbasis SQL pada acara Anda. CloudTrail Lake mengonversi peristiwa yang ada dalam format JSON berbasis baris ke format Apache ORC. ORC adalah format penyimpanan kolumnar yang dioptimalkan untuk pengambilan data dengan cepat. Peristiwa digabungkan ke dalam penyimpanan data peristiwa, yang merupakan kumpulan peristiwa yang tidak dapat diubah berdasarkan kriteria yang Anda pilih dengan menerapkan pemilih acara tingkat lanjut. Penyeleksi yang Anda terapkan ke penyimpanan data acara mengontrol peristiwa mana yang bertahan dan tersedia untuk Anda kueri. Untuk informasi lebih lanjut tentang CloudTrail Danau, lihat Bekerja dengan AWS CloudTrail Danau di Panduan AWS CloudTrail Pengguna.

CloudTrail Penyimpanan data acara danau dan kueri menimbulkan biaya. Saat Anda membuat penyimpanan data acara, Anda memilih opsi harga yang ingin Anda gunakan untuk penyimpanan data acara. Opsi penetapan harga menentukan biaya untuk menelan dan menyimpan peristiwa, dan periode retensi default dan maksimum untuk penyimpanan data acara. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga.

AWS Contoh acara TNB

Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang operasi API yang diminta, tanggal dan waktu operasi, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga peristiwa tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan CloudTrail peristiwa yang menunjukkan `CreateSolFunctionPackage` operasi.


```
{
    "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"
}
```

Untuk informasi tentang konten CloudTrail rekaman, lihat konten CloudTrail rekaman di Panduan AWS CloudTrail Pengguna.

AWS Tugas penyebaran TNB

Memahami tugas penerapan untuk secara efektif memantau penerapan dan mengambil tindakan lebih cepat.

Tabel berikut mencantumkan tugas penerapan AWS TNB:

Nama tugas untuk penerapan dimulai sebelum 7 Maret 2024	Nama tugas untuk penerapan dimulai pada dan setelah 7 Maret 2024	Deskripsi tugas
AppInstallation	ClusterPluginInstall	Menginstal plugin Multus di kluster Amazon EKS.
AppUpdate	tidak ada perubahan nama	Memperbarui fungsi jaringan yang sudah diinstal dalam instance jaringan.
-	ClusterPluginUninstall	Mencopot pemasangan plugin di kluster Amazon EKS.
ClusterStorageClassConfiguration	tidak ada perubahan nama	Mengkonfigurasi kelas penyimpanan (driver CSI) pada kluster Amazon EKS.

Nama tugas untuk penerapan dimulai sebelum 7 Maret 2024	Nama tugas untuk penerapan dimulai pada dan setelah 7 Maret 2024	Deskripsi tugas
FunctionDeletion	tidak ada perubahan nama	Menghapus fungsi jaringan dari sumber daya AWS TNB.
FunctionInstantiation	FunctionInstall	Menyebarkan fungsi jaringan menggunakan HELM.
FunctionUninstallation	FunctionUninstall	Mencopot pemasangan fungsi jaringan dari kluster Amazon EKS.
HookExecution	tidak ada perubahan nama	Mengeksekusi kait siklus hidup seperti yang didefinisikan dalam NSD.
InfrastructureCancellation	tidak ada perubahan nama	Membatalkan layanan jaringan.
InfrastructureInstantation	tidak ada perubahan nama	Ketentuan AWS sumber daya atas nama pengguna.
InfrastructureTermination	tidak ada perubahan nama	Pembatalan AWS sumber daya yang dimohonkan melalui TNB AWS .
-	InfrastructureUpdate	Memperbarui AWS sumber daya yang disediakan atas nama pengguna.
InventoryDeregistration	tidak ada perubahan nama	Deregisters AWS sumber daya dari TNB. AWS
-	InventoryRegistration	Mendaftarkan AWS sumber daya di AWS TNB.
KubernetesClusterConfiguration	ClusterConfiguration	Mengonfigurasi klaster Kubernetes dan menambahkan peran IAM tambahan ke Amazon EKS AuthMap seperti yang didefinisikan dalam NSD.

Nama tugas untuk penerapan dimulai sebelum 7 Maret 2024	Nama tugas untuk penerapan dimulai pada dan setelah 7 Maret 2024	Deskripsi tugas
NetworkServiceFinalization	tidak ada perubahan nama	Menyelesaikan layanan jaringan dan memberikan pembaruan status keberhasilan atau kegagalan.
NetworkServiceInstantiation	tidak ada perubahan nama	Menginisialisasi layanan jaringan.
SelfManagedNodesConfiguration	tidak ada perubahan nama	Bootstrap node yang dikelola sendiri dengan Amazon EKS dan bidang kontrol Kubernetes.
-	ValidateNetworkServiceUpdate	Menjalankan validasi sebelum memperbarui instance jaringan.

Kuota layanan untuk TNB AWS

Kuota layanan, juga disebut sebagai batas, adalah jumlah maksimum sumber daya layanan atau operasi untuk AWS akun Anda. Untuk informasi selengkapnya, lihat kuota layanan AWS di Referensi Umum Amazon Web Services.

Berikut ini adalah kuota layanan untuk AWS TNB.

Nama	Default	Dapat disesuaikan	Deskripsi
Operasi layanan jaringan yang sedang berlangsung bersamaan	Setiap Wilayah yang didukung: 40	Ya	Jumlah maksimum operasi layanan jaringan yang sedang berlangsung bersamaan di satu Wilayah.
Paket fungsi	Setiap Wilayah yang didukung: 200	Ya	Jumlah maksimum paket fungsi dalam satu Wilayah.
Paket jaringan	Setiap Wilayah yang didukung: 40	Ya	Jumlah maksimum paket jaringan dalam satu Wilayah.
Contoh layanan jaringan	Setiap Wilayah yang didukung: 800	Ya	Jumlah maksimum instance layanan jaringan dalam satu Wilayah.

Riwayat dokumen untuk panduan pengguna AWS TNB

Tabel berikut menjelaskan rilis dokumentasi untuk AWS TNB.

Perubahan	Deskripsi	Tanggal
Penambahan dan penghapusan grup node Amazon EKS di cluster yang ada	AWS TNB sekarang mendukung penambahan grup node baru dan menghapus grup node yang ada dari kluster Amazon EKS. Untuk informasi selengkapnya, lihat Parameter yang dapat Anda perbarui.	Juni 4, 2025
Ukuran volume akar	Anda dapat menentukan ukuran volume root Amazon EBS yang mendasari node pekerja Amazon EKS Anda melalui `root_volume_size` bidang di AWS.Compute. EKSManagedNode dan AWS.Compute. EKSSelfManagedNode Node TOSCA.	19 Mei 2025
Sumber referensi dalam skrip	Anda dapat mereferensikan sumber daya yang dibuat oleh AWS TNB untuk mengonfigurasinya dalam skrip Lifecycle Hook dan skrip data pengguna Anda.	2 Mei 2025
Kubernetes versi 1.32 sekarang didukung untuk node Amazon EKS dan grup node terkelola.	AWS TNB mendukung Kubernetes versi 1.32 untuk .compute.eks dan .Compute.AWSAWS EKSManagedSimpul.	April 24, 2025

Kubernetes versi 1.24 tidak lagi didukung untuk node Amazon EKS dan grup node terkelola	AWS TNB tidak lagi mendukung Kubernetes versi 1.24 untuk .compute.eks dan .Compute.AWSAWS EKSManagedSimpul.	17 April 2025
AL2023 Dukungan AMI untuk node terkelola Amazon EKS	AWS TNB mendukung AL2 023 tipe AMI untuk AWS.Compute. EKSManage dSimpul.	17 April 2025
Kubernetes versi 1.23 tidak lagi didukung untuk node Amazon EKS dan grup node terkelola	AWS TNB tidak lagi mendukung Kubernetes versi 1.23 untuk .compute.eks dan .Compute.AWSAWS EKSManagedSimpul.	April 4, 2025
ID AMI dapat diperbarui	Anda sekarang dapat memperbarui bidang ami_id selama panggilan `UpdateSol NetworkService` API.	Maret 31, 2025
Kubernetes versi 1.31 sekarang didukung untuk node Amazon EKS dan grup node terkelola.	AWS TNB mendukung Kubernetes versi 1.31 untuk .compute.eks dan .Compute.AWSAWS EKSManagedSimpul.	Februari 18, 2025
Versi Kubernetes untuk .Compute. AWS EKSManagedNode	AWS TNB mendukung Kubernetes versi 1.23 hingga 1.30 untuk membuat grup node terkelola Amazon EKS.	Januari 28, 2025
Versi Kubernetes untuk klaster	AWS TNB sekarang mendukung Kubernetes versi 1.30 untuk membuat cluster Amazon EKS.	Agustus 19, 2024

AWS TNB mendukung operasi tambahan untuk mengelola siklus hidup jaringan.	Anda dapat memperbarui instance jaringan yang dipakai atau diperbarui sebelumnya dengan paket jaringan baru dan nilai parameter. Lihat:
	• Operasi siklus hidup
	• Perbarui instance jaringan
	• AWS Contoh peran layanan TNB:
	  • Tambahkan tindakan Amazon EKS ini:`eks:UpdateAddon` ,`eks:UpdateClusterVersion` ,`eks:UpdateNodegroupConfig` ,`eks:UpdateNodegroupVersion` , `eks:DescribeUpdate`
	  • Tambahkan AWS CloudFormation tindakan ini: `cloudformation:UpdateStack`
	• Tugas Deployment baru:`InfrastructureUpdate` ,, `InventoryRegistration` `ValidateNetworkServiceUpdate`
	• Pembaruan API: GetSolNetworkOperation, ListSolNe
		Juli 30, 2024

	tworkOperations , dan UpdateSolNetworkInstance	
Tugas baru dan nama tugas baru untuk tugas yang ada	Tugas baru tersedia. Pada 7 Maret 2024, beberapa tugas yang ada memiliki nama baru untuk kejelasan.	7 Mei 2024
Versi Kubernetes untuk klaster	AWS TNB sekarang mendukung Kubernetes versi 1.29 untuk membuat cluster Amazon EKS.	April 10, 2024
Support untuk antarmuka jaringan `security_groups`	Anda dapat melampirkan grup keamanan ke node AWS.networking.ENI.	April 2, 2024
Dukungan untuk enkripsi volume root Amazon EBS	Anda dapat mengaktif kan enkripsi Amazon EBS untuk volume root Amazon EBS. Untuk mengaktif kan, tambahkan properti di AWS.compute. EKSManage dNode atau AWS.compute. EKSSelfManagedNodesimpul.	April 2, 2024
Support untuk node `labels`	Anda dapat melampirkan label node ke grup node Anda di AWS.compute. EKSManage dNode atau AWS.compute. EKSSelfManagedNodesimpul.	Maret 19, 2024
Support untuk antarmuka jaringan `source_de st_check`	Anda dapat menunjukkan apakah Anda ingin mengaktif kan atau menonaktifkan source/destination pemeriksa an antarmuka jaringan melalui node AWS.networking.ENI.	Januari 25, 2024

Support untuk EC2 instans Amazon dengan data pengguna kustom	Anda dapat meluncurkan EC2 instans Amazon dengan data pengguna khusus melalui AWS.Compute. UserData simpul.	Januari 16, 2024
Support untuk Grup Keamanan	AWS TNB memungkinkan Anda untuk mengimpor AWS sumber daya Grup Keamanan.	8 Januari 2024
Deskripsi yang diperbarui dari `network_interfaces`	Ketika `network_interfaces` properti termasuk dalam AWS.compute. EKSManagedNode atau AWS.compute. EKSSelfManagedNodenode, AWS TNB mendapat izin yang terkait dengan ENIs dari `multus_role` properti jika tersedia, atau dari `node_role` properti.	18 Desember 2023
Support untuk private cluster	AWS TNB sekarang mendukung klaster pribadi. Untuk menunjukkan klaster pribadi, atur `access` properti ke`PRIVATE`.	Desember 11, 2023
Versi Kubernetes untuk klaster	AWS TNB sekarang mendukung Kubernetes versi 1.28 untuk membuat cluster Amazon EKS.	Desember 11, 2023
AWS TNB mendukung grup penempatan	Ditambahkan grup penempatan untuk definisi `AWS.Compute.EKSManagedNode` dan `AWS.Compute.EKSSelfManagedNode` node.	Desember 11, 2023

AWS TNB menambahkan dukungan untuk IPv6	AWS TNB sekarang mendukung pembuatan instance jaringan dengan IPv6 infrastruktur. Periksa node AWS.networking.vpc, .networking.subnet, .networking.AWSAWS InternetGateway, AWS.Jaringan. SecurityGroupIngressRule, AWS.Jaringan. SecurityGroupEgressRule, dan AWS.compute.eks untuk konfigurasi. IPv6 Kami juga menambahkan node AWS.Networking. NATGatewaydan AWS.Networking.Route untuk konfigurasi. NAT64 Kami memperbarui peran layanan AWS TNB dan peran layanan AWS TNB untuk grup node Amazon EKS untuk IPv6 izin. Lihat Contoh kebijakan peran layanan.	16 November 2023
Menambahkan izin ke kebijakan peran layanan AWS TNB	Kami menambahkan izin ke kebijakan peran layanan AWS TNB untuk Amazon S3 AWS CloudFormation dan mengaktifkan instantiasi infrastruktur.	23 Oktober 2023
AWS TNB diluncurkan di lebih banyak wilayah	AWS TNB sekarang tersedia di Asia Pasifik (Seoul), Kanada (Tengah), Eropa (Spanyol), Eropa (Stockholm), dan Amerika Selatan (São Paulo).	27 September 2023

Tag untuk AWS.Compute. EKSSelfManagedNode	AWS TNB sekarang mendukung tag untuk definisi `AWS.Compute.EKSSelfManagedNode` node.	22 Agustus 2023
AWS TNB mendukung instans yang memanfaatkan IMDSv2	Saat meluncurkan instance Anda, Anda harus menggunakan IMDSv2.	Agustus 14, 2023
Izin yang diperbarui untuk MultusRoleInlinePolicy	MultusRoleInlinePolicy Sekarang termasuk `ec2:DeleteNetworkInterface` izin.	Agustus 7, 2023
Versi Kubernetes untuk klaster	AWS TNB sekarang mendukung Kubernetes versi 1.27 untuk membuat kluster Amazon EKS.	25 Juli 2023
AWS.compute.eks. AuthRole	AWS Dukungan TNB AuthRole yang memungkinkan Anda menambahkan peran IAM ke kluster Amazon EKS `aws-auth ConfigMap` sehingga pengguna dapat mengakses kluster Amazon EKS menggunakan peran IAM.	Juli 19, 2023
AWS TNB mendukung kelompok keamanan.	Menambahkan AWS.Networking. SecurityGroup, AWS.Jaringan. SecurityGroupEgressRule, dan AWS.Networking. SecurityGroupIngressRuleke template NSD.	Juli 18, 2023

Versi Kubernetes untuk klaster	AWS TNB mendukung Kubernetes versi 1.22 hingga 1.26 untuk membuat klaster Amazon EKS. AWS TNB tidak lagi mendukung Kubernetes versi 1.21.	11 Mei 2023
AWS.Menghitung. EKSSelfManagedNode	Anda dapat membuat node pekerja yang dikelola sendiri di in-region, AWS Local Zones, dan. AWS Outposts	29 Maret 2023
Rilis awal	Ini adalah rilis pertama dari Panduan Pengguna AWS TNB.	21 Februari 2023

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.